

Oracle® Demantra

Implementation Guide

Release 7.3

Part No. E05136-11

November 2010

Oracle Demantra Implementation Guide, Release 7.3

Part No. E05136-11

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Primary Author: Diane Westgate, Greg Watkins

Contributing Author: Kyle MacLean, Jason Lansdowne

Oracle is a registered trademark of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this software or related documentation is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS

Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, duplication, disclosure, modification, and adaptation shall be subject to the restrictions and license terms set forth in the applicable Government contract, and, to the extent applicable by the terms of the Government contract, the additional rights set forth in FAR 52.227-19, Commercial Computer Software License (December 2007). Oracle USA, Inc., 500 Oracle Parkway, Redwood City, CA 94065.

This software is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications which may create a risk of personal injury. If you use this software in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy and other measures to ensure the safe use of this software. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software in dangerous applications.

This software and documentation may provide access to or information on content, products and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third party content, products and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third party content, products or services.

Contents

Send Us Your Comments

Preface

Part 1 Concepts and Tools

1 Introduction to Demantra

Demantra Platform and Applications.....	1-1
Extending your Demantra Application.....	1-2
Elements of a Demantra Solution.....	1-2
Integration.....	1-6
Workflow.....	1-6
How the User Interfaces Can Be Configured.....	1-7

2 Core Concepts

Worksheets.....	2-1
The Basic Input Data.....	2-3
Time Resolution.....	2-3
Levels.....	2-3
Combinations.....	2-5
Series.....	2-6
Filtering and Exceptions.....	2-7
Methods.....	2-8
Security.....	2-9
Forecasting.....	2-9

3 How Demantra Calculates and Stores Data

How Data Is Stored.....	3-1
How Data Is Calculated at the Lowest Level.....	3-2
How Data Is Calculated at Higher Levels.....	3-3
How Proportions Are Chosen.....	3-5
How Proportions Are Used.....	3-6
How Splitting Handles Null Values.....	3-7
How Splitting Handles Zero Values.....	3-7
When and How Data Is Saved to the Database.....	3-8

4 Implementation Tools and Process

Overview of Tools.....	4-1
Initial Phase of Implementation.....	4-5
Middle Phase of Implementation.....	4-5
End Phase of Implementation.....	4-9
Hand-off Details.....	4-10

5 Levels

Introduction.....	5-1
Level Terminology.....	5-4
Hierarchy Variations.....	5-5
Types of Levels.....	5-6
Members.....	5-9
Member Identifiers.....	5-10
Introduction to the Data Model Wizard.....	5-12
Levels and the Forecast Tree.....	5-13
Filtered Levels.....	5-14
Other Concepts Related to Levels.....	5-14
Configuration Notes.....	5-15

6 Series

Introduction.....	6-2
Main Series Options.....	6-2
Data Types of Series.....	6-3
Types of Series.....	6-3
Update Field.....	6-5
Editability.....	6-5
Series Calculation: Server Expressions.....	6-6

Series Calculation: Client Expressions.....	6-8
Series Calculation: Using Both Server and Client Expressions.....	6-11
Series Calculation: Proportionality Option.....	6-11
Summary of Calculation Options.....	6-12
Display Properties.....	6-18
Color Expressions.....	6-18
Display Precision and Format.....	6-18
Display-Only Summaries.....	6-18
Configuring the Display-Only Summary.....	6-20
Other Basic Series Options.....	6-22
Advanced Series Options.....	6-23
Preserving Promotional Data While Moving or Copying.....	6-23
The Aggregated Base Level Option.....	6-27
Extra From and Extra Where.....	6-28
Note and Promotion Indicators.....	6-29
Series Groups.....	6-29
Configuration Notes.....	6-30

7 Units, Indexes, and Exchange Rates

Introduction.....	7-1
Unit Conversion Data.....	7-2
How Units Are Used.....	7-3
Time Units.....	7-3
Setting and Modifying the Base Time Unit.....	7-5
Unit-Level Association.....	7-7
Indexes and Exchange Rates.....	7-7
Configuration Notes.....	7-8

8 Worksheets

Introduction.....	8-1
Main Worksheet Options.....	8-2
Elements of a Worksheet View.....	8-5
Level Layout in a View.....	8-6
Filtering per View.....	8-8
Level and Worksheet Association: Embedded Worksheets.....	8-8
Worksheet and Layout Ownership.....	8-9
Worksheet Caching.....	8-11
Configuration Notes.....	8-13

9 Methods and Workflow

Introduction.....	9-1
Overview of Method Types.....	9-3
Overview of Workflow Step Types.....	9-6
Typical Step Properties.....	9-13
Passing Arguments to a Method.....	9-14
Workflow Schema Examples.....	9-18
Configuration Notes.....	9-19

10 Security

Data Security.....	10-1
Feature Security.....	10-4
Other Security Features.....	10-8
Program Groups.....	10-9
Configuration Notes.....	10-10
Password Policy.....	10-10

11 Proport

Overview.....	11-1
How You Can Tune the Proport Mechanism.....	11-3
Calculating the Rolling Average Demand.....	11-3
Calculating the Monthly Proportions.....	11-3
Calculating the Average Daily Demand for Each Month.....	11-4
Handling Data Issues.....	11-6
Which Combinations Are Affected.....	11-9
Other Notes on the Proport Mechanism.....	11-10
Proport when using Engine Profiles.....	11-11
Proport When Forecasting on General Levels.....	11-11

Part 2 Integration

12 General Integration Guidelines

"Open With" Worksheets.....	12-1
Worksheet Filters.....	12-1
Changing System Time Resolution.....	12-2
Analytical Engine Guidelines.....	12-4
DM application default users.....	12-5
Controlling System and Engine Max Sales Dates.....	12-6

13 Demantra Data Tables and Integration Processes

Demantra Data Tables.....	13-1
Data Assumptions and Requirements.....	13-3
Loading Basic Data.....	13-5
Loading Series and Promotions.....	13-6
Executing Integration Interfaces.....	13-6
Configure Series Load and Purge Options.....	13-13
Loading Supplementary Data.....	13-16
Exporting Data.....	13-17
Configuration Notes.....	13-17

14 Demantra Demand Management to EBS Integration

Demand Management Business Flow.....	14-2
Terms and Conventions Used in this Document.....	14-5
Integration Features.....	14-6
Seeded Demand Management Component.....	14-11
Summary of Integration Tasks.....	14-16
Initial Setup.....	14-17
Download E-Business Suite Data Into Demand Management.....	14-23
Download Collections.....	14-24
Combined Collections of Shipment and Booking History.....	14-25
Collecting Legacy Shipment and History Data.....	14-32
Collecting Returns History Data.....	14-35
Collecting Currency Conversion Data	14-38
Associating multiple Price Lists with a CTO worksheet.....	14-40
Collecting Unit of Measure Conversion Data.....	14-41
Collecting Price List Data.....	14-43
Downloading Calendars.....	14-45
Purging Data Before Import.....	14-46
Configure Series Load and Purge Options.....	14-48
Download to Oracle Demantra.....	14-48
Demand Management Functional Output.....	14-50
Upload from Oracle Demantra.....	14-51
Publish the Forecast to CP.....	14-53
Upload Forecast.....	14-54
Line Of Business Configuration and Execution.....	14-56
LOB Procedures.....	14-57
Troubleshooting the Line of Business Forecast Workflow.....	14-63
Setting Up the New Products List.....	14-64

Setting Up the Calendar List.....	14-65
Base Time Unit.....	14-70
Creating a New Leaf Level.....	14-70
Creating a New Top Level.....	14-71
Creating a New Intermediate Level.....	14-73
Deleting a Level.....	14-74
Approval and Upload Setup Process.....	14-74
Profile Options.....	14-76

15 Demantra Demand Management to EBS Service Parts Planning Integration

Introduction.....	15-1
Integration Points.....	15-3
Business Process.....	15-3
Service Parts Planning Navigator Menus.....	15-4
Integration Considerations.....	15-4
Initial Demantra Setup.....	15-6
Initial SPP/EBS Setup.....	15-7
Importing SPP Data into Demantra.....	15-12
Launching the Import Workflows from the Demantra Workflow Manager.....	15-15
Service Parts Forecasting in Demantra.....	15-16
Exporting SPF Data from Demantra to Service Parts Planning in EBS.....	15-17
Integration Workflows and Integration Profiles.....	15-18
Service Parts Forecasting Integration Import Profiles.....	15-21
Service Parts Forecasting Integration Export Profiles.....	15-23

16 Demantra Sales and Operations Planning to EBS Integration

Introduction.....	16-1
Integration Points.....	16-2
Sales and Operations Planning Navigator Menus.....	16-2
General Configuration.....	16-4
Integration Workflows and Interface Profiles.....	16-5
Demantra Sales and Operations Planning to Strategic Network Optimization Integration	16-14
Demantra Sales and Operations Planning to Advanced Supply Chain Planning Integration	16-17
Demantra Sales and Operations Planning to Rapid Planning Integration.....	16-19
SCI Data.....	16-22

17 Demantra Demand Management to EnterpriseOne Integration

Overview.....	17-1
---------------	------

Architectural Process.....	17-2
Integration Points Overview.....	17-2
Business Process.....	17-3
Mapping.....	17-5
18 Demantra Predictive Trade Planning to EnterpriseOne Integration	
Overview.....	18-1
Architectural Process.....	18-3
Integration Points Overview.....	18-3
Business Process.....	18-4
Promotion Method Configuration.....	18-7
Modeling Considerations.....	18-7
Mapping.....	18-11
19 Demantra Deductions and Settlement Management to EnterpriseOne Integration	
Overview.....	19-1
Architectural Process.....	19-2
Integration Points.....	19-2
Business Process.....	19-3
Modelling Considerations.....	19-5
Mapping.....	19-10
20 Demantra Predictive Trade Planning to Siebel Integration	
Introduction.....	20-1
Architectural Process.....	20-2
Integration Point Overview.....	20-3
Business Process.....	20-5
Siebel Trade Planning/Demantra Intelligence Business Process (Regular Synchronization)	20-5
Demantra Predictive Trade Planning/Siebel Trade Management (Near Real-time and Regular Synchronization).....	20-6
Setup Overview.....	20-9
Siebel Setup.....	20-10
ODI Setup.....	20-13
Overview of the ODI Applications.....	20-13
Understanding Packages and Scenarios.....	20-15
Configuring the Demantra Database Connection.....	20-15
Configuring the Demantra Agent.....	20-18
Configuring ODI Variables.....	20-19

Demantra Setup	20-24
Main Batch File Configurations	20-28
Running Integration	20-31
Siebel to Demantra ODI Packages and Batch Files	20-34
Account Hierarchy Planning.....	20-34
Plan Master List.....	20-38
Pricing and Cost.....	20-40
Product Planning Hierarchy.....	20-44
Promotion Status.....	20-48
Promotions.....	20-49
Promotional Tactics.....	20-63
Sales Activity.....	20-65
Demantra to Siebel ODI Packages and Batch Files	20-66
Baseline Volume Forecast.....	20-66
Promotional Lift Coefficients.....	20-68
Promotions.....	20-71

21 Demantra Sales and Operations Planning to Hyperion Integration

Demantra Sales and Operations Planning - Hyperion Planning Integration Overview	21-1
Architectural Process	21-2
Integration Points Overview	21-2
Business Processes	21-3

22 Integration with Demand Signal Repository

Solution Overview	22-1
Business Process.....	22-1
Overview of Procedures.....	22-2
Objects for DSR Integration.....	22-2
Importing DSR Data into Demantra	22-3
Running Multiple Forecasts	22-3

Part 3 Basic Configuration

23 Getting Started with the Configuration Tools

About Demantra Configuration Tools	23-1
Illegal Characters in Demantra	23-3
Desktop and Business Modeler Automatic Install	23-4
Logging Onto the Business Modeler	23-6
Refreshing the Data in the Business Modeler	23-6

Working with Lists.....	23-7
Configuring Parameters.....	23-7
Making Changes Available to Users.....	23-8
Quitting Business Modeler.....	23-9
Logging into the Workflow Manager.....	23-9
Quitting the Workflow Manager.....	23-9
Setting Browser Locale	23-9

24 Database Tools

Creating a Table.....	24-1
Modifying a Table.....	24-2
Recompiling the Database Functions and Procedures.....	24-3
Viewing the Procedure Error Log.....	24-3
Wrapping Database Procedures.....	24-5
Cleaning Up Temporary Tables.....	24-6
Monitoring Oracle Sessions.....	24-6
Index Adviser.....	24-6

25 Using the Data Model Wizard

About the Data Model Wizard.....	25-2
Before Using the Data Model Wizard.....	25-2
Getting Started with the Data Model Wizard.....	25-3
Describing the Staging Tables.....	25-4
Specifying the Structure of the Staging Tables.....	25-6
Joining Multiple Source Files or Tables.....	25-8
Defining the Minimal Required Data Model.....	25-9
Declaring the Sales Date.....	25-10
Declaring the Sales Quantity.....	25-10
Defining an Item Level and a Location Level.....	25-11
Saving the Data Model.....	25-12
Defining Additional Elements.....	25-13
Declaring the Unit Price.....	25-13
Defining a Unit of Measure.....	25-13
Adding Higher Levels.....	25-14
Adding Level Attributes.....	25-15
Defining Other Values.....	25-16
Impacts of Changing Levels, Series, and Units.....	25-16
Navigating the Data Model.....	25-21
Building the Data Model.....	25-22
Loading the Data into the Data Model.....	25-24

Manipulating Existing Data Models.....	25-24
----------------------------------------	-------

26 Configuring Levels

Before Configuring Levels.....	26-1
Creating a Level.....	26-2
Filtering a Level.....	26-4
Configuring an Existing Level.....	26-6
Adding Attributes to a Level.....	26-9
Dropdown Security.....	26-12
Filtering an Attribute Drop-down List.....	26-14
Specifying Default Parents for a New Member.....	26-15
Adding a Population Attribute to a General Level.....	26-18
Creating a time aggregation.....	26-20
Viewing the Members of a Level.....	26-21
Removing Levels.....	26-22

27 Configuring Series and Series Groups

Before Configuring Series.....	27-2
Creating a Series.....	27-2
Creating a New Series Based on an Existing Series.....	27-5
Specifying General Properties of a Series.....	27-5
Specifying How to Display a Series.....	27-8
Configuring a Dropdown-Style Series.....	27-12
Filtering a Series Drop-down List.....	27-14
Specifying Data Properties of a Series.....	27-16
Using the Expression Editors.....	27-21
Syntax of Server Expressions.....	27-23
Syntax of Client Expressions.....	27-24
Specifying Server and Client Expressions.....	27-25
Creating an Edit-Lock Expression.....	27-28
Creating a Color Expression.....	27-29
Controlling Access to Series.....	27-32
Configuring Desktop Group Expressions.....	27-33
Deleting a Series.....	27-35
Enabling Series Caching By Item.....	27-35
Specifying the Order of Series in Dynamic Open Link.....	27-35
Creating or Modifying a Series Group.....	27-37
Deleting a Series Group.....	27-37
Viewing Dependencies Among Series.....	27-37

28 Configuring Units, Indexes, and Update-Lock Expressions

Before Configuring Units and Indexes.....	28-1
Configuring Indexes and Exchange Rates.....	28-1
Editing Values for Indexes and Exchange Rates.....	28-4
Configuring Units.....	28-5
Associating Units with Levels.....	28-8
Configuring Time Units.....	28-10
Configuring Update-Lock Expressions.....	28-11

29 Series and Level Integration

Before Using the Integration Interface Wizard.....	29-1
Creating or Modifying an Integration Interface.....	29-2
Creating a Data Import Profile.....	29-4
Creating a Data Export Profile.....	29-10
Specifying Series Data to Import or Export.....	29-14
Creating an Export Profile for Any Level.....	29-19
Creating an Import Profile for a General Level.....	29-23
Deleting an Integration Interface.....	29-27
Details of the Staging Tables.....	29-28
Executing an Integration Interface.....	29-34
Checking the Integration Queue.....	29-35

30 Importing Supplementary Data

Creating or Modifying a File Load Interface.....	30-1
Deleting a File Load Interface.....	30-2
Creating an Import Profile.....	30-2
Creating the Import Batch File.....	30-4

31 Creating Workflows

Creating or Editing Workflow Schemas.....	31-1
Parameters Used as Arguments for a Workflow.....	31-4
Deleting Workflow Schemas.....	31-12

32 Configuring Methods

Configuring a New Level Method.....	32-1
Passing Arguments to a Method.....	32-9
Modifying a Level Method.....	32-11
Deleting a Level Method.....	32-12

33 Using the Desktop BLE User Interface

Overview of the Business Logic Engine.....	33-1
--------------------------------------------	------

34 Non-Engine Parameters

About These Parameters.....	34-1
System Parameters.....	34-1

35 Database Procedures

Recommended Procedure Scheduling.....	35-2
ANALYZE_SCHEMA.....	35-3
API_CONFIG_SETTLEMENT.....	35-3
APPPROC_REBUILD_CONTEXT_CACHES.....	35-3
CHAINING.....	35-5
CLEAN_LOG_TABLES.....	35-5
COMPUTE_STD_ERR.....	35-6
CREATE_OPT_STRUCT.....	35-6
CREATE_PE_STRUCT.....	35-6
DELETE_FORE_COL.....	35-6
DELETE_INVALID_GL_POPULATION.....	35-6
DROP_TEMP_TABLES.....	35-6
EP_LOAD_MAIN.....	35-7
EP_LOAD_MDP_LEVEL.....	35-7
EP_LOAD_ITEMS.....	35-10
EP_LOAD_LOCATIONS.....	35-13
EP_LOAD_SALES.....	35-16
EXECUTE_PROFILES.....	35-23
EXPOSE_PROMOTIONS.....	35-23
INSERT_UNITS.....	35-23
MANUALS_INS.....	35-25
MANUALS_INS_INTEGRATION.....	35-25
MANUALS_INS_RECOVERY.....	35-25
MDP_ADD.....	35-25
POP_ALL_MATCH_PROPOSAL.....	35-33
POP_OI_MATCH_PROPOSAL.....	35-34
PRE_LOGON.....	35-34
REBUILD_INDEXES.....	35-34
REBUILD_SCHEMA.....	35-34
REBUILD_TABLES.....	35-35
REPLACE_APOSTROPHE_LEVELS.....	35-35

SCHEDULED_CLEANUP_TASKS.....	35-35
UPGRADE_TO_SHAPE_MODELLING.....	35-35

36 Key Tables

Sales_Data.....	36-1
Mdp_matrix.....	36-3
Promotion_Data.....	36-10

37 Server Expression Functions and Operators

Supported SQL Functions.....	37-1
Operators in Server Expressions.....	37-4
Oracle Tokens.....	37-5

38 Client Expression Functions and Operators

About This Reference.....	38-2
Operators in Client Expressions.....	38-2
Abs Function.....	38-3
Case.....	38-4
CurrentRow.....	38-5
Date.....	38-6
Day.....	38-6
Exp.....	38-6
ForecastFirstRow.....	38-7
Fpos.....	38-7
Fsum.....	38-8
GetRow.....	38-9
GetWorksheetLevelNumber.....	38-10
If.....	38-12
Is_Modified.....	38-12
IsNull.....	38-13
Mod.....	38-13
Month.....	38-13
Pi.....	38-14
Rand.....	38-14
Round.....	38-15
RowCount.....	38-16
Sqrt.....	38-16
SummaryAVG.....	38-16
SummaryCount.....	38-17
SummaryMax.....	38-17

SummaryMin.....	38-17
SummarySum.....	38-17
SummaryWAVG.....	38-18
Today.....	38-18
Truncate.....	38-18
Year.....	38-19
Z_Val.....	38-20

39 Workflow Steps

Specifying a Task.....	39-2
BLE Step.....	39-3
Condition Step.....	39-5
Container Step.....	39-7
Create Member Step.....	39-9
Custom Step.....	39-10
Delete Member Step.....	39-13
Edit Member Step.....	39-13
Email Step.....	39-14
Exception Step.....	39-16
Executable Step.....	39-19
Group Step.....	39-21
Paste Member Step.....	39-24
Refresh Population Step.....	39-25
Selection Step.....	39-27
Simulation Step.....	39-30
Stored Procedure Step.....	39-33
Transfer Step.....	39-34
Update Data Step.....	39-35
User Step.....	39-36
Wait Until Step.....	39-38
Worksheet Cache Step.....	39-41
Worksheet Step.....	39-44

Part 4 Configuring Specific Applications

40 Configuring Predictive Trade Planning

Overview of Promotion Effectiveness.....	40-2
Overview of the Configuration Process.....	40-2
Configuring Levels for Promotion Effectiveness.....	40-2

Setting Parameters.....	40-3
Configuring the Activity Browser.....	40-3
Configuring the Application Server.....	40-4
Configuring Promotion Statuses.....	40-4
Loading Promotion Data.....	40-6
Reference: PE Levels.....	40-7
PTP Data Needs.....	40-9
Integration in PTP.....	40-10
Reference: Staging Tables.....	40-11
Reference: Integration Interfaces.....	40-17
Configuring the Default Promotion Start and Span.....	40-18
Changing the Default Retailer Profile.....	40-18
Configuring the Right-Click "Open With" Option.....	40-19
Removing Right-Click Menu Options.....	40-21
Replacing Collaborator Workbench Graphics.....	40-21
Configuring Promotion Status Behavior.....	40-22
Re-configuring Series if Syndicated Data Is Not Used.....	40-24

41 Configuring Promotion Optimization for Predictive Trade Planning

Overview of the Configuration Process.....	41-1
Set Up Promotion Optimization Without Using the Installer.....	41-2
Configuring the Optimization Step.....	41-2
Other Important Notes.....	41-6

42 Configuring Deductions and Settlement Management

Overview of DSM.....	42-1
Data Flow in DSM.....	42-4
Overview of the Configuration Process.....	42-5
Setting Up Database Structures.....	42-6
Configuring Promotions and Promotion Series.....	42-8
Identifying Key Promotion Series.....	42-11
Configuring Settlement Matching.....	42-11
Configuring Write-Offs.....	42-12
Loading Initial Data and Creating Possible Matches.....	42-13
Describing Customer Relationships.....	42-14
In Case of Problems.....	42-14
Reference: DSM Levels.....	42-15
Reference: DSM Series.....	42-23
Reference: DSM Integration Interfaces.....	42-23
Reference: Other DSM Tables.....	42-31

Part 5 Other Configuration

43 Fine Tuning and Scaling Demantra

Basic Parameters.....	43-2
Application Server.....	43-2
Collaborator Workbench.....	43-2
Database.....	43-5
Date/Time Formats.....	43-8
Email.....	43-9
Integration.....	43-11
Item Aggregation.....	43-12
Logs and Old Data.....	43-12
Proport Mechanism.....	43-13
EngineOutputThreshold.....	43-15
Simulation.....	43-16
Solution Branding.....	43-17
Threading.....	43-17
Workflow.....	43-21
Worksheets.....	43-21
Languages.....	43-26
Configuring User Productivity Kit (UPK)	43-30

44 Customizing Demantra Web Pages

Logging onto the Collaborator Workbench Administrator.....	44-1
Configuring Menus in Collaborator Workbench.....	44-2
Running Oracle Executables from Collaborator Menus.....	44-6
Configuring the Panes.....	44-7
Specifying Content Pane Security.....	44-10
Replacing Default Demantra Graphics.....	44-10
Customizing the Collaborator Workbench Login Page.....	44-12
Configuring Links in Collaborator Workbench.....	44-12

45 Configuring Rolling Data

About Rolling Data.....	45-1
Creating or Modifying Rolling Data Profiles.....	45-2
Executing Rolling Data Profiles.....	45-3
Creating Rolling Data Profile Groups.....	45-4

46	GL Synchronization	
	Overview.....	46-1
47	Performing Constraint Profit Optimization	
	About Constraint Profit Optimization.....	47-1
	Creating or Modifying an Optimization Profile.....	47-2
	Deleting a Constraint Optimization Profile.....	47-8
	Running a Constraint Optimization Profile.....	47-8
48	Creating a Standard Error Process	
	Creating a Standard Error Process.....	48-1

Part 6 Configuring and Running the Analytical Engine

49	Introduction to the Analytical Engine	
	Overview.....	49-1
	Engine Modes: DP and PE.....	49-2
	What the Engine Does.....	49-3
	Batch and Simulation Modes.....	49-3
	Engine Profiles.....	49-4
	Specifying the Demand Stream in an Engine Profile.....	49-7
	Maintaining Engine Versions.....	49-9
	Illegal Characters in Demantra.....	49-10
50	Basic Concepts	
	Overview of Forecasting.....	50-1
	Causal Factors.....	50-2
	Promotions (PE Mode Only).....	50-6
	Forecasting Models and the Engine Flow.....	50-8
	The Forecast Tree.....	50-8
	Influence and Switching Effects (PE Mode Only).....	50-13
	Combination-Specific Settings.....	50-16
	The Forecast Data.....	50-20
51	Configuring the Analytical Engine	
	Configuring the Engine.....	51-1
	General Data Requirements.....	51-2

Structure and Requirements of the Forecast Tree.....	51-3
Split Forecast by Series.....	51-5
Configuring SALES_DATA node-splitting.....	51-5
Guidelines for the Forecast Tree.....	51-6
Guidelines for Causal Factors.....	51-7
Setting Up Configure to Order.....	51-8

52 Configuring the Forecast Tree

Configuring the Forecast Tree.....	52-1
Pooled Time Series.....	52-5
Defining Influence and Competition (PE Mode Only).....	52-8
Defining the Forecast Tree for Service Parts Planning Supersessions.....	52-9
Specifying Additional Parameters.....	52-10

53 Configuring Causal Factors

Notes About Causal Factors.....	53-1
Creating a Global Factor.....	53-3
Creating a Local Causal Factor.....	53-5
Configuring Global and Local Causal Factors.....	53-6
About Activity Shape Modeling.....	53-10
Enabling Activity Shape Modeling.....	53-12
Deleting a Causal Factor.....	53-13

54 Configuring Promotions and Promotional Causal Factors

Base Behavior.....	54-1
Customizing the Promotion Levels.....	54-3
Loading Historical Promotions.....	54-3
How the Analytical Engine Uses Promotions.....	54-4
Configuring Promotional Causal Factors.....	54-8
Adjusting the Promotion Dates.....	54-13
About Promotion Shape Modeling.....	54-14
Enabling Promotion Shape Modeling.....	54-15

55 Tuning the Analytical Engine

Editing Engine Parameters.....	55-1
Creating or Renaming Engine Profiles.....	55-2
Tuning Analytics.....	55-3
Tuning Performance.....	55-6
Reconfiguring the sales_data_engine Table.....	55-11

Enabling Engine Models Globally.....	55-14
Configuring the Engine Mode.....	55-16
Advanced Analytics (Nodal Tuning).....	55-16
56 Using the Engine Administrator and Running the Engine	
Before Running the Analytical Engine.....	56-1
General Notes about Running the Analytical Engine.....	56-2
Registering the Analytical Engine.....	56-3
Starting the Engine Administrator.....	56-3
Configuring Engine Settings.....	56-4
Running the Engine from the Engine Administrator.....	56-8
Running the Engine from the Start Menu.....	56-11
Running the Engine from the Command Line.....	56-11
Troubleshooting.....	56-12
Viewing the Engine Log.....	56-15
Examining Engine Results.....	56-16
Running the Engine in Recovery Mode.....	56-17
Stopping the Engine.....	56-17
57 Engine Details	
Preparing the Database.....	57-1
Promotion Effectiveness Engine Phases	57-2
The Forecasting Process.....	57-6
Comparison of Batch and Simulation Modes.....	57-17
Engine Components and High-Level Flow.....	57-17
Details of the Distributed Engine.....	57-20
58 Engine Parameters	
About Engine Parameters.....	58-1
Analytical Engine Parameters.....	58-1
59 Theoretical Engine Models	
Introduction.....	59-2
Flags on Causal Factors.....	59-2
ARIX.....	59-3
ARLOGISTIC.....	59-4
ARX.....	59-4
BWINT.....	59-5
CMREGR.....	59-6

DMULT.....	59-8
ELOG.....	59-10
FCROST.....	59-12
HOLT.....	59-13
ICMREGR.....	59-15
IREGR.....	59-16
LOG.....	59-17
LOGISTIC.....	59-18
Moving Average.....	59-19
MRIDGE.....	59-20
NAIVE.....	59-21
REGR.....	59-23

Part 7 Administration

60 Administering Demantra

Keeping the System Running.....	60-1
Periodic Maintenance Tasks.....	60-2
Data Cleanup Script.....	60-3
Demantra URLs.....	60-5
Log Files and Tables.....	60-9
Illegal Characters in Demantra.....	60-10
Required Third-Party Software.....	60-10

61 Managing Security

Creating or Modifying a Component.....	61-1
Deleting a Component.....	61-10
Creating or Modifying a User.....	61-10
Copying a User.....	61-15
Deleting a User.....	61-16
Creating or Modifying a User Group.....	61-16
Deleting a Group.....	61-19
Providing Access to the Workflow Editor.....	61-19
Logging onto the Collaborator Workbench Administrator.....	61-20
Defining a Program Group.....	61-20
Redefining a Program Group.....	61-24
Deleting a Program Group.....	61-25
Specifying Permissions for Menu Items.....	61-25
Logging Out Users.....	61-28

62 Managing Workflows

Viewing Workflow Status.....	62-1
Starting Workflow Instances.....	62-3
Scheduling Workflow Instances.....	62-3
Stopping Workflow Instances.....	62-6
Creating or Editing a Schema Group.....	62-7
Deleting a Schema Group.....	62-8
Viewing the Workflow Process Log.....	62-9
Recovery and the Workflow Engine.....	62-9
Web Services for Demantra Workflows.....	62-10

63 Managing Worksheets

Viewing the Worksheets.....	63-1
Changing Worksheet Ownership.....	63-2
Changing Worksheet Access.....	63-2
Deleting a Worksheet.....	63-3

64 Other Administration

Other Web-based Administration Tools.....	64-1
Logging Messages of the Application Server.....	64-2
Managing Level Caching.....	64-2
Viewing and Modifying Cache Properties.....	64-5

65 Upgrading Using Oracle Demantra Business Application Language

Overview.....	65-1
The Upgrade Process.....	65-2
BAL Explorer Screen Elements.....	65-4
Using the BAL Explorer.....	65-6
Understanding Objects, Schemas, Folders, Files and Repositories.....	65-6
Schema Procedures.....	65-8
Folder Procedures.....	65-14
Upgrading Procedures.....	65-15
Troubleshooting.....	65-32

66 Tips and Troubleshooting

Initial Logon to Demantra.....	66-1
About Demantra Configuration Settings.....	66-5
Key Settings Controlled by the Installer.....	66-6

Redirecting Demantra to a Different Database.....	66-8
Java Tips.....	66-9
Tomcat Tips.....	66-10
Error Messages or Trouble Descriptions.....	66-12

A Key Tables

T_SRC Tables.....	A-1
-------------------	-----

Index

Send Us Your Comments

Oracle Demantra Implementation Guide, Release 7.3

Part No. E05136-11

Oracle welcomes customers' comments and suggestions on the quality and usefulness of this document. Your feedback is important, and helps us to best meet your needs as a user of our products. For example:

- Are the implementation steps correct and complete?
- Did you understand the context of the procedures?
- Did you find any errors in the information?
- Does the structure of the information help you with your tasks?
- Do you need different information or graphics? If so, where, and in what format?
- Are the examples correct? Do you need more examples?

If you find any errors or have any other suggestions for improvement, then please tell us your name, the name of the company who has licensed our products, the title and part number of the documentation and the chapter, section, and page number (if available).

Note: Before sending us your comments, you might like to check that you have the latest version of the document and if any concerns are already addressed. To do this, access the new Oracle E-Business Suite Release Online Documentation CD available on My Oracle Support and www.oracle.com. It contains the most current Documentation Library plus all documents revised or released recently.

Send your comments to us using the electronic mail address: appsdoc_us@oracle.com

Please give your name, address, electronic mail address, and telephone number (optional).

If you need assistance with Oracle software, then please contact your support representative or Oracle Support Services.

If you require training or instruction in using Oracle software, then please contact your Oracle local office and inquire about our Oracle University offerings. A list of Oracle offices is available on our Web site at www.oracle.com.

Preface

Intended Audience

Welcome to Release 7.3 of the *Oracle Demantra Implementation Guide*.

See Related Information Sources on page xxxiii for more Oracle E-Business Suite product information.

Deaf/Hard of Hearing Access to Oracle Support Services

To reach Oracle Support Services, use a telecommunications relay service (TRS) to call Oracle Support at 1.800.223.1711. An Oracle Support Services engineer will handle technical issues and provide customer support according to the Oracle service request process. Information about TRS is available at

<http://www.fcc.gov/cgb/consumerfacts/trs.html>, and a list of phone numbers is available at <http://www.fcc.gov/cgb/dro/trsphonebk.html>.

Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible to all users, including users that are disabled. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. Accessibility standards will continue to evolve over time, and Oracle is actively engaged with other market-leading technology vendors to address technical obstacles so that our documentation can be accessible to all of our customers. For more information, visit the Oracle Accessibility Program Web site at <http://www.oracle.com/accessibility/>.

Accessibility of Code Examples in Documentation

Screen readers may not always correctly read the code examples in this document. The

conventions for writing code require that closing braces should appear on an otherwise empty line; however, some screen readers may not always read a line of text that consists solely of a bracket or brace.

Accessibility of Links to External Web Sites in Documentation

This documentation may contain links to Web sites of other companies or organizations that Oracle does not own or control. Oracle neither evaluates nor makes any representations regarding the accessibility of these Web sites.

Structure

1 Introduction to Demantra

This chapter provides an overview of the Demantra platform.

2 Core Concepts

This chapter explains worksheets and other basic concepts.

3 How Demantra Calculates and Stores Data

As an implementor, you should understand how Demantra calculates and stores data, because this can affect how you set up your solution. This chapter provides an overview of most important details.

4 Implementation Tools and Process

This chapter provides a quick overview of the implementation and hand-off processes and tools you use during these processes.

5 Levels

This chapter describes levels and related concepts, outlines the primary configuration options, and summarizes the available tools.

6 Series

7 Units, Indexes, and Exchange Rates

This chapter describes units and related concepts, outlines the primary configuration options, and summarizes the available tools.

8 Worksheets

This chapter describes worksheets, outlines the primary configuration options, and summarizes the available tools.

9 Methods and Workflow

This chapter describes options that you can use to create automated actions in your application, outlines the primary configuration options, and summarizes the available tools.

10 Security

This chapter explains the Demantra security mechanisms.

11 Proport

This chapter explains the proportion mechanism.

12 General Integration Guidelines

A pre-made model supports out-of-the-box integration of the Oracle e-Business Suite or EnterpriseOne applications with the Oracle Demantra Demand Management application. This model contains necessary definitions to support both Oracle e-Business Suite and EnterpriseOne applications and uses many common building blocks. The Oracle e-Business Suite model serves as the core best practice for Oracle Demantra Demand Management integration. Where EnterpriseOne information is missing, e-Business Suite documentation applies.

13 Demantra Data Tables and Integration Processes

This chapter describes the Demantra table structure (at a high level) and gives the basic data requirements. It also describes ways you can import data into and export data from Demantra.

14 Demantra Demand Management to EBS Integration

This chapter overviews integration processes that synchronize or move data between the Oracle Demantra and E-Business Suite applications.

15 Demantra Demand Management to EBS Service Parts Planning Integration

This chapter describes the integration between Oracle Demantra Demand Management and EBS Service Parts Planning.

16 Demantra Sales and Operations Planning to EBS Integration

This chapter describes the integration between Sales and Operations Planning and EBS products, including Strategic Network Optimization (SNO), Advanced Supply Chain Planning (ASCP), and Rapid Planning (RP). It also describes the handling of SCI data.

17 Demantra Demand Management to EnterpriseOne Integration

This chapter describes how Demantra Demand Management (DM) integrates with EnterpriseOne.

18 Demantra Predictive Trade Planning to EnterpriseOne Integration

This chapter describes how Demantra Predictive Trade Planning (PTP) integrates with EnterpriseOne.

19 Demantra Deductions and Settlement Management to EnterpriseOne Integration

This chapter describes how Demantra Deductions and Settlement Management (DSM) integrates with EnterpriseOne.

20 Demantra Predictive Trade Planning to Siebel Integration

This chapter provides detailed information about how Demantra Predictive Trade Planning (PTP) integrates with Siebel Trade Promotion Management (TPM).

21 Demantra Sales and Operations Planning to Hyperion Integration

This chapter describes how Demantra Sales and Operations Planning integrates with Hyperion.

22 Integration with Demand Signal Repository

This chapter describes how Demantra integrates with Demand Signal Repository.

23 Getting Started with the Configuration Tools

This chapter introduces the primary tools you use to configure Demantra, namely, Business Modeler and Workflow Manager.

24 Database Tools

The Business Modeler provides a simple user interface for creating and modifying database tables that is useful during implementation.

25 Using the Data Model Wizard

This chapter describes how to use the Data Model Wizard.

26 Configuring Levels

This chapter describes how to configure levels with the Configure > Levels option.

27 Configuring Series and Series Groups

This chapter describes how to configure series and series groups.

28 Configuring Units, Indexes, and Update-Lock Expressions

This chapter describes how to perform miscellaneous configuration tasks.

29 Series and Level Integration

This chapter describes how to use the Integration Interface Wizard, which you use to import or export series data and level members.

30 Importing Supplementary Data

This chapter describes how to import data into the Demantra database by using Tools > Import File. You use this tool to import supplementary data such as lookup tables.

31 Creating Workflows

This chapter describes how to create Demantra workflows, which are automated or semi-automated processes that you can use for a wide variety of purposes.

32 Configuring Methods

This chapter describes how to configure methods that the users can run within worksheets or within a Members Browser content pane.bp

33 Using the Desktop BLE User Interface

This chapter describes how to configure and use the Business Logic Engine (BLE) to evaluate client expressions in worksheets.

34 Non-Engine Parameters

This chapter describes parameters unrelated to the Analytical Engine and lists their default values, if any. As indicated, most parameters are visible to all users; a few are visible only if you log in as the owner of the component.

35 Database Procedures

This chapter lists the most commonly needed database procedures.

36 Key Tables

This chapter provides reference information for some of the most important tables in Demantra, especially the data fields used by or written by the Analytical Engine. Unless otherwise noted, this information applies to all Demantra products.

37 Server Expression Functions and Operators

This appendix provides reference information for the operators and functions that are allowed in server expressions.

38 Client Expression Functions and Operators

This appendix provides reference information for the operators and functions that are allowed in client expressions.

39 Workflow Steps

This chapter provides reference information for the available workflow steps.

40 Configuring Predictive Trade Planning

This chapter describes how to configure Promotion Effectiveness, if an existing Demantra implementation is already in place.

41 Configuring Promotion Optimization for Predictive Trade Planning

This chapter describes how to configure the Promotion Optimization module. You can skip this chapter until you need to work with Promotion Optimization.

42 Configuring Deductions and Settlement Management

This chapter describes how to configure DSM and load an initial set of data.

43 Fine Tuning and Scaling Demantra

Typically you adjust parameters to control your solution's global behavior, including various defaults and performance settings. This chapter provides an overview of most of the parameters, grouped into specific areas.

44 Customizing Demantra Web Pages

This chapter describes how to customize the Demantra Web pages.

45 Configuring Rolling Data

This chapter describes how to roll selected data, saving a copy of the current version of that data.

46 GL Synchronization

This chapter describes the GL synchronization feature and how to configure this feature.

47 Performing Constraint Profit Optimization

This chapter describes how to use the Constraint Profit Optimizer.

48 Creating a Standard Error Process

This chapter describes the creation of a standard error process.

49 Introduction to the Analytical Engine

This chapter describes the role the analytical engine plays in the simulating forecasts. It also describes how the analytical engine functions generally and the different engine modes.

50 Basic Concepts

This chapter introduces the basic concepts involved with configuring the analytical engine.

51 Configuring the Analytical Engine

This chapter describes how to configure the analytical engine. It also introduces guidelines for configuring the forecast tree, causal factors, and the configure to order (CTO) feature.

52 Configuring the Forecast Tree

This chapter describes how to configure the forecast tree. In the case of PE mode, it also describes how to configure the influence relationships, and competition among the combinations.

53 Configuring Causal Factors

This chapter describes how to create causal factors, configure them, and populate them with data. It also describes the predefined causal factors provided by Demantra.

54 Configuring Promotions and Promotional Causal Factors

This chapter describes how to configure promotions and promotional causal factors in the Business Modeler.

55 Tuning the Analytical Engine

It is usually necessary to adjust some parameters to configure the Analytical Engine correctly before running it the first time. Other adjustments can be made later to optimize the behavior and performance.

56 Using the Engine Administrator and Running the Engine

Before you run the Analytical Engine for the first time, it is useful to ensure that you have configured it correctly. This chapter describes how to administer the analytical engine.

57 Engine Details

This chapter provides details on the Analytical Engine, for the benefit of advanced users.

58 Engine Parameters

This chapter describes the Analytical Engine parameters that you can see in Business Modeler and lists their default values, if any.

59 Theoretical Engine Models

This chapter contains reference information for the theoretical models that the Analytical Engine uses.

60 Administering Demantra

This chapter briefly introduces the tasks that the system administrator for Demantra would perform. It also lists all the URLs that Demantra uses.

61 Managing Security

The Demantra data and features are secured, so that not all users have access to the same data and options. This chapter describes how to maintain security.

62 Managing Workflows

This chapter describes how to use the Workflow Manager to start, stop, and view workflow instances and to manage schema groups.

63 Managing Worksheets

Worksheets are created within the user interfaces, but you can manage them from the Business Modeler.

64 Other Administration

Demantra provides a Web-based interface to perform other, less common administrative tasks, described here.

65 Upgrading Using Oracle Demantra Business Application Language

This chapter describes the Business Application Language tool (BAL) that helps you to compare old and new schemas, and make choices about how to handle differences. The BAL Explorer is launched during the installation process when you select to manually upgrade to the latest application version.

66 Tips and Troubleshooting

For reference, the first section describes the first-time login procedure for Demantra applications, followed by several sections of tips. After that, this chapter lists possible errors that users may encounter and describes how to resolve them. The errors are listed alphabetically by message text or general description.

See Also

Oracle Demantra Release Notes Oracle Demantra Installation Guide

A Key Tables

Related Information Sources

Do Not Use Database Tools to Modify Oracle E-Business Suite Data

Oracle **STRONGLY RECOMMENDS** that you never use SQL*Plus, Oracle Data Browser, database triggers, or any other tool to modify Oracle E-Business Suite data unless otherwise instructed.

Oracle provides powerful tools you can use to create, store, change, retrieve, and maintain information in an Oracle database. But if you use Oracle tools such as SQL*Plus to modify Oracle E-Business Suite data, you risk destroying the integrity of your data and you lose the ability to audit changes to your data.

Because Oracle E-Business Suite tables are interrelated, any change you make using an Oracle E-Business Suite form can update many tables at once. But when you modify Oracle E-Business Suite data using anything other than Oracle E-Business Suite, you may change a row in one table without making corresponding changes in related tables. If your tables get out of synchronization with each other, you risk retrieving erroneous information and you risk unpredictable results throughout Oracle E-Business Suite.

When you use Oracle E-Business Suite to modify your data, Oracle E-Business Suite automatically checks that your changes are valid. Oracle E-Business Suite also keeps track of who changes information. If you enter information into database tables using database tools, you may store invalid information. You also lose the ability to track who has changed your information because SQL*Plus and other database tools do not keep a record of changes.

Part 1

Concepts and Tools

Introduction to Demantra

This chapter provides an overview of the Demantra platform.

This chapter covers the following topics:

- Demantra Platform and Applications
- Extending your Demantra Application
- Elements of a Demantra Solution
- Integration
- Workflow
- How the User Interfaces Can Be Configured

Demantra Platform and Applications

The Demantra Platform provides a flexible data model on which we build the comprehensive business logic of our applications.

The platform includes a number of services that are leveraged to provide out-of-the-box application functionality in four key areas:

- Demand Management
- Real Time Sales & Operations Planning
- Trade Promotion Management
- Deductions and Settlement Management

These standard out-of-the-box applications can be tailored to reflect a customer's specific business logic in these business areas. Please see "Extending your Demantra Application."

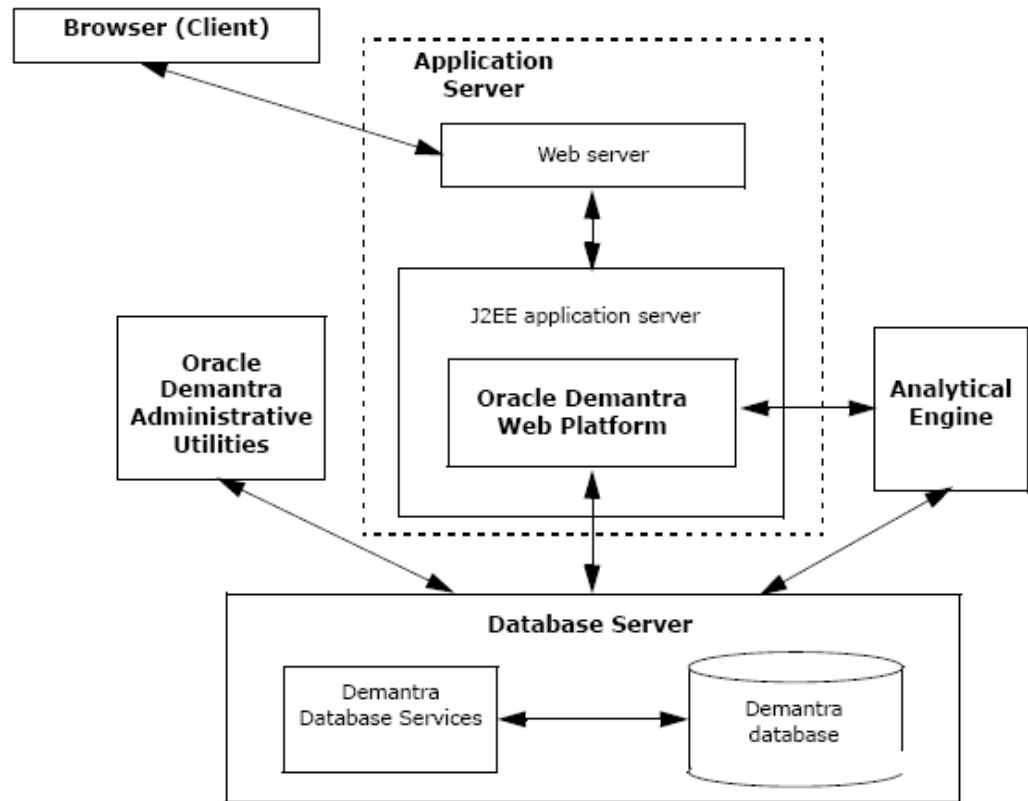
Extending your Demantra Application

There are many ways to extend your pre-configured Demantra Application. For example, most implementations will create additional worksheets to capture and review data in alternate formats. We provide the following guidelines around extensions to the data model:

1. Please review all documentation fully. It includes important details on the standard application configurations and how they support integration to other products in the Oracle family (for example, E-business Suite (EBS) or JD Edwards EnterpriseOne(E1)).
2. Wherever possible, please work within the standard application hierarchy. It has been pre-seeded with a comprehensive business hierarchy. In addition, there are a number of generic 'Category Code' levels in the item, location and Matrix hierarchies that can be leveraged to capture additional business levels in the hierarchy. This approach avoids the need to upgrade the model. If additional levels are required, please only add parent levels to the hierarchy. Do not add lowest levels as these fundamentally alter the data model. Sufficient lowest levels have been pre-seeded to capture many-to-many relationships.
3. Please review series details in the implementation guide before creating additional series. Comprehensive business logic has been configured for each of our applications and should be fully understood to avoid building extraneous series. Please try to avoid series modifications as this may affect other series referencing it.
4. Please review workflow and integration details in the implementation guide and within the workflow engine itself before creating additional logic. Comprehensive integration flows have been configured for each of our applications and should be fully understood to avoid building extraneous workflows. Note that workflows are generally grouped by the integration they are supporting (for example, "EBS Integration").
5. Adding additional Levels or Series to the data model will require enhancements to pre-configured integration logic (with E-Business Suite or EnterpriseOne, for example) to populate these fields.

Elements of a Demantra Solution

Whether you use a Demantra application as-is or you use the Application Platform, a Demantra solution consists of the following elements:



- This figure is not meant to show specific hardware architecture.
- A Demantra solution can also include a Citrix server or other software for terminal emulation, not shown here.
- For information on supported databases, platforms, and configurations, see the *Oracle Demantra Installation Guide*.

Clients

A Demantra solution includes multiple client machines, each running a browser to access Demantra.

Supported Web Browsers

Demantra is compatible with Internet Explorer or Mozilla Firefox. Please refer to the Demantra Installation Guide for supported versions.

Configuring the Firefox Browser

When using the Firefox Web browser, if a Demantra user closes the browser using the "

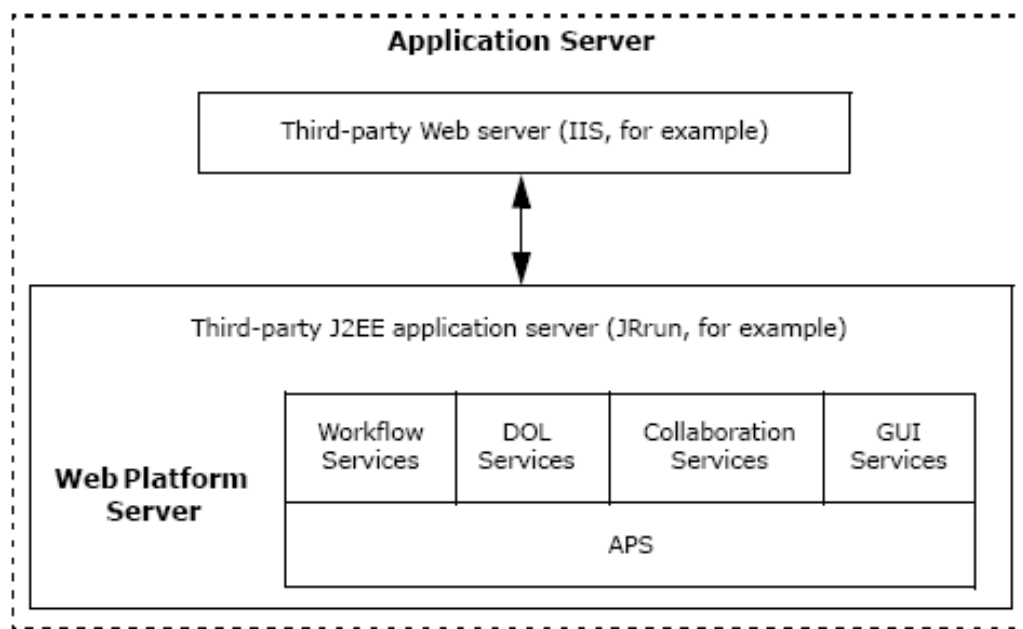
X" icon, it is possible to restore the session by re-launching a new Firefox browser. In this scenario, the login page is not displayed and the user is not required to enter a username and password.

Modify Firefox configuration settings as follows:

1. Select Tools > Options > Startup > Main tab
2. Verify that the "When Firefox starts" setting is **not** set to "Show my windows and tabs from last time." Change this setting, if required.
3. Navigate to the 'Security' tab, and then verify that "Remember password for sites" is **not** selected.
4. Save the changes.

Application Server

Any Web-based solution includes the Application Server:



This server includes a Web server and a J2EE application server, which can be on the same or different machines; for supported software, see the Oracle Demantra Installation Guide.

Within the J2EE server, Demantra runs the **Oracle Demantra Web Platform Server**, which includes the following:

- Workflow Services

- DOL Services
- Collaboration Services
- GUI Services
- APS (this is the main support layer and handles all communications with the database, as needed by the other services)

Database Server

Every Demantra solution includes a database server.

Oracle Demantra Administrative Utilities

The Oracle Demantra Administrative Utilities include the desktop configuration and maintenance utilities, as well as Member Management, Chaining Management, and Allocation Management tools. Administrative Utilities coordinate running the Analytical Engine, communicate with the database as needed, schedule database procedures, and run other background processes as needed.

Analytical Engine

Most Demantra solutions include the Analytical Engine.

Note: In general, the documentation refers to either mode as the "Analytical Engine". Wherever the distinction is necessary, the documentation is more specific.

Note: Oracle provides two different modes for the Analytical Engine:

- In PE mode, the engine is suitable for use with Promotion Effectiveness.
- In DP mode, the engine is suitable for use in demand planning applications.

You may have access to the Distributed Engine (a mode in which the Analytical Engine automatically distributes its work across multiple machines).

SSL Security

The Demantra Web products can use either http or SSL protocol. You can deploy a Demantra solution in either of two ways:

- Use http for all pages
- Use SSL for all pages (so that Web addresses start with https instead of http)

Pure Desktop Solutions

A desktop-based solution is different from a Web-based solution in two key ways:

- Users access Demantra via the desktop Demand Planner, which communicates directly with the database.
- There is no Application Server. Instead, you use the Stand-Alone Integration Tool (the aps.bat executable), which handles import and export. The Oracle Demantra Installation Guide assumes that most solutions are Web-based.

Integration

You can import and export data either with core Demantra tools or with Demantra Enterprise Integrator (powered by Pervasive).

Core Demantra Tools

The core Demantra tools allow you to do the following:

- Import lowest-level item, location, and sales data
- Import or export series data at any aggregation level, with optional filtering
- Import promotions and promotional series
- Export members of any aggregation level
- Import supplementary data into supporting tables as needed

Workflow

The Application Platform provides the Workflow Manager. A workflow is a logically connected set of steps. Each step can be automated or can require interaction from one or more users or groups.

Workflows can do all the following kinds of actions:

- Run integration interfaces.
- Run stored database procedures.

- Run external batch scripts and Java classes.
- Pause the workflow until a specific condition is met, possibly from a set of allowed conditions. For example, a workflow can wait for new data in a file or in a table.
- Send tasks to users or groups; these tasks appear in the My Tasks module for those users, within Collaborator Workbench. A typical task is a request to examine a worksheet, make a decision, and possibly edit data. A task can also include a link to a Web page for more information.

Special workflow steps programming logic. For example, one step type provides a user with a selection of choices to direct the continuation of the workflow instance.

How the User Interfaces Can Be Configured

Whether you start from a Demantra application as-is or from the Application Platform, you can configure the user interfaces in the following complementary ways:

- You typically create worksheets to meet the needs of specific users. A worksheet is a working environment that shows specific data, aggregated and filtered as needed. Users can view, sort, edit, print, and so on. The next chapter, "Core Concepts", describes the elements of worksheets.
- You can create methods that the users can execute from within worksheets. The methods appear in the worksheets as options on the right-click menu. Demantra also provides default methods that you can redefine or disable. These allow users to create, edit, and delete level members.
- You create components that subdivide the data as needed for different organizational roles. Each component has an owner, who acts as the administrator of the component. In turn, the owner can log onto the Business Modeler and further restrict data access for particular users.
- You apply security so that different users have access to different menu options. See "Managing Security".
- You can configure the default layout of Collaborator Workbench, access to different elements of Collaborator Workbench, and the links and menus in Collaborator Workbench. You can also substitute custom graphics throughout the Web products. See "Customizing Demantra Web Pages".

Core Concepts

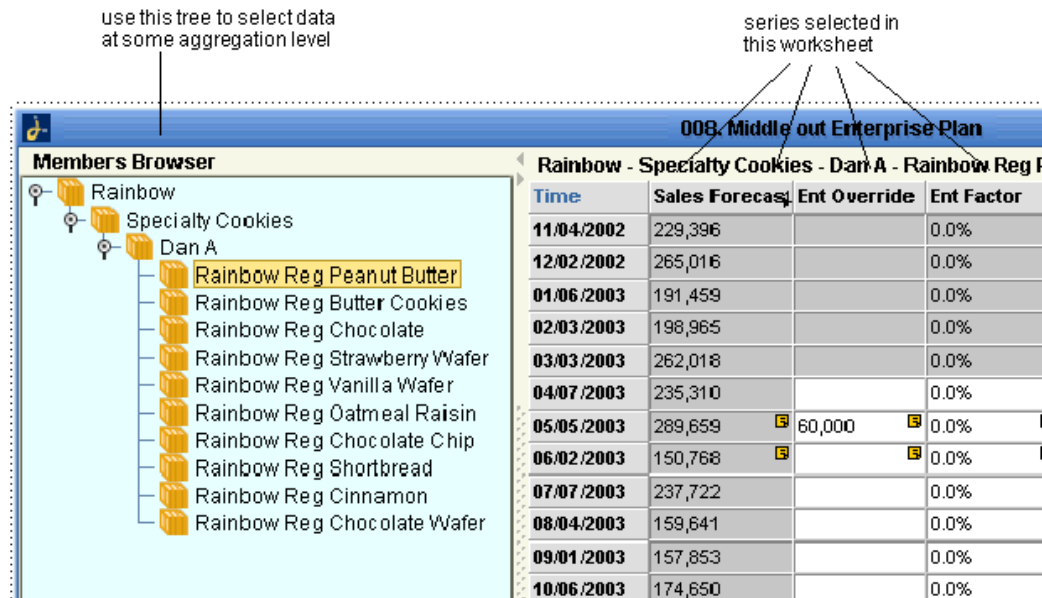
This chapter explains worksheets and other basic concepts.

This chapter covers the following topics:

- Worksheets
- The Basic Input Data
- Time Resolution
- Levels
- Combinations
- Series
- Filtering and Exceptions
- Methods
- Security
- Forecasting

Worksheets

A worksheet (sometimes known as a query) is the primary user interface to Demantra data. A typical worksheet might look like this:



Within a worksheet, a user can examine and edit data as needed, view the forecast, run simulations, and save changes back to the database, for the benefit of other users and downstream operations. The precise details vary from application to application, but worksheets share the following characteristics:

- Most of the worksheet data is usually based on imported data.
- The data is organized in a set of multi dimensional hierarchies that enable users to slice and dice data in any way. These hierarchies are completely configurable and are easily extended.
- A worksheet displays series of data, usually time-dependent data for specific items and locations. Some series are editable, and other are not.
- A worksheet can display series at an aggregated level, based on any of the hierarchy levels in the system.
- A user can zoom in and out in time, viewing data aggregated into different buckets of time.
- At any given time, the worksheet uses a single unit of measure, which applies to some or all of the displayed series (some series do not include units). The worksheet can also use a financial index or exchange rate. The user who is working with the worksheet can switch to another unit of measure or another index as needed.
- A worksheet can be filtered. In addition, users generally have access to only some of the data, and that filters the data further.
- Multiple users can access the data, depending on their authorization.

For details, see "Worksheets".

The Basic Input Data

When fully configured, Demantra imports the following data, at a minimum, from your enterprise systems:

- Item data, which describes each product that you sell.
- Location data, which describes each location to which you sell or ship your products.
- Sales history, which describes each sale made at each location. Specifically this includes the items sold and the quantity of those items, in each sale.
- For Promotion Effectiveness: Historical information about promotional activities.

Demantra can import and use other data such as returned amounts, inventory data, orders, and settlement data.

For details, see "Data Assumptions and Requirements".

Time Resolution

Sales data is typically available at the daily (or sometimes hourly) level, but demand plans do not usually go down to that level of detail. When sales data is imported into Demantra, it is automatically binned into time buckets corresponding to the base time unit, depending on how you configure the system.

Specifically, when the sales data is imported, each sale date is changed automatically to the start date of the appropriate time bucket. For example, suppose you use a weekly base time unit, starting on Monday. If a sale actually happened on Wednesday, May 7, the sale date in Demantra is changed to Monday, May 5.

The base time unit is specified during configuration to a length that is appropriate for your planning cycle. Oracle provides three sizes of base time unit (day, week, or month) and can support hourly time units if needed.

Oracle also provides larger time units for use in worksheets, and you can define additional time units. You specify the time unit to use in a given worksheet, and the worksheet shows data for the time buckets that correspond to that time unit.

For details, see Time Units, page 7-3.

Levels

The first interesting feature of any worksheet view is the aggregation level or levels that it uses. For example, you might want to view data at the account level, as follows:

Accounts		Accounts and SKUs				
Members Browser		CVS				
<ul style="list-style-type: none"> BJ CVS McKessen Rainbow Company Ralphs Stop and Shop Walmart 		Time	Demand	Price \$	Revenue \$	Market Plan \$
		02/04/2002	5,757,460	\$10.00	\$57,574,600	\$19,771,290
		05/06/2002	7,674,924	\$10.00	\$76,749,240	\$19,771,290
		08/05/2002	7,285,994	\$9.99	\$72,606,928	\$19,609,128
		11/04/2002	5,736,909	\$9.97	\$57,049,760	\$19,609,128
		02/03/2003	1,571,396	\$9.99	\$15,649,532	\$36,329,952
		05/05/2003		\$10.00	\$67,828,136	\$58,347,280
		08/04/2003		\$10.00	\$63,045,580	\$57,779,168
		11/03/2003		\$10.00	\$47,297,200	\$46,307,320
		02/02/2004		\$10.00	\$49,651,088	\$48,727,416

The worksheet might include a drop down list instead of this tree control.

For example:

Accounts Accounts and SKUs

Page Items

Account CVS

All

Time	Demand	Price \$	Revenue \$	Market Plan \$
02/04/2002	5,757,460	\$10.00	\$57,574,600	\$19,771,290
05/06/2002	7,674,924	\$10.00	\$76,749,240	\$19,771,290
08/05/2002	7,285,994	\$9.99	\$72,606,928	\$19,609,128
11/04/2002	5,736,909	\$9.97	\$57,049,760	\$19,609,128
02/03/2003	1,571,396	\$9.99	\$15,649,532	\$36,329,952
05/05/2003		\$10.00	\$67,828,136	\$58,347,280
08/04/2003		\$10.00	\$63,045,580	\$57,779,168
11/03/2003		\$10.00	\$47,297,200	\$46,307,320
02/02/2004		\$10.00	\$49,651,088	\$48,727,416

In either case, you can view data for any account. For example, for the quarter that started on February 3, 2003, the Demand for the CVS account was 1, 571, 396 units, and the unit price was \$9.99. You can edit any data that is shown in white, such as the price and market plan.

In generic terminology, the word *member* refers to a unit within a level. For example, CVS is a member of the account level.

Levels are also used in import and export, security, and forecasting.

For details, see "Levels".

Combinations

When users explore their sales data, they generally examine data associated with some item (or aggregation of items) at some location (or aggregation of locations). Each possible pairing of item and location is known as a combination.

Note: In theory, some implementations may have more than two chief dimensions. For example, you might track sales for items, locations, and demand types. In this case, a combination is an item, a location, and a demand type.

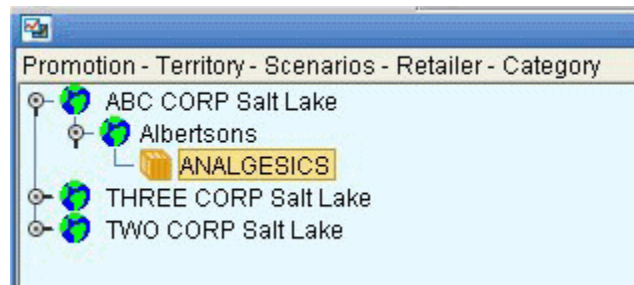
Combinations are central to Demantra. At any given time, a worksheet displays data for one combination at any aggregation level, for example:

- Low fat items in Northeast stores
- SKU PLLF202FCPB at CVS 0051
- Private Label Brand cookies at the account Retailer D
- Ice cream, aggregated at all locations

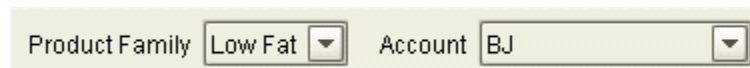
Selecting Combinations

Apart from completely aggregated worksheets, each worksheet provides a way to select the combination to view. Demantra provides two equivalent mechanisms, as in the following examples.

Members Browser (available only in Web worksheets)



Drop down lists



In either case, the selected combination is "Low fat products at the BJ account." The rest of the worksheet shows data for that combination.

In some cases, you view data that is aggregated across one dimension. For example, if the worksheet contains only the product family level, and you select the Low Fat member, that means that you have selected the combination "Low fat products at all locations."

Combination Status

Not all items are sold at all locations. By default, Demantra stores only those combinations that have actual sales, and the Analytical Engine considers only those combinations. You can enable users to create new combinations, for simulation purposes; to do so, they use tools called Member Management and Chaining Management.

The Analytical Engine also considers the relative age of each combination, as well as other combination-specific details. For the details, see "Combination-Specific Settings".

Series

A series is a set of data that represents some value that varies over time or that varies between item-location combinations (or most commonly, that varies in both ways). A worksheet displays the series data in a table, or in a graph, or both. The following shows an example of a worksheet table:

Private Label LF Butter - BJ Store # 0006							
Time	Demand	Final Plan	Pseudo	Simulation	Sales Forecast	Sales Fcst Bias	Stat Frcst (Y/N)
04/08/2002	1,258,700				1,240,202	-18,498	Do Forecast
07/08/2002	1,232,800				1,161,719	-71,081	Do Forecast
10/07/2002	1,326,200				1,057,580	-268,620	Do Forecast
01/06/2003	488,500				903,675	415,175	Do Forecast
04/07/2003		1,193,227			1,193,227		Do Forecast
07/07/2003		1,123,295			1,123,295		Do Forecast
10/06/2003		1,040,942			1,040,942		Do Forecast
01/05/2004		820,737			820,737		Do Forecast
04/05/2004		280,121			280,121		Do Forecast
Summary	4,306,200	4,458,322			8,821,497	14,244	

The example here shows series at the lowest level, but you can generally view data for any given series at any aggregation level. The definition of the series controls how the data is aggregated. Data can be aggregated in various ways, for example by totalling it, or by taking the maximum or the minimum value.

Series have many properties, including the following:

- It is editable or non editable. Some series are editable only within certain spans of time or when certain conditions are met.
- The data type: numeric, string, or date.

- The series type. Most series apply to sales data, which means that the series data can be different for each item-location combination at each time bucket. Demantra provides other types of series as well; see "Types of Series".
- The series may or may not be stored in the database.
- Its definition specifies how data for that series is to be calculated at any level or at any higher level time aggregation.

The Analytical Engine directly populates the data used in some of the base series:

- For Demand Management, Demand Planner, and Demand Replenisher, these series include information about the forecast and related information such as markers that indicate regime changes, outliers, and so on.
- For Promotion Effectiveness, these series include information about the forecast and switching effects that describe the impact of various promotions.

For details, see "Series".

Filtering and Exceptions

Both filters and exceptions both limit the members that users can see. Filters act directly, and exceptions act indirectly.

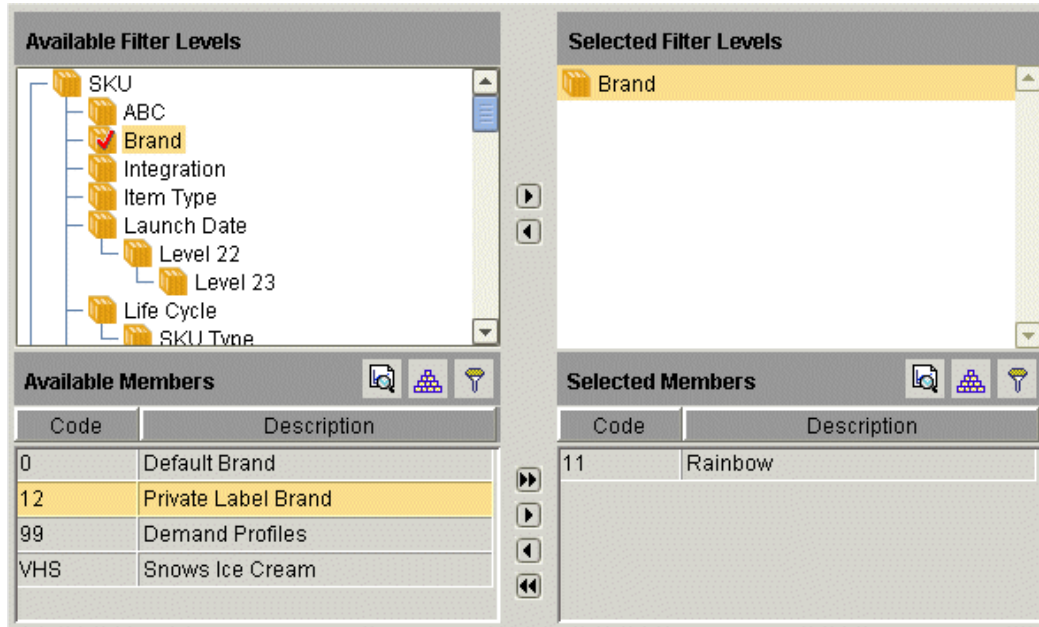
Filters

Filters specify the members that users can see. When you apply a filter, you specify the following:

- An aggregation level. You can filter data at any level in any dimension.
- Specific members of that aggregation level that are allowed through the filter; other members are not included.

The net result is that a filter allows Demantra to display only certain item-location combinations.

Demantra uses filters in various contexts. In all cases, it uses a standard user interface to display a filter. In the following example, the filter blocks data for all brands except for the Rainbow brand.

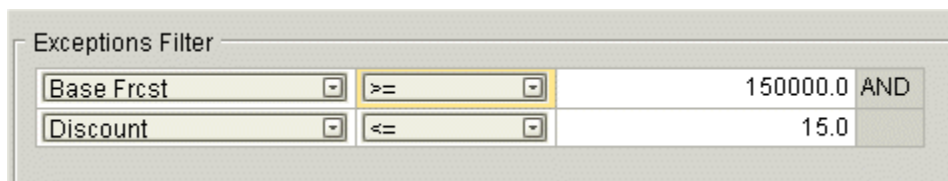


As a result, the worksheet will display only those item-location combinations that are associated with the Rainbow brand. You can filter data at any level, whether or not it is chosen as an aggregation level of the worksheet.

Exceptions

Exceptions (or exception filters) indirectly control which members the users can see. When you apply an exception, you specify a true/false expression that specifies a series, an operator, and a value, for example: Sales > 50000. A combination is displayed only if this expression is true for at least some of the time buckets in the time range of interest.

You can specify multiple expressions and relate them by logical AND or logical OR.



Methods

You can define level methods, which the user sees as ordinary right-click menu options in Demantra (either in worksheets or in Members Browser content panes). Each level can have its own set of methods. Demantra provides a set of default methods (Create, Edit, and Delete) that you can redefine or disable as needed.

Each method runs a workflow. In Demantra, a workflow is a logically connected set of

steps. Each step can be automated or can require interaction from one or more users or groups. Demantra provides a set of workflow steps, each with predefined behavior.

Workflows can also be used for general automation purposes.

For details, see "Methods and Workflow".

Security

The Demantra data and menus are secured, so that not all users have access to the same data and options. The security model includes the following features:

- The Oracle *license* controls which menus are secured, so that not all users have access to the same data and options. The security model includes the following features:
- The data is partitioned into components, which generally correspond to organizational roles. In the definition of a component, you can control the following:
 - The levels that can be seen
 - The degree of access for members of each level: no access, read-only access, read/write access, or full control (including the ability to delete members)
 - The series that can be seen

Each component has an owner, who acts as the administrator and who can create additional users:

- Within a component, you can restrict each user to a subset of the data associated with that component. You can control the same data elements as previously described.
- You can control access to menu items at the component level, the group level, or the user level. This includes both the menu bar and the right-click menu.
- You can define program groups, or sets of menu items, and apply security at that level, for greater convenience.

For details, see "Security".

Forecasting

The Analytical Engine reads data from the database, generates a forecast and performs other analyses, and writes the forecast to the database. This section provides a brief overview of the concepts.

Demand and Causal Factors

The forecast considers both the historical demand and the causal factors (such as seasons, price changes, and specific events such as promotions).

In a Demantra solution, the demand data is ultimately imported from external systems. Typically, the data that is actually imported needs to be adjusted by the forecasters, as they apply their own knowledge to better describe the history.

Causal factors provide information about historical events that are expected to recur in the future. Causal factors cause demand to deviate from a trend. More specifically, a causal factor is a time-varying quantity (such as promotions, price, season, or day of the week) that affects demand. Demantra requires historical data for causal factors, as well as future data that describes expected occurrences that will affect demand.

In the case of Promotion Effectiveness, you also configure promotional causal factors, influence ranges, and influence groups, all of which control how the Analytical Engine determines the effects of promotions on the forecast.

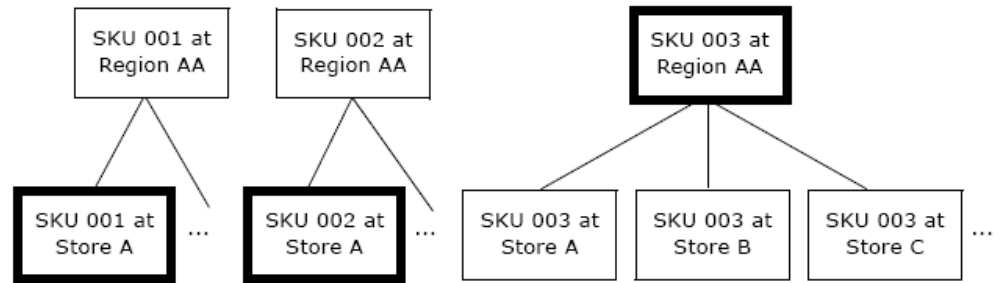
Engine Coefficients

As a result of the forecasting process, the Analytical Engine calculates a set of *coefficients* that describe how each causal factor affects demand for each item-location combination, over time. The Analytical Engine then uses those coefficients, along with future values for the causal factors, to calculate the forecast. The Promotion Optimization module also makes use of these coefficients.

Forecasting Models and the Forecast Tree

The Analytical Engine uses a set of mathematical forecast models. When forecasting, the engine follows a specific process of examining the data, checking for outliers and so on, evaluating the usefulness of each model, and generating the forecast.

In general, forecasting is most accurate when it can be performed at the lowest possible allowed aggregation level. However, sometimes there is not enough data at that level for all combinations. For those combinations, the Analytical Engine aggregates the data to a higher level and tries to generate a forecast there. The purpose of the forecast tree is to organize data for this process. Each node in the forecast tree aggregates both by items and by locations. The following example shows a small part of a forecast tree.



The bold boxes show the nodes at which the Analytical Engine is forecasting in this example.

Parameters and Engine Profiles (PE)

Demantra provides parameters to control both the theoretical models and the overall engine flow.

The engine uses engine profiles, which are sets of engine parameters with specific values. Demantra provides some predefined profiles for different purposes, and you can define additional engine profiles, as needed. When you run the engine, you specify the engine profile to use.

Batch and Simulation Modes

The Analytical Engine runs in two modes:

- When a forecast is made in the batch mode (in the background), the Analytical Engine creates a forecast for all item-location combinations within the forecast tree. You configure Demantra to run the Analytical Engine periodically, usually after importing new data.
- If the Analytical Engine is running in simulation mode, it waits for simulation requests and then processes them.

From within a worksheet, a user submits a simulation request to create a tentative forecast for a subset of the data. The user can then accept or reject the results of the simulation. In this mode, the user is usually performing a "what if" analysis. That is, he or she has made some changes within the worksheet and then performs the simulation to see whether those changes have the desired effect.

See Also

"Introduction to the Analytical Engine".

How Demantra Calculates and Stores Data

As an implementor, you should understand how Demantra calculates and stores data, because this can affect how you set up your solution. This chapter provides an overview of most important details.

This chapter covers the following topics:

- How Data Is Stored
- How Data Is Calculated at the Lowest Level
- How Data Is Calculated at Higher Levels
- How Proportions Are Chosen
- How Proportions Are Used
- How Splitting Handles Null Values
- How Splitting Handles Zero Values
- When and How Data Is Saved to the Database

How Data Is Stored

In order to understand how Demantra works, you should be aware of two central facts about how Demantra stores data:

- Demantra stores data only at the lowest possible level.
- Demantra stores data only where the sales data is non-null.

These facts have wide-ranging implications. The following subsections expand these two facts, and the sections after that provide further details.

Data Is Stored at the Lowest Level

Demantra stores data only at the lowest possible item level, lowest possible location level, and base time unit. This has the following implications:

- When data is viewed or exported at a higher level, that data must be calculated.
- When data is edited or imported at any higher level, Demantra must calculate and then write the data for the appropriate lowest-level members.
- If the Analytical Engine generates a forecast at a higher level, Demantra must split that forecast and write it into the appropriate lowest-level members.

Data Is Stored Only Where the Sales Data Is Non-Null

Theoretically there might be a very large number of item-location combinations, but some items just might not be sold at some locations. It would be incorrect for Demantra to assume that all items could be sold at all locations during any time buckets. If Demantra assumed that, the result would be a very inaccurate forecast.

Instead, Demantra assumes (by default) that if there were no sales for a given combination during a specific time bucket, sales were not possible for that combination and time. Specifically, Demantra uses this assumption as follows:

- The Demantra database does not, by default, contain records that represent null sales. That is, for a given item-location combination, if no sales occurred during a given time bucket, Demantra does not contain a record for that combination and time bucket.
- If there is no sales record for a given combination and time bucket, Demantra ignores that combination and time bucket when performing various operations, such as forecasting.

In some cases, however, a null value for sales really does mean zero sales. Depending on the context, you can force Demantra to treat null values as zero values.

How Data Is Calculated at the Lowest Level

In general, the definition of a series also specifies how to calculate data at the lowest level, in the case when data changes at a higher level. Data can potentially change at a higher level either when it is imported at a higher level or when users edit a series while displaying data at a higher level.

Each series can be configured as proportional or non-proportional.

- If a series is proportional, the parent value is split among the child members according to the proportions of those child members.
- If a series is non-proportional, the value for each child member is set equal to value of parent.

Other series are neither proportional nor non-proportional. Data for these series should not be changed except at the lowest level. For details, see "Summary of Calculation

Options".

How Data Is Calculated at Higher Levels

The definition of a series specifies how to calculate data at any level. A series can have either or both of the following expressions:

- A server expression, which aggregates data from the lowest level in the database.
- A client expression, which calculates the series data, based on other series at the same level. If a series has a client expression, that series is automatically made non editable.

Server Expressions: Aggregation from Lower Levels

A server expression is an SQL expression that calculates the series data at any level by aggregating the associated lowest-level data. A very common server expression has the following form:

`sum (table_name.update_column_name)`

Here *table_name.update_column_name* is the table and column that stores data for this series.

If you view a combination at the lowest level, this expression reads the series value for one row in the given table. On the other hand, if you view a combination at a higher level, this expression sums the series values associated with all the rows that correspond to the child members.

- Parent: muffins
 - Expected: 200
 - Actual: 220

Note: The values for Expected and Actual series are calculated by aggregating from child members

- Child: apple muffins
 - Expected: 100
 - Actual: 110
- Child: blueberry muffins
 - Expected: 100

- Actual: 110

Similarly, if you view a combination at a larger time bucket, this expression sums the series values associated with all the rows that correspond to the smallest time buckets for the combination.

A server expression can also aggregate by averaging or taking the minimum or maximum value of the child members.

Client Expressions: Calculations within a Level

A series can also have a client expression, which calculates data in a different way. In contrast to server expressions, a client expression always refers to data at the same level. You use client expressions to calculate numbers that cannot be calculated by aggregation from lowest-level data. For example, consider the following client expression for a hypothetical series called ErrorSquared:

$(\text{Expected} - \text{Actual}) * (\text{Expected} - \text{Actual}) / (\text{Expected} * \text{Expected})$

For a given combination at a given time bucket, this expression calculates the ErrorSquared series directly in terms of the Expected and Actual series for the same combination at the same time bucket. As you can see from the following example, it would not be possible to compute this series by aggregating lowest-level members:

- Parent: muffins
 - Expected: 200
 - Actual: 220
 - ErrorSquared: 0.0001

Note: The value for the ErrorSquared series is calculated at this level, not aggregated from child members

- Child: apple muffins
 - Expected: 100
 - Actual: 110
 - ErrorSquared: 0.01
- Child: blueberry muffins
 - Expected: 100
 - Actual: 110

- ErrorSquared: 0.01

In addition to formulas as these, you can use client expressions to perform the following kinds of computations, which are not possible with server expressions:

- Conditional expressions, including nested conditional expressions
- Expressions that refer to data at earlier or later time buckets

Using Both Server and Client Expressions

A series can have both a server expression and a client expression. The client expression always takes precedence. Therefore, if a series has both expressions, the client expression is usually of the following form:

If (*condition*, *client-expression-value*, *series-name*)

Here *series-name* is a reference to the same series to which this client expression applies. This reference directs Demantra to the server expression that this series uses. Depending on whether the condition is true for a given cell, this expression returns either the client expression value or server expression value.

How Proportions Are Chosen

Demantra provides three general ways to specify the relative proportions of different combinations:

- Actual proportions. This option splits higher-level data according to the proportions of the Demand series. This is an option when importing data.
- Proportions of a reference series. When you configure a series as proportional, you also specify a reference series (the Proportion Calculation series). You typically use one of the following series:
 - Demand (suitable for a historical series)
 - Final Plan (suitable for a forecast series)

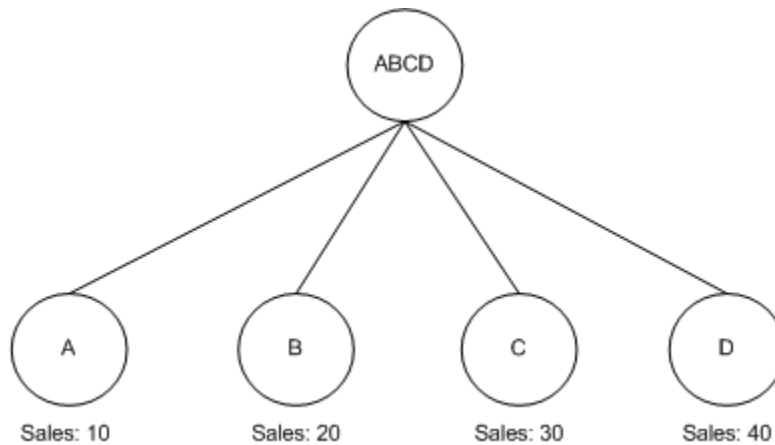
This option is available when you define a series and is used when data for that series is edited at an aggregated level.

- Matrix proportions, which are stored proportions that Demantra calculates and saves. (The mechanism that performs the calculation is called *proport*.) The calculation is based upon the demand, but also considers recent average demand, month-to-month variations, and so on. See Chapter 12, "Proport". These proportions are available in multiple cases:
 - When importing data.

- Automatically used when forecast must be created at higher level.
- When a user performs a chaining operation (data copy) at a higher level. A user performs chaining in order to use existing data for a given combination as a basis for a new combination.

How Proportions Are Used

The following figure shows an upper-level member, ABCD, and its four child members. It also shows a reference series (Sales), and it shows the value of that series for each child member, all within the same time bucket.

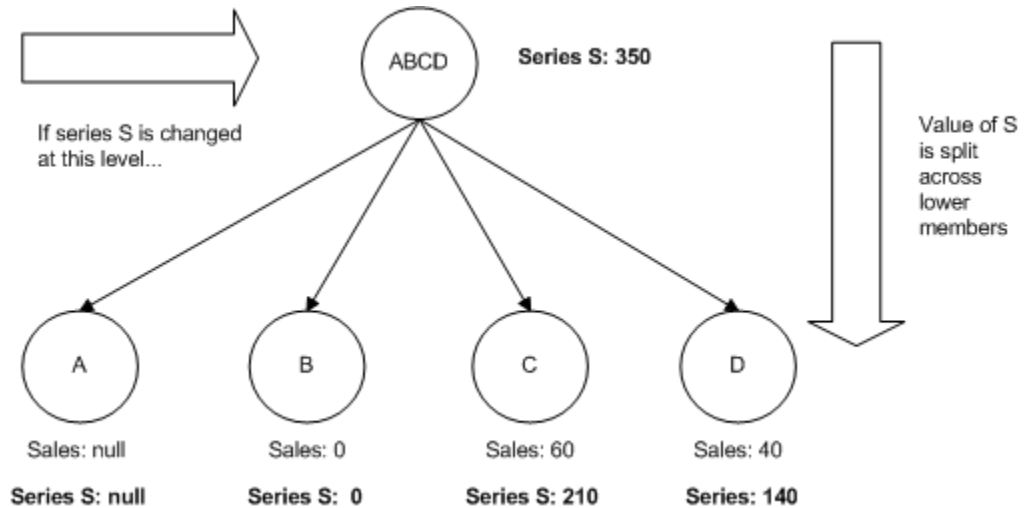


Now suppose that series S is a proportional series that uses Sales as its reference series, and suppose that the value of S is changed to 350 for the parent member. In this case, the series S is split across the child members as follows:



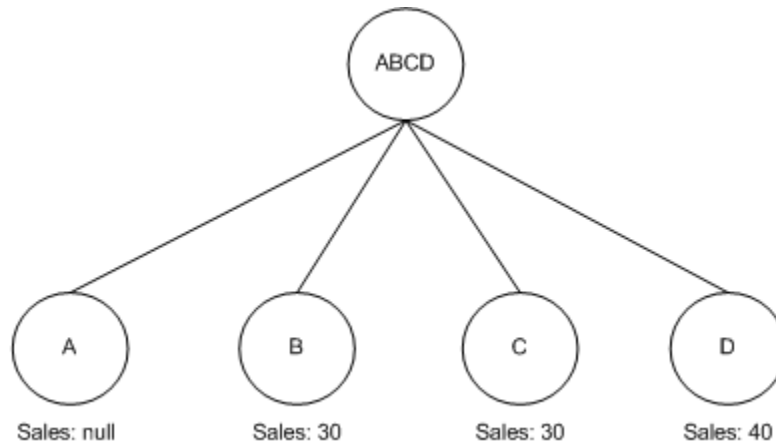
How Splitting Handles Null Values

Now consider a case where the reference series has a null value for one of the child member. The proper mechanism ignores that member, as follows:



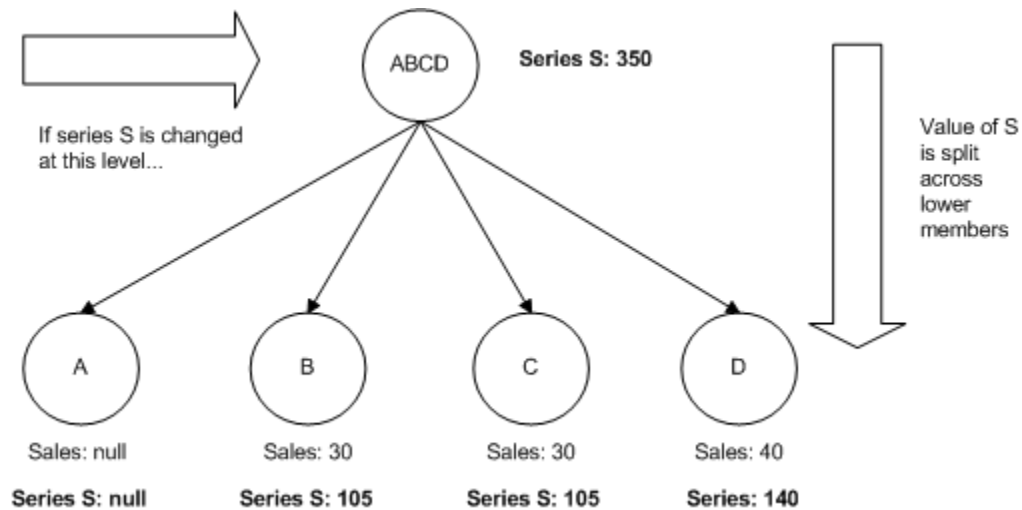
How Splitting Handles Zero Values

Now let us consider two cases where child members have zero values. In the first case, the reference series is zero for one of the child members, but has non zero numbers for other child members. Any member with 0 sales receives 0% of the split, as follows:



Notice that member A that has a null value for the reference series; for this member, the value of series S is null, rather than 0.

In the second case, none of the child members has a non zero value. In a case like this, the parent value is split equally among all members that have zero values for the reference series.



As always, if a child member has null for the reference series, the proprot mechanism ignores that member entirely.

When and How Data Is Saved to the Database

A series may or may not be stored in the database. If it is stored, its data is saved in the series update field. Because you can potentially have different values from splitting at different levels, it is important to understand when and how Demantra saves data to the update field for a series.

For a series that has an update field, Demantra saves data to that update field as

follows:

- When you import data for that series, Demantra splits it down to the lowest level and saves it in the update field.
- When a user edits data for that series, Demantra splits it down to the lowest level and saves it in the update field. If the user is working with the desktop products (Demand Planner or Demand Replenisher), the MANUALS_INS procedure performs the splitting and saving. If the user is working with the Web products, no database procedure is needed.

If this user changes causes a change to the value of another series in the worksheet, Demantra splits that other series value down to the lowest level and saves it in the update field for that other series.

Demantra ignores any series whose values have not been changed.

If the series also has a client expression, Demantra also saves data to that update field as follows:

- When a user runs a worksheet that contains the series, Demantra evaluates the client expression for that series, computing the values at that level. Demantra then splits it down to the lowest level and saves it in the update field. If the user is working with the desktop products (Demand Planner or Demand Replenisher), the MANUALS_INS procedure performs the splitting and saving. If the user is working with the Web products, no database procedure is needed.
- When the Business Logic Engine evaluates a worksheet, Demantra evaluates the client expression for all series in that worksheet, computing the values at that level. Demantra then splits the values down to the lowest level and saves them in the update fields of the appropriate series. In this case, the MANUALS_POPULATE_INS procedure performs the splitting and saving.

Caution: Demantra does not automatically launch the database procedures. Instead, you must make sure to schedule the procedures to run periodically or otherwise automate running them. Use the Workflow Engine. See "Workflows".

Implementation Tools and Process

This chapter provides a quick overview of the implementation and hand-off processes and tools you use during these processes.

This chapter covers the following topics:

- Overview of Tools
- Initial Phase of Implementation
- Middle Phase of Implementation
- End Phase of Implementation
- Hand-off Details

Overview of Tools

This section provides a brief overview of the tools you use during implementation. All these tools are available later for use by system administrators.

Setup Scripts

Some Demantra products, such as DSM, assume that your database contains specific levels, parameter settings, and other configuration options. Demantra provides setup scripts that perform all the required configuration for these products. See "Other Configuration".

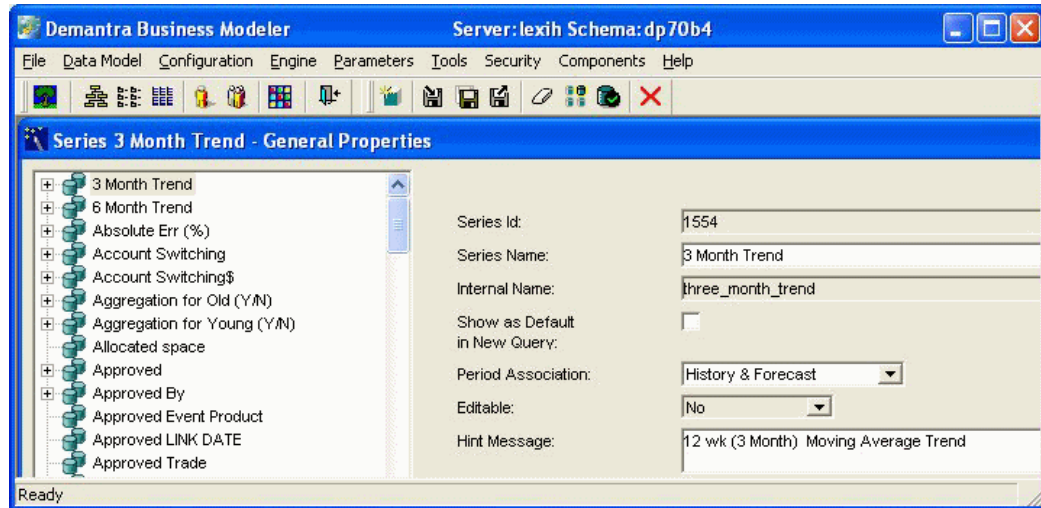
Business Modeler

You use the Business Modeler for most configuration tasks, including the following tasks:

- Defining all elements used in worksheets: levels, series, units of measure, indexes and exchange rates

- Defining level methods
- Defining integration
- Defining components and users
- Setting parameters

The Business Modeler is desktop-based and looks like this; this example shows the wizard for configuring series:



Workflow Manager

You use the Web-based Workflow Manager enables you to create and manage workflows or automated processes. The Workflow Manager looks like the following:

name of currently displayed schema group

use these buttons to manage schema groups

View according to Schema Groups:

Schema ID	Schema name	Owner	Creation Date	Last Modified	Instances	Status	Action			
1	Partner Plan Collaboration	dp	Feb 28 20:10:02 2002	Aug 21 16:55:48 2003	0		Edit	Start	Schedule	Delete
2	Space consumption Alert	guy_yehiaiv	Apr 14 21:21:46 2003	May 07 18:06:46 2003	0		Edit	Start	Schedule	Delete
22	Stockout Alert	guy_yehiaiv	Apr 14 22:40:47 2003	Sep 22 16:31:00 2003	0		Edit	Start	Schedule	Delete
23	CPFR Step 1	guy_yehiaiv	Apr 29 17:46:04 2003	Apr 29 17:46:34 2003	0		Edit	Start	Schedule	Delete
43	Stockout Alert Per Store	guy_yehiaiv	May 08 13:05:53 2003	May 08 13:05:53 2003	0		Edit	Start	Schedule	Delete
63	Export Dynamic Data	inv	May 16 09:58:59 2003	May 16 10:00:14 2003	0		Edit	Start	Schedule	Delete

Process Log [New Schema](#) [Refresh](#)

view workflow process details

create a new schema

refresh this screen

schemas in the currently displayed group

For information on creating and managing workflows, "Creating Workflows" and "Managing Workflows".

Analytical Engine

The Analytical Engine reads data from the database, generates a forecast and performs other analyses, and writes the forecast to the database.

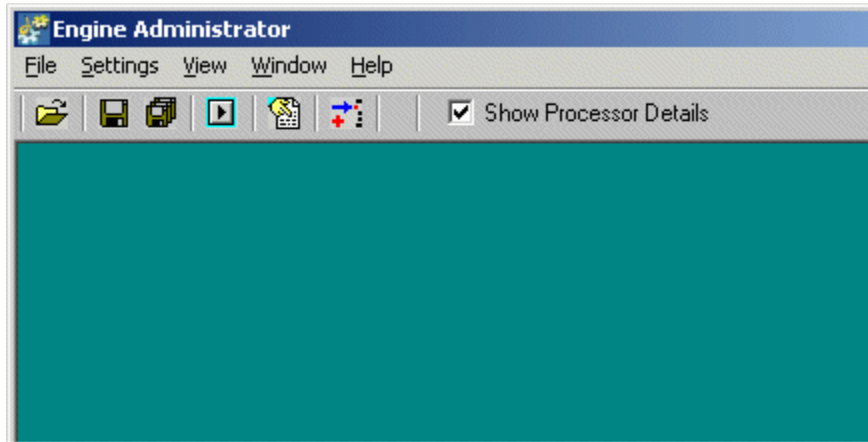
You use the following tools to configure the Analytical Engine:

- The Business Modeler, where you configure causal factors, the forecast tree, and parameters that control the engine. Here you can also define engine profiles.
- The Engine Administrator, where you specify additional engine settings. You save the settings in an XML file for convenience. You can open and use settings files that you have previously saved.

Some of the configuration details are different for the two engine versions (Promotion Effectiveness and demand planning), as noted in the engine manuals.

You use the Engine Administrator to specify logging options, choose machines to run the engine (if you have the Distributed Engine), to choose batch or simulation mode, and to start the engine run.

The Engine Administrator looks like the following:



For information on configuring and running the engine, see "Configuring and Running the Analytical Engine".

Business Logic Engine

The Business Logic Engine evaluates client expressions in the background so that Demantra can update series that use client expressions. The main way to run the Business Logic Engine is to call it from within a workflow step. You specify the worksheet to be evaluated. See "BLE Step".

Collaborator Workbench Administrator

You can use the Web-based Collaborator Workbench Administrator to configure Collaborator Workbench as follows:

- Specify the items on the Planning Applications and Tools and Applications menus.
- Specify which of those menu items are available to each user.
- Specify the default layout of the core content panes (such as My Tasks).
- Specify which core modules are available to each user.

The Collaborator Workbench Administrator looks like the following:

[Define Menus](#)
[Define Program Groups](#)
[Define Program Permissions](#)
[Define Content Security](#)
[Personalize Component Templates](#)

[Logout](#)
[Login to Collaborator Workbench](#)
[Login to Demantra Anywhere](#)

See "Administration".

Initial Phase of Implementation

In the initial phase of an implementation, you gather information and perform groundwork. It is important to analyze the business and demand planning requirements of the enterprise. Generally, you complete a questionnaire that outlines the enterprise business model, products, workflow, sales data, and distribution channels. The information usually includes the following:

- Sales history, including what was sold, where, the quantity and the dates on which an item was shipped
- Other operational/logistics data relating to sales history
- Item and location information
- Information about the various item and location hierarchies that are meaningful to this organization
- For Promotion Effectiveness: Information on sales promotions
- Required lowest-level time resolution

After gathering this information, you should create a detailed design document for later use during the implementation process. The design process itself is beyond the scope of the Demantra documentation.

Middle Phase of Implementation

The main implementation phase uses many tools. Here the steps are grouped loosely into three areas:

- Configuring data and the engine

- Setting up integration and workflow
- Setting up users; configuring and customizing the user interfaces

Data and the Engine

For any implementation, you typically perform all the following tasks:

Task	Tool used	For details, see
Build the data model.	Business Modeler	"Using the Data Model Wizard"
Add more levels if needed	Business Modeler	"Configuring Levels"
Configure the series and groups of series	Business Modeler	"Configuring Series and Series Groups"
Configure units of measure, financial indexes, and conversion rates for use in series and worksheets	Business Modeler	"Configuring Units, Indexes, and Update-Lock Expressions"
Configure Promotion Effectiveness	Database setup scripts	"Configuring Promotion Effectiveness"
Configure DSM	Database setup scripts	"Configuring DSM"
Configure Promotion Optimization	Database setup scripts	"Configuring Promotion Optimization for PTP"

Task	Tool used	For details, see
Configure the engine: <ul style="list-style-type: none"> Set up causal factors Set up the forecast tree For Promotion Effectiveness: Configure the influence groups and influence ranges that affect how the engine works. Tune the Analytical Engine 	Business Modeler	"Configuring and Running the Analytical Engine"
Run the Analytical Engine and check the results Specify additional engine settings and save them in an XML file for convenience.	Engine Administrator	
Set parameters that control Demantra behavior in many ways.	Business Modeler	"Configuring Parameters"

Integration, Workflow, and Automation

During a typical implementation, you typically perform at least some of the following tasks:

Task	Tool used	For details, see
Define import and export mechanisms.	Business Modeler	"Series and Level Integration"
	Demantra Enterprise Integrator	Demantra Enterprise Integrator documentation, available separately

Task	Tool used	For details, see
Load sample data and test the import and export processes.		
Write database procedures to maintain data as needed.	Text editor	Outside the scope of this documentation
Define workflows.	Workflow Editor	"Creating Workflows"

Users and User Interfaces

During a typical implementation, you typically perform at least some of the following tasks:

Task	Tool used	For details, see
Create components, or subdivisions of Demantra data	Business Modeler	"Creating or Modifying a Component"
Specify which levels and series are displayed in the each component		
Create additional users for the components, as needed.	Business Modeler	"Creating or Modifying a User"
Create user groups for collaboration.		
Define security for menu options.	Collaborator Workbench Administrator	"Specifying Permissions for Menu Items"
Define worksheets. Worksheets are visible only within the component where they were defined.	Worksheet designer	Oracle Demantra Demand Management User's Guide or other user manual
Define methods that the users can execute from within the worksheets; redefine or disable default methods.	Business Modeler Workflow Editor	"Configuring Methods"

Task	Tool used	For details, see
Optionally customize Collaborator Workbench.		"Customizing Demantra Web Pages"

End Phase of Implementation

The end phase of an implementation includes the following general steps.

Fine-Tuning

After you define the data model and the components, it is often necessary to iterate by making adjustments to both the data model and the components.

- You may need to use the Business Modeler for the following tasks:
 - Make series and levels available or unavailable
 - Further customize the components
- You may need to adjust the worksheets. To do so, you use the worksheet wizard. See the Oracle Demantra Demand Management User's Guide or other user manual.

Delivering the Solution

To deliver the solution, you must perform the following general tasks:

- Wrap the database procedures. See "Wrapping Database Procedures".
- Make sure that all server machines are configured correctly.
- Set up the client machines and verify that each can run the Demantra software.
- Run the Analytical Engine, examine the results, and tune the Analytical Engine as needed.
- Perform acceptance testing with the qualified users.
- Train users and provide hand-off details. Here you must train end users and one or more designated administrators who will be responsible for maintaining the system in the future.

Hand-off Details

When the system goes online, you should provide the following information to the end users and one or more designated administrators who will be responsible for maintaining the system in the future.

Hand-off Details for the Users

When you hand off the solution to the users, be sure to provide details on the following implementation-specific topics:

- The worksheets that you have pre-configured for the solution: the purpose of each and intended users, as well as the levels, series, and other details that describe those worksheets.
- The level hierarchies and the purpose of each level. Make particular note of the levels that are used in the forecast tree, as those affect the Analytical Engine.
- The series and their interrelationships.
- Which data each user can see.
- How often new data is imported.
- How often the engine runs.
- Workflows that require user participation.

Also, make sure that users are familiar with the basic concepts, as documented in the user manuals. In particular, make sure they know how to make and save changes in the worksheets, as well as understand why changes do not always immediately appear in the worksheets.

Hand-off Details for the System Administrator

When you hand off the solution to an administrator, be sure that the administrator understands how to keep the solution running. Depending on how Demantra is configured, it needs some or all of the following:

Component	When needed	See
Database	Always	Information for Oracle or SQL Server

Component	When needed	See
Workflow Engine	If workflows are being used	"Managing Workflows"
Web server	If solution uses any Web-based components	Documentation for the Web server
Possible other background processes	Varies	Contact the implementors of your Demantra system

Also, be sure to provide details on the following implementation-specific topics:

- The specific automated processes that the solution uses, including any database procedures that must be scheduled to run periodically
- How often the Analytical Engine runs
- Any workflows that are in the solution
- How many components have been defined and who owns them; user IDs and initial passwords; permissions for each user
- User groups, their memberships, and their purposes

The administrator will probably have to add, remove, or change permissions for users, also described in "Managing Security".

Levels

This chapter describes levels and related concepts, outlines the primary configuration options, and summarizes the available tools.

This chapter covers the following topics:

- Introduction
- Level Terminology
- Hierarchy Variations
- Types of Levels
- Members
- Member Identifiers
- Introduction to the Data Model Wizard
- Levels and the Forecast Tree
- Filtered Levels
- Other Concepts Related to Levels
- Configuration Notes

Introduction

Levels control how data is aggregated and organized. They are used in worksheets, in filters, in import and export, and in forecasting. In a worksheet, for example, you can display data at the account level, as follows:

Accounts		Accounts and SKUs				
Members Browser		CVS				
<ul style="list-style-type: none"> BJ CVS McKessen Rainbow Company Ralphs Stop and Shop Walmart 		Time	Demand	Price \$	Revenue \$	Market Plan \$
		02/04/2002	5,757,460	\$10.00	\$57,574,600	\$19,771,290
		05/06/2002	7,674,924	\$10.00	\$76,749,240	\$19,771,290
		08/05/2002	7,285,994	\$9.99	\$72,606,928	\$19,609,128
		11/04/2002	5,736,909	\$9.97	\$57,049,760	\$19,609,128
		02/03/2003	1,571,396	\$9.99	\$15,649,532	\$36,329,952
		05/05/2003		\$10.00	\$67,828,136	\$58,347,280
		08/04/2003		\$10.00	\$63,045,580	\$57,779,168
		11/03/2003		\$10.00	\$47,297,200	\$46,307,320
		02/02/2004		\$10.00	\$49,651,088	\$48,727,416

The worksheet might include a drop down list instead of this tree control. For example:

Accounts Accounts and SKUs

Page Items

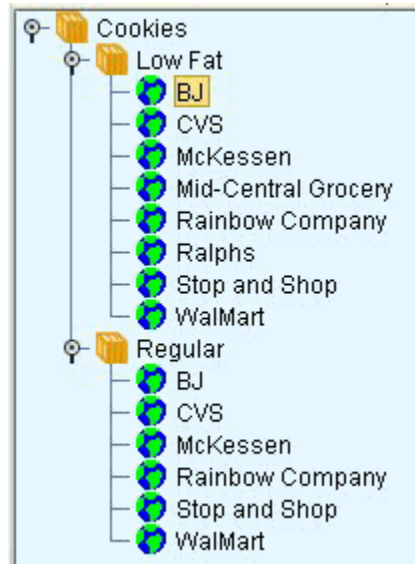
Account CVS

All

Time	Demand	Price \$	Revenue \$	Market Plan \$
02/04/2002	5,757,460	\$10.00	\$57,574,600	\$19,771,290
05/06/2002	7,674,924	\$10.00	\$76,749,240	\$19,771,290
08/05/2002	7,285,994	\$9.99	\$72,606,928	\$19,609,128
11/04/2002	5,736,909	\$9.97	\$57,049,760	\$19,609,128
02/03/2003	1,571,396	\$9.99	\$15,649,532	\$36,329,952
05/05/2003		\$10.00	\$67,828,136	\$58,347,280
08/04/2003		\$10.00	\$63,045,580	\$57,779,168
11/03/2003		\$10.00	\$47,297,200	\$46,307,320
02/02/2004		\$10.00	\$49,651,088	\$48,727,416

In either case, you can display data for any account.

You can use multiple levels together, for example:



In generic terminology, the word *member* refers to a unit within a level. For example, CVS is a member of the account level. When the user hovers the mouse over a member, Demantra displays a hint indicating the name of the level to which that member belongs.

Levels and Filtering

Within Demantra, you generally apply filters by specifying a level and the members of that level to include. For example, the following filter includes only the Rainbow brand.

Available Members		Selected Members	
Code	Description	Code	Description
Default	Default Brand	11	Rainbow
12	Private Label		
99	Demand Profiles		
VHS	Snows Ice Cream		

level members not included in the filter

level members included in the filter

(and therefore displayed in the worksheet)

You can apply multiple filters at the same time. For example, for the preceding worksheet, you could also filter by account.

You can apply use filters in worksheets, in user security, and in import and export.

Levels and Member Management

Within a worksheet, a user can right-click and operate on a member. For example, a

user can edit a member, displaying a dialog box like the following:

The screenshot shows a dialog box titled "Edit SKU : Rainbow Reg Chocolate Chip (Input)". It contains the following fields and values:

Field	Value
Brand	Rainbow
Item Type	0
Life Cycle	Catalog
Make or Buy	M
Measurement Unit	unit
Name	Rainbow Reg Chocolate Chip
Product Family	Regular
Unit Cost	1.25
Unit Volume	1

At the bottom right, there are "Cancel" and "Ok" buttons.

Here the user can edit attributes of the member, including its parent members. Most level members are imported from external systems, and users generally create or change members only if they expect the same change to occur in the imported data.

You can disable or hide the right-click menu options that permit these activities.

Custom Methods

As the implementer, you can define custom methods to perform operations on a selected member, for users to access as an option on the right-click menu. You can apply security to your methods, just as you do with the core right-click actions.

You can define a user security threshold for visibility of that method. For example, you can state the method should only be visible to users who have 'Full Control' of the member from which you launch the method. To control this, you log into the Business Modeler, select 'Configure > Configure Methods'. For 'Method Type'= Custom, you can select from the Security Threshold of Full Control, Read & Write or Read Only.

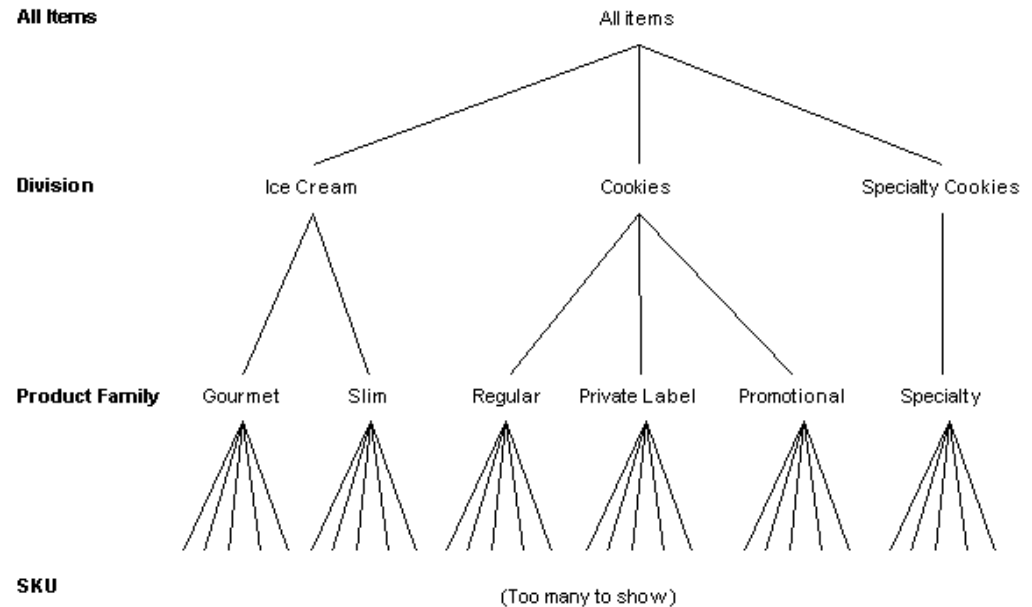
For information on methods, see "Methods and Workflow".

Level Terminology

Demantra uses standard terminology to describe level hierarchies. The following figure shows an example of item levels:



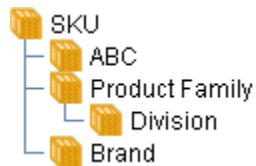
This hierarchy corresponds to the following data:



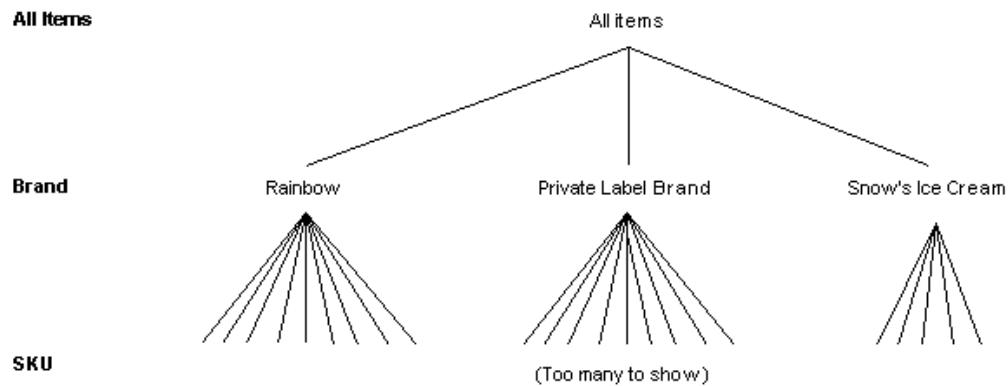
The Product Family level is the parent of the SKU level, and conversely, SKU is the child of Product Family.

Hierarchy Variations

Your application can include multiple, independent hierarchies. For example, you could include the following three independent hierarchies:



The Product Family hierarchy is described above. The Brand hierarchy is as follows:



Note that this hierarchy is independent of the Product Family hierarchy. That is, there is not necessarily any relationship between brands and product families. The ABC hierarchy is not shown here, but is also independent of the other hierarchies.

Each hierarchy can contain as many levels as needed. Some hierarchies are typically much more complex than others.

Types of Levels

Demantra supports the following types of levels, most of which are indicated with different icons:

- *Item levels*, which group data according to characteristics of the items you sell. Item levels typically include brand, item type, product category, and so on.
- *Location levels*, which group data according to characteristics of the locations where you sell. For example, location levels could describe a hierarchy of geographical places or of company organization. Another location hierarchy could organize the locations in some other manner, such as by type of store.
- *Combination levels*, which group data according to time-independent characteristics of the item-location combinations. Combination levels are also known as matrix levels.
- *Time aggregations*, which group data by sales date. You typically use time aggregations on the x-axis of a worksheet, instead of the predefined time units.
- *General levels*, which group data by item, location, and date. General levels are provided as a modeling tool to be used as needed. Demantra does not provide a standard icon for this type of level, because the possible uses can vary a great deal.

Demantra also provides special-purpose levels for use with specific products; see "Special-Purpose Levels".

Note: The desktop products (Demand Planner and Demand Replenisher) can display only item, location, and combination levels. The Web products can support all kinds of levels.

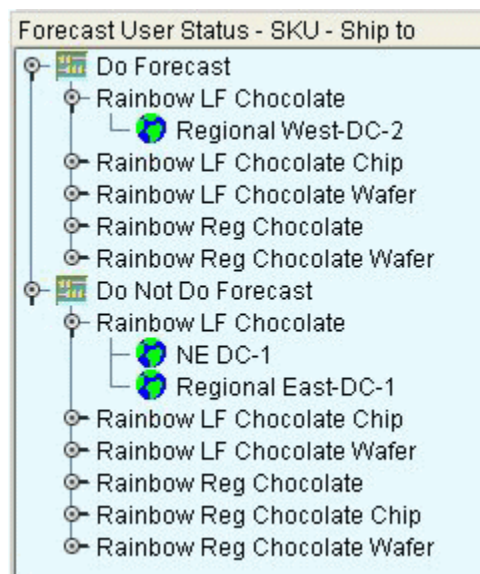
Item and Location Levels

Every application includes at least one item level and one location level. These are useful levels because generally our applications are interested in products and how those products perform at different locations.

Item and location levels are also used in the forecast tree; see "Levels and the Forecast Tree".

Combination Levels

As noted earlier, combination (matrix) levels group data according to time-independent characteristics of the item-location combinations. For example, the following partial worksheet uses a combination level that groups data by a user-controlled flag that determines which combinations the Analytical Engine will forecast:



Time Aggregations

A time aggregation groups data by time and are often used for custom calendars. Your solution can use time aggregations, custom time units, or a combination of both. Use the following guidelines to determine which you need:

na	Names	Uses in worksheet
time aggregation	Each member can have a user-friendly name that you create.	You can use a time aggregation like a level. For example, you can place it on a worksheet axis.
time unit	Each time bucket in the worksheet is automatically labeled with the start date of that bucket.	You can use time units only on the x-axis (the time axis) of the worksheet.

See also "Time Units".

General Levels

A general level groups data by the item, location, and time bucket. Promotion Effectiveness uses general levels to model marketing promotions, but they could be used in other ways.

In addition to ordinary attributes, a general level can have a population attribute, which specifies a set of item-location combinations and consecutive time buckets with which the general level is associated; see "Population Attributes".

Special-Purpose Levels

Demantra also provides the following special-purpose levels:

- Promotion levels, which are used by Promotion Effectiveness and which group data by sales promotions. Depending on how your needs, you may have a hierarchy of promotional levels (to organize the promotions), and the higher levels might use different icons.
- Settlement levels, which are used only by DSM. In general, a settlement is an outstanding sum of money that needs to be resolved, related to a promotion. If you use a settlement level in a worksheet, you cannot use levels from any other hierarchy in that worksheet.
- Check request levels, which are used only by DSM. A check request is an instruction to send a check to a customer or designated third party. Check requests are exported to the accounting systems that actually perform them.
- Configure to Order levels, which support the display and forecasting of a base model's manufacturing components.
- Service Parts Forecasting levels, which are used to support the integration between

EBS Service Parts Planning (SPP) and Demantra Demand Management. These levels support the display and forecasting of a base model's maintenance components.

Members

Each level includes a set of members, each with different characteristics. For example, the SKU level includes a set of members, each corresponding to a different value of SKU.

Member Attributes

A level can have attributes, which are descriptive properties associated with the level (and stored in the table associated with the level). Each member can have different values for each attribute. You use attributes in several different ways:

- To provide extra information for the end users to view and edit, within the worksheets. To view attributes of a member, the user can right-click the member within a worksheet; see "Levels and Member Management".
- To act as levels, that is, to provide a further subdivision of the level data. To do this, you add an attribute to a level and select an option to create it as a child level. For example, suppose you create an attribute called ABC. If ABC can have the values A, B, or C, and if you create this attribute as a level, then the ABC level would have three members: A, B, and C. The member A, for example, would consist of all the data that had the A value for this attribute, within the parent level.
- For integration.
- In the case of promotions, promotion attributes are converted into promotional causal factors for use by the Analytical Engine. See Chapter 44, "Configuring Causal Factors".

You can have different types of attributes (numeric, character, or date), and you can specify a default value for each.

Population Attributes

As noted earlier, a general level can also have a population attribute, which specifies a set of item-location combinations and consecutive time buckets with which the general level is associated.

Because it is more complex than an ordinary attribute, a population attribute is stored in several additional tables that Demantra automatically creates.

Member Defaults

Users can create new members by using the right-click menu in a worksheet. To simplify this work, it is useful to provide default values for the attributes and for the parent members.

For parent members, Demantra provides a predefined default member for each level, and that member is initially named Default level name. You can choose whether to display this member and you can rename it. This predefined default member is not normally associated with any data, however; it is provided for convenience.

If you have data loaded in the system, you can instead choose an existing member to use as the default member.

So, for example, you could use any of the following as the default member of the Brand level:

- The predefined default member: Default Brand
- The predefined default member, renamed by you: Unspecified Brand
- An existing member: Acme

A given level can have multiple parent levels. This means that you can specify a default within each of those parent levels. For example, in the default setup for Promotion Effectiveness, the Promotion level has three parents: Promotion Status, Promotion Type, and Scenario. When a user creates a new promotion, you may want the user to have a default value for each of these.

Member Identifiers

Whenever Demantra retrieves data for a worksheet, it uses the levels in that worksheet to determine how to group the database rows. For example, consider the following set of rows in a table of items.

SKU	SKU_DESC	...	FAMILY	...
RLF0013OR	Rainbow LF Oatmeal Raisin	...	Low Fat	...
RLF0016CH	Rainbow LF Chocolate	...	Low Fat	...
RLF0018VW	Rainbow LF Vanilla Wafer	...	Low Fat	...
RLF0019CW	Rainbow LF Chocolate Wafer	...	Low Fat	...

SKU	SKU_DESC	...	FAMILY	...
PLRG0209C	Private Label Reg Chocolate Wafer	...	Regular	...
PLRG0210S	Private Label Reg Strawberry Wafer	...	Regular	...
RLF0011CC	Rainbow LF Chocolate Chip	...	Low Fat	...
RLF0012PB	Rainbow LF Peanut Butter	...	Low Fat	...
RRG0007CN	Rainbow Reg Cinnamon	...	Regular	...
RRG0008VW	Rainbow Reg Vanilla Wafer	...	Regular	...
RRG0010SW	Rainbow Reg Strawberry Wafer	...	Regular	...
RSP0021AC	Rainbow Spc Animal Crackers	...	Specialty	...

Here the FAMILY field indicates the product family to which each SKU belongs. When aggregating to the family level, Demantra groups together all the rows with the same value of FAMILY.

The field that can be used in this way is called the code field for the level. When you define a level, you identify the field that Demantra should use as the code field for that level. For each unique value of the code field, all records with that value are grouped together as one member. (The nature of the grouping is controlled by the series definitions, as described in "Series".)

Code and Description Fields

For each level, the enterprise data must have at least one unique field that can be used to distinguish level members; this is used as the code. In some cases, the enterprise may have two fields: an internal identifier and the corresponding user-friendly, "pretty name," to be used as the description field. The SKUs in the previous example have two

such fields: SKU and SKU_DESC.

If the enterprise data includes only a single field that is unique to a given level, you use that field as the code and the description.

ID Field

Internally, Demantra generates and uses its own unique numeric ID, which is not meant to be exported to external systems.

Code Display Field

Within the Demantra user interface, users see and use two unique labels for each level member, the code display field and the description field:

Code	Description
Default	Default SKU
PLLF0211C	Private Label LF Cho
PLLF0212P	Private Label LF Pear
PLLF0213O	Private Label LF Oatr
PLLF0214B	Private Label LF Butti

63 rows retrieved.

As the implementor, you can configure the code display field in three different ways.

- The code display field can be equal to the internally generated numeric ID.
- The code display field can be equal to the code field.
- The code display field can be equal to the description field.

Introduction to the Data Model Wizard

Level definitions are generally coupled tightly with integration, because each level is defined by a code field, and most code fields are imported from corporate systems. Consider, for example, a level such as Brand. Any SKU belongs to a brand, and Demantra imports that information. Adding an item or location level usually requires a new field in the data, which also requires changes to the integration.

The Data Model Wizard (in the Business Modeler) therefore has two related, primary purposes:

- To create batch scripts to load the lowest level item, location, and sales data.
- To create database procedures that define Demantra levels based on that data.

Note: The Data Model Wizard also performs other configuration tasks, not discussed here, that can be performed only within the Data Model Wizard.

The Data Model Wizard prompts you for the location and format of the raw data. It helps you describe the contents of the staging tables so that the data can be moved into the Demantra internal tables. You then specify how to use the fields in the staging tables, generally using each field in a level definition or in a series definition. Demantra ignores any field for which you do not specify a use.

The Data Model Wizard is discussed further in "Loading Basic Data".

As a final result, the Data Model Wizard creates a batch script and database procedures. The script executes the procedures, which load the data into the Demantra internal tables and synchronize the tables as needed.

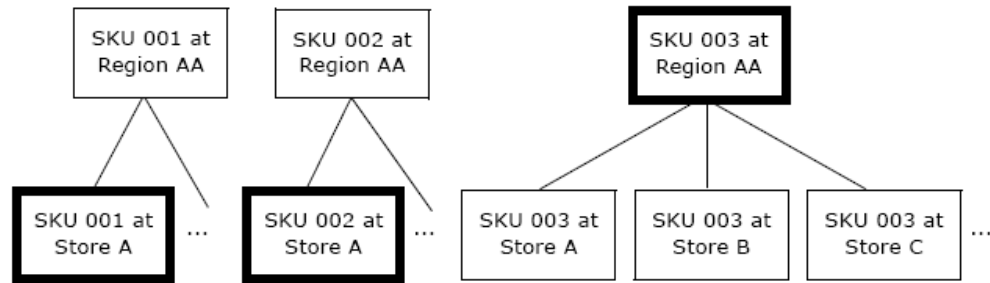
Note: You can define additional levels later, outside of the Data Model Wizard, but you should do so only if you do not need a corresponding change in the load scripts. The Data Model Wizard automatically removes any levels that you define outside of the wizard.

Levels and the Forecast Tree

If your application uses the Analytical Engine, you will also need to consider what sort of forecast tree you will need. The forecast tree organizes data for use by the Analytical Engine. In this hierarchy, each node aggregates by both item and location. General levels can also be included in the forecast tree.

Demantra supports multiple analytical engines, with associated forecast trees. Demantra supports multiple analytical engines, with associated forecast trees. Depending on which engine profile you choose, Demantra will apply different settings and can provide different forecasting results. For example, when using Booking Forecast profile, the forecast generated will be focused on bookings for an item. Similarly, the Forecast Spares Demand profile will generate a forecast for service parts.

In general, forecasting is most accurate when it can be performed at the lowest possible allowed aggregation level. However, sometimes there is not enough data at that level for all combinations. For those combinations, the Analytical Engine aggregates the data to a higher level and tries to generate a forecast there. Consider the following example, showing a small part of a forecast tree.



The bold boxes show the nodes at which the Analytical Engine is forecasting.

- In this example, there is enough data at the SKU-store level for SKU 001 and SKU 002; the Analytical Engine can generate a forecast at that level for those SKUs.
- On the other hand, there is less data for SKU 003, so the Analytical Engine aggregates data for that SKU across all the stores in Region AA, generates the forecast for those SKUs at the SKU-region level, and then splits to the store level.

See *Proport When Forecasting on General Levels*, page 11-11 for more information about how the proportion handles supersessions.

Filtered Levels

By default, a level of a given type groups all the associated data; for example, an item-type level groups all the item data. You can, however, create filtered levels. A filtered level contains a filtered subset of the data. To create a filtered level, you join the underlying data to another table of your choice, and you then add an SQL WHERE clause to filter the data.

To do this, you use the EXTRA FROM and EXTRA WHERE options for the level.

Other Concepts Related to Levels

After you configure levels, you associate them with several other kinds of Demantra objects.

Levels and Units

In Demantra, you associate each unit with the levels where it makes sense to use that unit. For example, a relatively small unit might make sense only at lower levels.

Demantra uses this association within worksheets. If a worksheet contains three levels, for example, then only the units associated with those levels can be used in that worksheet.

For information on units, see "Units, Indexes, and Exchange Rates".

Levels and Methods

It is useful to be able to right-click a level member within a worksheet and perform some operation on it. With Demantra, you can define methods, which the user sees as an ordinary right-click menu options. Demantra also provides a set of default methods that you can redefine or disable as needed; these allow the users to view, edit, delete, and so on.

Each method is associated with a specific level. Also, a method can be available in all worksheets or in a single specific worksheet. You can apply security to all methods.

For information on methods, see "Methods".

Level and Worksheet Association

It is useful to be able to examine a level member more closely, to launch a worksheet from that member that is filtered to show only that member. But typically, a Demantra application includes a large number of worksheets, and most of those worksheets would not be useful in this way. So Demantra provides an option for associating each level with any number of worksheets. Demantra uses this association in two ways:

- A user can start from a level member and launch a worksheet that is filtered to that member. To do so, the user right-clicks the member and clicks the Open or Open With option. Alternatively, this worksheet can show just the combination from which the user started.
- A worksheet can include a sub tab worksheet that is associated with any of the levels in the main worksheet. Then when a user selects a member in the main worksheet, the sub tab shows the details.

For information on worksheets, see "Worksheets".

Configuration Notes

This section contains configuration notes related to levels.

Standard Levels

Some Demantra products, such as DSM, assume that your database contains specific levels, parameter settings, and other configuration options. Demantra provides setup scripts that perform all the required configuration. See Part IV, "Other Configuration".

Dependencies

Before you can configure levels, you will need some sample data.

Because level definitions are generally coupled tightly with integration, you typically

need to define levels at the same time as your basic loading scripts. For this, you use the Data Model Wizard.

You can define additional levels later, outside of the Data Model Wizard, but you should do so only if you do not need a corresponding change in the load scripts. The wizard automatically removes any levels that you define outside of the wizard.

Tools

Demantra provides the following tools for creating and configuring levels and related objects:

Tool	Purpose/Notes
Data Model Wizard*	Defines levels (and other model elements) and creates a script to import data, particularly sales, item, and location data, at the lowest level.
Configure > Levels option*	Defines additional levels.
CREATE_PE_STRUCT procedure	Creates basic promotion levels needed by Promotion Effectiveness. You can customize these levels to some extent, after running the procedure.
UPGRADE_TO_DSM procedure	Creates the settlement and check request levels required by DSM.
Configure > Units for Levels option*	Associates units with levels.
Configure > Methods option*	Defines methods.
Components > Open/Create Component option*	Creates components, which define the associations between levels and worksheets.
*These options are in the Business Modeler.	

For information on these tools, see Part II, "Basic Configuration".

Series

This chapter covers the following topics:

- Introduction
- Main Series Options
- Data Types of Series
- Types of Series
- Update Field
- Editability
- Series Calculation: Server Expressions
- Series Calculation: Client Expressions
- Series Calculation: Using Both Server and Client Expressions
- Series Calculation: Proportionality Option
- Summary of Calculation Options
- Display Properties
- Color Expressions
- Display Precision and Format
- Display-Only Summaries
- Configuring the Display-Only Summary
- Other Basic Series Options
- Advanced Series Options
- Preserving Promotional Data While Moving or Copying
- The Aggregated Base Level Option
- Extra From and Extra Where
- Note and Promotion Indicators

- Series Groups
- Configuration Notes

Introduction

A series is a set of data that can be displayed in a worksheet table or graph, at any aggregation level. The following example shows several series displayed at the lowest level:

Private Label LF Butter - BJ Store # 0006							
Time	Demand	Final Plan	Pseudo	Simulation	Sales Forecast	Sales Fcst Bias	Stat Frcst (Y/N)
04/08/2002	1,258,700				1,240,202	-18,498	Do Forecast
07/08/2002	1,232,800				1,161,719	-71,081	Do Forecast
10/07/2002	1,326,200				1,057,580	-268,620	Do Forecast
01/06/2003	488,500				903,675	415,175	Do Forecast
04/07/2003		1,193,227			1,193,227		Do Forecast
07/07/2003		1,123,295			1,123,295		Do Forecast
10/06/2003		1,040,942			1,040,942		Do Forecast
01/05/2004		820,737			820,737		Do Forecast
04/05/2004		280,121			280,121		Do Forecast
Summary	4,306,200	4,458,322			8,821,497	14,244	

Main Series Options

When you define a series, you specify many options. To start, the following list gives an overview of the main, interrelated options:

- An external and an internal identifier. The external identifier appears in worksheets, and you use the internal identifier when you refer to the series in server expressions.
- The data type of the series.
- An option that controls the type of series. For example, some series are associated with sales; these series potentially have different values for each time bucket and each item-location combination. (If you are familiar with database terminology, note that this property determines the primary key of the series.)
- An option that controls whether the series is stored in the database or not. If a series is stored in the database, the series has an *update field*, which is the database column that stores the series data.
- Options that control whether the series is editable, and if so, under what conditions.
- Options that control how the series is calculated at any aggregation level and how series data is split to the lowest level. These options include the server expression,

the client expression, the proportionality option, and the proportions reference series.

The following sections discuss these options. Additional options are discussed later in the chapter.

Data Types of Series

Demantra supports the following data types for series:

- Numeric
- String
- Date

Note: The desktop products (Demand Planner and Demand Replenisher) can display only numeric series.

Types of Series

Demantra supports the following types of series:

sales series	Consists of time-dependent data for each item-location combination. That is, each data point in the series corresponds to a given item-location combination at a given point in time. This type of series is the most common.
matrix series	Consists of time-independent data for each item-location combination. That is, each data point in the series corresponds to a given item-location combination. You use matrix series to store and maintain information about item-location combinations.
promotion series	Consists of data for each promotion at each item-location combination, at each time bucket.

level series	<p>Stores data associated with a specific level. Each data point in the series corresponds to a given member of that level. For example, suppose that a level is the page size in a catalog, which lets you view and group items by their assigned page sizes. If you created an editable level series, you could easily reassign items to different page sizes.</p>
update-by series	<p>The Business Modeler allows for the creation of a series with server expressions that retrieve data from one of several columns, according to a CASE statement based on the combination's context. To better support this functionality, a non-fixed update field is required. This allows for an update to go to a dynamic, context-based column.</p> <p>Series that require this functionality must have an Update-By series specified. The series specified as the Update-By series must be of type String, and should return a specific column name. Updates generated from the updated series will be stored in the column string results of the Update-By series. The updated and Update-By series are handled as pairs, both in the retrieve process and the update process. For example:</p> <p>Series S1: Server expression= 'sum(case when Level_id = 1 then column1 when Level_id = 2 then column2 when Level_id = 3 then column3 else column4 end)' Update by series= S2.</p> <p>Series S2: Type=String Server expression= 'sum(case when Level_id = 1 then 'column1' when Level_id = 2 then 'column2' when Level_id = 3 then 'column3' else 'column4' end)' Note: Series S2 returns stings which match the actual column names.</p>

Note: The desktop products (Demand Planner and Demand Replenisher) can display only sales and matrix series.

Update Field

A series may or may not be stored in the database. If it is stored, its data is saved in the series update field. This option is known as the update field because it refers to the field that is updated when changes are saved to the database.

When you use the Business Modeler to configure a series, it automatically adds the update field if needed.

Although you generally should avoid working directly in the database, when you configure series, you need to write SQL expressions to aggregate data from the tables in which the series are stored. Depending on the type of series, the update field is in one of the following tables:

For sales series	sales_data
matrix series	mdp_matrix
promotion series	promotion_data (not promotion as implied by the Business Modeler)
For level series	Table associated with the level.

Demantra provides an alias (branch_data), which you can use to refer to sales_data or promotion_data.

Editability

You control whether a series is editable in a combination of the following ways:

- You can make an entire series editable or non-editable with a single setting.
- You can control the time periods during which a series is editable. To do so, you specify whether the series is associated with history, forecast, or history and forecast. For an editable series:
 - If the series is configured as history, then it is editable only in the current time bucket and previous time buckets.
 - If the series is configured as forecast, then it is editable only in the current time bucket and future time buckets.
 - If the series is configured as history and forecast, then it is editable for all time buckets.

- You can apply an edit-lock expression to the series to further restrict editing. An edit-lock expression evaluates to true or false; for each cell where the expression is true, that cell is locked.

Series Calculation: Server Expressions

A server expression must be an aggregating SQL expression that returns a value with length greater than zero for each element. (If you never plan to use the series within a cached worksheet, it can return null or a zero-length value; but you may not be able to prevent the series from being misused.)

A server expression must have one of the following forms:

aggregate_function (branch_data.database_column * #UNIT#)

aggregate_function (branch_data.database_column)

aggregate_function (mdp_matrix.database_column * #UNIT#)

aggregate_function (mdp_matrix.database_column)

aggregate_function (other_expression)

Here:

- *aggregate_function* is one of the SQL aggregating functions, most commonly sum.
- *database_column* is a column of the branch_data or mdp_matrix table, most often the update field that corresponds to this series. That is, if SeriesA is associated with branch_data.SeriesA, then the server expression for SeriesA could be sum(branch_data.SeriesA)

Note: branch_data is a synonym for the sales_data table or the promotion_data table.

- #UNIT# represents the unit conversion factor. Within a worksheet, this token is automatically replaced by the conversion factor that corresponds to the unit that the worksheet is currently using.

In turn, *other_expression* can be made up of some or all of the following components:

- Other SQL functions.
- Constants and expressions that have numeric, string, date, and true/false values.

Note: Enclose any literal negative value within parentheses, as in this example: (-0.33)

- Operators such as +-*./.
- Demantra tokens such as #UNIT#.
- Columns of the branch_data and mdp_matrix tables.

Note: These column references must explicitly have the table name (or branch_data) included.

You can use parentheses to control the precedence of calculations, according to standard algebraic rules.

Caution: SQL expressions have a limit of 3000 characters. To avoid reaching this limit, use small field names.

For information on the supported operators, tokens, and SQL functions, see "Server Expression Functions and Operators".

Forecast Versions

Each time the Analytical Engine runs, it generates a forecast. Each forecast is associated to the engine profile used to create it. The default profile being used is the *Batch* profile. In addition, each forecast generated receives a forecast version. The most recent forecast is version 0, the previous one is version 1, and so on.

Each series can be implicitly or explicitly associated with a specific forecast profile and version, or possibly with several. Typically, the large majority of series are associated with the most recent forecast generated using the base profile, but it is often useful to configure some series to capture information associated with a previous forecast, or to compare forecasts generated using different profiles.

You can include forecast profiles and versions, if needed, in the server expression for the series. When you specify a server expression, you should specify the forecast version and engine profile used to generate it. To do so, you use the #FORE@<Version>@<Engine Profile># token. The worksheet mechanism will dynamically replace this token with the appropriate forecast. For example, #FORE@0@25# is replaced by the current forecast version generated by using engine profile 25, and #FORE@1@52# is replaced by the most recent previous forecast version generated using engine profile 52. If the engine profile is not designated in the token, the token will default to the base forecast profile.

The server expression can refer to multiple forecast versions, for example, to compare them.

In the case of Promotion Effectiveness, the forecast details are more complex, because the Analytical Engine decomposes the forecast into multiple effects. Therefore, Oracle Demantra provides tokens such as #SW_BRAND@<Version>@<Engine Profile># and

#SW_CHANNEL@<Version>@<Engine Profile># for these separate effects. See "Server Expression Functions and Operators".

Note: Within the hint message for a series, you can include the token #FDATE@<Version># to refer to the date on which a forecast version was generated. This can be very useful to the users of the worksheets.

Units of Measure

You can include the #UNIT# token, if needed, in the server expression for the series. At any time, a worksheet uses one unit of measure, which is used by most of the numeric series in that worksheet. The user can switch to another unit of measure, and all those series are correspondingly scaled by the appropriate conversion factors.

Note: You can instead hard code the unit into a series definition, so that it expresses, for example, the buyback per case. Whatever your choice is, be sure to give useful names and hints to the series.

For more information on units, see "Introduction".

Series Calculation: Client Expressions

Expressions

A client expression uses Demantra functions. The client expression can be made up of some or all of the following components:

- Constants and expressions that have numeric, date, true/false or null values.

Note: Enclose any literal negative constant within parentheses, as in this example: (-0.33)

- Demantra functions.
- Operators such as +-*/*.
- References to other series. To refer to a series, you use the name of the series.
- References to series at other time periods. Here, you use the following syntax:
series_name [relative-time-bucket]

An expression like this is sometimes called a vertical formula. For example: Sales [-1] refers to the Sales series for the previous period. Sales [1] refers to the Sales series for the next period. [0] is not allowed.

Here relative-time-bucket must be any of the following:

- An integer
- A series name
- A simple expression using integers, series names, and the basic mathematical operators.

For information on the supported operators and functions, "Client Expression Functions and Operators".

Time-Shifted Client Expressions (Vertical Formulas)

When a client expression includes a reference to another series, by default, Demantra uses data from the same time bucket. You can refer to data from earlier or later time buckets, however. The following example shows three series, each of which has a client expression that refers to the Example series.

Date	Example	Unshifted	Shifted1	Shifted2
12/2/2002	2	2		3
12/9/2002	3	3	2	4
12/16/2002	4	4	3	5
12/23/2002	5	5	4	6

Client expression:
Example Series

Client expression:
Example Series[-1]

Client expression:
Example Series[1]

Notice that the series Shift1 is null for 11/25/3002. This is because this cell refers to the Example series at a time bucket that is not displayed in the worksheet.

Null Sales Records and Time-Shifted Client Expressions

You do not typically have sales records for all combinations for all dates. This affects client expressions that refer to series at other time buckets. When a client expression refers to a time bucket that does not have sales data, Demantra automatically uses the next available non-null sales data. The following figure shows an example:

Date	Example	Shifted1
2/24/2003	2	
3/3/2003	3	2
3/10/2003	4	3
8/11/2003	5	4
8/18/2003	6	5
8/25/2003	7	6

For 8/11/2003, Shifted1 series refers to Example series at last non-null time bucket, 3/10/2003

Client expression:
Example Series[-1]

Using Expressions to Refer to Time Buckets

The previous examples have used the simplest syntax for time-shifted client expressions, in which you use an integer to refer to the relative time bucket. You can instead use simple expressions that include series names, integers, and mathematical operators. For example, if you have series A, B, C, and D, the client expression for D could be something like the following: A[B+C]

For example, suppose you want to know how much inventory your warehouse will contain on a given date. The date is determined relative to today based on both the production lead time and the transportation lead time. That is, you want to know Inventory[Production lead time + Transportation lead time].

Precedence of Calculations for Client Expressions

The following rules apply for the recalculation order, and will be performed recursively:

1. The system looks for vertical formulas, which use data in other time buckets. Such as formulas either use a function such as FSUM or they reference data in time buckets: Demand [2].
2. The system calculates the data series that are the source for those in the Step 1.
3. The system calculates the data series of Step 1.
4. The system calculates the series that use the series in Step 1 as source.
5. The system calculates the series that are the result of Step 4, and so on.

Series Calculation: Using Both Server and Client Expressions

It is important to understand how server and client expressions are used in combination. All series must have a server expression, although the expression is not always important. The client expression always takes precedence. That is, the client expression, if present, is evaluated, displayed, and stored in the database, instead of the server expression.

If a series has a client expression, the series should be configured in one of the following ways:

- The server expression is trivial. For example, it is a constant such as 0. Because this value is never meant to be seen or stored, the specific value is irrelevant.
- The server expression is meaningful and useful in some cases. In this case, the client expression consists of a conditional expression of the following general form:

If (*condition*, *client-expression-value*, *series-name*)

Here *series-name* is a reference to the same series to which this client expression applies. This reference directs Demantra to the server expression that this series uses. Depending on whether the condition is true for a given cell, this expression returns either the client expression value or server expression value.

Note: In some cases, it is useful for a client expression to null out data in the worksheet table in situations where the data might be confusing.

There is another important difference between server and client expressions, as far as end users are concerned. Server expressions are evaluated in the background and the resulting changes are not available instantly in the worksheets. Client expressions, on the other hand, are evaluated immediately.

In many cases, a server expression and a client expression can be mathematically equivalent, but the client expression might be more user-friendly.

Series Calculation: Proportionality Option

When you define a series as proportional, you need to choose the proportional by series from a series dropdown. For a series to show up in the dropdown, it must have a non-zero/not null server expression. It does not have to be proportional; a proportion by series does not need to be proportional itself.

In general, the definition of a series also specifies how to calculate data at the lowest level, in the case when data changes at a higher level. Data can potentially change at a higher level either when it is imported at a higher level or when users edit a series while displaying data at a higher level.

Each series can be configured as proportional or non-proportional.

- If a series is proportional, the parent value is split among the child members according to the proportions of those child members.
- If a series is non-proportional, the value for each child member is set equal to value of parent.

When you configure a series as proportional, you also specify a proportions reference series. For best performance, Oracle recommends the following:

- Proportions from the same table are better than proportions from a different table.
- If the proportions are not in the same table that stores the series that uses those proportions, consider caching the proportions into the same table that stores the series. For example: create a cache of `GLOB_PROP` in `sales_data` and `promotion_data`.
- Use `PROPORTION_COLUMN` when the proportions are from the same table and do not require a server expression.
- Use a series that is not empty (most of the time) for the proportion reference.

Supported Proportion Configuration

The series available in the proportional series dropdown depend upon the table on which the series is defined.

The series listed in the dropdown are also filtered by the table on which the proportional series exists:

- Series on `sales_data`: Can be split by any series with Item or Location levels, for example, `sales_data`, and `mdp_matrix`,
- Series on a data table: For example, `promotion_data` can be split by `sales_data`, `mdp_matrix`, Item or Location levels, and its own base level.
- Series on level: All levels with the same base level as the level and lower or equal aggregation to the series level.
- Series on `mdp_matrix`: Can be split by any series with Item or Location levels, for example, `mdp_matrix` except for `sales_data`,

Summary of Calculation Options

When you configure a series, you have many options to set, and not all the combinations are useful. This section summarizes the useful combinations of the most important series options.

Combinations of Key Series Options

The following table summarizes the combinations of the most important series options:

Expression	Update field	Proportional	Editable
Server only or Server and client	Yes	Depends on the nature of the server expression	Editable or non-editable*
	No	Non-proportional only	Probably should be non-editable.
Client only (server expression is trivial and its value is never seen)	Yes	Depends on the nature of the client expression	Non-editable only
	No	Non-proportional only	

*Depending on how the series is configured, it may be necessary to ensure that data changes only at the lowest level. Apart from those cases, these series can be either editable or non-editable.

When to Configure a Series as Proportional

The following table indicates when to make a series proportional:

Update field	Form of expression*	Proportional	Editable	If data changes at a higher level...
Yes	SERVER: sum (<i>table_name.update_column_name</i>)	Should be proportional	Editable or non-editable	Lower levels are calculated by splitting the higher-level value according to the proportions in the Proportion Calculation series.

Update field	Form of expression*	Proportional	Editable	If data changes at a higher level...
	SERVER: avg (table_name.update_column_name) max (table_name.update_column_name) or min (table_name.update_column_name)	Should be non-proportional	Editable or non-editable	Value for each lower level is set equal to value of parent level.
	Any other expression	Should be non-proportional	Should not be editable except at lowest level	Undesirable behavior occurs.
	CLIENT: Name of a proportional series	Should be proportional	Non-editable	Lower levels are calculated by splitting the higher-level value according to the proportions in the Proportion Calculation series.
	CLIENT: Any other expression	Should be non-proportional	Non-editable	Value for each lower level is set equal to value of parent level.
No	Any	Must be non-proportional	Non-editable	Undesirable behavior occurs.

*Where *table_name.update_column_name* is the update field for this series. In all cases, the expression can also include the token #UNIT#, which represents the unit conversion factor. For example: sum (table_name.update_column_name * #UNIT#)

Useful Series Configurations

For any series, data can safely be changed at the lowest level. Depending on how the series is configured, it may or may not be safe to change data at higher levels.

The following table indicates which series configurations support data changes at higher levels:

Update field	Form of expression*	Proportional	If data changes at a higher level...
Yes	Server expression: sum (<i>table_name.update_column_name</i>)	Should be proportional	Lower levels are calculated by splitting the higher-level value according to the proportions in the Proportion Calculation series.
	Server expression, any of the following: avg (<i>table_name.update_column_name</i>) max (<i>table_name.update_column_name</i>) or min (<i>table_name.update_column_name</i>)	Should be non-proportional	Value for each lower level is set equal to value of parent level.
	Any other expression	Should be non-proportional	Undesirable behavior occurs.
	Client expression: Name of a proportional series	Should be proportional	Lower levels are calculated by splitting the higher-level value according to the proportions in the Proportion Calculation series.
	Client expression: Any other expression	Should be non-proportional	Value for each lower level is set equal to value of parent level.
No	Any	Must be non-proportional	Undesirable behavior occurs.
<p>*Where <i>table_name.update_column_name</i> is the update field for this series. In all cases, the expression can also include the token #UNIT#, which represents the unit conversion factor. For example: sum (<i>table_name.update_column_name</i> * #UNIT#)</p>			

Series That Can Be Changed at Any Level

For any series, data can safely be changed at the lowest level. Depending on how the series is configured, it may or may not be safe to change data at higher levels.

The most common series are the ones that are configured so that the data can be changed at any level. Remember that data can change for many reasons, by editing within a worksheet, by importing changed data, or by changing data from which the series is derived.

Update field	Proportional	Form of expression	*If data changes at a higher level...
Yes	Proportional	<p>SERVER:</p> <p>sum (<i>table_name.update_column_name</i>) Where <i>table_name.update_column_name</i> is the update field for this series.</p> <p>CLIENT:</p> <p>Name of a proportional series</p>	Lower levels are calculated by splitting the higher-level value according to the proportions in the Proportion Calculation series.
Yes	Non-proportional	<p>SERVER:</p> <p>avg (<i>table_name.update_column_name</i>) max (<i>table_name.update_column_name</i>) or min (<i>table_name.update_column_name</i>)</p> <p>CLIENT:</p> <p>Any other expression</p>	Value for each lower level is set equal to value of parent level.

*Where *table_name.update_column_name* is the update field for this series. In all cases, the expression can also include the token #UNIT#, which represents the unit conversion factor. For example: sum (*table_name.update_column_name* * #UNIT#)

Series That Must Be Changed Only at the Lowest Level

If a series is configured in the following ways, it should be edited or changed only at the lowest level:

Update field	Form of expression	Proportional
Yes	Any expression other than the ones in "Series That Can Be Changed at Any Level".	Should be non-proportional. Otherwise, undesirable behavior occurs.
No	Any	Must be non-proportional.

Calculating Data at Lower Levels

For a series that has a server expression and that is stored in the database, Demantra needs to know how to calculate data at lower levels if data changes at a higher level. When you configure a series, you specify whether the series is proportional. The following table shows which series should be proportional and explains how these series behave.

Form of server expression	Proportional	If data changes at a higher level...
sum (<i>table_name.update_column_name</i>) Where <i>table_name.update_column_name</i> is the update field for this series.	Series should be proportional.	Lower levels are calculated by splitting the higher-level value according to the proportions in the Proportion Calculation series.

Form of server expression	Proportional	If data changes at a higher level...
<code>avg (</code> <code>table_name.update_column_name</code> <code>) max (</code> <code>table_name.update_column_name</code> <code>) or min (</code> <code>table_name.update_column_name</code> <code>)</code> Where <code>table_name.update_column_name</code> <code>e</code> is the update field for this series.	Series should be non-proportional.	Value for each lower level is set equal to value of parent level.
Any other expression	Series should be non-proportional.	Data should not be changed except at lowest level.

Display Properties

You can control how Demantra displays each series in a variety of ways.

Color Expressions

Any series can have a color expression, which controls the background color of the series when displayed in a worksheet table.

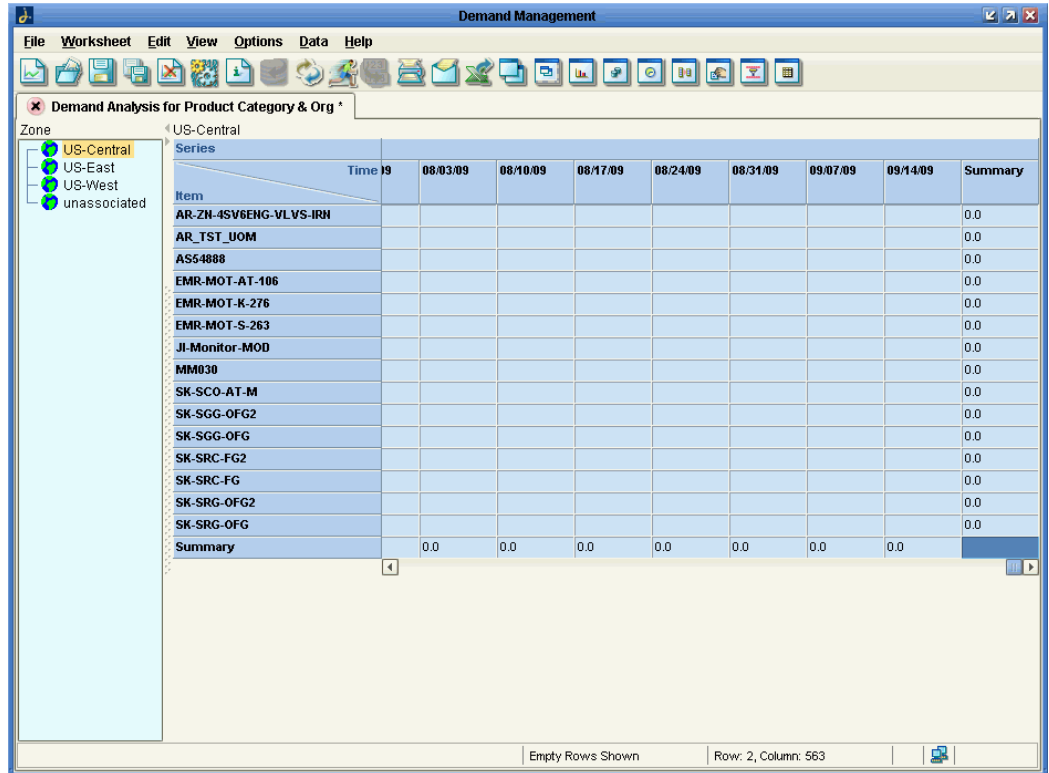
Display Precision and Format

For each numeric series, you can specify the format that worksheet tables should use when displaying the series. By specifying this format, you are also implicitly specifying the maximum possible size of numbers in the series.

For example, if the display format of a series is `##,###.##`, the maximum size of a number in this series is 99999.99.

Display-Only Summaries

Summary rows may appear in a worksheet as either rows or columns, or both as shown in the example below:



For each series, you also can specify a summary function or expression for use only within the worksheet. The following figure shows examples of Total and Average, in a worksheet:

	Sales Forecast	Ent Factor	Fixed Plan Lift	Ent Forecast
10/06/2003	278	0.0%	100	378
11/03/2003	264	0.0%		264
12/01/2003	304	0.0%	50	354
01/05/2004	438	25.0%		547
02/02/2004	668	0.0%		668
03/01/2004	184	0.0%		184
Summary	2,136	4.2%	150	2,395

↑
Total
↑
Average
↑
Total
↗

The summary is only for display and the results are not stored in the database. However, to avoid user confusion, you should probably summarize data in a manner consistent with the server or client expressions you define for this series; see "Series Calculation: Using Both Server and Client Expressions".

For example, if you define this series by a server expression that sums data, the summary function should probably be Total.

The worksheet table may also include subtotal rows. The following shows an example:

Product Family	Account	Time	Price \$	Revenue \$	Discount	Approved
Gourmet	Stop and Shop	03/10/2003	\$10.00	\$6,963,948	9.00%	<input type="checkbox"/>
		09/08/2003	\$10.00	\$3,190,742	0.00%	<input type="checkbox"/>
		03/08/2004	\$10.00	\$3,141,407	0.00%	<input type="checkbox"/>
		Summary	\$10.00	\$13,296,097	3.00%	3
	Summary		\$10.00	\$13,296,097	3.00%	1
Regular	Stop and Shop	09/08/2003	\$10.00	\$52,827,440	0.00%	<input type="checkbox"/>
		03/10/2003	\$9.14	\$47,052,748	9.00%	<input type="checkbox"/>
		03/08/2004	\$10.00	\$13,265,270	0.00%	<input type="checkbox"/>
		Summary	\$9.71	\$113,145,464	3.00%	3
	McKessen	03/10/2003	\$10.00	\$769,117	0.00%	<input checked="" type="checkbox"/>
		09/08/2003	\$10.00	\$1,168,671	0.00%	<input type="checkbox"/>
		03/08/2004	\$10.00	\$308,523	0.00%	<input type="checkbox"/>
		Summary	\$10.00	\$2,246,312	0.00%	3
	Summary		\$9.86	\$115,391,776	1.50%	2
Slim	Stop and Shop	03/10/2003	\$10.00	\$22,640,676	9.00%	<input type="checkbox"/>
		09/08/2003	\$10.00	\$21,201,430	0.00%	<input type="checkbox"/>
		03/08/2004	\$10.00	\$7,785,908	0.00%	<input type="checkbox"/>
		Summary	\$10.00	\$51,628,012	3.00%	3
	Summary		\$10.00	\$51,628,012	3.00%	1
Summary			\$9.93	\$180,315,888	2.25%	4

A given series can be summarized in different ways within a single worksheet table, although that usually means that the series is useful only within that worksheet.

Configuring the Display-Only Summary

You can customize the summary line to provide flexibility in cross tabulating columns/rows, and reporting purposes. These customizations include:

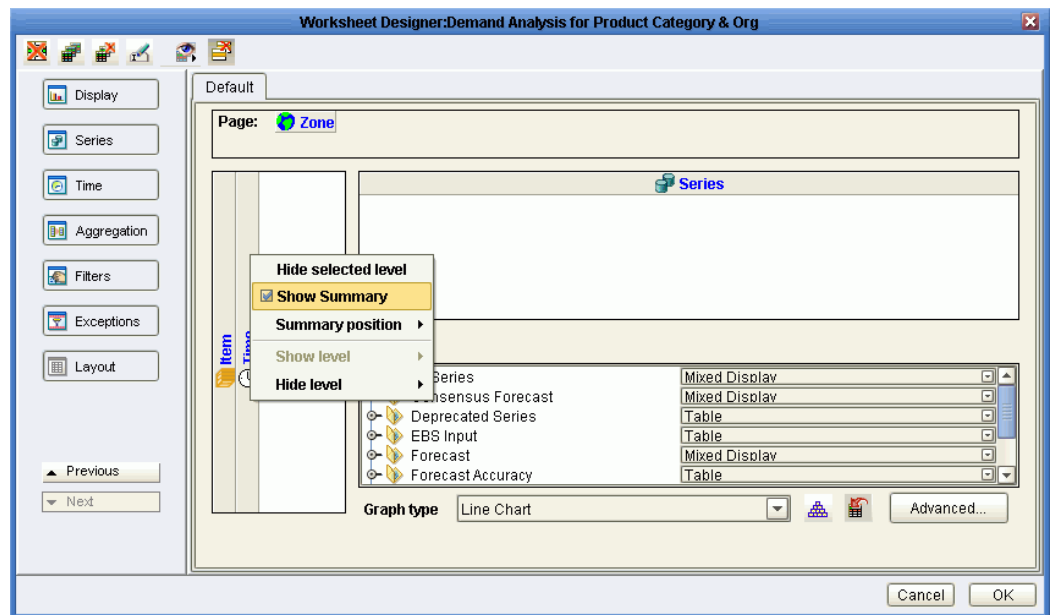
- Hiding or showing the summary
- Freezing the summary in a particular place on the worksheet
- Setting the summary location

To toggle the Summary Line:

It is often desirable to eliminate the summary line within a given section of the cross-tabulated worksheet. For example, you may not wish to see a summary of all promotions within a scenario because a separate view is used for this purpose.

1. From the Collaborator Workbench, open the worksheet whose summary line you want to configure.
2. From the Worksheet menu, choose Layout Designer.
The Worksheet Designer appears.
3. Click the Layout button.
4. Right-click on the series that you want to configure, and either enable or disable the Show Summary check box.
5. Click Ok.

The worksheet is refreshed to show or hide the summary line.



Freezing the Summary Line:

If there are many rows in a table, it is useful to display the worksheet summary as the top row and then freeze that row so that it remains in position as you scroll down the page. Only the overall summary of a worksheet is freezable. The overall summary corresponds to the outermost summarized level. Only summary rows on the top row or left-most row can be frozen. Otherwise, the menu option is disabled.

1. From the Collaborator Workbench, open the worksheet whose summary line you want to configure.
2. From the View menu, choose Freeze Overall Summary.

To set the Summary Line position:

1. 1. From the Collaborator Workbench, open the worksheet whose summary line you want to configure.
2. 2. From the Worksheet menu, choose Layout Designer.
The Worksheet Designer appears.
3. Click the Layout button.
4. Do one of the following:
 1. To set the summary row to the right or left of the worksheet, right-click on a series in the horizontal series list and then choose Summary Position> Left or Right.
 2. To set the summary row to the top or bottom of the worksheet, right-click on a series in the vertical series list and then choose Summary Position> Top or Bottom.

The worksheet refreshes to display the summary position in its new location.

Other Basic Series Options

This section discusses other basic options you can use when configuring series.

Drop-down Lists

A series can be configured as a drop-down list, which means that when the series is displayed in a worksheet, each series cell includes the same drop-down list of choices. When a user includes the series in a worksheet, he or she can set the value of a series element by selecting from the list, if the series is editable.

Typically each list element is a text string that has a corresponding numeric value. When the user chooses a list element, Demantra finds the corresponding numeric value and sets the series value equal to that.

To configure the drop-down list for a series, you can use any of the following:

- A table that exists in the Demantra database.
- A level.
- A list that you enter directly in the Business Modeler for use by this series.

All three variations behave in the same way.

Scaling

If the series is numeric, it can be configured as scaled. At any time, a given worksheet uses a single scaling factor. The user chooses this factor and Demantra automatically divides all numbers in the worksheet by that factor, except for any series that are marked as "unscaled".

Caching by Item

A series can be cached (aggregated by item and cached in the `branch_data_items` table). This improves performance of worksheets that are aggregated across locations and that do not have any location or matrix filtering.

Advanced Series Options

On occasion, you may need to consider the more advanced options for series.

Preserving Promotional Data While Moving or Copying

When you copy and paste a promotional level, Demantra copies data for the promotional series, as well. The span of time of the new copy might not be the same as the span of time of the original, so the definition of each series needs to specify how to perform the computation. Similarly, when a user changes the length of a promotion, Demantra adjusts the associated promotional series data.

There are two preservation types:

- Copy/Paste preservation type
- Move preservation type

The settings should be consistent with the rest of the settings for the series. The following guidelines are suggested:

Option	Meaning	Suggested Series Type	Suggested Aggregation Function
As Is Preservation	Demantra shifts the data to the new dates but makes no other changes. If the new date range is longer than the original date range, Demantra uses nulls for the dates at the end. If the new date range is shorter than the original date range, Demantra omits the "extra" dates.	Any	Any
Do Nothing	Demantra ignores the series during the copy/paste and move operations. Use this option to retain values of any price series of a promotion when using the Refresh Population method and workflow.	Any, suggested	Any
Most Common	Demantra ensures that the pasted data closely resembles the source data. Use this setting for any kind of series; the other settings apply only to numeric series.	Any, but not usually appropriate for proportional numeric series	Any function other than Sum
None	Demantra does not copy the data for this series.	Any	Any

Option	Meaning	Suggested Series Type	Suggested Aggregation Function
Percentage Preservation	Demetra first aggregates the data according to the Aggregation Function of the series. It then ensures that the pasted data generally has the same level, over time, as the source data.	Numeric; not proportional	Any function other than Sum
Volume Preservation	Demetra first aggregates the data according to the Aggregation Function of the series. It then ensures that in the pasted data, the overall volume is the same (area under the curve) as the volume of the source data. Choose Volume Preservation for the Copy/Paste Preservation option when you use a Refresh Population method and workflow to realign promotion data after an item is moved from one promotion group to another. This option works in conjunction with the CopyPasteIntersect system parameter.	Numeric; proportional	Sum

For most series, you will want to use the same setting for both options. However, for some series, it does not make sense to copy the data when you create a new promotion (so you would use the setting None for copy/paste), although it does make sense to preserve the data if you are just moving a promotion. In such cases, it is useful to have two separate options.

The following figure shows examples of series that are configured with each of these preservation types:

Promotion	Time	Preserve As Is	Preserve Vol	Preserve Percent	Preserve None	Preserve Most Common
Promo 1	01/02/2006	100	90	80	70	a
	01/09/2006	100	90	80	70	a
	01/16/2006	100	90	80	70	b
	01/23/2006	100	90	80	70	a
	01/30/2006	100	90	80	70	a
	Summary	500	450	80	350	
Copy (1) of Promo 1	01/02/2006	100	26	80		a
	01/09/2006	100	26	80		a
	01/16/2006	100	26	80		a
	01/23/2006	100	26	80		a
	01/30/2006	100	26	80		a
	02/06/2006		26	80		a
	02/13/2006		26	80		a
	02/20/2006		26	80		b
	02/27/2006		26	80		b
	03/06/2006		26	80		b
	03/13/2006		26	80		a
	03/20/2006		26	80		a
	03/27/2006		26	80		a
	04/03/2006		26	80		a
	04/10/2006		26	80		a
	04/17/2006		26	80		a
	04/24/2006		26	80		a
	Summary	500	450	80		
Summary		1,000	900	80	350	

This worksheet table shows two promotions, Promo 1 and a copy which spans more time. Notice the following in the copy:

- The Preserve As Is series contains the same numbers for the first five time buckets, which the length of the original promotion. After that, this series is null in the promotion copy.
- For the Preserve Vol series, the level of this series is lower in the copy so that there is the same overall volume as in the original.
- For Preserve Percent, the pasted data is at the same level as the original, and is extended for the length of the pasted promotion.
- For Preserve None, there is no pasted data.
- For Preserve Most Common, the pasted data mimics the original data in overall pattern.

See Also

- "System Parameters" for more information about the CopyPasteIntersect system parameter that works in conjunction with the Copy/Paste Preservation series configuration.

- "Specifying Data Properties of a Series Table" for additional information about configuring series.

The Aggregated Base Level Option

This option lets you specify how this series is aggregated in a worksheet that includes a promotion level:

- If you choose sales_data, this series is aggregated by the items, locations, and dates selected in the worksheet. Most series are aggregated this way in a typical implementation.
- If you choose promotion, this series is aggregated by the items, locations, dates, and promotions selected in the worksheet. That is, when the series is aggregated, any data that is not associated with a promotion is ignored.

Within a worksheet that does not include a promotion, the series is aggregated based on the series setting; that is, it is aggregated by the items, locations, and dates if it aggregates by sales_data only, and additionally by promotions if aggregated by promotion.

The following shows two series that are defined almost identically. The Orders series is aggregated by sales_data and the Orders for Promotions series is aggregated by promotion.

Private Label Brand - BJ				
Promotion	Time	Orders	Orders for Promotions	Promo0 ▾
3. Q2 2002 B.	07/08/2002	442,956	233,135	100
	07/15/2002	589,276	310,145	100
	07/22/2002	665,507	350,267	100
	07/29/2002	572,546	301,340	100
	08/05/2002	455,318	239,641	100
	08/12/2002	829,641	436,653	100
	Summary	3,555,244	1,871,181	600

This worksheet is aggregated to the Brand, Account, and Promotion levels. The worksheet is filtered to show only the Private Label brand and two specific BJ's locations (these locations are children of the Account level):

- BJ 0005 ran a promotion on all Private Label products during the time span of the worksheet.
- BJ 0003 did not run any promotion.

Notice that the values are greater for Orders than for Orders for Promotions. This is because only one of the locations ran the promotion.

Extra From and Extra Where

Normally the server expression can refer only to fields in the following tables:

For sales and matrix series	branch_data and mdp_matrix tables. Note that branch_data is a synonym for the sales_data table or the promotion_data table.
For promotion series	branch_data table.
For level series	Table associated with the level.

In rare cases, you may need to refer to data in other tables. In such a case, use the Extra From field. In this field, specify an optional list of additional tables (separated by commas) that contain data relevant to this series.

If you include a table here, the server expression can refer to columns in that table.

Note: Internally, these tables are added to the From clause in the SQL query that retrieves data for this series.

If you need to filter the data further, use the Extra Where field. The syntax of this field is as follows:

table.column operator other_table.other_column

Here *operator* is a comparison operator, one of the following

- :=
- <>
- >
- >=
- <
- <=

and *table.column* and *other_table.other_column* are key columns in the database.

Note: Internally, the Extra Where field is added to the WHERE clause in the SQL query that retrieves data for this series.

Note and Promotion Indicators

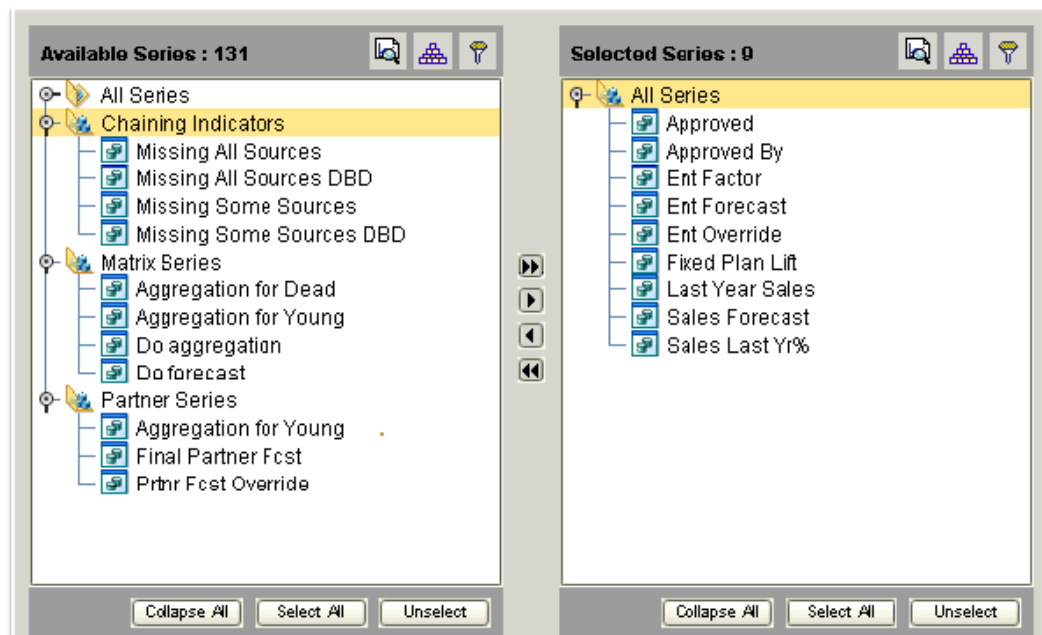
Within a worksheet, a user can attach a promotion (in the case of Promotion Effectiveness) or a note to a given item-location combination, at a given date. Depending on how the series was configured, the series will be displayed with an indicator in all worksheet cells that correspond to that item-location combination and date.

You control these indicators when you define components, within the Business Modeler

Note: If your solution uses other types of general levels, you can associate an indicator for any general level that does not have child levels.

Series Groups

You can define optional groups of series, in order to make the lists of series more manageable, especially in cases where there are a large number of series. For example, the Worksheet Designer includes a screen like the following.



Well-defined series groups can make it easier to place related series on a worksheet.

A series can belong to any number of groups.

You define, modify, and delete series groups in the Business Modeler. The series groups are visible in the Worksheet Designer, within in the Web-based products (Demand Planner Web, Promotion Effectiveness, and Settlement Management).

Note: Series groups are not visible in the desktop products (Demand Planner and Demand Replenisher).

Configuration Notes

This section contains configuration notes related to series.

Dependencies

Before you can configure series, you will need to load some sample data for items, locations, and sales (and promotions, if you want to create promotion series).

Before creating a dropdown-type series, you must consider where to look up the dropdown choices. You may need to create the table for the lookup data and then load that data (as indicated in "Loading Supplementary Data"). Or you can use an existing level or you can enter the choices directly in the Business Modeler.

Series can be added fairly easily throughout implementation and later as needed.

Tools

Demantra provides the following tools for creating and configuring series:

Tool*	Purpose/Notes
Data Model Wizard	Can define series, although this wizard provides only a small subset of the series options.
Configure > Series option	Defines series.
Configure > Series Group option	Defines series groups.
Components > Open/Create Component option	Creates components. Among other things, a component defines the associations between series and indicators.
*These options are in the Business Modeler.	

Units, Indexes, and Exchange Rates

This chapter describes units and related concepts, outlines the primary configuration options, and summarizes the available tools.

This chapter covers the following topics:

- Introduction
- Unit Conversion Data
- How Units Are Used
- Time Units
- Setting and Modifying the Base Time Unit
- Unit-Level Association
- Indexes and Exchange Rates
- Configuration Notes

Introduction

At any time, a worksheet uses one unit of measure, which is used by most of the series in that worksheet. The user can switch to another unit of measure; any series that uses a unit of measure is correspondingly multiplied by the appropriate conversion factors. For example, a worksheet can express sales and forecast in units or in cases or dollar value.

Note: You do not need to use units in this way. You can instead hard code the unit into a series definition, so that it always expresses, for example, the buyback per case. Whatever your choice is, be sure to give useful names and hints to the series.

Similarly, at any time, a worksheet can use one index or exchange rate, which is used by any series that express financial quantities. The user can switch to a different index (such as CPI) or exchange rate, and the worksheet automatically multiplies those series

by the index or exchange rate.

Unit Conversion Data

The imported data contains the item quantity per sales record, expressed as the number of units sold. Note that you can rename units.

The imported data also includes the unit price, which depends on the item, location, and date. You use the item price as a conversion unit, to represent monetary values.

Typically, you define additional units of measure, of two general kinds:

- Size units, which measure the size of a sale: cases, truckloads, and so on. When you define these units, you provide a conversion factor by which the base item quantity is automatically multiplied. This conversion factor does not have to be the same for all items.
- Monetary units, which measure the value of a sale. When you define these units, you provide a conversion factor (the imported unit price), which depends on the item, location, and date. You can also specify time-dependent indexes and exchange rates that can be applied to monetary units within a worksheet.

Size Units

When you define a size unit, you specify the following:

- A name, used on the vertical axis of worksheet graphs.
- The table and data field that contains the associated conversion factor, which is generally different for different products.

The unit conversion factors must be supplied in the imported data. For example, the `t_ep_sku` table might include a column that indicates the number of cases per unit, as follows:

SKU	...	Cases	Pallets	...
109784	...	0.01	0.001	...
109785	...	0.015	0.0015	...
109786	...	0.005	0.0005	...
...

This means that the SKU 109784 has 0.01 cases per unit, or inversely, 100 units per case.

When you define the Case unit, you would specify the Cases column of t_ep_sku as the source of the conversion factor for this unit.

Monetary Unit

When you define a monetary unit, you specify the following:

- A name, used on the vertical axis of worksheet graphs.
- The table and data field that contain the price per unit.
- An optional expression for the conversion factor, if the factor cannot be simply read from the table.
- Optional time-dependent exchange rates and indexes that can be applied to this unit.

How Units Are Used

The token #UNIT# represents the unit conversion factor. You can include this token within the server expression for a series, which should have the following general form:

quantity * #UNIT#

Within a worksheet, this token is automatically replaced by the conversion factor that corresponds to the unit that the worksheet is currently using. For example, if the Demand was 1200 units, and if the worksheet is using cases instead, then Demand is displayed as 12 cases.

To configure a series to use units, do either of the following:

- Create a server expression with the form shown previously.
- Create a client expression that refers to another series that uses units.

Time Units

Any Demantra solution has a base time unit, such as weeks or months. Demantra provides some larger predefined time units, and you can add others. In general, there are two types of time units:

- Simple time units (such as quarters) are simple multiples of the base time unit. For these, you just provide a scaling factor. For example, for a weekly system, a quarter consists of 13 time units. These time units are assumed to divide evenly into one year, and Demantra automatically figures out which base time bucket each date belongs to.
- Data-dependent time units, such as 4-4-5 time units, require explicit data. That is,

they must be assigned explicitly to each date in the system, in the Inputs table.

Note that by default, in any worksheet, the date and label for a given bucket is the first date in that bucket. Within a worksheet, another date format can be used.

Data-Dependent Time Units

The following example represents rows in the Inputs table. It shows a set of dates from a weekly system and shows how those dates are mapped into quarters and 4-4-5 periods. (A 4-4-5 time system creates quarters that consist of a four-week "month," followed by another four-week "month," and then followed by a five-week "month." In practice, 4-4-5 calendars vary from company to company.) The second and third columns show the bucket numbers associated with each date, depending on the date system.

Date	Bucket number when quarters are used	Bucket number when 4-4-5 periods are used
1/3/05	100	122
1/10/05	100	122
1/17/05	100	122
1/24/05	100	122
1/31/05	100	123
2/7/05	100	123
2/14/05	100	123
2/21/05	100	123
2/28/05	100	124
3/7/05	100	124
3/14/05	100	124
3/21/05	100	124
3/28/05	100	124

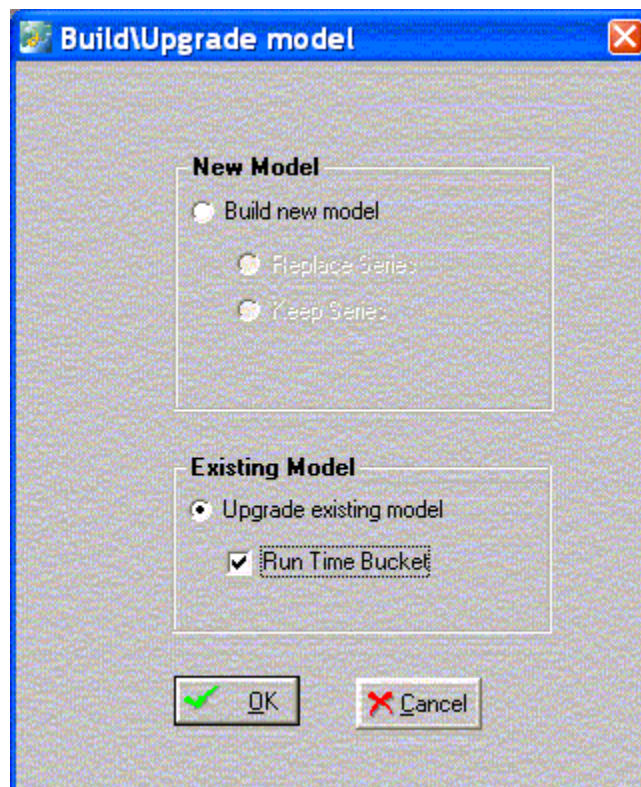
Date	Bucket number when quarters are used	Bucket number when 4-4-5 periods are used
4/7/05	101	125

The first thirteen dates belong to a single quarter, and the last date belongs to the following quarter. The first four dates belong to the first 4-4-5 "month" and so on.

Setting and Modifying the Base Time Unit

The base time unit is used by the Data Model to aggregate the source data to the specified time bucket size. Allowed settings of the base time unit (time bucket size) are:

- day
- week
- Gregorian month



Impacts of Changing the Base Time Unit:

If the time bucket is re-configured, the time aggregation set for all worksheets is modified to match the new time aggregation. A review of all worksheets is strongly recommended. See Worksheets, page 8-1.

After making changes, the Data Model should be upgraded, *not Rebuilt*, with the Run Time Bucket option checked. See Building the Data Model, page 25-22 and Manipulating Existing Data Models, page 25-24.

The erased member and fact data in Demantra must then be downloaded again. See Loading the Data into the Data Model, page 25-24.

Integration profiles are required to be redefined by the user, if the unit of time specified therein becomes invalid. For details, see Loading Series and Promotions, page 13-6.

Common changes to the Base Time Unit:

The Business Modeler allows the Demantra System Administrator to change the base time unit at any time after initial installation. Common changes include:

- Setting the start date of the weekly time bucket from Monday to Sunday
- Changing the base time unit to month or day from week

To set or change the Base Time Unit:

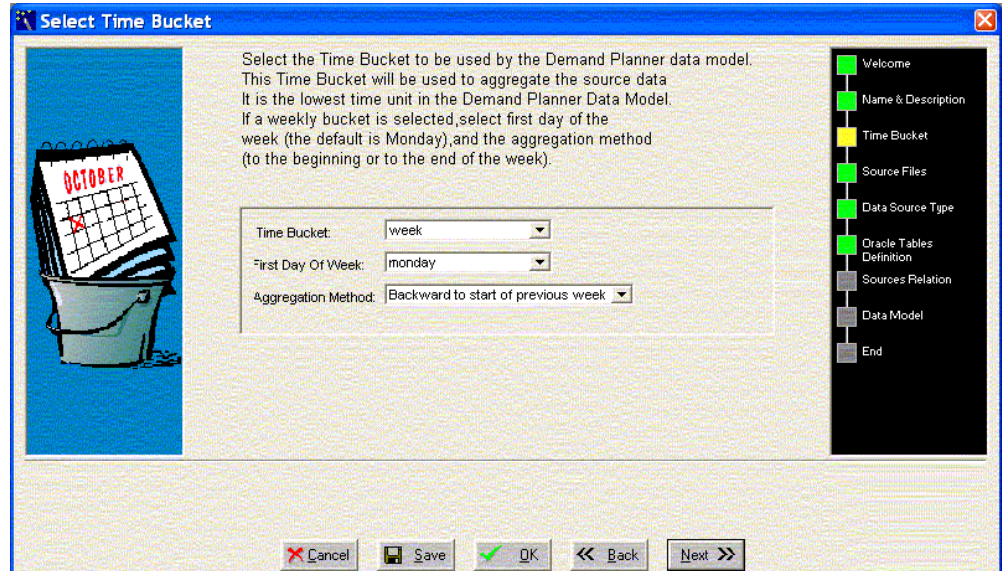
Prerequisite

Install the Business Modeler.

1. Navigate to the data model.

Business Modeler > Data Model > Open Data Model

The Select Time Bucket window appears.



2. Select the Base Time Unit from the time bucket list of values. The default value is: week
3. If the Time Bucket field is set to week, then choose the day that represents the starting day of the week from the First Day Of Week list of values. The default value is Monday. User may change the default but must then rebuild the model.
4. If the Time Bucket field is set to week, then select the Aggregation Method from the list of values to determine whether events that occur mid week are aggregated to the start date or the end date of the weekly time bucket.
5. Save your work.
6. Click OK.

Unit-Level Association

In Demantra, you associate each unit with the levels where it makes sense to use that unit. For example, a relatively small unit might make sense only at lower levels.

Demantra uses this association within worksheets. If a worksheet contains three levels, for example, then only the units associated with those levels can be used in that worksheet.

Indexes and Exchange Rates

Monetary units of measure can use financial indexes and exchange rates. This means that when users display data in a worksheet, they can apply any of those associated

indexes or exchange rates.

Each index and exchange rate is stored in a different table, except for the placeholder index (constant, equals one for all dates).

The placeholder index is used to switch a worksheet back to the same monetary units that are used in the imported data. By default this is called **dollar \$**, because monetary values are usually imported in dollars.

Configuration Notes

This section contains configuration notes related to units, indexes, and exchange rates.

Dependencies

Before you can configure units, you will need to load some sample data for items, including unit conversion data.

If a unit requires an index or exchange rate, you must configure that index or exchange rate first.

Tools

Demantra provides the following tools for creating and configuring units:

Tool	Purpose/Notes
Data Model Wizard*	Can define units, although this wizard provides only a subset of the options.
Configure > Display Units option*	Defines units.
Data Model > Global Factors option*	Allows you to add columns and values to the Inputs table.
Configure > Configure Units for Levels option*	Allows you to associate units with levels.
*These options are in the Business Modeler.	

Worksheets

This chapter describes worksheets, outlines the primary configuration options, and summarizes the available tools.

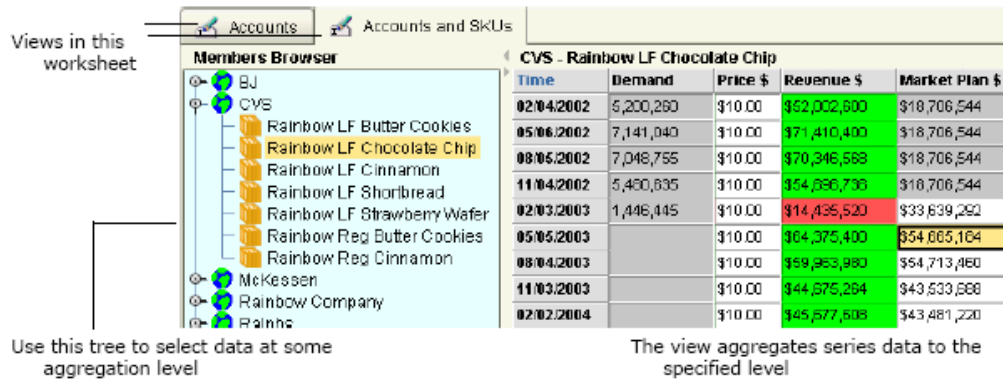
This chapter covers the following topics:

- Introduction
- Main Worksheet Options
- Elements of a Worksheet View
- Level Layout in a View
- Filtering per View
- Level and Worksheet Association: Embedded Worksheets
- Worksheet and Layout Ownership
- Worksheet Caching
- Configuration Notes

Introduction

Within Demantra, users work almost entirely within worksheets. A worksheet is a customized working environment where users can view and edit data. When users save changes back to the database, they become available to other users and to downstream operations.

A worksheet consists of one or more views, usually displayed as tabs within the worksheet. Each view retrieves a set of data that is aggregated in a specific way and that may also be filtered. The following shows an example:



Views in this worksheet

Use this tree to select data at some aggregation level

The view aggregates series data to the specified level

You can also display views as child windows of the worksheet.

Main Worksheet Options

This section provides a quick overview of the main worksheet options:

- Levels in the worksheet
- Series in the worksheet
- Time resolution and time span
- Optional filters
- Optional exception filters
- View definition and layout

Levels in a Worksheet

A worksheet usually includes aggregation levels. Based on the levels included in a worksheet, Demantra automatically determines which item-location combinations the worksheet should include. Depending on which combination you select, the worksheet displays series data associated with that combination. For example, if you select one location level (city) and one item level (SKU), the worksheet will contain series data associated with each city-SKU combination. On the other hand, if you select one location level (city) and you do not specify an item level, the worksheet aggregates data for all items. That is, the worksheet will contain series data associated with each city, aggregated for all products.

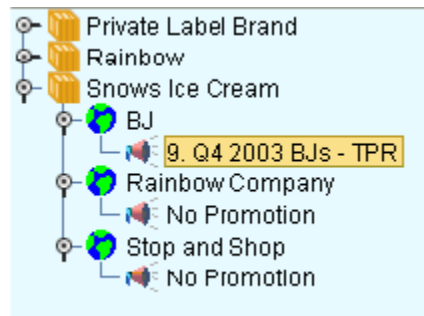
- If you do not specify any aggregation levels in a worksheet, the data is completely aggregated across all selected items and locations.
- If you use a settlement level in a worksheet, you cannot use levels from any other hierarchy in that worksheet.

Advanced Selection Options

By default, if a worksheet includes a promotion level, the worksheet includes all the following types of combinations:

- Combinations that have both sales data and promotions
- Combinations that have sales data, but no promotions
- Combinations that have promotions, but no sales data

The worksheet displays placeholders for combinations that do not have promotions. For example:



You can exclude some of these combinations. For example, you might want the worksheet to include only the combinations that have both sales and promotions, as follows:



Series in a Worksheet

Every worksheet must include at least one series. You can display series in the worksheet table, the graph, both, or neither. (It can be useful to add series to a

worksheet but leave them undisplayed, so that the series are available for any client expressions that depend on them.)

Note: If you use a settlement level in a worksheet, all series in the worksheet must refer to tables used by the settlement hierarchy.

Time Criteria

Each worksheet selects data for a specified span of time and optionally aggregates it in time using a time unit.

You can specify the span of time as a fixed range of dates, a time range relative to today, or a time range relative to the last sales date in the loaded data.

To aggregate data in time, you can also include a time aggregation in the worksheet.

Filters

Within Demantra, you generally apply filters by specifying a level and the members of that level to include. For example, the following filter includes only the Rainbow brand.

Available Members		Selected Members	
Code	Description	Code	Description
Default	Default Brand		
12	Private Label		
99	Demand Profiles		
VHS	Snows Ice Cream		
		11	Rainbow

level members not included in the filter

level members included in the filter
(and therefore displayed in the worksheet)

You can apply multiple filters at the same time. For example, for the preceding worksheet, you could also filter by account.

In contrast to an exception filter ("Exception Filters"), this type of filter is static and behaves the same no matter how the data changes.

Exception Filters

See Applying Exception Filters in *Oracle Demantra User's Guide*.

View Layout

A worksheet contains one or more views, which the user can display either as tabs or as sub windows. For each view, you specify the following options:

- Name of the view
- Elements to include in the view
- Layout of levels and series in the view
- Additional filtering of the view
- Sub tab worksheets in the view

The following sections provide more details on view layout.

Elements of a Worksheet View

For each worksheet view, you can control which of the following elements are included in that view:

- The Members Browser or combination-selection lists. A worksheet view usually includes either a Members Browser or a set of drop down menus, with which the user chooses the data to display in the rest of the worksheet:
- The worksheet table, which shows series data for the item-location combination that is currently selected in the view. Depending on how the layout is configured, this may appear as an ordinary table or it may appear as a cross tab; see "Level Layout in a View".

By default, each row in the table corresponds to a point in time, and each column displays the data for a series. As noted earlier, the table also has a summary row. If the worksheet is in cross tab layout, the table also includes subtotal rows.
- The graph, which displays data for the current selection. By default, the horizontal axis shows time, and the vertical axis shows one or more series.
- The Notes/Attachments sub tab, which displays notes and attachments related to the selected combination.
- The Activity Details sub tab, which displays promotions and the promotion hierarchy. The Activity Browser displays an expandable tree view of the promotions associated with the currently selected combination. The Gantt chart displays the promotions associated with the currently selected combination.
- Sub tabs that contain related worksheets. When a selection is made in the worksheet, the related worksheet shows further details. This related worksheet potentially includes different series than the rest of the worksheet and may also be filtered further.

Level Layout in a View

When you include levels in a worksheet, that means you can see data associated with each member of those level. In each view of a worksheet, you can use any of those levels in any of the following ways:

- Use it within the Members Browser or combination-selection lists, as in the previous examples.
- Use it on one of the worksheet axes, creating a crosstab layout. Each worksheet view has an x-axis and a y-axis.
 - In the graph, the x-axis is shown horizontally and the y-axis is shown vertically.
 - In the table, the x-axis is displayed vertically and the y-axis is displayed horizontally. (This way, the x-axis displays the same data in the table and in the graph.)
- Hide it, causing Demantra to aggregate data to that level.

Crosstab Layouts

In a crosstab layout, you include a level on an axis. The table (also known as a pivot table) provides a cross tabulation of all the members.

The following figure shows a worksheet table in crosstab layout, with a row for each SKU member within each time bucket:

Time	SKU	Demand	Price \$	Revenue \$	Market Plan \$
02/04/2002	Rainbow LF Butter Cookies	155,250	\$10.00	\$1,552,500	\$231,660
	Rainbow LF Chocolate Chip	5,200,260	\$10.00	\$52,002,600	\$18,706,544
	Rainbow LF Shortbread	67,000	\$10.00	\$670,000	\$243,243
	Rainbow LF Strawberry Wafer	265,000	\$10.00	\$2,650,000	\$162,162
	Rainbow Reg Butter Cookies	69,950	\$10.00	\$699,500	\$427,680
	Summary	5,757,460	\$7.14	\$57,574,600	\$19,771,290
05/06/2002	Rainbow LF Butter Cookies	133,934	\$10.00	\$1,339,340	\$231,660
	Rainbow LF Chocolate Chip	7,141,040	\$10.00	\$71,410,400	\$18,706,544
	Rainbow LF Shortbread	46,717	\$10.00	\$467,170	\$243,243
	Rainbow LF Strawberry Wafer	330,000	\$10.00	\$3,300,000	\$162,162
	Rainbow Reg Butter Cookies	23,233	\$10.00	\$232,330	\$427,680
	Summary	7,674,924	\$7.14	\$76,749,240	\$19,771,290

Notice that the Members Browser does not include the SKU level, because all SKUs are displayed at the same time.

For another example, the worksheet could instead display the SKU members across the top of the table rather than down the side, as in the following example:

Accounts Accounts and SKUs Accounts and SKUs crosstab

Members Browser

CVS

BJ

CVS

McKesson

Rainbow Company

Ralphs

Stop and Shop

WalMart

SKU	Rainbow LF Butter Cookies				Rainbow LF Chocolate Chip			
Time	Demand	Price \$	Revenue \$	Market Plan \$	Demand	Price \$	Revenue \$	Market Plan \$
02/04/2002	155,250	\$10.00	\$1,552,500	\$231,660	5,200,260	\$10.00	\$52,002,600	\$18,760,000
05/06/2002	133,934	\$10.00	\$1,339,340	\$231,660	7,141,040	\$10.00	\$71,410,400	\$18,760,000
08/05/2002	169,484	\$10.00	\$1,679,586	\$231,660	7,048,755	\$10.00	\$70,348,568	\$18,760,000
11/04/2002	156,500	\$10.00	\$1,549,350	\$231,660	5,480,635	\$10.00	\$54,696,736	\$18,760,000
02/03/2003	75,400	\$10.00	\$746,460	\$1,310,872	1,446,445	\$10.00	\$14,435,520	\$33,600,000
05/05/2003		\$10.00	\$1,718,029	\$1,900,434		\$10.00	\$64,375,400	\$54,660,000
08/04/2003		\$10.00	\$1,470,648	\$1,689,744		\$10.00	\$59,963,960	\$54,660,000
11/03/2003		\$10.00	\$1,173,626	\$1,515,606		\$10.00	\$44,675,264	\$43,500,000
02/02/2004		\$10.00	\$2,030,132	\$2,261,640		\$10.00	\$45,677,608	\$43,440,000
Summary	690,568	\$9.00	\$13,259,671	\$9,624,935	26,317,136	\$9.00	\$477,584,064	\$304,000,000

Other variations are possible.

Hidden Levels

Hidden Levels allow users to exclude worksheet selected levels from specific worksheet views. This allows flexibility of what individual worksheet tabs can show without having to use embedded worksheet functionality. When used, one or more levels are removed from a worksheet view this causes data to aggregate across the level. In order to support this aggregation the worksheet data will not be editable

To hide a level:

- Access Worksheet Designer -> Layout.
- Navigate to the tab where the level should be hidden
- Right click on any level
- Navigate to Hide level and click on the desired level

To show a hidden level:

- Access Worksheet Designer -> Layout.
- Navigate to the tab where the level should be hidden
- Right click on any level
- Navigate to Show level and click on the desired level

Editing Data

- Data in a view with hidden level will not be editable

Advanced Analytics

- Levels appearing in one or more view will be shown in the advanced analytics screen. Levels hidden in all views will not be shown

Information Retrieval

- Notes and other information being retrieved or updated will include all filters on the worksheet and view. This includes filters applied to level which are subsequently hidden.
- The ability to filter data based on level members is not affected by hiding a level.

Open With

- Levels not being displayed in the view being used to "open with" will not part of context used to the filter opened worksheet.

Filtering per View

In some cases, you create multiple views so that you can show different series in each view. In other cases, you might need to show different combinations in each view. You can separately filter each worksheet view. In this case, you filter a view by choosing a subset of the members of the levels included in the worksheet.

DSM uses this feature to segregate settlements with different statuses. Settlements of each status are on a different worksheet tab.

Level and Worksheet Association: Embedded Worksheets

It is useful to be able to examine a level member more closely, to launch a worksheet from that member that is filtered to show only that member. But typically, a Demantra application includes a large number of worksheets, and most of those worksheets would not be useful in this way. So Demantra provides an option for associating each level with any number of worksheets. Demantra uses this association in two ways:

- A user can start from a level member and launch a worksheet that is filtered to that member. To do so, the user right-clicks the member and clicks the Open or Open With option.

Alternatively, this worksheet can show just the combination from which the user started.

The worksheet appears in a new window.

Note: Demantra indicates the filtering as follows:

- If the worksheet is filtered by member, the name of the worksheet is preceded by the name of the member by which you are filtering it.
- If the worksheet is filtered by combination (full context), the name of the worksheet is preceded by the word "Filtered".

- A worksheet can include an embedded worksheet that is associated with any of the levels in the main worksheet. Then when a user selects a member in the main worksheet, the embedded worksheet shows the details. The embedded worksheet is displayed in a sub tab.

If you open a worksheet as an embedded worksheet:

- It retrieves filters based on the parent worksheet selection.
- The update does not filter unless you explicitly include the level in the embedded worksheet. Oracle Demantra treats the embedded worksheet as if you opened it independently; it filters the population to update only by the worksheet filters itself, and doesn't consider the parent worksheet filters

If you open a worksheet as an embedded worksheet using Open With, Oracle Demantra retrieves and updates using both the worksheet filters and the parent worksheet filters.

Worksheet and Layout Ownership

In general, any worksheet is available as follows:

- A private worksheet is available only to the user who created it.
- A public worksheet is available to all users but can be changed only by the user who created it.

In any case, Demantra automatically prevents any user from seeing data for which he or she does not have permissions.

Worksheet Definition, Layout, and Local Adjustments

As users work with a Demantra worksheet, they often sort columns, hide or display features, and make various other changes. It is useful to understand how these settings are saved.

Base Demantra configuration	<ul style="list-style-type: none"> • Display format for each series
These settings affect all users and all worksheets.	<ul style="list-style-type: none"> • Initial display width of series and levels • Colors and graph style for each series • Other display colors (generally dependent on a condition)

Worksheet definition

These settings are saved through the File > Save Worksheet menu option. Only the worksheet owner can make these changes.

- Initial number of views within the worksheet and their initial names
- Initial elements (Members Browser, table, graph, and so on) in worksheet view
- View synchronization setting
- Aggregation levels used in worksheet and initial level layout; advanced selection options
- Series used in worksheet and initial series layout
- Time aggregation; time span; time formatting
- Filtering and exception filtering
- Unit of measure used in worksheet; overall scale used in worksheet, if any; index or exchange rate, if any

Layout changes

These settings are saved separately for each user if the user clicks File > Save Worksheet. Any user can save these changes, not just the worksheet owner.

- Additional views in the worksheet
 - New names of worksheet views
 - Level layout: order of levels; placement on axes in each view; whether level is hidden in each view
 - Series layout: order of series; where each series is displayed (table, graph, both)
 - Hide/show time axis
-

Local adjustments	<ul style="list-style-type: none"> • Use of windows or tabs for views within a worksheet • Size and position of the Members Browser, table, graph, and so on in each view • Sorting in the worksheet table • Graph type; legend; grid lines • Hide/show empty rows setting • Activity synchronization setting (Options menu)
These settings are saved automatically separately for each worksheet and each user.	
Not saved	<ul style="list-style-type: none"> • Changes to column widths in the worksheet table • Initial view focus; focus in each worksheet view • Expansion state in the Members Browser and Activity Browser • Focus and scroll in all areas • Zoom setting in Gantt

The auto run option (Options menu) is saved separately for each user, but applies to all worksheets that the user sees.

Worksheet Caching

This describes:

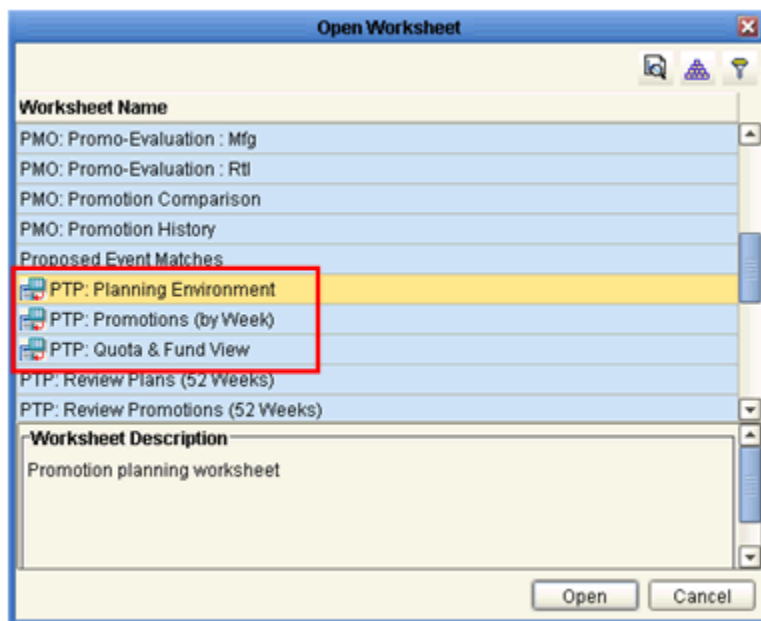
- Enabling Worksheet Caching
- Creating a Workflow to Build Worksheet Caches Automatically

Enabling Worksheet Caching

Perform the following to enable worksheet caching:

1. Verify that the System Parameter **EnableWorkSheetCaching** is set to **True** (**Business Modeler > Parameters > System Parameters**).
2. For each Worksheet that you want to cache, open Worksheet Designer (**Display** section) and then select **Cache Worksheet Data**.
3. Set **Refresh Type** to either **Manual** or **Automatic**:
 - Manual: Users must manually refresh the cache.
 - Automatic: Changes to source data will be detected when the worksheet is opened and the cache will automatically be refreshed as needed

The Open Worksheet dialog displays an icon next to all worksheets that are cached.



Default Worksheet Caching Behavior

By default, all users can create cached Worksheets. However, a system administrator can revoke this privilege for specific users. To do this, the administrator must modify the `CAN_CREATE_CW` column on the `USER_ID` table (this table contains one row for each User). This setting is not visible in Business Modeler; therefore it must be updated directly in the database.

Set the `CAN_CREATE_CW` column to:

- 1 to indicate that the user can create cached Worksheets (this is the default)
- 0 (or null) to indicate that the user cannot create cached Worksheets.

See also [Creating a Workflow to Build Worksheet Caches Automatically](#).

Creating a Workflow to Build Worksheet Caches Automatically

You can create a workflow that automatically creates caches for worksheets that can be opened via the Open With method from another worksheet. This workflow creates worksheet caches in a batch process thereby eliminating the need to build the cache the first time the user opens the worksheets.

Note: This procedure supports building caches only for worksheets that are opened using Open With and where the Open With Context for the worksheet is set to Selected Member.

Create the workflow as follows:

1. **Stored Procedure Step:** This step runs the APPPROC_REBUILD_CONTEXT_CACHES procedure, passes the ID of the worksheet, the Open With Level, the User Group to be cached, and Run Mode equal to 1. For details about this stored procedure, see APPPROC_REBUILD_CONTEXT_CACHES, page 35-3.
2. **Worksheet Cache Step:** Specify the following settings:
 - Worksheet Name: The Worksheet to be cached (must match Worksheet ID from Step 1).
 - User/Group: The User Group to cache for the selected Worksheet (must match User Group Id from Step 1).
 - Cache Type: Open With Context.

For details about this step, see Worksheet Cache Step, page 39-41.

3. **Stored Procedure Step:** This step runs the APPPROC_REBUILD_CONTEXT_CACHES procedure passing the same values as Step 1 for the ID of the worksheet, the Open With Level, and the User Group to be cached, but with a Run Mode equal to 2.

The Worksheet, Open With Level, and User Group must be the same for each of the steps listed above. Only one Worksheet/Open With Level/User Group combination can be processed in each group of these steps. Multiple combinations can be processed by repeating these steps in a single workflow or by creating separate workflows.

Configuration Notes

This section contains configuration notes related to worksheets.

Dependencies

Before you can create worksheets, you will need to load some sample data, create any needed levels, and create any needed series.

Design Considerations

- It is common practice to create a master worksheet, which is public and meant for multiple users. Different users typically have permission to see different subsets of the worksheet data, such as different accounts. In addition, users can launch the worksheet from a level member, to further filter the worksheet results.
- For performance reasons, don't select too much data to view, unless there is no other choice.
- If you receive a message saying "out of memory," try the following techniques to reduce the amount of memory that your worksheet selects:
 - Remove series if possible
 - Reduce the span of time
 - Apply filters
- If you do need to select a large amount of data, use the levels to your advantage. Specifically, use the levels in the Members Browser or selector lists rather than moving them to a worksheet axis. If levels are in the Members Browser or selector lists, each combination in the worksheet is relatively smaller and will load more quickly.
- Remember that you can filter the worksheet by any level, including levels that are not shown in the worksheet. For example, you might want to see data at the region level, but exclude any data that does not apply to the Acme territory. To do this, you would filter the worksheet to include only the Acme member of the Territory level, but you would select data at the Region level.
- A multi-view worksheet is useful in following cases:
 - If you need to edit data at one aggregation level and see easily how that affects higher aggregation levels.
 - If you need to display a large number of series without having to scroll to see each one.
- To make sure that all client expressions in a worksheet are always evaluated correctly, make sure that the worksheet includes all series to which those client expressions refer. (Note that you can add series to a worksheet but leave them undisplayed.)

Known issue regarding evaluating client side exceptions

For series that contain both client and server expressions, filtering of exceptions occurs first on the server expression, causing it to take precedence over the client expression. Therefore, if the critical value is coming from the client side, you must configure the exception series with '0' as the server expression (thereby removing it).

Tools

Demantra provides the following tools for configuring worksheets and related objects:

Tool	Purpose/Notes	See
Worksheet wizard in the Web client (Demand Planner Web, Promotion Effectiveness, and Settlement Management)	Define worksheets and Collaborator Workbench content panes.	Oracle Demantra Demand Management User's Guide or other user guide
Content wizard in Collaborator Workbench	Define worksheets and Collaborator Workbench content panes.	Oracle Demantra Demand Management User's Guide or other user guide
Components > Open/Create Component option in the Business Modeler	Creates components. Among other things, a component defines the associations between levels and worksheets.	"Creating or Modifying a Component"

Methods and Workflow

This chapter describes options that you can use to create automated actions in your application, outlines the primary configuration options, and summarizes the available tools.

This chapter covers the following topics:

- Introduction
- Overview of Method Types
- Overview of Workflow Step Types
- Typical Step Properties
- Passing Arguments to a Method
- Workflow Schema Examples
- Configuration Notes

Introduction

Demantra provides two closely related options that you can use to create automated actions in your application:

- A method is an object-specific action, which the user sees as an ordinary right-click menu option in a worksheet. With the method, the user can view or edit attributes of a level member and automatically launch any processing steps that you create. In Demantra, methods are typically used for any of the following purposes:
 - Viewing, editing, copying, pasting, or deleting level members.
 - Running the Promotion Optimization engine, from a selected promotion.
 - Performing custom processing as needed by the DSM product (for example, matching a settlement to a promotion).

- A workflow is an automated or semi-automated software process. In Demantra, a workflow can perform any kind of processing needed in a Demantra application. These are typically used for any of the following purposes:
 - To automate routine work such as loading data, running the Business Logic Engine, and maintaining the Demantra database.
 - To define activities that require organized participation from multiple users. Multiple users can participate in an automated workflow, receiving tasks at the appropriate times and sending tasks to others as needed.

These two options are closely related because most methods actually include a workflow.

Methods

Each method is associated with a specific level. Also, a method can be available in all worksheets or in a single specific worksheet.

Demantra provides a set of default methods that you can redefine or disable as needed. When you create a level, Demantra automatically creates the following default methods for it:

- New level_name
- Edit level_name
- View level_name
- Delete level_name
- Copy (only for promotional levels)
- Paste (only for promotional levels)
- Paste from Clipboard (only for promotional levels)

You can customize, disable, or delete these methods. You can add other methods as needed.

Method Security

Within the Collaborator Workbench Administrator, you can specify user access to all methods (as well as to all menu bar items). See "Specifying Permissions for Menu Items".

Workflows

In Demantra, a workflow is a logically connected set of steps. Each step can be

automated or can require interaction from one or more users or groups. Demantra provides a set of workflow steps, each with predefined behavior.

A workflow is associated with one component, the component to which its creator belongs. A workflow can include any users of the component, as well as any groups (groups can be defined across multiple components).

Overview of Method Types

To execute a method, the user right-clicks a level member in a worksheet (or in a Members Browser content pane) and then selects the method name from the menu. The behavior after that depends on the type of method.

Method Types

Demantra provides the following method types:

Constructor	Prompts the user for values of the attributes of the new member and then adds the member in the database.
Destructor	Removes the member from the database.
Edit	Prompts the user for new values of the attributes for this member and then saves the changes.
View	Displays the values of the attributes for this member.
Custom	Optionally prompts the user for new values of the attributes for this member and then runs a workflow.
Open	Opens the default worksheet, filtered to the selected member. You specify a default worksheet when defining the method.
Open With	Similar to Open, but enables the user to select from a list of available worksheets.
Copy	Copies the selected member to the clipboard. Valid only for promotional levels.

Paste	Pastes the member that was last copied to the clipboard. Before pasting, the user can modify the dates of the promotion or choose to derive them from the worksheet. This option is valid only for promotional levels.
Paste From Clipboard	Enables the user to specify which member to paste from the clipboard. Before pasting, the user can modify the dates of the promotion or choose to derive them from the worksheet. This option is valid only for promotional levels.
Add Note	Add a comment or additional information about the selected member.

Constructor, Destructor, and Edit type methods can also run workflows. The workflow is run after the level member is created, removed, or edited.

Constructor, Edit, and Custom Methods

If the method type is Constructor, Edit, or Custom, the following occurs:

1. Depending on how the method is configured, Demantra may save the worksheet data immediately.
2. Demantra optionally displays a dialog box that prompts the user for values of attributes for this level member, as follows:

The screenshot shows a Windows-style dialog box titled "New SKU - Input". Inside the dialog, there is a section titled "Specify attributes for this new SKU". Below this title are five input fields arranged vertically. The first field is labeled "Name" in red text and is a text box. The second field is labeled "Brand" in red text and is a dropdown menu currently showing "Rainbow". The third field is labeled "Control Type" in red text and is a dropdown menu currently showing "NA". The fourth field is labeled "Unit Cost" and is a text box. The fifth field is labeled "Unit Volume" and is a text box containing the value "0.0". At the bottom right of the dialog are two buttons: "Cancel" and "Create".

For a Custom type method, this dialog box includes an additional check box at the bottom. With this, the user can specify whether to save the attribute changes to the

database.

You can customize the following:

- Text at the top of the dialog box
 - Attributes to list and the order in which to show them. The possible attributes include Name, each parent of this level, and any additional attributes. You can specify which attributes are required and which are editable. Required attributes are shown in red.
 - Label on the OK button ("Create" in this example).
3. Demantra creates or edits the member as indicated, and saves the changes to the database.
 4. If the method includes a workflow schema, Demantra continues as follows:
 1. Demantra constructs the set of arguments to pass to the workflow in memory. Specifically, it constructs an array that consists of the following:
 2. Name-value pairs of the attributes of the member, using the values that the user provided.

Note: Demantra does not necessarily pass all the attribute values to the method itself. For a custom method, you specify which attributes should be passed to the workflow.

3. Additional name-value pairs that describe the context in which the method was invoked; see "Passing Arguments to a Method".
4. Demantra runs the associated workflow.

Within the Business Modeler, the method configuration may include additional parameters that control how the method behaves. See "Passing Arguments to a Method". If any of these parameters have the same name as the arguments that are passed to the method, the values that are passed in memory take precedence over the values in the method definition.

For example, suppose the configuration for a given method includes a parameter called `unit_cost` and specifies the value for this parameter as 3.00. If the user invokes this method and specifies `unit_cost` as 3.50, then the value of 3.50 is saved in the database and is used in the method execution. On the other hand, if the user does not specify a value for `unit_cost`, the value of 3.00 is used in the method execution.

5. Demantra optionally displays an output dialog box. The output dialog box is similar to the input dialog box, except that the attributes are not editable.

View Methods

If the method type is View, Oracle Demantra does the following:

1. Oracle Demantra displays a dialog box that displays the values of attributes for this level member. This dialog box is a read-only version of the one shown in "Constructor, Edit, and Custom Methods".
2. When the user clicks OK, Oracle Demantra closes the dialog box.

Destructor Methods

If the method type is Destructor, Oracle Demantra does the following:

1. If the method includes an associated workflow, Oracle Demantra invokes that workflow.
2. It displays the standard deletion confirmation dialog box.
3. When the user clicks OK, Oracle Demantra closes the dialog box and removes the member from the database.

Overview of Workflow Step Types

When you create or edit a workflow schema, the Workflow Manager displays a list of the available kinds of steps:

This section provides a brief overview of all the kinds of available steps, grouped into rough categories. Some kinds of steps fall into multiple categories.

Responding to External Events

A common use of workflows is to wait for specific conditions or external events and then launch actions. For example, a workflow could wait for a file to be updated and then import data into Demantra.

The primary tool here is the Wait Until Step, which pauses the workflow until a specific condition is met, possibly from a set of allowed conditions. When the condition is true, the Workflow Engine continues with the next step in the workflow. In this step, the Workflow Engine can look for a specific file and wait until the file is created, or is modified, or reaches a certain size. Or the Workflow Engine can execute an SQL query repeatedly until it returns a new value (for example, when a price changes). You control the frequency of testing, as well as the time-out period and other timing properties.

The Workflow Engine can also respond to user interaction. Several steps (User Step, Group Step, and Exception Step) send tasks to users and then wait until those users either mark the tasks as done or until the task times out. If the task times out, the

workflow can continue with an alternative step.

Finally, the Workflow Manager itself can be used to schedule workflows.

Sending Tasks and Email

Some steps send tasks to users or groups; these tasks appear in the My Tasks module for those users, within Collaborator Workbench. A typical task is a request to examine a worksheet, make a decision, and possibly edit data. A task can also include a link to a Web page for more information. A task can be accompanied by email. Also, a workflow step can simply send email.

The following kinds of workflow steps support tasks and email:

- User Step sends tasks to a specific user, to ask the user to review and update a worksheet, or to prompt for a workflow decision.
- Group Step sends tasks to a group of users using just one step instead of sending the task individually to each one. This allows you to coordinate your workflow processes with responses from whole groups of users.
- Email Step sends an email to a user that will arrive in the user's standard email application. This step allows integration with the organizational messaging system.
- Exception Step runs a worksheet on which an exception has been defined. If the worksheet returns data, the step then sends tasks to users to resolve the exception.
- Selection Step provides a user with a selection of choices to direct the continuation of the workflow instance. For instance, selection step can be used to obtain approval, rejection, or postponement of workflow activities, or selection of a priority from a list of activities.

Integration

The Transfer Step initiates transfer procedures for the import and export of data. This kind of step is associated with an integration interface, as defined within the Business Modeler.

Managing Demantra Objects

The following workflow steps should be used only as methods. Each of them uses arguments that are available when a user launches a method from a level member or a Members Browser:

- Create Member Step creates the specified level member.
- Edit Member Step makes changes to the specified level member.

- Delete Member Step creates the specified level member.

Other Demantra Actions

The following specialized workflow steps perform actions that are specific to Demantra needs:

- Stored Procedure Step runs a stored database procedure on the database that holds the Demantra data. Demantra provides a set of predefined database procedures, some of which must run regularly in any Demantra solution. See "Database Procedures".
- BLE Step runs the Business Logic Engine directly on a worksheet to evaluate all the client expressions, split the resulting data to the lowest level, and save it to the database. This step automatically starts the Business Logic Engine if necessary.
- Simulation Step runs simulations and then either automatically accepts the results or displays the results in a worksheet for review by a user.
- Worksheet Cache Step refreshes the caches for specified worksheets, for some or all users.
- Refresh Population Step refreshes the population of a promotion to ensure that the promotion data is aligned correctly when items or locations grouped in a promotion (and defined at the aggregate item or location level) change. For example, items can be grouped under a promotion group. If a promotion is defined for a given promotion group and an item grouped under this promotion group is moved to a different promotion group, then running the Refresh Population Step ensures that the promotion data for this promotion is realigned according to the current promotion group definition.

Logic

The following kinds of steps support programming logic within the workflow:

- Selection Step, which was introduced previously, provides a user with a selection of choices to direct the continuation of the workflow instance.
- Condition Step directs the course of the workflow instance depending on condition results obtained from worksheets run on the Demantra database. Instead of testing a worksheet, you can test an SQL query or a custom Java class.
- Exception Step runs a worksheet on which an exception has been defined. If the worksheet returns data, the step then send tasks to users to resolve the exception.
- Container Step runs multiple steps in parallel and then proceeds when all are completed.

- Wait Until Step waits until a specific condition is met before allowing the workflow to continue. The condition can be the existence or modification of a given file, or a change in a value in the database.

Work Step to Launch a Workflow

This work step allows another workflow to be encapsulated within a single step. The two workflows are then linked. When the process flow in the calling workflow reaches this step, the called workflow executes. This only functions synchronously, which means that the two workflows cannot execute simultaneously. The calling workflow waits for the called workflow to complete its work, then resumes execution at the next work step.

The capability to call another workflow adds significant flexibility for designing efficient workflows. For example, let's say there is a Daily Workflow, and a Weekly Workflow that includes the work of the Daily Workflow plus some extra tasks. The Weekly Workflow can just call the Daily Workflow as one of its work steps. It does not need to redefine those tasks.

Creating a Work Step to Call Another Workflow

1. Double click on "Launch Workflow Step" to bring up the work step dialog.

Launch Workflow Step's Properties

Launch Workflow Step

Step Id Step2

Workflow Schema Group: All

Workflow: Import Promo Data

Recovery: Abort

General Time

- Mandatory Fields

OK Cancel

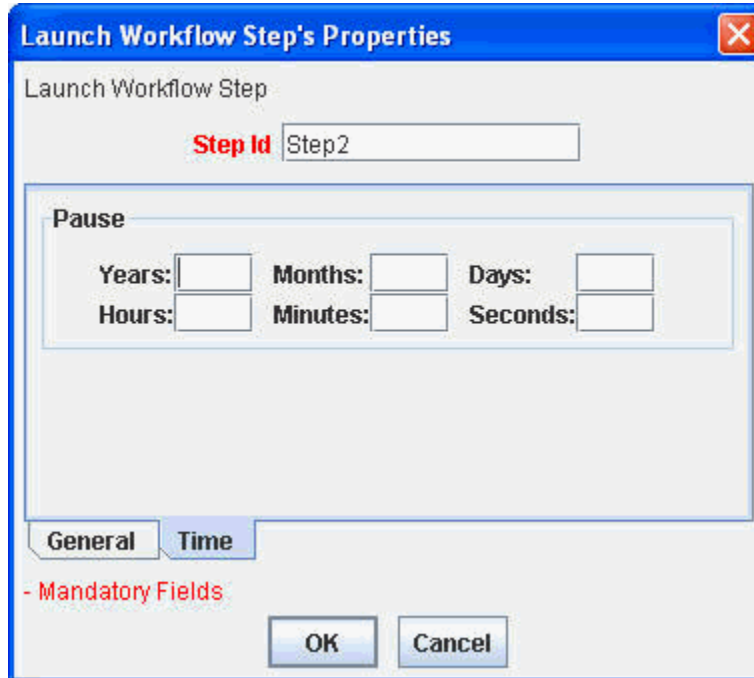
2. In the Step Id text field, enter an appropriate work step name.
3. In the Workflow Schema Group dropdown you may select a group by which the items in the second dropdown are filtered.

4. In the Workflow dropdown, select which workflow shall be called by this work step.
5. In the Recovery dropdown, select what action to take upon recovery if a server crash takes place during this work step.

Ask	The engine follows a Fail-To-Execute procedure for the step.
Abort	The engine executes the step again.
Retry	The engine continues with the next step.
Continue	The engine terminates the workflow instance.

Note: For the Launch Workflow Step, the recommended recovery setting is "Ask."

You may optionally set a pause that will occur between this work step and the beginning of the next one. Select the "Time" tab, then configure the text boxes in the Pause section (Years, Hours, etc.).



Launch Workflow Step's Properties

Launch Workflow Step

Step Id

Pause

Years: Months: Days:

Hours: Minutes: Seconds:

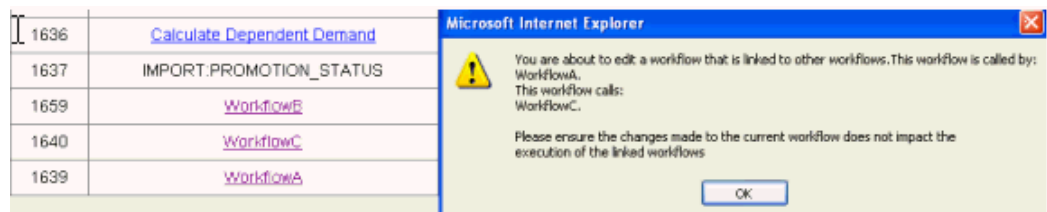
General **Time**

- Mandatory Fields

OK **Cancel**

Editing Linked Workflows

When two workflow are linked, future modifications can alter both of their behaviors, and this may not be immediately apparent to a user editing the workflow without being alerted to this fact. So when a user clicks on 'Edit' for a workflow that is linked to another workflow (it either calls another workflow, or is called by another workflow), the system displays a warning message that identifies the name(s) of the linked workflow(s).



Note that only workflows that are directly linked are named. So if the calling scenario is: wf A > wf B > wf C, when a user starts to edit workflow A, the warning message will provide the name of workflow B, but not workflow C, because it is not directly linked to workflow A.

Note: If a linked workflow is deleted, then the other workflows that were linked to it become invalid. They must now be modified so that they no longer make calls to it or expect to be called by it.

Workflow Recovery

When a server that is running linked workflows crashes, then on recovery, the outer workflow (the calling workflow) is restarted from the current step. For example, if it was at work step C when the crash occurred, then upon recovery execution would begin again at work step C, not work step A.

An inner workflows (a called workflow) however, is simply terminated and must start from the beginning again. So if the outer workflow was at step C, and step C was a call to an inner workflow, and the inner workflow was in the middle of its execution during the crash, then upon server recovery, the inner workflow would always be terminated and the outer workflow would recover step C again per its recovery setting

Since the inner workflow is always terminated upon recovery (if running at the time of the crash), this recovery situation allows the possibility for an inner workflow to be re-run on an unknown state of data caused by the previous run of the inner workflow. In order to mitigate this situation, the user may want to set the recovery setting of the calling workstep of the outer workflow be set to a value of: 'Ask.' On recovery, before calling the inner workflow the system will display a dialog asking the user to choose whether to Retry, Continue, or Abort calling the inner workflow. This provides an opportunity for the user to validate the state of the data caused by the incomplete run of the inner workflow and take any necessary steps before re-running the inner workflow

Workflow Recursion

The ability to launch another workflow from a workflow step may lead to a condition that causes recursion among the workflows. For example: Two workflows can call each other recursively. For example: workflow A calls workflow B, and workflow B calls workflow A.

Since, this could potentially lead to an infinite loop, there is a new system parameter that controls the upper limit of the number of recursive calls that the workflow engine will allow. The parameter is described as follows:

PNAME	PTYPE	VALUE_NUMB ER	DEFAULT_NU MBER	DESCRIPTION
-------	-------	------------------	--------------------	-------------

workflow.recursive.limit	0	2	2	Controls the upper limit on the number of recursive calls during run-time of a workflow. Upon reaching this upper limit, the system will not initiate any more workflows instances that are part of the recursive call. The user who initiated the workflow will be sent a relevant message to the "My Task" pane.
--------------------------	---	---	---	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Caution: It is generally recommended not to configure recursive workflows as it results in constant running of the workflows. Recursive workflows should therefore be configured and used only if the business need warrants it.

External Functions

Finally, other kinds of steps call external functions:

- Executable Step runs executables such as Demantra executables (for example, the Analytical Engine), or external executable and batch files. This step allows interaction between Demantra and external software.
- Custom Step runs a Java class and is typically used to define custom methods. If a workflow is configured as a method, then a user can launch it from within a worksheet. In that case, Demantra automatically passes arguments to the workflow, which Custom Step can use.

Typical Step Properties

This section provides an overview of the properties of a typical step.

Connection Handles

Each step has connection handles that you use to connect it to other steps.

Common Properties

When you add a step to a workflow, the Workflow Editor displays a popup page where you can specify properties for that step. Common properties include the following; not all steps have all these properties.

- The User and Group properties specify users and groups, respectively, associated with the step. Generally, the Workflow Engine sends tasks to these users or groups. Some kinds of steps have both these properties, some have only the User property, and some have neither. In some cases, you can specify only a single user, while in other cases, multiple users are permitted.
- The Worksheet Name property specifies an associated worksheet, from the set defined within Demantra. Different kinds of steps use worksheets in different ways. For example, BLE Step evaluates the client expressions in the worksheet.
- Several properties specify built-in processing delays with short default values. For example, the Pause property specifies how long the Workflow Engine should wait after the step is completed, before starting the next step. In this way, you can coordinate workflow activities by making the engine wait for defined periods of time.
- The Timeout>Timer property specifies when the step times out. For example, if the user does not mark a task as done before its due date, then the task will expire or time out. You use this property to prevent a step from stalling the workflow. If you specify a timeout period, you also specify an alternative following step that the Workflow Engine should execute.

Note: When a step times out, the Workflow Engine executes the timeout step immediately without waiting for the pause counter to finish.

- The Timeout>Alert Time property specifies when the step enters its alert phase.
- The Recovery property specifies the recovery action for the Workflow Engine to use if the system crashes while performing this step.

Passing Arguments to a Method

Demantra can pass arguments in memory to the method. Considered as a group, these arguments are the context dictionary. For each argument, Demantra passes a variable

name and its associated value.

Available Arguments

The available arguments fall into three general categories:

- System information, for example, the ID of the worksheet from which the method was launched.
- Member information, that is, information that indicates the member from which the method was launched.
- User inputs, that is, all arguments shown on the Input dialog box. These arguments are generally attributes of the member from which the method was launched.

The following table lists the possible variables.

Category	Variable*	Value	Data Type
System	ws_id	Identifier of the worksheet from which the method was launched.	Java.util.String
System	worksheet_filter	The filter population of the worksheet from which the method was launched. Represented as a list of pairs of level_id and member_id.	java.util.String level_id,member_id; pairs separated by comas and semicolons.
System	view_name	The name of the active view from which the method was called.	java.util.String
Member	level_id	Identifier of the level from which the method was launched.	java.util.String
Member	member_id	Identifier of the member from which the method was launched.	java.util.String

Category	Variable*	Value	Data Type
Member	Combination_path	The context of the selected member for the method. Will be represented as a list of pairs of level_id and member_id.	java.util.String level_id,member_id
Member	population.filters (example)	Applies only to promotion levels. The population attribute of the selected member. The name of this variable is based on the name of the population attribute as follows: <i>population_attribute_name.filters</i>	Array of com.demantra.applicationServer.metaDataObjects.level.levelFilters.LevelFilterGetters
Member	population.from_date (example)	Applies only to promotion levels. The from_date attribute of the selected member. The name of this variable is based on the name of the population attribute as follows: <i>population_attribute_name.from_date</i>	java.util.Date
Member	population.to_date (example)	Applies only to promotion levels. The to_date attribute of the selected member. The name of this variable is based on the name of the population attribute as follows: <i>population_attribute_name.to_date</i>	java.util.Date

Category	Variable*	Value	Data Type
Inputs	Attribute_column_name	Values of the attributes of the selected member that are specified as inputs to the method (all attributes on the Select Input Arguments screen). The name of each variable is the same as the name of the column in which the attribute is stored.	java.util.Object

Passing Arguments

In order to pass arguments to the method, you must explicitly configure the variables that each workflow step should receive. To do so, you type the parameter names on the Parameters list for that step; see "Properties Used as Arguments for a Method".

Note: The parameter names are case-sensitive.

For the input variables, you also specify which variables to pass when you configure the method. Specifically you select the desired attributes on the Select Input Arguments screen.

Properties Used as Arguments for a Method

When you configure a workflow as a method, Demantra can pass arguments in memory to the method. Considered as a group, these arguments are the context dictionary. For each argument, Demantra passes a variable name and its associated value.

In order to make these arguments available to a workflow step, you must explicitly configure the variables that each workflow step should receive. To do so, you type each variable name in the Name column of the Parameters list for that step, as follows:

Parameters		
Description	Name	Value
	level_id	
	member_id	
Product Family	t_ep_Fam_EP_ID	
Brand	t_ep_Brand_EP_ID	
Name	SKU_D	

In this example, the first two arguments are standard member variables, from the table in "Available Arguments". These arguments can be used in any method.

The remaining three arguments are input variables; these variables refer to attributes of the member. Specifically these are the names of the columns in which these attributes are stored (Product Family, Brand, and Name).

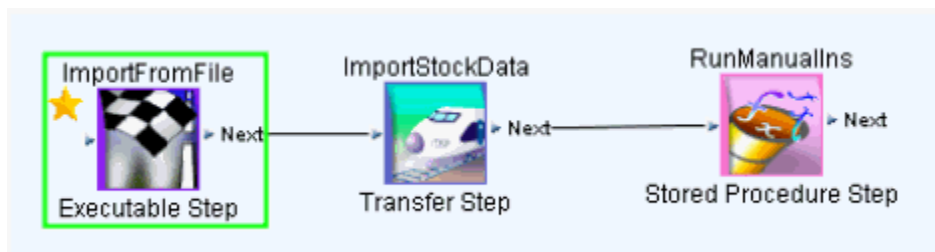
Note: In the Parameters list:

- The parameter names are case-sensitive.
- The descriptions are not used by the method.
- If a value is null in this table, then the value is taken from the member from which the method was launched. If the value is not null, then it is used instead of the value taken from that member.

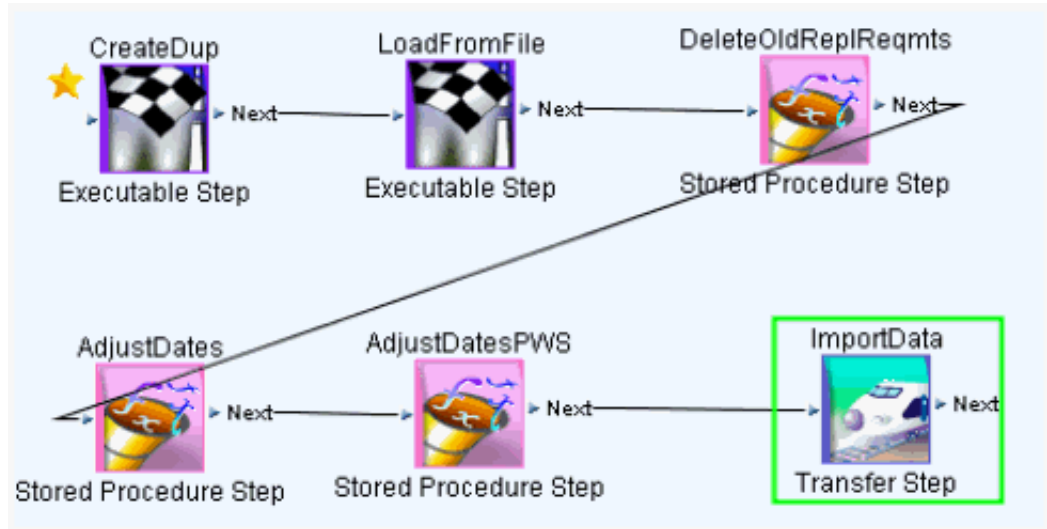
Workflow Schema Examples

Workflows are very often used to import data. In the process of importing data, it is often necessary to perform various kinds of integrity checking and data cleanup, which you do within database procedures. As a consequence, many workflow schemas include the Stored Procedure Step in addition to the Transfer Step.

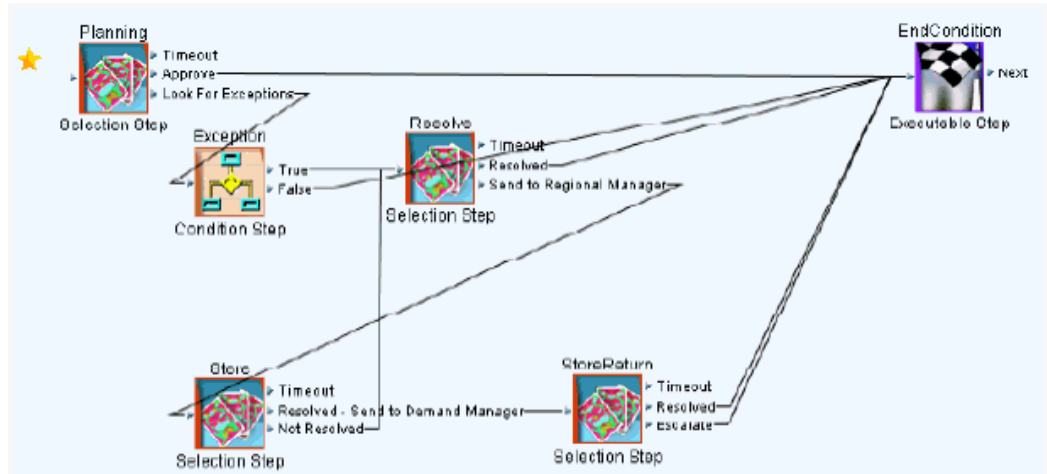
Example 1



Example 2



Example 3



Configuration Notes

This section contains configuration notes related to methods, workflows, and automation in general

Dependencies

Before you can configure a workflow, you may need to define the following elements that it might use:

- Stored procedures
- Java classes

- Batch scripts
- Worksheets

Before you create a method, you may need to define the following elements that it might use:

- Worksheets
- Workflows

Design Considerations

Before you begin creating a workflow schema, you should have a clear idea of the intended purpose, flow, and users of the schema. Also, you should consider the following issues:

- Remember that client expressions that are affected by data changes should not be used until the Business Logic Engine evaluates them. The Workflow Manager does provide a kind of step that submits worksheets to the Business Logic Engine.
- A worksheet must be public in order to be visible from a task. Also, users must have the correct security privileges to view worksheet results.
- Initiators of a workflow instance must have permissions to view worksheet results for all worksheets that the workflow schema includes. If the workflow instance is to be initiated by more than one user, the worksheet must be a public worksheet, and the users of the group must have permissions to view the worksheet results.
- Properties of workflow steps cannot be changed within or by the workflow instance. If you make changes to a workflow schema, those changes do not affect any instances that are currently running.
- Only the owner of a workflow schema can edit that schema. Anyone with login access to the Workflow Manager can launch it.

Tools

Demantra provides the following tools for creating and configuring methods and workflows:

Tool	Purpose/Notes
Configure > Method option in the Business Modeler	Defines methods.

Tool	Purpose/Notes
Workflow Manager	Defines workflows.

Security

This chapter explains the Demantra security mechanisms.

This chapter covers the following topics:

- Data Security
- Feature Security
- Other Security Features
- Program Groups
- Configuration Notes
- Password Policy

Data Security

Demantra data is secured as follows:

- The data is partitioned into components, which generally correspond to organizational roles, which can overlap. Each component has an owner, who acts as the administrator and who can create additional users. (See "Creating or Modifying a Component".)
- Each user is authorized for one component. In addition, you can further restrict a specific user's access to data by applying filters so that the user can see only specific level members as well as only certain series.
- Users can belong to groups, and group members can collaborate, inside or outside of workflows. When a user creates a note, he or she can control access to that note by user or by group.

The following table summarizes how Demantra controls access to data elements.

Data Element	Options	Controlled by Component	Controlled by User Group	Controlled by User ID
Series	Visible or not visible	Yes	No	Yes
Series indicators (which indicate the presence of a note or promotion within the worksheet table.)	Visible or not visible	Yes	No	No
Levels	Visible or not visible	Yes	No	No
Level members	Full control, including ability to delete members Read/write existing members Read existing members No access	Yes	No	Yes
Units of measure	Visible or not visible	Yes	No	No
Indexes and exchange rates	Visible or not visible	Yes	No	No
Notes	Similar to level member options	No	As specified by creator of note	

It is useful to remember that each user of a component sees a subset of the data associated with that component. You cannot give user access to data that is not contained in the component.

Components

For more information about components see: [Creating or Modifying a Component](#),

page 61-1

Each component has the following properties:

- Series, levels, units of measure, indexes, and exchange rates. For each level, you define permissions that the users have for the members of that level. The choices are as follows:
 - Full control, including ability to delete members
 - Read/write existing members
 - Read existing members
 - No access
- An owner. This owner acts as the administrator of the component.
- Possible additional users, created by the owner. The owner can also further restrict data access for particular users.

Users

User details are defined using the Business Modeler (Security > Create/Modify User). For more information about users see: [Creating or Modifying a User](#), page 61-10

For users, you can specify the following details:

- Overall permission level, which can enable the user to log onto Demantra administrative tools and modify the component.
- Series that the user can access, generally a subset of the series included in the component.
- Optional permissions to control which level members the user can see and edit. The choices are as follows:
 - Full control, including ability to delete members
 - Read/write existing members
 - Read existing members
 - No access (the members are filtered out entirely for this user)
- Group or groups to which the user belongs.

User Groups

For user groups, you can specify the following details:

- Which users are in the group.
- Whether this user group is also a collaboration group (for use by the Workflow Engine).
- Whether users of this group can log into the Workflow Editor.

Security for Deleting Members

Most level members are created by integration and it would generally be undesirable to delete them. Most users, therefore, do not have delete access to these members. The exception is a user with System Manager permission; see "Permission Levels".

Level members can be created directly within Demantra (through Member Management). For any these members, the user who created the member has permission to delete it.

Data Security at Higher Levels

When a user views data at an aggregation level that is higher than where the permissions are set, it is necessary to resolve how to aggregate editable members and uneditable members. Demantra uses the following rules:

- If all lower-level members are editable (either as read/write or full control), the member is editable.
- If some of the lower-level members are visible but read-only, the member is not editable.
- If some of the lower-level members are not visible, those members are filtered out and do not affect the aggregation. The upper-level member may or may not be editable, depending on the preceding rules.

Feature Security

Demantra features are secured as follows:

- Permission levels control access to administrative tools and to menu items. Demantra provides four predefined permission levels that you can customize. You can control access to all of the Demantra menus:
 - Menus on the Collaborator Workbench menu bar
 - Menus on the DSM menu bar
 - Menus on the Promotion Effectiveness menu bar
 - Menus on the Demand Management menu bar

- Right-click menus associated with each level in your system
- You can also control access to all the same menu items at the group and user ID level.

For convenience, you control access to individual menu items, to predefined collections of menu items, or to your own collections of menu items (your own program groups).

Permission Levels

Demantra defines four permission levels, as follows:

- System Manager
- Supervisor
- Power user
- Casual user

The table below shows the default rights for these four permission levels. Note that only the System Manager has a different set of permissions from the other three. However, users with the System Manager permission level can utilize the Collaborator Workbench Administration tool to modify the access restrictions for specific menu items, or sets of menu items, thereby changing these defaults. See the section Specifying Permissions for Menu Items., page 61-25

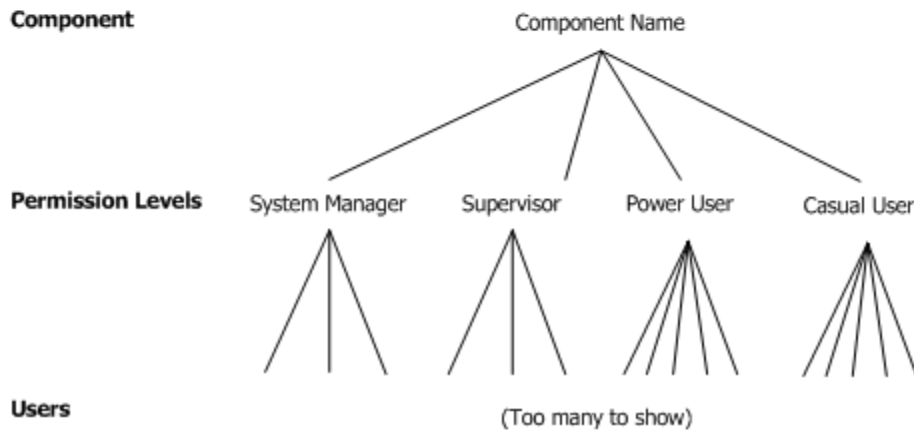
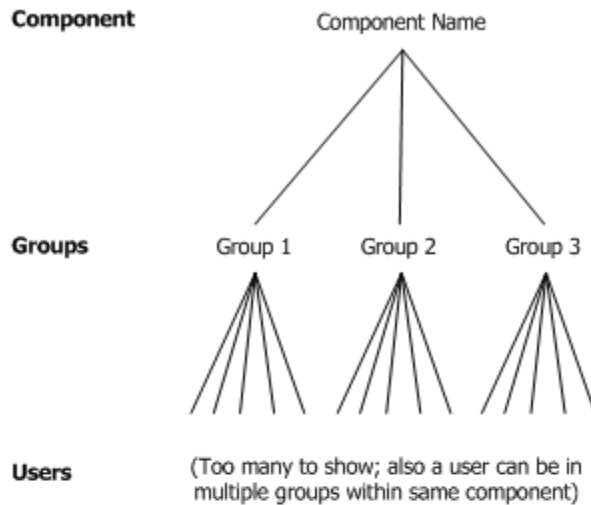
Permission Level	Business Modeler – login / change pwd	Business Modeler – All Menus	Collaborator or Workbench Administration tool	Collaborator or Workbench - view public and own worksheets	Collaborator or Workbench - view all worksheets	Demand Planner - System menu
System Manager	X	X	X	X	X	X
Supervisor	X	-	-	X	-	-
Power User	X	-	-	X	-	-
Casual Supervisor	X	-	-	X	-	-

Permission Hierarchies

In order to understand how Demantra determines a given user's access to a given menu

item, it is necessary to understand the permission hierarchies and how Demantra combines them.

Demantra has two independent permission hierarchies. In the first hierarchy, each component includes groups, and each group includes users. A user can belong to multiple groups, provided that all those groups belong to the same component. In the second hierarchy, each component includes four permission levels, and each user has one permission level.



Explicit and Implicit Permissions

In Collaborator Workbench you can display or hide any menu item. You can also display but disable a menu item, which can provide a useful clue about advanced features that are available to other users. Each permission is either explicit or implicit (inherited).

Note: For more information see: Logging into the Collaborator Workbench Administrator, page 61-20.

You define permissions in an expandable hierarchy like the following. For now, let's focus on the three check boxes:

Program Object	Hidden	Disabled	Inherited Permission
Settlement Management			
File	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Worksheet	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Edit	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
View	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Options	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Data	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Help	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Program Groups			
Object Menu			

The following table describes how to use these check boxes:

Desired outcome	Hidden	Disabled	Inherited Permission
Menu option is explicitly hidden	Checked	Irrelevant	Unchecked
Menu option is explicitly displayed but disabled	Unchecked	Checked	Unchecked
Menu option is explicitly displayed and enabled	Unchecked	Unchecked	Unchecked
Use implicit permissions for this menu item	Unchecked	Unchecked	Checked

How Demantra Combines Multiple Permissions

For a given user and a given menu item, Demantra checks for all the following

permission descriptions:

- For the component
- For each group to which the user belongs
- For the permission level that the user has
- For the user ID
- For each program group to which the menu item belongs

To determine whether a user has access to a given menu item, Demantra searches for and combines the permission descriptions as follows.

1. Demantra checks to see if the user has an explicit permission setting (for a given menu item). If so, that setting is used, and all others are disregarded.
2. If the user does not have an explicit permission setting for a given menu item, then Demantra looks at the settings for the groups to which the user belongs, the permission level that the user has, and each program group that the menu item is in. Here, the following rules apply:
 - An explicit permission takes precedence over an implicit permission.
 - Among explicit permissions, the most liberal permission takes precedence.
 - Among implicit permissions, the most liberal permission takes precedence.
3. If no explicit permission setting for the menu item has been found so far, then Demantra uses the permission setting at the component level, if any.
4. If there is no setting at the component level, Demantra displays and enables the menu item.

See Also

"Data Security"

"Specifying Permissions for Menu Items"

Other Security Features

Note the following additional security features:

- Assigning a user to a user group that is flagged as a 'Collaboration Group' provides access to the Workflow Manager.
- After adding a user to a Collaboration Group, the Web server must be restarted

before that user can access the Workflow Manager. For more information about User Groups see: Creating or Modifying a User Group, page 61-16

- A user with the System Manager permission level can see all public worksheets and all private worksheets. Users with lower permission levels can see all public worksheets and all private worksheets created by themselves.
- A user with the System Manager permission level can see the System menu in the desktop Demand Planner, in addition to the other menus.
- Any user can log onto the Business Modeler. If the user's permission level is lower than System Manager, the user can only change his or her own password, as documented in the user guides.

Program Groups

For more information about Program Groups see: Defining a Program Group, page 61-20

Redefining a Program Group, page 61-24

Deleting a Program Group, page 61-25

A program group is a collection of menu items, typically related to each other in some way. You create program groups so that you can easily control access to all the menu items in the group.

Demantra provides several predefined program groups, for convenience. These program groups contain only menu items from the right-click menus.

Program group	Menu items in this group, by default
Add	New <i>member</i> right-click menu option for every level in the system.
Edit	Edit <i>member</i> right-click menu option for every level in the system.
Delete	Delete <i>member</i> right-click menu option for every level in the system.
View	View <i>member</i> right-click menu option for every level in the system.

Program group	Menu items in this group, by default
Copy	Copy, Paste, and Paste from Clipboard right-click menu options for every applicable level in the system. (Note that this option is available only for promotional-type levels.)
Open	Open and Open With right-click menu options for every level in the system.

Configuration Notes

The following table summarizes the Demantra security tools.

Tool	Purpose/Notes
Components > Open/Create Component option*	Creates components, which are usually created as part of basic implementation.
Security > Create/Modify User option*	Creates users and configures all information except for access to menu items.
Security > Create/Modify Group option*	Creates user groups and configures all information except for access to menu items.
Collaborator Workbench Administrator	Controls access to menu items; defines program groups.
*These options are in the Business Modeler.	

Password Policy

You can set up Demantra to enforce password policies and ensure that passwords are well-formed and are changed frequently. By default, password policies are not enforced. To enable password policy verification, an administrator must modify the system parameter PasswordRulesEnforcedGlobal (see System Parameters).

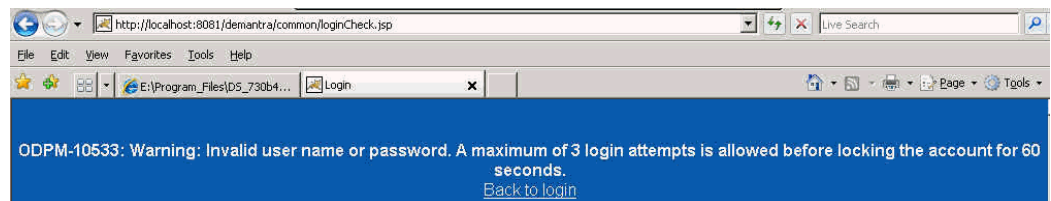
Once enabled, the password policies are:

- Password length must be 8 to 12 characters.
- At least one character must be in UPPER CASE.

- At least one digit or special character must be used in the password.
- At least one digit or special character must be used in the password.
- Password should NOT be a Security Dictionary Word (please contact your administrator for details).
- Password should NOT be the same as User name.
- Password should NOT be the same as current password.

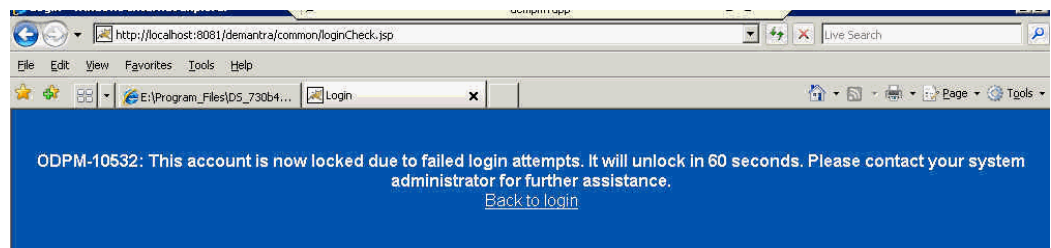
If a user attempts to create a new password that does not follow these policies, a message notifies the user of the password policies.

If the user attempts to login and fails, a message similar to the following appears:



The number of tries allowed by the password policy is determined by the system parameter "AccountLockoutThreshold". (see System Parameters).

If the user is locked out because of too many failed attempts, the following message appears:



An administrator can unlock the user's account by logging into Business Modeler, navigating to Security > Create/Modify User, and then deselecting the Locked check box.

If an administrator explicitly locks a user's account, a different message appears, saying that the account is locked and to please contact your system administrator.

Note that this locking applies to Collaborator Workbench, Workflow Manager, Administrator Login, Demand Planner Web, Dynamic Open Link (DOL) , Demantra Anywhere. Locking does not apply to the Business Modeler, Member Management, or Chaining Management.

When a user's password expiration date is within 10 days, a message displays

prompting the user to change his password.

For more information see these system parameters:

- PasswordHistoryCount
- PasswordRulesEnforcedGlobal
- AccountLockoutThreshold
- AccountLockoutDuration
- PasswordResetTime

Proport

This chapter explains the proposit mechanism.

This chapter covers the following topics:

- Overview
- How You Can Tune the Proport Mechanism
- Calculating the Rolling Average Demand
- Calculating the Monthly Proportions
- Calculating the Average Daily Demand for Each Month
- Handling Data Issues
- Which Combinations Are Affected
- Other Notes on the Proport Mechanism
- Proport when using Engine Profiles
- Proport When Forecasting on General Levels

Overview

The proposit mechanism computes and stores information about the average demand per day for each item-location combination. Demantra uses this information whenever it needs to split higher-level data across the relevant lowest-level members.

For example, if one item-location combination had four times as many sales as another, the former combination should receive four times as much of the forecast.

When Proportions Are Used

In general, Demantra splits data whenever necessary, including the following occasions:

- When the Analytical Engine generates a forecast at an aggregated level.

- When data is imported at an aggregated level.
- When users perform chaining at an aggregated level.

This chapter describes how matrix proportions are calculated.

Kinds of Proportions

Demantra provides three general ways to specify the relative proportions of different combinations:

Kind of proportions	Details	When used
Matrix proportions or stored proportions	Proportions that Demantra calculates and stores for later use. The calculation is based upon the demand, but also considers recent average demand, month-to-month variations, and so on. Various parameters and combination-specific flags control exactly how proportion works. These proportions are averages and are not as good as actual proportions.	Option when importing data Automatically used when forecast must be created at higher level
Actual proportions	Use the proportions of the Demand series.	Option when importing data
Proportions of a reference series	Use the proportions of a reference series, typically: Demand (suitable for a historical series) Final Plan (suitable for a forecast series)	When data is edited at an aggregated level
SALES_DATA based proportion	Proportion that Demantra uses for future forecast splits. This proportion is used when the SKUs have insufficient historical demand data, and the engine forecasts several SKUs aggregated together with older SKUs providing history for the SKUs in use. In nodes where all participants are marked for SALES_DATA based proportion, the engine splits the forecast using the lowest level value of the series called Future Proportions.	When the future forecast is split based on configurations set by the users.

How You Can Tune the Proport Mechanism

You can tune the proportion mechanism as follows;

- Tuning how the proportion mechanism smooths data variations from month to month. You can tune these settings globally or separately for individual combinations.
- Specifying how the proportion mechanism handles null data and other data issues.
- Specifying which combinations the proportion mechanism should consider when it runs.

The following sections describe how proportion handles these steps.

Calculating the Rolling Average Demand

For each combination, Demantra computes a rolling average of the most recent demand over some span of time. This rolling average (`glob_prop`) depends on the following:

<code>hist_glob_prop</code>	Number of base time buckets to include when calculating the running average demand. Usually, you use one season's worth of data. Each combination can have a different value for this setting.
<code>quantity_form</code>	Expression that Demantra uses to calculate demand, based on sales data and various overrides. The default expression transforms negative values to zero and should be modified if business needs require negative demand.
<code>proport_missing</code>	Specifies what value to use for dates with null sales. See "Specifying How to Treat Null Data".

Calculating the Monthly Proportions

After calculating the rolling average demand for each combination, Demantra calculates the average demand per day (P1, P2, ..., P12) averaged over a month's time. This calculation consists of three steps:

1. Calculating the average daily demand for each month of the year.

2. Adjusting the level of these averages to account for any overall trend. This calculation uses the rolling average demand.
3. Smoothing these averages to account for month-to-month variations. This calculation also uses the rolling average demand.

Note: In weekly and daily systems, the proportion mechanism scales the monthly proportions (P1, ..., P12) by dividing by the number of days in the month, as appropriate.

Calculating the Average Daily Demand for Each Month

For each combination, Demantra calculates the following average demand per day averaged over a month's time, for each month of the year. (This data is stored in `mdp_matrix`):

P1	Average demand per day for the month of January
P2	Average demand per day for the month of February
and so on	

Smoothing Out Variations

Depending on your business, you may want to smooth out the month-to-month variations. The `delta` field in `mdp_matrix` specifies a weight for a given item-location combination. Demantra uses this weight to even out these variations as in the following example:

$$P1 = \text{glob_prop} * \text{delta} + (\text{old } P1) * (1 - \text{delta})$$

- The `delta` field must be a floating-point number, anywhere between 0 and 1, inclusive. The larger it is, the more you smooth out the day-to-day variations.
- The `def_delta` parameter specifies the default value for the `delta` parameter for any new combinations.

These smoothed proportions are stored in `mdp_matrix` follows, overwriting the old P1, P2, ... fields:

P1	Smoothed, level-adjusted average daily demand for the month of January
P2	Smoothed, level-adjusted average daily demand for the month of February
and so on	

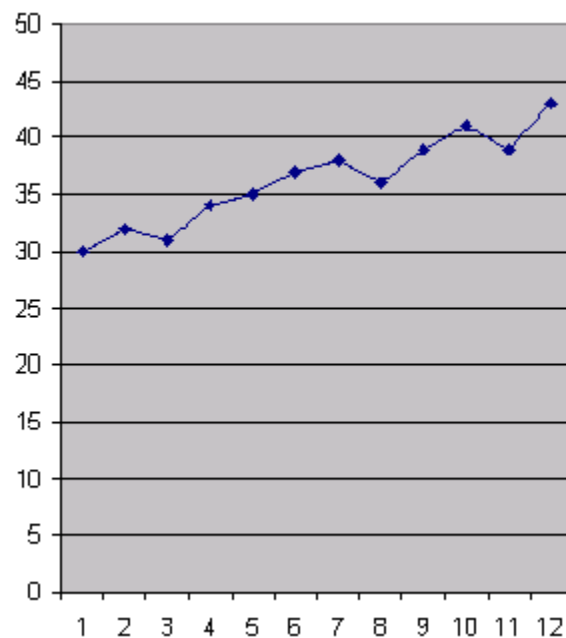
Adjusting the Level

Starting with the average daily demands for each item-location combination for each month of the year, Demantra considers the change in level over the past year and adjusts the level of the proportions accordingly.

For a simple example, consider the following historical daily demand, over the last year:

Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
30	32	31	34	35	37	38	36	39	41	39	43

A slight upwards trend is fairly obvious, especially when this demand is graphed.



To keep matters simple, suppose that we have only one year's worth of data. If we just used demand from last January (that is, 30) for the next January, we would

underestimate the demand, because the overall level of demand has increased over the last year. Therefore, Demantra calculates the level-adjusted daily demand for January as follows:

$$P1 = (\text{smoothed } P1) * (\text{rolling average}) * 12 / (\text{sum of all average demand})$$

Here smoothed P1 is the average demand per day for January as calculated previously and then smoothed as described in "Smoothing Out Variations". These level-adjusted, "normalized" proportions are stored in mdp_matrix, overwriting the old P1, P2, ... fields:

P1	Level-adjusted, smoothed average daily demand for the month of January
P2	Level-adjusted, smoothed average daily demand for the month of February
and so on	

Handling Data Issues

You can control how the proposit mechanism handles various data issues.

Specifying How to Treat Null Data

You must specify how to treat missing dates, which can have a large effect on the averages. If there are no sales for an item-location combination for a given month, that can either mean there truly were zero sales or it can indicate a problem with the data. You specify the proposit_missing parameter as follows.

- If this parameter is equal to 0, dates with null data are treated as dates with zero sales. That is, suppose that you have three months worth of data as follows: 30, null, 60. If proposit_missing equals 0, the average of these three months is calculated as 30 (or $[30+0+60]/3$).
- If this parameter is equal to 1, dates with null data are ignored. Using the old example, if proposit_missing equals 1, the average of these three months is calculated as 45 (or $[30+60]/2$). This is mathematically equivalent to assuming that the missing month has average sales (45).

Similarly, suppose you have weekly sales data, but you do not have data for all the weeks in a given month. If proposit_missing equals 0, the weeks with null sales are treated as having zero sales. If proposit_missing equals 1, the weeks that have null sales are considered as having average sales.

For data with many missing observations, it is likely that the null sales actually represent no sales; in this case, it is suitable to specify proposit_missing as 0.

For data with only a few missing observations, it may be more likely that the missing observations represent data problems. In this case, it would be better to specify `proport_missing` as 1 and ignore the missing observations.

Determining Coverage of the Months of the Year

For any given item-location combination, the sales may not include data for every month of the year. For example, for a given item-location combination, suppose that you have 24 months worth of data, but that there were no sales in November or December—for any year. This means that you have ten distinct months represented in the history.

The `proport_threshold` parameter specifies the minimum number of distinct months that must be present in the sales data for any given item-location combination. For example, if you have data for three Januaries, that counts as one observation for January.

Then:

- If the history does contain enough distinct months, the averages are calculated as normal for the months that have non-null data. You must specify what values to use for the other months; see "Specifying How to Handle Missing Time Buckets".
- If the history does not contain enough distinct months, Demantra checks the value of the `proport_missing` parameter and then does the following:
 - If `proport_missing` equals 0, Demantra sets the averages equal to the `glob_prop * delta`.
 - If `proport_missing` equals 1, Demantra sets all averages equal to the rolling average (`glob_prop`).

In the preceding example, suppose that you have monthly data and suppose that `proport_threshold` is 11. In this case, this combination does not have data for enough distinct months, and all monthly proportions are equal to `glob_prop`. In contrast, suppose that `proport_threshold` equals 8 instead. In this case, the monthly averages are calculated as normal for the months with non-null data.

Specifying How to Handle Missing Time Buckets

You use the `proport_spread` parameter to specify what value to use for any bucket that has null data.

Suppose that we have the following data for a given item-location combination:

Nov 2002	Dec 2002	Jan 2003	Feb 2003	Mar 2003	Apr 2003	May 2003	Jun 2003	Jul 2003	Aug 2003
7	0	1	5	9	3	2	null	1	no data yet
no data before this month									

Before we examine the `proport_spread` parameter, it is worthwhile to rearrange this information and identify which months are missing:

P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12
1	5	9	3	2	missi ng	1	missi ng	missi ng	missi ng	7	0

The `proport_spread` parameter can equal any of the following values:

- If `proport_spread` is 0, missing months receive 0 proportions. In this case, Demantra calculates the monthly averages as follows:

P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12
1	5	9	3	2	0	1	0	0	0	7	0

- If `proport_spread` is 1, Demantra checks the value of the `proport_missing` parameter and then does the following:
 - If `proport_missing` equals 0, then missing months receive `glob_prop*delta`. In this case, Demantra calculates the monthly averages as follows:

P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12
1	5	9	3	2	glob _pro p*de lta	1	glob _pro p*de lta	glob _pro p*de lta	glob _pro p*de lta	7	0

- If `proport_missing` equals 1, then missing months receive the rolling average (`glob_prop`). In this case, Demantra calculates the monthly averages as follows:

P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12
1	5	9	3	2	glob _pro p	1	glob _pro p	glob _pro p	glob _pro p	7	0

- If `proport_spread` is 2, Demantra considers whether a missing month *could* have been included within the history. (This setting has an effect only if you have data for less than a full year as in our example.)

First, Demantra uses 0 for missing months that *could* have been included within the partial year.

For missing months that *could not* have been included, Demantra checks the value of the `proport_missing` parameter and then does the following:

- If `proport_missing` equals 0, then missing months receive `glob_prop*delta`.
- If `proport_missing` equals 1, then missing months receive the rolling average (`glob_prop`).

In our example, the history started in November 2002 and continues through July 2003. That span of time does not include the months of August, September, and October, so those missing months receive `glob_prop`; the missing month of June, on the other hand, receives 0. In this case, Demantra calculates the monthly averages as follows (assuming that `proport_missing` does not equal 1)

P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12
1	5	9	3	2	0	1	glob_ prop	glob_ prop	glob_ prop	7	0

Note: If you set `proport_missing` to consider omitted values as null, there is no reason to set the `Run_full_matrix_proport` parameter to 1.

Which Combinations Are Affected

By default, the `proport` mechanism loads only combinations with non-null values, and it recalculates proportions based only on the loaded combinations. In some cases, this is

correct; it may not be desirable to recompute proportions frequently if your data is intermittent.

In other cases, however, a null value really does mean zero sales, and the split should be recalculated accordingly. Assume that proposit is considering three months of history data (hist_glob_prop equals 3). Consider this example:

Combination	January	February	March	April
1 (Product A at store A)	500	500	400	(null)
2 (Product A at store B)	50	60	40	70

In this case, if you calculated proportions in March, you would calculate the split between combination 1 and combination 2 as $(500+500+400)/(50+60+40)$, which is appropriate.

However, if you calculated proportions in April, you would not load new member data for combination 1. In this case, the system would calculate the split between combination 1 and combination 2 as $(500+500+400)/(60+40+70)$, which means that combination 1 would receive most of the split, even though there were no sales of this product in this store in April. This is probably not what you want.

The Run_full_matrix_proposit parameter specifies whether to run the proposit mechanism on all the item-location combinations.

- If no (0), run proposit only on the combinations that have newly loaded sales data or that have been flagged (prop_changes=1) in the database.
- If yes (1), the proposit mechanism calculates proportions for all nodes at loading time and assumes that null values represent zero. This takes longer (possibly much longer), but avoids the miscalculation outlined previously.
- If equal to 2, the proposit mechanism calculates proportions for all combinations that have new_member=1.

The proposit mechanism then recomputes the rolling average, individual monthly averages, and individual monthly proportions for each of those combinations, as described in "Calculating the Rolling Average Demand".

Other Notes on the Proposit Mechanism

The proposit mechanism considers only real combinations. That is, it ignores combinations for which is_fictive equals 1.

The proposit mechanism calculates the prediction status of each combination, in addition to calculating the proportions.

Proport when using Engine Profiles

Proport supports all engine profiles. The proposit procedure can be passed an engine profile id (the engine profile ID is a system-generated value generated when the user creates the engine profile). Users must query the DB table engine_profiles to determine the ID of the profile that they want to use (i.e. insert this value for the new 'ProportDefaultEngineProfile' system parameter described below).

Proport then searches the corresponding parameter table for parameters currently used to drive proportion and prediction_status calculations. These parameters include but are not limited to:

- Start_date
- Quantity_form
- Proport_threshold
- Def_delta
- Last_date
- Last_date_backup
- HistoryLength
- Timeunit
- Dying_time
- Mature_age
- Hist_glob_prop

Any parameters not found in the profile specific parameter table will be retrieved from the base profile parameter table.

In instances where proposit is called from another process, including data load or data updates, the profile called will be based on the system parameter ProportDefaultEngineProfile. This parameter should be modified if the proposit engine profile differs from the base profile.

Proport When Forecasting on General Levels

When determining which combinations require a forecast, general level-based

combinations behave differently than sales_data-based combinations. General Level (GL) combinations have an effectivity start and end date. Combinations are only forecast for dates when they are active.

When the proposit procedure is called, it is passed a profile_id to be used. Proposit will first search for definitions on the init_params table matching the passed profile. Any engine parameters that are not found in this profile use the parameter values defined in init_params_0.

In cases where the data and combination tables are not specified as SALES_DATA and MDP_MATRIX, proposit will generate GL-relevant proportions and prediction_status values on the specified tables.

If a supersession is detected by the use of the EngKeyDef_supersession parameter, then the proportion is split as follows:

- Fit Split (Historical Forecast)
- Forecast Split (Future Forecast)

For both, the split is done according to the allocation rules from the forecast level down to the lowest GL level, defined by the lowest level of the forecast tree. Once the forecast is allocated down to the lowest GL level, any underlying combinations receive equal split.

Note: When forecasting above the supersession level, each supersession group receives its part of the forecast based on historical volume proportions, similar to the split used for non-supersession groups.

Fit Split

When allocating the historical forecast to the lowest GL level, the split is done in the usual manner, using the historical demand values.

For example:

- End of history is defined for 12/1/2009

If fit forecast generated is 4000 per month, the forecast is allocated as follows based on historical values for each combination and month:

Total Forecast	Date	Items					
		1	2	3	1	2	3
		History				Historical Forecast	
4000	1/1/2009	500	1500		1000	3000	
4000	2/1/2009	300	700		1200	2800	
4000	3/1/2009	800	1200		1600	2400	
4000	4/1/2009	1200	800		2400	1600	
4000	5/1/2009	700	300	1000	1400	600	2000
4000	6/1/2009	500	500	1000	1000	1000	2000
4000	7/1/2009		800	1200		1600	2400
4000	8/1/2009		1200	800		2400	1600
4000	9/1/2009		900	1100		1800	2200
4000	10/1/2009		400	1600		800	3200
4000	11/1/2009		200	800		800	3200
4000	12/1/2009		1000	1000		2000	2000

In the situation where there are multiple members under the GL level, the allocation of fit among those members should be based on an equal split.

Forecast Split

Splitting the forecast is driven by proportions. General level forecasting can be more complicated because of the requirement to respect effectivity start and end dates as well as product revisions logic. In addition, all members belonging to the same revision are defined by parameter EngKeyDef_Supersession. If this parameter is defined at the same aggregation level as the EngKeyDefPK parameter, then no supersessions are being used. If the EngKeyDef_Supersession parameter is defined above the EngKeyDefPK parameter, all nodes belonging to the same EngKeyDef_Supersession node are part of

the supersession.

Note: When enabled, split by series takes precedence over the split rules used below.

Split from Supersession Level to GL Level

When allocating a forecast generated at a supersession level, the forecast is allocated to dates that are defined as active for each member of the supersession. In this scenario, when forecasting for a specific date to underlying combinations, two options are available:

- Allocate forecast amongst all active revisions
- Allocate forecast only to the latest active revisions

When allocating the forecast amongst all active revisions, the forecast should be split equally among the GL members whose activity dates specify that the member is active during this period.

Example: Using All Active Revisions

- End of history defined as 12/1/2009
- Allocation in the supersession is set to All Active Revisions
- Amount of history used to calculate proportions is 12 months

A specific forecast node has five underlying combinations:

Latest Revision	Item	Effectivity Start	Effectivity End
AA	1	1/1/2008	6/1/2009
AA	2	1/1/2009	12/1/2009
AA	3	5/1/2009	6/1/2010
AA	4	1/1/2010	10/1/2010
AA	5	6/1/2010	12/1/2011

The history is as follows:

Item				
1	2	3	4	5
History				
500	1500			
300	700			
800	1200			
1200	800			
700	300	1000		
500	500	1000		
	800	1200		
	1200	800		
	900	1100		
	400	1600		
	200	800		
	1000	1000		

A forecast of 4000 is generated for each period. The future one-year forecast is allocated equally among the revisions active at each date as follows:

Proportions					
Item	1	2	3	4	5
Date	Active Revisions				

Proportions						
1/1/2010	3,4	0	0	4000/2	4000/2	
2/1/2010	3,4	0	0	4000/2	4000/2	
3/1/2010	3,4	0	0	4000/2	4000/2	
4/1/2010	3,4	0	0	4000/2	4000/2	
5/1/2010	3,4	0	0	4000/2	4000/2	
6/1/2010	3,4,5	0	0	4000/3	4000/3	4000/3
7/1/2010	4,5	0	0	0	4000/2	4000/2
8/1/2010	4,5	0	0	0	4000/2	4000/2
9/1/2010	4,5	0	0	0	4000/2	4000/2
10/1/2010	4,5	0	0	0	4000/2	4000/2
11/1/2010	5	0	0	0	0	4000
12/1/2010	5	0	0	0	0	4000

When the option of latest revision has been chosen, the forecast generated at a supersession level is only allocated to the latest revision active during that period. The latest revision is defined as the active GL member who has the latest effectivity start date.

Example: Using Latest Revision

- End of history defined as 12/1/2009
- GLPropSupersessionMethod set to Latest Revision

A specific forecast node has five underlying combinations:

Latest Revision	Item	Effectivity Start	Effectivity End
AA	1	1/1/2008	6/1/2009

Latest Revision	Item	Effectivity Start	Effectivity End
AA	2	1/1/2009	12/1/2009
AA	3	5/1/2009	6/1/2010
AA	4	1/1/2010	10/1/2010
AA	5	6/1/2010	12/1/2011

The history is as follows:

Item	1	2	3	4	5
Date	History				
1/1/2009	500	1500			
2/1/2009	300	700			
3/1/2009	800	1200			
4/1/2009	1200	800			
5/1/2009	700	300	1000		
6/1/2009	500	500	1000		
7/1/2009		800	1200		
8/1/2009		1200	800		
9/1/2009		900	1100		
10/1/2009		400	1600		
11/1/2009		200	800		
12/1/2009		1000	1000		

A forecast of 4000 is generated for each period. The future one-year forecast is allocated

as follows:

		Proportions				
Item		1	2	3	4	5
Date	Latest Revision					
1/1/2010	4	0	0	0	4000	
2/1/2010	4	0	0	0	4000	
3/1/2010	4	0	0	0	4000	
4/1/2010	4	0	0	0	4000	
5/1/2010	4	0	0	0	4000	4000
6/1/2010	5	0	0	0		4000
7/1/2010	5	0	0	0		4000
8/1/2010	5	0	0	0		4000
9/1/2010	5	0	0	0		4000
10/1/2010	5	0	0	0		4000
11/1/2010	5	0	0	0		4000
12/1/2010	5	0	0	0		4000

Note: In cases where supersessions exist, Oracle recommends that EngKeyDef_Supersession be set at a level aggregating the different supersession nodes together.

Part 2

Integration

General Integration Guidelines

A pre-made model supports out-of-the-box integration of the Oracle e-Business Suite or EnterpriseOne applications with the Oracle Demantra Demand Management application. This model contains necessary definitions to support both Oracle e-Business Suite and EnterpriseOne applications and uses many common building blocks. The Oracle e-Business Suite model serves as the core best practice for Oracle Demantra Demand Management integration. Where EnterpriseOne information is missing, e-Business Suite documentation applies

This chapter covers the following topics:

- "Open With" Worksheets
- Worksheet Filters
- Changing System Time Resolution
- Analytical Engine Guidelines
- DM application default users
- Controlling System and Engine Max Sales Dates

"Open With" Worksheets

"Open With" worksheets should be unfiltered. If you wish to show a filtered version of the worksheet, you will need to create a duplicate for "My Worksheets". If you place a filter on a worksheet to be used by "Open With", the "Open With" filter will be applied to the already filtered population which may not provide a result set. For example, if the worksheet is filtered to Member 1 of Level 1, and "Open With" is launched from Member 2 of Level 1, the result set will be null.

Worksheet Filters

The Demand Management worksheets have a default filter. This filter is to ensure that when first run in a large production environment, the worksheet will not attempt to run

on the entire data population. The filter added is pointing to the default members of all levels configured as aggregation levels in the worksheet. When implementing, go into all the worksheets and their embedded worksheets, change the filters to match the business process and scope. Remember that very large worksheets are typically not representative of one user's business process and will typically be accompanied by degradation in performance.

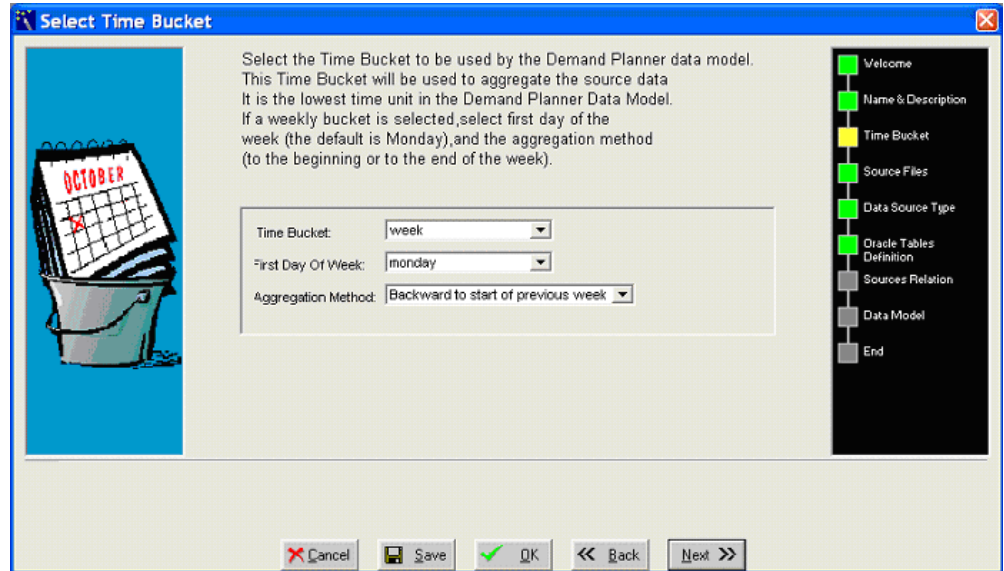
Changing System Time Resolution

Demantra uses a base time resolution. All other time displayed in the system is an aggregation of this base time resolution. The default time of the Demand Management application is weekly beginning on Monday. There may be several business reasons to change this:

- Starting your week on a different day.
- Aggregating the week based on the ending day not the beginning day.
- Daily or monthly base time.

To change the base time:

1. Access the Business Modeler.
2. From the Data Model menu, choose Open Data Model.
3. Select the data model you want to modify and click next until the Time Bucket screen appears.

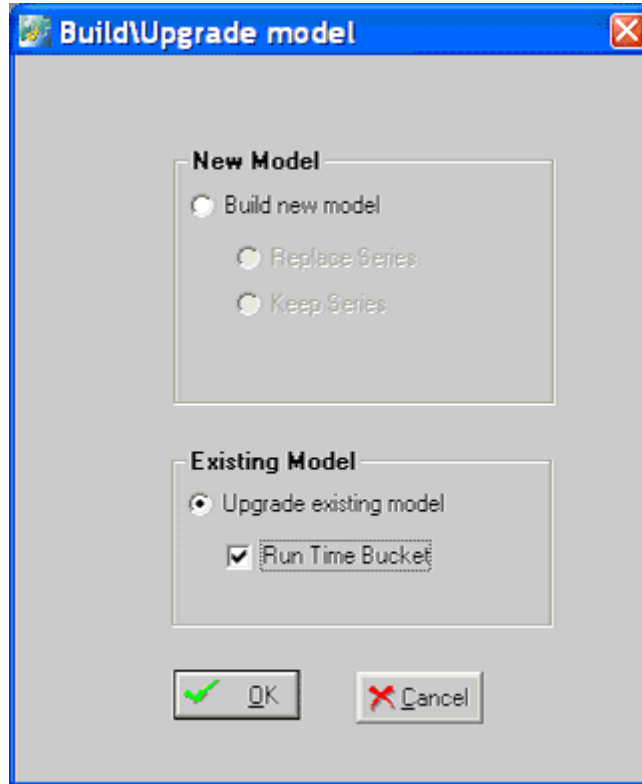


4. Complete the following fields:

- Time Bucket
- First Day of the Week
- Aggregation Method

Note: The day and month time unit do not designate the first day of the period. Months are assumed to begin on the first and end on the last day of the Gregorian month.

5. After your changes have been saved, the data model should be upgraded, not rebuilt using the Run Time Bucket option selected.



Note: If the time bucket is re-configured, the time aggregation set for all worksheets is modified to match the new time aggregation. A review of all used and embedded worksheets is strongly recommended.

Changing time resolution and engine parameters:

Many engine parameters set for a weekly system do not comprise best-practice setting in a monthly and daily system. A good source of default values can be found in `init_params_0_daily` and `init_params_0_monthly` tables. It is recommended that you review engine parameters and change time relevant parameters if you change the time bucket setting.

Parameter `MetricsPeriod` defines the length of history for which accuracy is calculated as an engine output. Default for weekly system is 26. A monthly system is set to 24 while a daily system is set to 60.

Analytical Engine Guidelines

The batch engine generates a new forecast for a system-wide population or a line of business. It uses distributed processing, analyzes very large amounts of data at night

and on the weekends when users are not logged into the system. By contrast, the simulation engine is used to generate or regenerate a forecast for a very specific population subset. Simulations can be run on an as needed basis, and several users may run simulations concurrently. Due to the large amount of processing used by the batch engine and the fact that it typically regenerates the entire forecast, the batch and simulation engine are not enabled to run at the same time.

The analytic engine outputs several accuracy metrics when running the batch engine. They are:

- MAPE
- BIAS
- MRE
- RMSE
- And a number of historical observations used to calculate these metrics.

The length of history serving as a basis for the first 4 metrics is set by INIT_PARAMS_0 parameter MetricsPeriod. This parameter defines the number of periods of history, starting with the last and moving backward when calculating the accuracy metrics. Since the accuracy metrics are based on comparison of a back-cast with history they can serve as indicators of likely accuracy but are not best practice calculations. Best practice calculations should be based on comparison of an archived forecast with actual history as it becomes available. These metrics are stored on table MDP_MATRIX and are generated by the engine at the level a node is forecast. This implies that nodes not receiving a forecast will not have these numbers and all MDP_MATRIX combinations under a specific node will have the same engine metric values.

Additional out-of-sample error calculations can be done using a stand alone error calculation process. For more details refer to Out of Sample Error Calculations, page 50-11.

DM application default users

The DM application is pre-configured with template users. These are necessary to allow system administrator access as well as set up the default process workflows and notifications.

User name	Password	Permission Level	Description
-----------	----------	---------------------	-------------

dm	dm	System Manager	Demand Management component owner. User necessary to add or delete users from component. Owns all DM workflows.
Analyst1	analyst1	Power	Typical system day-to-day user
Analyst2	analyst2	Power	Typical system day-to-day user
Analyst3	analyst3	Power	Typical system day-to-day user
Analyst4	analyst4	Power	Typical system day-to-day user
Analyst5	analyst5	Power	Typical system day-to-day user
Admin1	admin1	Power	Typical system administrator. Will typically oversee download and upload process and should be notified of any issues
Manager1	manager1	Power	Manager of analysts. Notified when exceptions occur in forecast approval process

Controlling System and Engine Max Sales Dates

When loading future dates in the EP_LOAD process, it is important to populate a control parameter to determine how you would like the end of history populated. The control parameter can be found in the Business Modeler and is called MaxSalesGen.

Populating MaxSalesGen:

1. Access the Business Modeler.
2. From the Parameters menu and choose System Parameter.
3. Click the System tab and scro down until you find the MaxSalesGen parameter.

Name	Value	Default Value
DSMPromotionBudgetSeries	3668	
DSMWriteOffThreshold	100	100
EnableIncrementalLoading	true	true
EnableWorksheetCaching	true	true
FIRSTDAYINWEEK	2	
ImportDataMode	1	1
Integration1CalendarCode	None	
Integration1CalendarLoad	APPS.MSD_DEM_TIME	APPS.MSD_DEM_TIME
Integration1CalendarReset	no	no
Integration1E1DaysLoaded	3000	40
LoadDataStop	yes	yes
mail_recipient	no send	no send

Description

Historical data integration - Does the system load delta data (accumulate) or summary data (update)

4. For the MaxSalesGen parameter, enter the value you want. Some considerations:

- Null. Leaving the parameter blank causes the system to continue to behave as it does today. The last date loaded into the system is compared to the current last system date, and the latest of the two set is the last date of history. It is recommended in cases where only historical dates are being loaded.
- Sysdate. Entering Sysdate as the parameter causes the last date of history to be based on the period containing today's date (date in the DB server). In a weekly system with weeks beginning Monday, if run on February 16, 2007, the last date of history is set to the previous Monday February 12, 2007. For a monthly system run on the same date, the end of history is set to February 1, 2007. This option is good for a production environment where the system date should match the current date while allowing future information to be loaded.
- 01-01-1900 00:00:00. Setting the parameter to this value sets the end of history to the last date in the sales_data table where the actual_quantity column>0. For very large systems, this could add time to loading availability. It is critical that the data used to drive the engine be stored in the actual_quantity column.
- Any date other than 01-01-1900 00:00:00. Any other date will cause the last date of history to be based on the entered date. In a weekly system with weeks beginning Monday, if the date entered is January 16, 2007, the last date of history would be set to the previous Monday January 15, 2007. For a monthly system run with the same parameter setting, the end of history would be set to January 1, 2007. This option is ideal for testing systems where the desired end of history date does not match the executed date. This allows users full control on dates assigned as end of history and beginning of forecast.

Note: All dates must be entered in the MM-DD-YYY 00:00:00 format.

Demantra Data Tables and Integration Processes

This chapter describes the Demantra table structure (at a high level) and gives the basic data requirements. It also describes ways you can import data into and export data from Demantra.

This chapter covers the following topics:

- Demantra Data Tables
- Data Assumptions and Requirements
- Loading Basic Data
- Loading Series and Promotions
- Executing Integration Interfaces
- Configure Series Load and Purge Options
- Loading Supplementary Data
- Exporting Data
- Configuration Notes

Demantra Data Tables

When you configure Demantra, it is not necessary to work directly with the database, except in minor ways such as creating standalone tables. Also, you should use the Demantra configuration tools as much as possible, to avoid making database changes that could damage your configuration. However, it is useful to have a general understanding of the Demantra table structure and its many interconnections.

First, Demantra stores most of its basic data in the following internal tables:

items	Contains one record for each lowest-level item, with fields to indicate the membership of that item in every higher item-type level.
location	Contains one record for each lowest-level location, with fields to indicate the membership of that item in every higher location-type level.
mdp_matrix	Contains one record for each known combination of lowest-level item and lowest-level location. This includes all combinations that have had sales, as well as any combinations created manually in the user interfaces. Matrix series select data from this table.
sales_data	Contains one record for each lowest-level item, lowest-level location, and date—for all combinations and dates where sales actually occurred. Sales series select data from this table.
promotion_data	Contains one record for each lowest-level item, lowest-level location, promotion, and date—for all combinations, promotions, and dates where promotions actually occurred. Promotion series select data from this table.
Inputs	<p>Contains one record for each date, through a date in the far future. You use this table to store purely time-dependent information, such as 4-4-5 calendars and global causal factors.</p> <p>The Inputs table extends through the year 2030 for internally maintained records. The columns representing the time aggregations are not populated. The Inputs table includes the following holidays as observed in the USA:</p> <ul style="list-style-type: none"> • Easter Monday • Easter Sunday • Father's Day • Mother's Day • Independence Day • Labor Day • Memorial Day • Thanksgiving Day • Christmas Day

Demantra also has the following tables:

- Several tables (promotion_dates, promotion_levels, and promotion_members) that indicate the combinations and dates to which each promotion applies.
- A table (group_attributes) that lists all the levels defined in Demantra.
- One table for each level, which lists the members of that level and their attributes. If you have defined series on this level, data for that series is also stored here.
- A table that describes all the series defined in Demantra. For each series, this information includes details such as the table from which the series data should be retrieved.
- Tables that describe causal factors that the Analytical Engine should use.

As you can see, you should never directly edit any of these tables. These tables are interconnected and must be kept synchronized whenever new data is loaded or whenever structural changes are made.

You can, however, add your own tables to use in drop-down series or other purposes.

Data Assumptions and Requirements

Demantra requires input data that describes the items, the locations at which the items are sold, and all sales of those items, at each location over time. It is important to understand the Demantra requirements for this data.

Lowest-Level Item Data

Demantra requires the following fields for each record in the item data:

- A unique code that can be used as the identifier for any lowest-level item.
- A unique description, which is a short string that serves as a user-friendly name of the item. If no description is available, use the code.
- Additional codes that indicate the membership of this item within all levels of the item hierarchy. See "Member Identifiers".
- Additional fields that describe this item, as needed.
- Additional fields that specify unit conversion factors for this item, if needed. See "Unit Conversion Data".

Also, it is useful to be able to display quantities in different units of measure. The default unit of measure is called units, which simply counts the number of individually packaged product units. Sometimes you need to convert quantities to another unit such

as case, carton, or truckload. In order to do so, you need a conversion factor, and that can be different for different items. This means that the item data usually also includes unit conversion factors.

Lowest-Level Location Data

Demantra requires the following fields for each record in the location data:

- A unique code that can be used as the identifier for any lowest-level location.
- A unique description, which is a short string that serves as a user-friendly name of the location. If no description is available, use the code.
- Additional codes that indicate the membership of this location within all levels of the location hierarchy. See "Member Identifiers" .
- Additional fields that describe this location, as needed.

Sales Data

Demantra requires the following fields for each record in the sales data:

- The unique code of the item being sold.
- The unique code of the location of the sale.
- The date of the sale.
- The number of units that were sold for this item, location, and date. This field must be numeric. See "Unit Conversion Data".
- Price per unit for this item, at this location and date. This field must be numeric. "Unit Conversion Data".
- Additional fields as needed.

Aggregation in Time

You must choose the smallest time unit that you will use within Demantra. Correspondingly, you must also specify the start of that time unit (such as the starting day of the week) and an aggregation method for Demantra to use when importing data (backwards or forwards).

However, it is not necessary to pre-aggregate the raw data in time. The Demantra loading and integration tools can perform that aggregation if needed. That is, if you import multiple sales records for different dates for the same item-location combination, Demantra automatically sums them up into the time unit to which those dates belong.

Note: Together, the item, location, and date will form the primary key for the sales record. That is, Demantra stores no more than one record for each combination of item, location, and date.

Data Denormalization

As you build the data model, you will probably import data from multiple sources in the enterprise. Some of these sources probably store data in a normalized manner. For example, one table would store the relationship between a product group and the product family, and another table would store the relationship between a product family and the marketing class.

Before you import data into Demantra, you will need to denormalize the data and get it into the flattened formats described in this section.

Loading Basic Data

To load the basic data, you use the Data Model Wizard, which helps you describe the location, format, and structure of your raw data.

The Raw Data

Before you can build a Demantra data model, you must have some sample data. You then use the Data Model Wizard to describe that data so that the system can load it.

This data can be in the form of either text files or database tables:

- If you use text files, the files must be either comma-delimited or tab-delimited.
- If you use database tables, you must create these tables before you start the Data Model Wizard. These tables must be within the same database user name as the Demantra database.

The Data Model Wizard assumes that you have one, two, or three source tables or files as follows:

Number of sources	First source	Second source	Third source
1	sales, locations, and items		
2	sales and locations	items	
3	sales	locations	items

What the Data Model Wizard Does

The Data Model Wizard prompts you for the location and format of the raw data, as well as the number of sources. If you have two or three sources, the wizard prompts you for details on how to join them.

Then if your sources are text files, the wizard helps you map them into staging tables. If your sources are database tables, your tables are the staging tables. In either case, the wizard helps you describe the contents of the staging tables so that you can build a model on them.

You specify how to use the fields in the staging tables, generally using each field in a level definition or in a series definition. Demantra ignores any field for which you do not specify a use.

As a final result, the Data Model Wizard creates a batch script and database procedures. The script executes the procedures, which load the data into the Demantra internal tables and synchronize the tables as needed.

Loading Series and Promotions

To load series and promotions, you use the Integration Interface Wizard, which provides a flexible way to import data into Demantra. (It also can be used to export data; see "Exporting Data".)

Integration Interfaces

Within the Integration Interface Wizard, you create integration interfaces. An integration interface consists of at least one of the following:

- A data profile, which specifies how to import Demantra series, at the aggregation levels that you choose. You can import sales series, promotion series and other level series, but not matrix series.
- A level profile, which specifies how to import promotions and their attributes.

Note: When you import promotions, any existing promotions are not changed.

Executing Integration Interfaces

Once you have created an integration interface, you can execute it in either of two ways:

- You can incorporate the integration interface in a workflow controlled by the Workflow Manager. See "Overview of Workflow Step Types".

- You can use the separate Standalone Integration Tool, which is Demantra_root/Demand Planner/Integration/aps.bat. (This tool consists of a subset of the APS, packaged as an executable file.)

API to Modify Integration Interface in Workflow Description

When integrating data from a source instance into Demantra, there may be a need to modify an existing integration interface. For example: a pre-seeded integration profile is defined to import ten price lists by item. It is desired to use this pre-seeded integration profile, but to only import three of price lists. To accomplish this, the price list import workflow is triggered by collecting 3 price lists in APS Collections, modifying the integration interface such that the remaining 7 price list series are set to 'No load' and 'No purge,' and then running the transfer step.

The API provides control over the following data profiles:

- Load options:
 - Override
 - Accumulate, or
 - No load
- Purge options:
 - No purge
 - All dates
 - All dates within a time range
- Time definitions
- Time Filter:
 - Relative
 - Fixed
- Date Range:
 - From
 - Until

Setup

Notify the Application Server that you've modified Integration Profiles so it refreshes its cache.

```
API_NOTIFY_APS_INTEGRATION(ii_profile_id INT)
```

The following procedural APIs can be leveraged within the workflow to modify a pre-defined integration interface:

This is the API to modify an Integration Profile's Series Attributes:

```
API_MODIFY_INTEG_SERIES_ATTR  
(ii_profile_id INT, ii_series_id INT, ii_load_option INT,  
ii_purge_option INT)
```

Where,

- ii_profile_id is transfer_query.id
- ii_series_id is transfer_query_series.series_id (and also computed_fields.forecast_type_id)
- ii_load_option is:
 - "0" Override
 - "1" Accumulate
 - "2" No Load
- ii_purge_option is:
 - "0" No Purge
 - "1" Purge All
 - "2" Purge Within

This is the API to modify an Integration Profile's Fixed Date:

```
API_MODIFY_INTEG_SERIES_FDATE  
(ii_profile_id INT, id_from_date DATE, id_until_date DATE)
```

Where,

- ii_profile_id is transfer_query.id
- id_from_date is transfer_query.from_date
- id_until_date is transfer_query.until_date

This is the API to modify an Integration Profile's Relative Date:

```
API_MODIFY_INTEG_SERIES_RDATE
(ii_profile_id INT, id_rel_from_date INT, id_rel_until_date INT)
```

Where,

- id_rel_from_date is transfer_query.relative_from_date
- id_rel_until_date is transfer_query.relative_until_date

To configure and deploy APS on Unix/Linux:

For this procedure, please refer to the latest Demantra Installation Guide.

Maintaining Data Consistency

If you import data using actual proportions (rather than matrix proportions), be sure to run the MANUALS_INS_INTEGRATION procedure after you run the integration interface. (Note that you can run database procedures from a workflow.)

Depending on your database, it may be necessary to run additional database scripts.

API to Create, Modify or Delete Users

The API_CREATE_ORA_DEM_USER command is available to create, modify or delete users from the Demantra application using the command prompt or batch routine. In addition, you can add level filters to an existing user using the API_ADD_ORA_DEM_LEVEL_FILTERS command.

This is the API to create, modify or delete users:

```
API_CREATE_ORA_DEM_USER
(User name, Password, Permission, First name, Last name, Company, Phone
number, Fax number, Email, Language, User group, Component, Series
group, Update action)
```

Where,

- User name is the Demantra user name (for example, jsmith)
- Password is the user's password (for example, pw1234).
- Permission is the user's permission level (for example, DP).

Possible permission categories are:

Category	Description	Product Modules
DM	Demand Manager (casual user)	2, 3, 4

Category	Description	Product Modules
DMA	Demand Management Administrator	1, 2, 3, 4
PTP	Predictive Trade Planning user	2, 3, 4, 6
PTPA	Predictive Trade Planning Administrator	1, 2, 3, 4, 6

Product modules are:

- 1 - Demantra Administrative Tools
- 2 - Demantra Demand Planner
- 3 - Demantra Demand Planner Web
- 4 - Demantra Collaborator Workbench
- 6 - Demantra Promotion Effectiveness

- First name is the user's first name (for example, John).
- Last name is the user's last name (for example, Smith).
- Company is the company name (for example, ABC Ltd.).
- Phone number is the user's phone number (for example, 012 3456789).
- Fax number is the user's fax number (for example, 012 1111111).
- Email is the user's email address (for example, jsmith@ABC.com).
- Language is the language preference of the user (for example, 0=English).
- User group is the group to which the user is assigned (for example, p_portal).
- IS_COMPONENT is the product component (for example, 156=Demand Management)
- Series group provides the overriding allocation of series available to the user (for example, 204=Sales)
- Update action indicates whether the user is added, modified or deleted (for example, ADD)

Options include:

ADD - Used only to add users. An error message appears if the user already exists.

UPDATE - Used to update existing user details.

REPLACE - If the user already exists, then the user along with all related data is first deleted. Then the process uses the ADD method and the user is recreated. All parameters must have values. The three id values are all integer values.

The User name and Password parameters are mandatory.

Running the API_CREATE_ORA_DEM_USER procedure from the command prompt:

```
EXEC API_CREATE_ORA_DEM_UESR ('jsmith', 'pw1234', 'DP', 'John', 'Smith',  
'ABC Ltd.', '012 3456789', '012 11111111', 'jsmith@ABC.com', '0',  
'p_portal', '4', NULL, 'ADD')
```

Running the API_CREATE_ORA_DEM_USER procedure from a script:

```
DECLARE
  IS_USER_NAME VARCHAR2(200);
  IS_PASSWORD VARCHAR2(200);
  IS_USER_PERMISSION VARCHAR2(200);
  IS_FIRST_NAME VARCHAR2(200);
  IS_LAST_NAME VARCHAR2(200);
  IS_COMPANY VARCHAR2(200);
  IS_PHONE_NUM VARCHAR2(200);
  IS_FAX_NUM VARCHAR2(200);
  IS_EMAIL VARCHAR2(200);
  IS_LANGUAGE VARCHAR2(200);
  IS_USER_SECURITY_GROUP VARCHAR2(200);
  IS_COMPONENT VARCHAR2(200);
  IS_SERIES_GROUP VARCHAR2(200);
  IS_UPDATE_ACTION VARCHAR2(200);
BEGIN
  IS_USER_NAME := 'jsmith';
  IS_PASSWORD := 'pw1234';
  IS_USER_PERMISSION := 'DP';
  IS_FIRST_NAME := 'John';
  IS_LAST_NAME := 'Smith';
  IS_COMPANY := 'ABC Ltd.';
  IS_PHONE_NUM := '012 3456789';
  IS_FAX_NUM := '012 1111111';
  IS_EMAIL := 'jsmith@ABC.com';
  IS_LANGUAGE := '0';
  IS_USER_SECURITY_GROUP := 'p_portal';
  IS_COMPONENT := '4';
  IS_SERIES_GROUP := '1';
  IS_UPDATE_ACTION := 'REPLACE';

  API_CREATE_ORA_DEM_USER(
    IS_USER_NAME => IS_USER_NAME,
    IS_PASSWORD => IS_PASSWORD,
    IS_USER_PERMISSION => IS_USER_PERMISSION,
    IS_FIRST_NAME => IS_FIRST_NAME,
    IS_LAST_NAME => IS_LAST_NAME,
    IS_COMPANY => IS_COMPANY,
    IS_PHONE_NUM => IS_PHONE_NUM,
    IS_FAX_NUM => IS_FAX_NUM,
    IS_EMAIL => IS_EMAIL,
    IS_LANGUAGE => IS_LANGUAGE,
    IS_USER_SECURITY_GROUP => IS_USER_SECURITY_GROUP,
    IS_COMPONENT => IS_COMPONENT,
    IS_SERIES_GROUP => IS_SERIES_GROUP,
    IS_UPDATE_ACTION => IS_UPDATE_ACTION);
END;
```

This is the API to add level filters to an existing user:

API_ADD_ORA_DEM_LEVEL_FILTERS
(User name, Level Id, Member Id, Permission Id)

Where,

- User name is the Demantra user name (for example, jsmith).
- Level Id is the internal level id value (for example, 6)

- Member Id is the internal member id value (for example, 1)
- Permission id is the permission id value (for example, 4)

Permission id value options:

- 1 - Not Visible
- 2 - Read Only
- 3 - Read & Write
- 4 - Full Control

Permission id values are derived from the table SECURITY_PERMISSION.

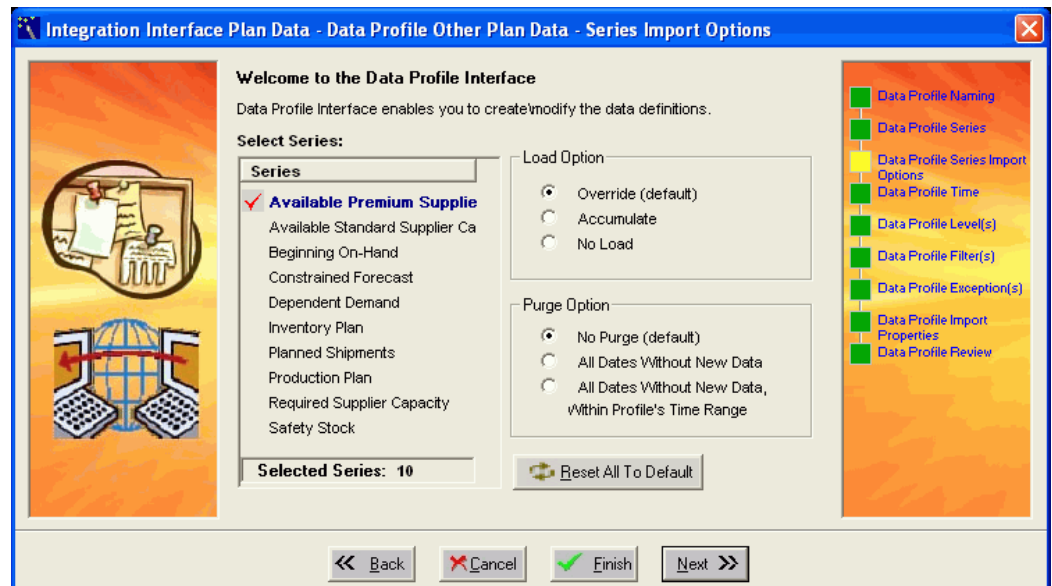
Running API_ADD_ORA_DEM_LEVEL_FILTERS from the command prompt

```
EXEC API_ADD_ORA_DEM_LEVEL_FILTERS ('jsmith', 6, 1, 4);
```

Configure Series Load and Purge Options

The Integration Interface provides the ability to set purge and load options for each series. This controls whether the import integration profile overrides, accumulates, or purges (nulls out) the data for the specified integration profile date range.

The location of the Purge Data Before Import profile option within the integration profile wizard is shown:



Selected Series: This list box displays the selected series in the profile that were checked on the Data Profile Series page.

Load Option: The selected radio button indicates the load option for the series that is adjacent to the red check mark. The available load options are Override, Accumulate, or No Load. The default selection is Override.

Purge Option: The selected radio button indicates the purge option for the series that is adjacent to the red check mark. The default selection is No Purge.

Reset All: This button resets all series that appear in Selected Series list box to the default settings.

Note: To maintain backwards compatibility with old profiles, when upgrading:

- Set the Load Option to Override, and
- Set the Purge Option to No Purge.

Load Options

Setting	Description
Override	Override values on existing dates
Accumulate	Add values from the profile data to the values on existing dates
No load	Don't load data

Purge Options

Setting	Description
No purge	Do not purge
Purge (null out) all data on existing dates	Purge (null out) all data on existing dates
Purge (null out) all data on existing dates	Purge (null out) all data on existing dates

Load and Purge Option Combinations

Settings	Option Explanation	Results
Override, No Purge	Override values on existing dates. Do not purge.	Loads data for series in the date range of the profile. Overrides values on existing dates. Inserts the dates and values that are in the profile data, but not in the system.
Accumulate, No Purge	Add values from the profile data to the values on existing dates. Do not purge.	Loads data for series in the date range of the profile. Adds the values from the profile data to the values on existing dates. Inserts the dates and values that are not in the system.
No Load, No Purge	Don't load data. Do not purge.	Does nothing to this series
Override, All Dates Without New Data, Within Profile's Time Range	Override values on existing dates. Purge (null out) data within profile dates.	Loads data for series in the date range of the profile. Overrides values on existing dates. Purges (null out) values for dates in the range of the profile that are not in the loading data.
Accumulate, All Dates Without New Data, Within Profile's Time Range	Add values from the profile data to the values on existing dates. Purge (null out) data within profile dates.	Loads data for series in the date range of the profile. Adds the values from the profile data to the values on existing dates. Purges (null out) values for dates in the range of the profile that are not in the loading data.
No Load, All Dates Without New Data, Within Profile's Time Range	Don't load data. Purge (null out) data within profile dates.	Purges (null out) all values in the system within the time range of the profile.

Settings	Option Explanation	Results
Override, All Dates Without New Data	Override values on existing dates. Purge (null out) data within profile dates.	<p>Loads data for series in the date range of the profile.</p> <p>Overrides values on existing dates.</p> <p>Inserts the dates and values that are in the profile data, but not in the system.</p> <p>Purges (null out) values for all dates that are not in the loading data.</p>
Accumulate, All Dates Without New Data	Add values from the profile data to the values on existing dates. Purge (null out) data within profile dates.	<p>Loads data for series in the date range of the profile.</p> <p>Adds the values from the profile data to the values on existing dates.</p> <p>Inserts the dates and values that are not in the system.</p> <p>Purges (null out) values for all dates that are not in the loading data.</p>
No Load, All Dates Without New Data	Don't load data. Purge (null out) data within profile dates.	Purges (null out) all values for all dates in the system.

Caution: Only combinations with some data in the integration profile will be purged. Combinations with no data will not be purged or overridden.

Loading Supplementary Data

To load other data, such as lookup tables, you use the Demantra import tool (Tools > Import File).

In contrast to the Integration Interface Wizard, this tool does not load the data into Demantra internal tables; it is inappropriate for importing series or levels. Nor does it provide a way to export data.

Unlike the Data Model Wizard, this tool does not create the tables into which you are importing data. You must first create the tables.

Import Interfaces

Within the import tool, you create import interfaces. An import interface consists of at least one profile. Each profile corresponds to one table; note that multiple files can be loaded into a single table.

The import tool creates a batch script that executes the import interface.

Executing Import Interfaces

To execute an import interface, you run the corresponding batch script. If the data needs to be loaded only once, you can run the script manually. If the data needs periodic refreshing, you can run the batch script from a workflow controlled by the Workflow Manager. See "Overview of Workflow Step Types".

Exporting Data

To export series and level members, you use the Integration Interface Wizard, which is introduced in "Loading Series and Promotions". When you define an interface, you specify how that interface will be used: for import, for export, or for both import and export. You can execute the interface in a couple of ways; see "Executing Integration Interfaces".

The Integration Interface Wizard provides slightly different functionality for export than for import:

- You can export sales series, promotion series and other level series, but not matrix series.
- You can export any kind of level, not just general levels.
- You can export members and attributes of a general level, but you cannot export the population attributes of the members. (The population attributes specify the item-location combinations to which each promotion applies.)

Note that an export profile creates a database view, and the data in that view is then exported to the specified export file. The view is created when you run the export process, not before.

Also note that if you want to export a series that uses a client expression, you must first run the Business Logic Engine to evaluate the expression, split the resulting data to the lowest level, and save it to the database. You can run the Business Logic Engine from within a workflow; see "Overview of Workflow Step Types".

Configuration Notes

This section contains configuration notes related to dependencies.

Dependencies

Before you can set up integration, you will need sample data.

Tools

The following table summarizes the core Demantra import and export tools:

Data	To import, use...	To export, use...
Lowest level item and location data; sales data	Data Model Wizard*	N/A
Series data at any aggregation level	Integration Interface Wizard*	Integration Interface Wizard
Sales promotions	Integration Interface Wizard*	Integration Interface Wizard
Members and attributes of other levels	N/A	Integration Interface Wizard
Other data, for example, lookup tables	Demantra import tool*	N/A
*These options are in the Business Modeler.		

You can also use the Demantra Enterprise Integrator (powered by Pervasive).

Demantra Demand Management to EBS Integration

This chapter overviews integration processes that synchronize or move data between the Oracle Demantra and E-Business Suite applications.

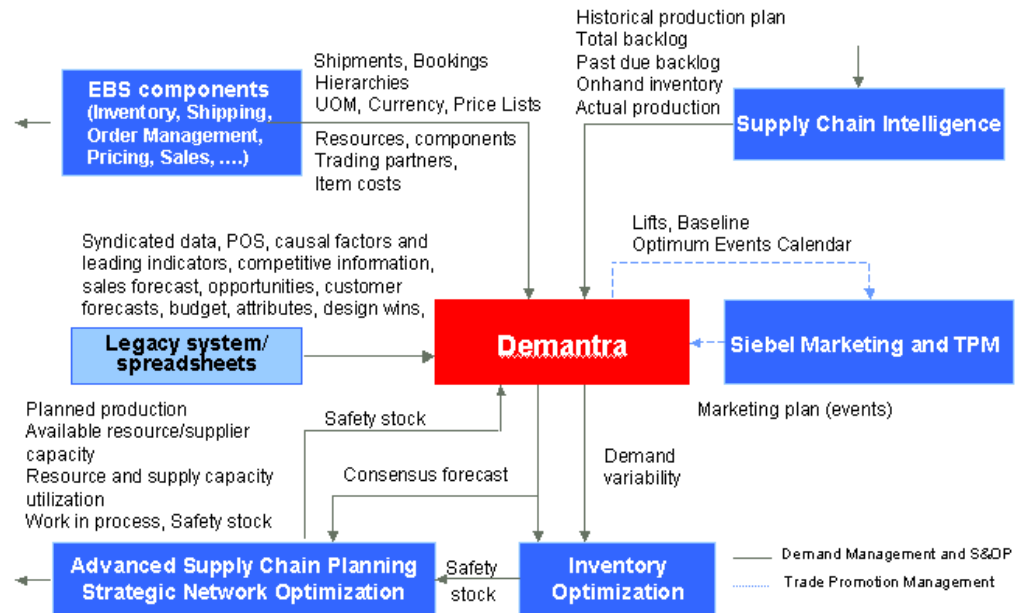
This chapter covers the following topics:

- Demand Management Business Flow
- Terms and Conventions Used in this Document
- Integration Features
- Seeded Demand Management Component
- Summary of Integration Tasks
- Initial Setup
- Download E-Business Suite Data Into Demand Management
- Download Collections
- Combined Collections of Shipment and Booking History
- Collecting Legacy Shipment and History Data
- Collecting Returns History Data
- Collecting Currency Conversion Data
- Associating multiple Price Lists with a CTO worksheet
- Collecting Unit of Measure Conversion Data
- Collecting Price List Data
- Downloading Calendars
- Purging Data Before Import
- Configure Series Load and Purge Options
- Download to Oracle Demantra

- Demand Management Functional Output
- Upload from Oracle Demantra
- Publish the Forecast to CP
- Upload Forecast
- Line Of Business Configuration and Execution
- LOB Procedures
- Troubleshooting the Line of Business Forecast Workflow
- Setting Up the New Products List
- Setting Up the Calendar List
- Base Time Unit
- Creating a New Leaf Level
- Creating a New Top Level
- Creating a New Intermediate Level
- Deleting a Level
- Approval and Upload Setup Process
- Profile Options

Demand Management Business Flow

The Integration Data Flow diagram shows, at a general level, the sources and destinations of data.



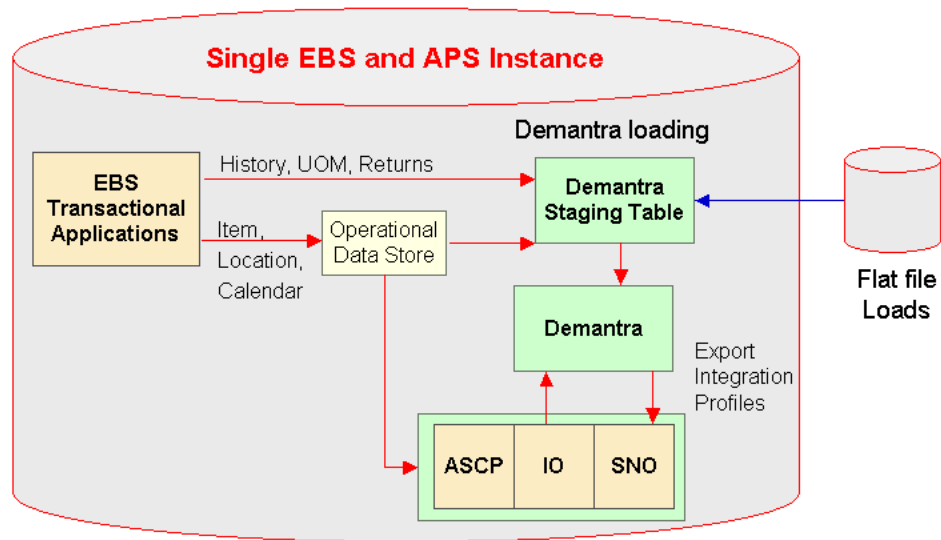
For details describing user business flow details, see the *Oracle Demantra Demand Management User's Guide*.

Supported Integration Configurations

Integration between Oracle Demantra Demand Management and the E-Business Suite leverages Oracle Demantra Foundation functionality to the extent possible. Booking history, price list, currency, calendars, users, and items collected from the E-Business Suite applications are loaded into Oracle Demantra. Forecasts and accuracy measures return.

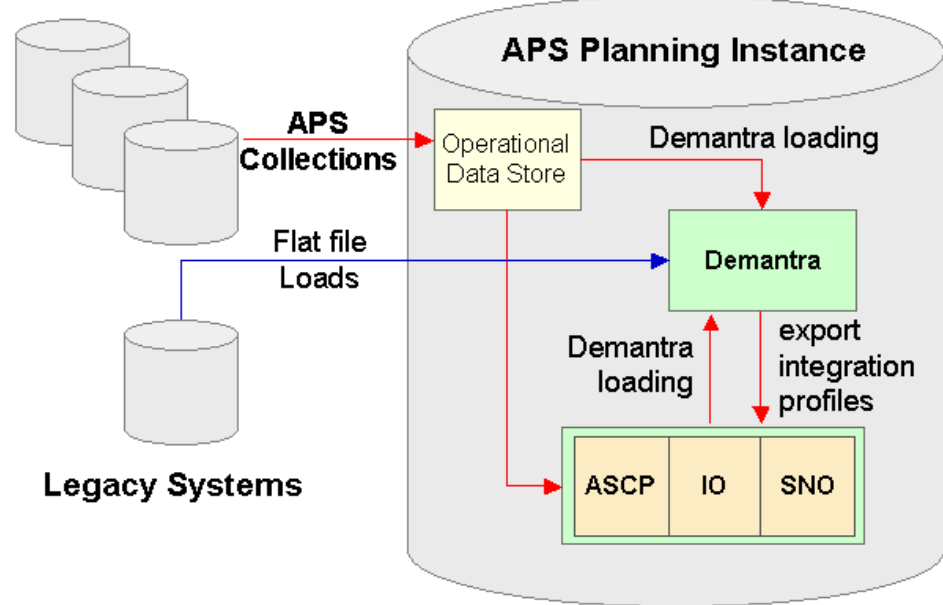
Oracle supports the following configurations:

- **Single instance.** The single instance would include ERP, Advanced Planning and Scheduling, and Oracle Demantra. It must be a supported combination of E-Business Suite and Oracle Demantra releases. Please contact support for an up-to-date list of certified versions for integration.



- **Separate destination and non-legacy source instances.** The destination instance and the source instance must both be Demantra-certified combinations of ERP and Advanced Planning and Scheduling.

EBS Source Instances



- **Separate destination and legacy source instances.** The destination instance must be a Demantra-certified version of Advanced Planning and Scheduling.

Terms and Conventions Used in this Document

Levels control how data is aggregated and organized. Levels are used in worksheets, in filters, in import and export, and in forecasting. A level *member* refers to a unit within a level. For example, "tollhouse" is a member of a level named "cookies". A *hierarchy* organizes levels into ranks. The top level in the hierarchy provides the most aggregate, general view of information. The bottom level provides the most disaggregate, specific view of information. Your application can include multiple, independent hierarchies. Each hierarchy can contain as many levels as needed.

Within Oracle Demantra, you generally apply a *filter* by specifying a level and the members of that level that you want to display in a worksheet.

A *worksheet*, sometimes known as a query, is the primary user interface to Oracle Demantra data. For example, within a worksheet, a user can examine and edit data as needed, view the forecast, run simulations, and save changes back to the database.

Our use of the terms *download* and *upload* here are always relative to the E-Business Suite, or a similar legacy system. In other words, *download* procedures move information *from* an E-Business Suite application, while *upload* procedures move information from Oracle Demantra *to* the E-Business Suite, or a legacy system.

When we discuss the *source* we are referring the E-Business Suite Enterprise Resource

Planning applications. *Destination* refers to the Advanced Planning and Scheduling applications.

Note: Oracle supports legacy collections for level members and history. The user must define and apply Oracle Demantra import integration profiles.

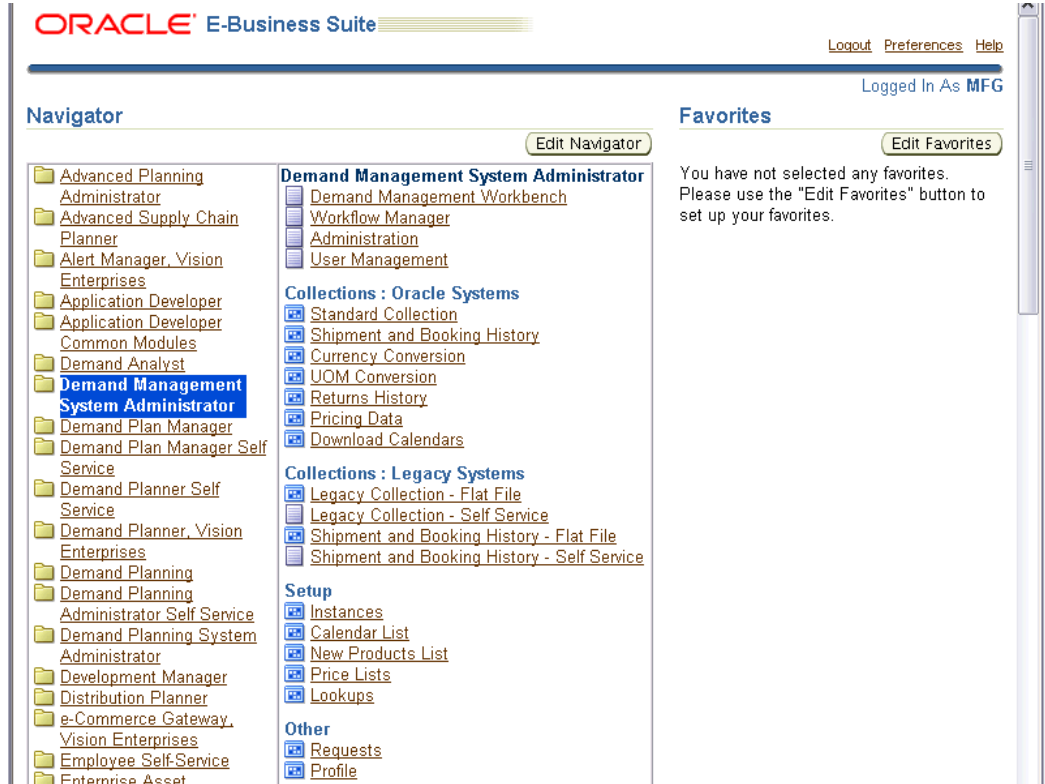
Integration Features

Demand Management Navigator Menus

The Oracle E-Business Suite Navigator provides the following two responsibilities:

- Demand Management System Administrator
- Demand Analyst

Note: EBS users can have both Demand Management and Sales and Operations Planning responsibilities, and therefore access both Demantra Demand Management and Sales and Operations Planning.



The Oracle E-Business Suite Navigator menu for the Demand Management System Administrator Responsibility provides the following links to integrate with the respective Oracle Demantra functionality:

- Demand Management System Administrator > Demand Management Workbench – opens the Oracle Demantra Collaborator Workbench user interface
- Demand Management System Administrator > Workflow Manager – opens the Oracle Demantra Workflow Manager user interface
- Demand Management System Administrator > Administration – accesses the Oracle Demantra Administration page
- Demand Management System Administrator > User Management – accesses the Oracle Demantra User Management page
- Collections: Oracle Systems - accesses collections programs to obtain data entities from the E-Business Suite, Advanced Supply Chain Planning, Operational Data Store applications with an option to download data into Oracle Demantra:
 - Standard Collections
 - Shipment and Booking History

- Currency Conversion
- UOM Conversion
- Returns History
- Pricing Data
- Download Calendars
- Collections: Legacy Systems: > Shipment and Booking History - Flat File – allows access to legacy shipment and booking history flat file, if applicable
- Setup > Instances - Allows setting up multiple Instances from where collections can be run to obtain data entities.
- Setup > Calendar List - allows setting up calendars to be downloaded into Oracle Demantra
- Setup > New Products List – allows setting up new products to be downloaded into Oracle Demantra
- Setup < Price Lists

The Oracle E-Business Suite Navigator menu for the Demand Analyst Responsibility provides the following link:

- Demand Analyst > Demand Management Workbench – opens the Oracle Demantra Collaborator Workbench user interface

User Synchronization

When an E-Business Suite user is granted any responsibility containing the Demand Management Workbench (MSD_DEM_DEMPLANR) function grant, an Oracle Demantra user of the same username is created in the Oracle Demantra Demand Management component.

If the E-Business Suite user additionally has the Setup > Instances (MSD_DEM_DEMADMIN) function grant, the corresponding Oracle Demantra user has the following Oracle Demantra function security grants:

- Business Modeler
- Run batch engine
- Run workflow that launches Oracle Demantra EP_LOAD
- Run workflows that launch import and export integration profiles

- Run workflow that archives current Final Forecast for waterfall analysis

If the E-Business Suite user does not have the MSD_DEM_DEMADMIN function grant, the corresponding Oracle Demantra user has none of the previously listed Oracle Demantra function security grants.

As E-Business Suite users' function grants change over time, the corresponding Oracle Demantra users' function grants automatically change to match. For example, if at any time the E-Business Suite user loses the Demand Management (MSD_DEM_DEMPLANR) E-Business Suite function grant, the corresponding Oracle Demantra user is deleted.

Important: If the user is a customer contact, then restrict the contact's Oracle Demantra data security scope to that customer in the customer class hierarchy.

Users assigned E-Business Suite Demand Management System Administrator or Demand Analyst responsibilities are automatically assigned mirrored responsibilities in Oracle Demantra. When a user is created in the E-Business Suite and mirrored in Oracle Demantra, the default password is a randomly-generated sequence of 10 characters. A Demantra System Administrator/component owner must then go into the Business Modeler and change that user's password before the user can access Demantra.

Single Sign-on (SSO)

Single Sign-on means that users who log into the E-Business Suite can access the Oracle Demantra system without requiring an additional login to Oracle Demantra. When users log out from Oracle Demantra they are also logged out from the E-Business Suite. For logout purposes, Oracle Demantra invokes E-Business Suite logout procedures. More information can be found in the *E-Business Suite SSO Developer's Guide*.

SSO Process

The Single Sign-on process in E-Business Suite is managed via a *mod_osso* plug-in on the HTTP server. Basically, it receives a request to access an application and makes sure that the current user is authenticated with the Oracle SSO Server. On the Oracle Demantra side, the SSO process consists of getting a user name and forwarding it to an appropriate login page.

1. From the E-Business Suite Home Page Navigator, the User clicks an Oracle Demantra responsibility:
 - Demand Management System Administrator
 - Demand Analyst
2. Oracle Demantra Login JSP obtains the user information cookie, and then initializes the session. Depending on user role, Oracle Demantra offers up to four Single

Sign-on enabled pages to log the user into an appropriate application:

- Demand Management Workbench
 - Workflow Manager
 - Administration
 - User Management
3. The user selects an application, and is redirected to the single sign-on server.
After verifying credentials in Oracle Internet Directory, the server passes these credentials on to the Oracle Demantra application.
 4. The application serves up the requested content.

SSO Setup

There are two different setups based on whether Oracle Demantra is deployed into the same Application Server as the E-Business Suite:

- Oracle Demantra Deployed Together with E-Business Suite
- Oracle Demantra Deployed Separate from E-Business Suite

Oracle Demantra Deployed Together with E-Business Suite.

Assumptions: SSO server and Oracle Internet Directory (OID) are available with E-Business Suite and can be used by Oracle Demantra without requiring new licenses for SSO and OID.

After authenticating the user, mod_osso transmits the header values that iAS applications require to validate the user. These include the following:

1. User name - User nickname as entered by user on Single Sign-On login page
2. User DN - Single Sign-On user's distinguished name
3. User GUID - Single Sign-On user's globally unique user ID (GUID)
4. Language and territory - User selects Language and Territory on the login page

To configure the application to use mod_osso for SSO, the following lines need to be added in the mod_osso.conf file in the IfModule tag:

```
<Location/MyLogin>
    require valid-user
    authType Basic
</Location>
```

where /MyLogin is the mapping URL (context root).

The Mod_osso.conf file can be located in <Ora10iAS_home>/Apache/Apache/conf.

There should be one configuration block per responsibility in E-Business Suite (or login page in Oracle Demantra). Currently, there should be four similar entries pointing to Collaborator Workbench, Workflow Engine, Administrator, and User Management login pages.

These values are transmitted in HTTP request and can be extracted as following:

```
//User name as entered in EBS SSO
String userName = request.getRemoteUser();
//Osso-User-Dn
request.getHeader("Osso-User-Dn");
//Osso-User-Guid
request.getHeader("Osso-User-Guid");
```

Oracle Demantra Deployed Separate from E-Business Suite.

If Oracle Demantra is deployed into a different Application Server instance from E-Business Suite, then the mod_osso plug-in should be configured to serve Oracle Demantra via configuration files (mod_osso.conf). If Oracle Demantra is not deployed into an iAS, then mod_osso plug-in needs to be installed on the relevant HTTP server. This latter case requires an additional license for the plug-in.

The Oracle Demantra login URL needs to be registered with the Oracle application server (ssoreg.sh). Such registration is a one time activity.

Once this is done, the process to enable SSO is the same as described previously.

Seeded Demand Management Component

The seeded Demand Management component contains the default owning user levels, organized as hierarchies; series; and workflows required for the demand management business functions. The default owning User ID / Password for the Demand Management component is 'dm' / 'dm'.

Seeded Levels and Hierarchies

Oracle Demand Management provides several seeded levels organized into several seeded hierarchies:

Item levels: hierarchy roll-up sequence

- *Product Category:* Item > Category > All
- *Product Family:* Item > Product Family > All
- *Demand Class:* Demand Class > All
- *Resources:* > Resource Group > All

Location level: hierarchy roll-up sequence

- *Zone:* Site > Trading Partner Zone > Zone > All

- *Geography*: Site > Region > Country > Area > All
- *Trading Partner Class*: Site > Trading Partner > Trading Partner Class > All
- *Ship From*: Organization > Operating Unit > Legal Entity > Business Group > All
- *Business Group*: Organization > Operating Unit > Business Group > All
- *Legal Entity*: Organization > Legal Entity > All
- *Sales Channel*: Sales Channel > All
- *Customer Class*: Site > Account > Customer > Customer Class

Time:

- *Manufacturing Calendar*: Day > Week(calendar_id) > Period(calendar_id) > All
- *Gregorian Calendar*: Day > Month > Quarter > Year > All
- *Fiscal Calendar*: as collected from the E-Business Suite

Note: This set of notes applies to Manufacturing Calendars and Fiscal Calendars, but not Gregorian Calendars.

- Dynamically construct a separate hierarchy for each collected calendar. See Dynamic Creation of Calendar Hierarchies.
- If the base time unit is set to 'week', then the hierarchy is Week (calendar_id) > Period (calendar_id)
- If the installed base time unit is set to 'week', then only those Manufacturing Calendars with matching week definitions are collected.
- Manufacturing and Fiscal Calendars are not supported if the base time unit is set to 'month'

See the "Levels" chapter.

Seeded Series

A *series* is a set of data that represents some value that varies over time or that varies between item-location combinations, or most commonly, that varies in both ways. A worksheet displays the series data in a table, or in a graph, or both. You can generally view data for any given series at any aggregation level. The definition of the series controls how the data is aggregated.

See the "Series" chapter.

- Booking History - Booked Items – Booked Date
- Booking History – Requested Items – Booked Date
- Booking History – Booked Items – Requested Date
- Booking History – Requested Items – Requested Date
- Shipment History – Shipped Items – Shipped Date
- * Shipment History – Requested Items – Shipped Date
- Shipment History – Shipped Items – Requested Date
- Shipment History – Requested Items – Requested Date
- Return History

* Shipment History - Requested Items - Shipped Date is the default series for the base forecast and historical demand.

Loaded level for all seeded series:

- Product: Item
- Demand Class: Demand Class
- Organization: Organization
- Geography: Site
- Channel: Sales Channel
- Time: Week

The seeded default values are null.

Seeded Workflows

Oracle provides the following out-of-the-box workflows:

- Seeded workflows to run downloads using EP_LOAD for:
 - Items
 - Locations
 - History (sales data)

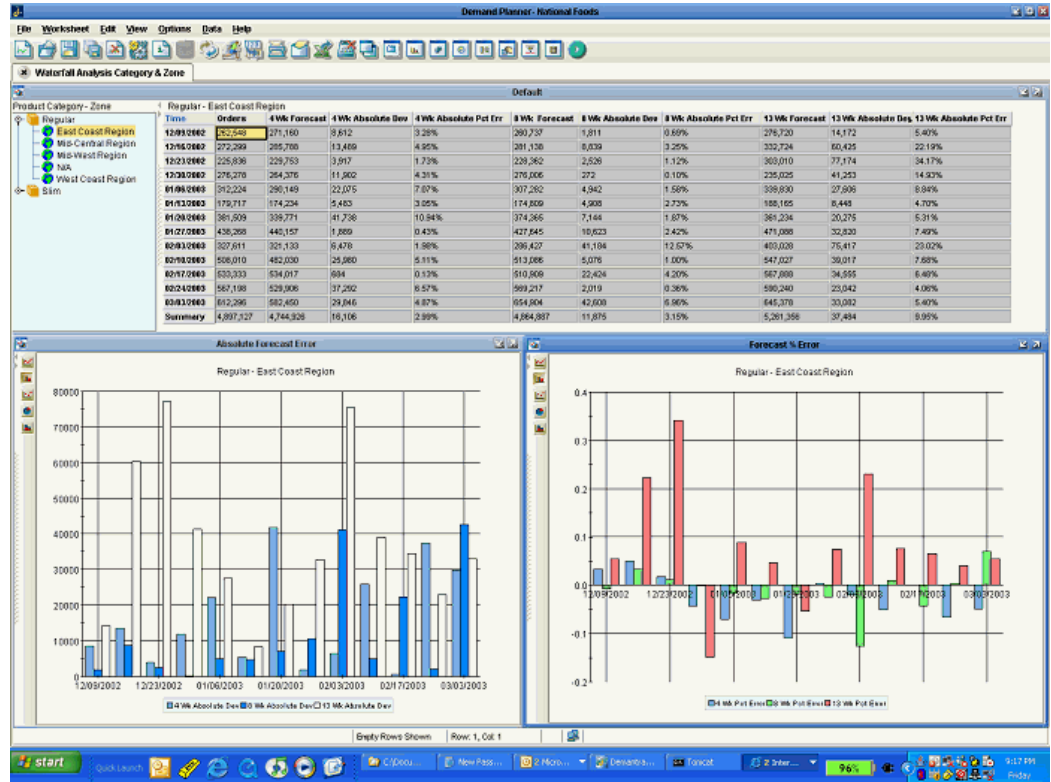
- Calendars
- Seeded workflows to run uploads to Advanced Supply Chain Planning or other sources
- Seeded workflow to download return history
- Seeded workflow to download price lists
- Seeded workflows to run the Demand forecast
- Seeded workflows to:
 1. Set the Final Approval series to NULL
 2. Run the statistical forecast, by default based on Shipment History – Requested Date, and
 3. Notify all users when the forecast is finished.
- Seeded workflows and seeded user groups used in the approval process. The default setting for the user step is: 'Check Finish Every Day'. The default setting for 'Timeout' is: after 10 days. The Planning Group Workflow should time out 5 days after the Final Approver's targeted time range has passed.
- Seeded workflow to archive forecasts for the Waterfall Analysis
- Seeded workflow to calculate forecast for Line of Business

These workflows may be changed depending on the business need. For example, the Administrator wants to ensure that the relevant user groups and users are notified of a change the timeout process. The Administrator does this by editing the relevant workflow and editing the steps.

Predefined Worksheets

Predefined worksheets with the appropriate series for analysis and modification of the forecast are provided. For more information about predefined worksheets, see the *Oracle Demantra Demand Management User's Guide*.

A predefined waterfall worksheet with the forecast and accuracy series is available for the analyst at the beginning of each cycle. A default level is specified for every hierarchy in the Aggregation tab, although only a subset of these hierarchies will be in a Component.



To produce this worksheet, historical final forecasts must be available. For implementations with Weekly time periods, the final forecast from the current quarter must be kept on a rolling basis moving forward. The following archived forecasts are used in the worksheet:

- The forecast series for the current week minus 4, named 4 Week Lag Forecast
- The forecast series for the current week minus 8, named 8 Week Lag Forecast
- The forecast series for the current week minus 12, 12 Week Lag Forecast
- The Mean Absolute Percentage Error (MAPE) is calculated for each of the historical forecast series named appropriately, for example 4 Week Lag MAPE
- The Mean Absolute Deviation (MAD) is calculated for each of the historical forecast series named appropriately, for example 4 Week Lag MAD

For implementations with Monthly time periods, the following forecasts must be kept for the current year on a rolling basis moving forward:

- The forecast series for the previous month, named 1 Month Lag Forecast
- The forecast series for the current month minus 2, named 2 Month Lag Forecast

- The forecast series for the current month minus 6, named 6 Month Lag Forecast
- MAPE calculated for each of the historical forecast series named appropriately, for example 1 Month Lag MAPE
- MAD calculated for each of the historical forecast series named appropriately, for example 1 month Lag MAD

MAPE calculation:

$$\frac{\text{summation (absolute value | Actual Demand - Lagged Forecast |)}}{\text{Number of Observations}}$$

MAD calculation:

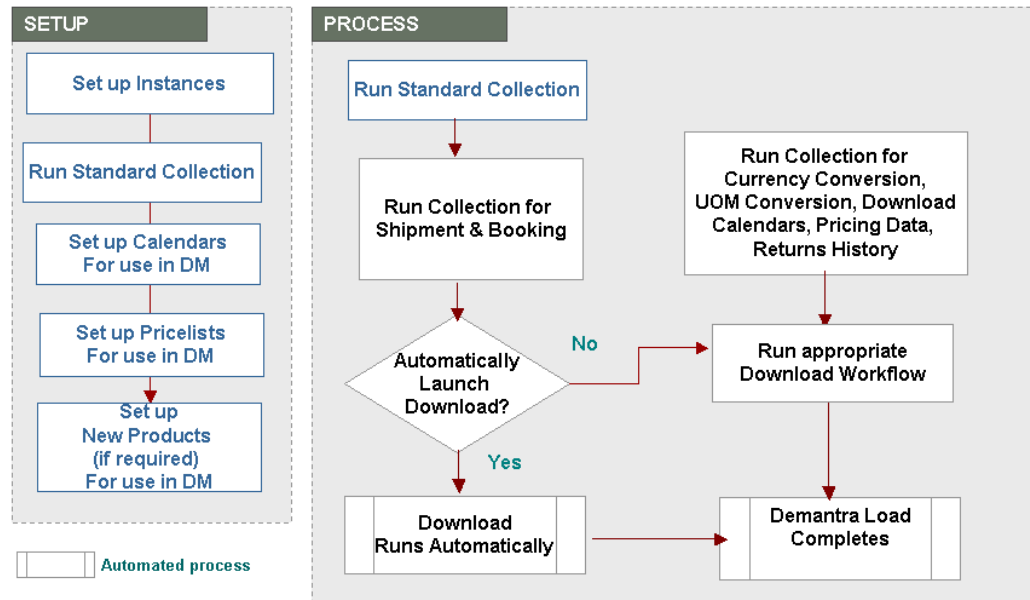
$$\frac{\text{summation (absolute value | Actual Demand - Lagged Forecast |)}}{\text{Number of Observations}}$$

Summary of Integration Tasks

This section lists integration tasks in the appropriate sequence.

Note: The Demantra schema must be in the same instance as the APS instance.

1. Initial Setup, page 14-17
2. Collect Data and Download to Three Staging Tables. See Download Collections, page 14-24.
3. Transfer data to Oracle Demantra schema See Download to Oracle Demantra, page 14-48.
 - EP_LOAD
 - Import Integration Profiles
4. Generate forecasts
5. Export Output from Oracle Demantra. See Demand Management Functional Output, page 14-50.
 - Export Integration Profiles
6. Upload Forecast, page 14-54



Initial Setup

Initial setup encompasses the following steps:

- Set up Instances
- Run Standard Collections
- Set up Calendars, Price Lists, New Products

Important: After Demantra has been installed you must run the 'Update Synonyms' concurrent request, which is under the responsibility 'Demand Management System Administrator.' This program defines the value of the MSD_DEM: Schema profile option..

This program has a parameter 'Schema Name' that is not displayed by default. If there is only one schema, (the most likely production scenario), the program will automatically update that one. But if there are multiple schemas (possible in a development environment), an administrator can log in using the 'Applications Developer' responsibility and choose to display the Schema Name parameter. The Demand Management System Administrator will then be able to select a schema when running Update Synonyms.

Setup Instances

An instance is a database and a set of applications. Setup Instances is run before running Standard Collections to specify the Instances from which Standard Collections obtains data.

Oracle Advanced Planning can plan a single instance or multiple instances. For information about setting up instances, see "Instances" in the Cross-Instance Planning chapter of *Oracle Advanced Supply Chain Planning Implementation and User's Guide*.

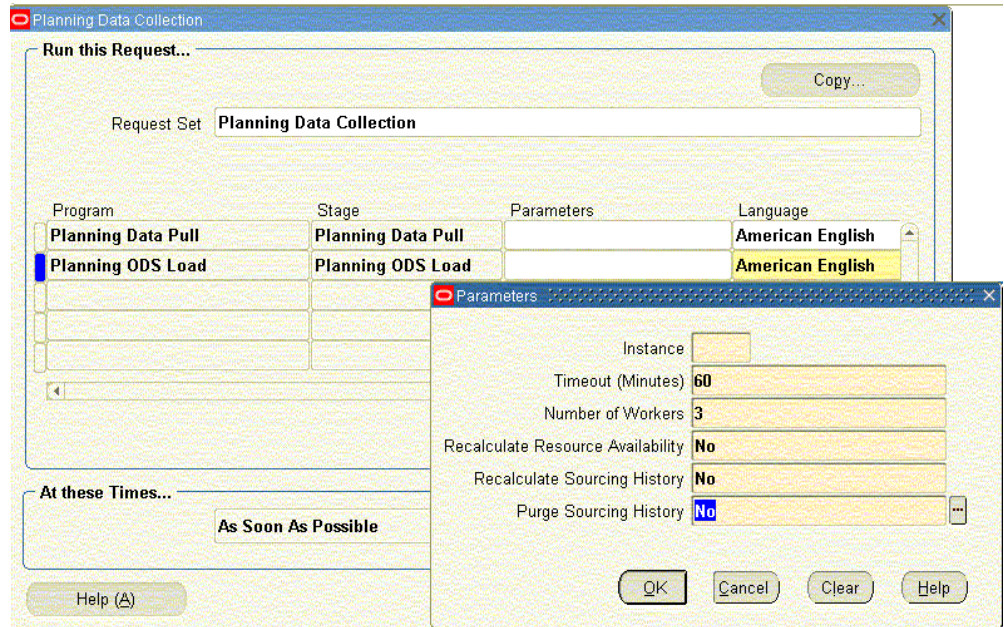
Run Standard Collections

"Standard" Collections refer to the Advanced Supply Chain Planning (ASCP) concurrent program for collecting new or changed information from the E-Business Suite to the Oracle Data Store (ODS). For information about collections, see "Collections" in the "Cross-Instance Planning" chapter and "Running Standard Collections" in the "Running Collections" chapter of the *Oracle Advanced Supply Chain Planning Implementation and User's Guide*.

Tip: You are recommended to set the MSC: Configuration profile option to 'APS & CP' if you want to collect item descriptions when standard collections are run. See "Profile Options" for more information about this profile option.

1. Sign on using the Advanced Supply Chain Planner responsibility or the Advanced Planning Administrator responsibility.
2. Navigate to the Planning Data Collection window by selecting Collections > Oracle Systems > Standard Collection.

The Planning Data Collection window appears.



3. This window shows that the collections process consists of two sequentially executed concurrent programs. The first program, *Planning Data Pull*, copies information from the source instance into the APS staging tables on the planning server. The second program, *Planning ODS Load*, copies information from the APS staging tables into the operation data store on the planning server,
4. To select the Data Pull Parameters to use during Standard Collections, select the Parameters field for the Planning Data Pull program.
The Planning Data Pull Parameters window appears.

Parameters

Instance	D9i
Collection Group	All All Enabled Organizations
Number of Workers	2
Timeout (Minutes)	180
Purge Previously Collected Data	No
Collection Method	Targeted Refresh
Analyze Staging Tables	No
Approved Supplier Lists (Supplier Capacities)	No
ATP Rules	No
Bills of Materials/Routings/Resources	No
Bills Of Resources	No
Calendars	Yes
Demand Classes	Yes
End Item Substitutions	No
Forecasts	No
Items	Yes

OK Cancel Clear Help

Parameters

Key Performance Indicator Targets	No
Master Demand Schedules	No
Master Production Schedules	No
On Hand	No
Planning Parameters	No
Planners	No
Projects/Tasks	No
Purchase Orders/Purchase Requisitions	No
Reservations	No
Resources Availability	No
Safety Stock	No
Sales Orders	No
Sourcing History	No
Sourcing Rules	No
Subinventories	No
Supplier Responses	No

OK Cancel Clear Help

Parameter	Value
Resources Availability	No
Safety Stock	No
Sales Orders	No
Sourcing History	No
Sourcing Rules	No
Subinventories	No
Supplier Responses	No
Suppliers/Customers/Orgs	Yes
Transportation Details	No
Unit Numbers	No
Units of Measure	Yes
User Company Association	No
User Supplies and Demands	No
Work in Process	No
Sales Channel	Yes
Fiscal Calendar	Yes

Buttons: OK, Cancel, Clear, Help

- Set the parameters as shown in the previous figures.
- Select the Parameters field for the Planning ODS Load program.
The Parameters window appears.

Instance	
Timeout (Minutes)	60
Number of Workers	3
Recalculate Resource Availability	No
Recalculate Sourcing History	No
Purge Sourcing History	No

Buttons: OK, Cancel, Clear, Help

- Set the parameters as shown in the previous figure.

Note: If the item descriptions have not been collected after you

complete this process, set the MSC: Configuration profile option to 'APS & CP' and rerun the collection. See "Profile Options" for more information.

Set up Price Lists, Calendars, and New Products

Setup Calendar, Price Lists and New Products are run initially, and on an as needed basis in ongoing cycles.

See

Collecting Price List Data, page 14-43

Setting Up the Calendar List, page 14-65

Downloading Calendars, page 14-45

Setting Up the New Products List, page 14-64

Ongoing Collections

After Setup is complete, the remaining Collections are run. All other Collection choices under the Oracle Systems menu are used to collect the specified data from the planning server for download to Oracle Demantra.

See:

- Combined Collections of Shipment and Booking History, page 14-25
- Collecting Currency Conversion Data, page 14-38
- Collecting UOM Conversion Data, page 14-41
- Collecting Returns History Data, page 14-35

Legacy Collection loads Shipment and History data into Oracle Demantra

See Collecting Legacy Shipment and History Data, page 14-32.

Download E-Business Suite Data Into Demand Management

Downloading Oracle E-Business Suite data into Oracle Demantra Demand Management involves a two-stage process:

1. **Collection.** Login as Demand Management System Administrator and navigate to Collections. From there, the Administrator can collect shipment and booking history, returns history, and so on. If you select the Download Now check box option to start the download once collection successfully complete, an automated process transforms collected data into staging table structures and formats that are

amenable to the following Oracle Demantra native download processes:

- **EP_LOAD** download procedures are used for booking history streams and level members. For example, the EP_LOAD procedures are used to load booking history by organization-site-sales channel and item-demand class into staging tables.
 - **Data Import Integration Profiles** are used for all other data streams. A data import profile describes how to import series data aggregated to a specific aggregation level or levels, with optional filtering. For example, integration profiles are used to load returns history. See *Creating a Data Import Profile*.
2. **Transfer.** If the Download Now check box was not selected during the collections process, run EP_LOAD and Import Integration Profiles to move data from the staging tables into the Oracle Demantra Demand Management schema.

Download Collections

Collections use existing Oracle Demand Planning and Oracle Advanced Supply Chain Planning user interfaces, accessed from a single Navigator menu structure.

Available collections:

- Standard Collections, - designates existing ASCP collections of reference data, Items, Location, and Calendars that are collected from the Instances specified in Setup > Instance. Sales Orders, which is an entity inside Advanced Supply Chain Planning Standard Collections, provides the data stream representing future demand. For information about standard collections, see "Collections" in the "Cross-Instance Planning" chapter and "Running Standard Collections" in the "Running Collections" chapter of the *Oracle Advanced Supply Chain Planning Implementation and User's Guide*.
- Shipment and Booking Data, - provides the data stream representing past demand. See Combined Collections of Shipment and Booking History, page 14-25, and Collecting Legacy Shipment and History Data, page 14-32
- Return History, See Collecting Returns History Data, page 14-35
- Currency Conversion, See Collecting Currency Conversion Data., page 14-38.
- Units of Measure (UOM) Conversion, See Collecting Unit of Measure Conversion Data, page 14-41
- Pricing Data, See Collecting Price List Data, page 14-43

Combined Collections of Shipment and Booking History

The Collection Utility merges programs that collect data streams for both Shipment and Booking History.

Prerequisites

- ☐ Define database Instances.
- ☐ Set up appropriate source and destination profiles.
- ☐ Run Standard Collections.

To run Shipment and Booking Data Collections:

1. Navigate to the Collection Utility.

Collections > Oracle Systems > Shipment and Booking History

The Collections Utility window appears, listing several collections programs.

Program	Stage	Parameters	Language
Collect Shipment and Booking Data	Collect Shipment and B		American English
Push Setup Parameters	Push Setup Parameters		American English
Collect Level Type	Populate Staging Table		American English
Update Level Codes	Populate Staging Table		American English
Collect Time	Populate Staging Table		American English

- **Collect Shipment and Booking Data.** This program collects shipment and booking history data from the E-Business Suite Order Management source application based on the collection parameters specified, and then inserts the data into the Oracle Demantra sales staging table.

- **Push Setup Parameters.** This program pushes destination data into the E-Business Suite source, such as source profiles, organizations in the collection group, and time data from Oracle Demantra.
- **Collect Level Type.** There are two Collect Level Type programs, one for items and the other for locations. These programs generate distinct item and location intersections, as defined in Oracle Demantra, from the shipment and booking history, and then insert the data into Oracle Demantra item and location staging tables.
- **Update Level Codes.** This program updates the site level codes in the Oracle Demantra sales staging table as present in the Advanced Supply Chain Planning Operational Data Store.
- **Collect Time.** This program collects Manufacturing and Fiscal calendars from the Advanced Supply Chain Planning Operational Data Store, as setup in the Calendar List. See Setting Up the Calendar List, page 14-65.

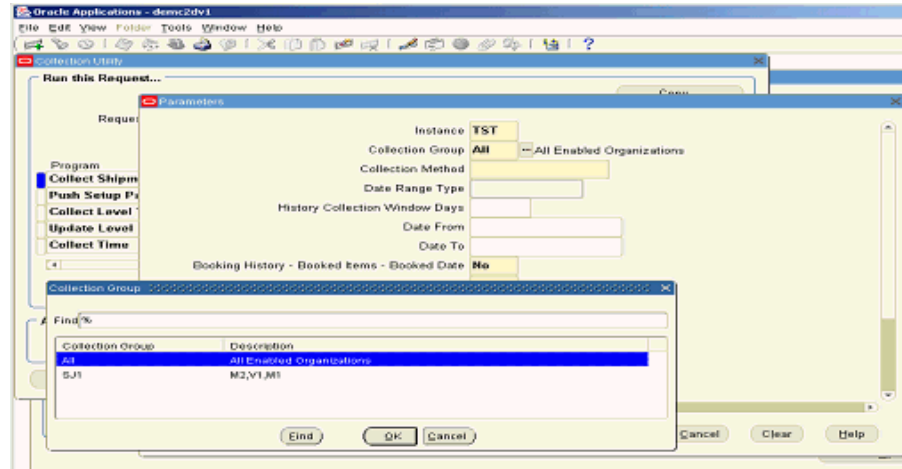
- **Launch EP LOAD.**

Historical information and level data are imported into Oracle Demantra via the EP_LOAD procedure. All other series data are imported into Oracle Demantra via Import Integration Profiles. An assumption of the EP LOAD procedure is that the series are populated into the Oracle Demantra staging tables before the load process begins. To ensure this occurs, the collection programs for all eight historical series have been merged so that these streams are always collected simultaneously. See Seeded Series, page 14-12.

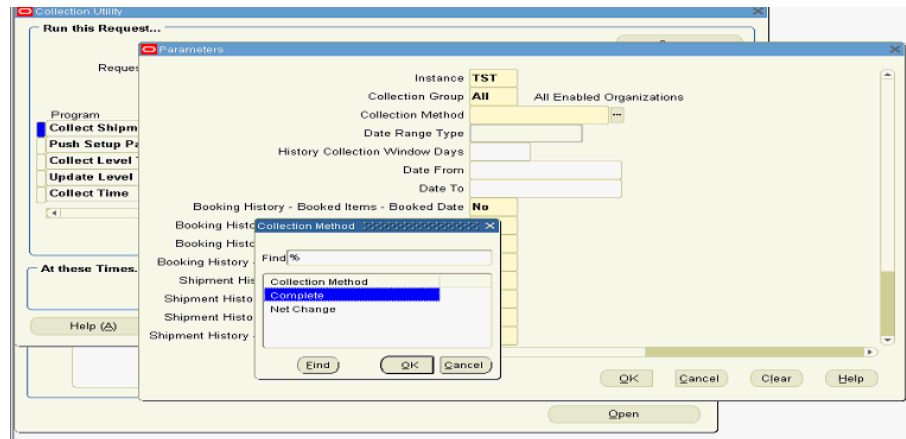
2. Highlight the Collect Shipment and Booking Data program.
3. In the row for the Collect Shipment and Booking Data program, click within the Parameters field.

The Parameters window appears. The parameters are described following the image.

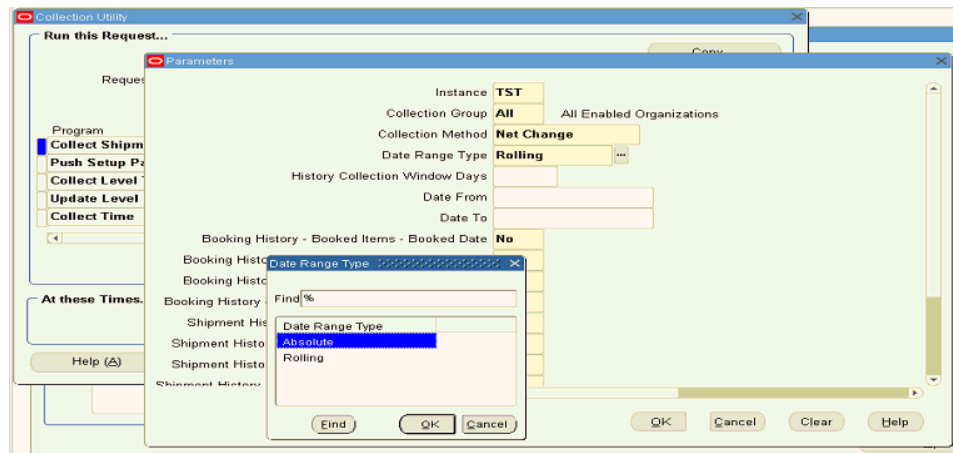
- **Instance.** Select the instance code of the E-Business Suite source instance from the list of values as defined in Instances form.
- **Collection Group.** Collection group is a group of Organizations. The parameters screens of all demand planning entities: Level Values, Manufacturing Forecast, Return History, Sales Forecast, Shipment and Booking Data, include a Collections Group parameter. This is to support line of business-specific collections.
 - Currency, Unit of Measure, and Price List do not have a Collections Group parameter. Currency is dimensioned by time only. Unit of Measure is dimensioned by item only. Price List is dimensioned by item and time.
 - The default value is 'All', which implies data for all Oracle Advanced Supply Chain Planning- and all Oracle Demand Planning-enabled organizations available for the specified instance are to be collected.
 - User-defined values can be specified if user-defined collection groups have been created in the Instances form. Only Oracle Advanced Supply Chain Planning- and Oracle Demand Planning-enabled organizations can be added to the user-defined collection groups.



- **Collection Method.** Allowed values are: 'Complete' or 'Net Change'
 - 'Complete' clears the data in the Oracle Demantra sales staging table, collects all available records from the E-Business Suite source, and inserts data into the Oracle Demantra sales staging table. No date filters apply for Complete collection. For first time collection of history data, you typically select the 'Complete' Collection Method.
 - 'Net Change' clears the data in the Oracle Demantra sales staging table, collects data by applying the specified date filters, and inserts the fetched data into the Oracle Demantra sales staging table. Net Change is typically selected for ongoing, periodic collection of history data, say on a weekly basis.
 - The Complete and Net-Change Collection Methods are mutually exclusive.



- **Date Range Type.** For Net Change collection, the allowed values for Date Range Type are 'Rolling' or 'Absolute'. Date Range type is not valid for 'Complete' collections.
- 'Rolling' implies the history data is collected from the system date up to the number of days in the past specified in the 'History Collection Window Days' field.
- 'Absolute' implies the user specifies the date range for which collection is to be done in the Date From and Date To fields.
- The Rolling and Absolute Date Range Types are mutually exclusive.



- **History Collection Windows Days.** This field specifies the number of days in the past from the system date for which history data is to be collected. This field is valid when the Date Range Type is 'Rolling'.

- **Date From and Date To.** These fields specify a date range for the collection of history data. They are valid when the Date Range Type is 'Absolute'.
- **Collected Series.** The next eight parameters are the names of the eight seeded history series. Specifying 'Yes' causes a series to be collected. Specifying 'No' will not collect it.

Note: For history collections, at least one series must be specified. The Shipment History – Requested Items – Shipped Date series is the default value.

- **Collect Internal Sales Orders.** Specifying 'Yes' collects any internal sales orders available in the source. By default, internal sales orders are not collected.
- **Collect All Order Types.** Specifying 'Yes' collects sales orders for all order types. Specifying 'No' enables the Include Order Types field.
 - **Include Order Types.** Specifying 'No' in the Collect All Order Types parameter enables entry of Order Types in the Include Order Types field, for which sales orders are collected. Enter Order Types one after another with a comma separated delimiter.
 - **Exclude Order Types.** To exclude certain order types from being collected specify the list of Order Types that are not to be collected in the Exclude Order Types field. Enter Order Types one after another with a comma separated delimiter.
 - Either 'Include Order Types' or 'Exclude Order Types' can be specified, but not both.

- **Type Selection Method**

The Type Selection Method is available through the EBS Menu under Demand Management System Administrator > Collections > Oracle Systems > Shipment and Booking History. It is disabled if the Collect All Order Types parameter is set to YES (Default). If Collect All Order Types is set to NO, then Type Selection Method is enabled and mandatory.

The available settings are defined in the table below.

List Item	Description
-----------	-------------

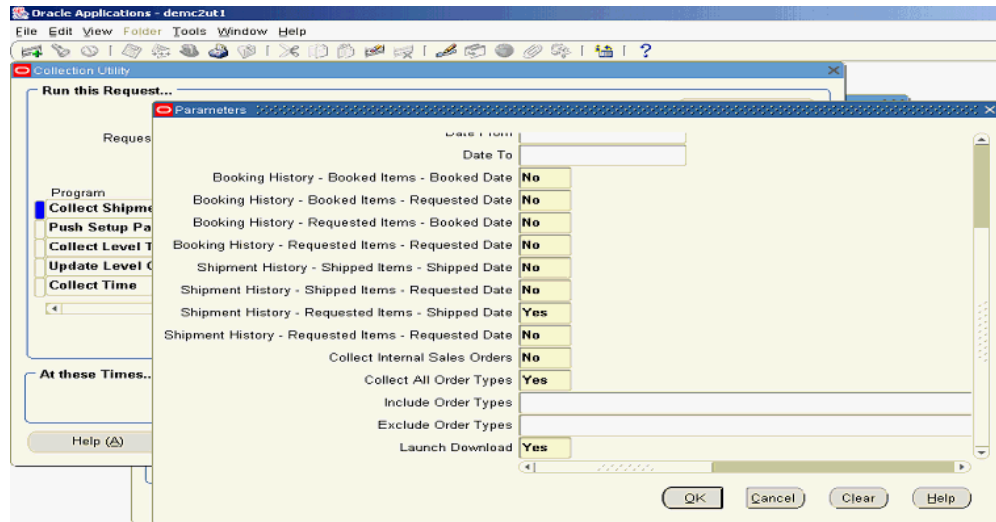
Comma (,) separated list of order types (Default value)	Select either "Include Order Types" or "Exclude Order Types" with a list of order types. The list must be less than 240 characters.
SQL query name	SQL query name from Table MSD_DEM_ENTITY_QUERIES. When selected; user has to enter the query name stored in Table MSD_DEM_ENTITY_QUERIES in the "Include Order Types" or "Exclude Order Types" parameter fields which will return valid order type ids. It is the customer's responsibility to maintain entity_name, sql query & order types ids. Note: Sql query must be executable on source side instance.
Value Set	When selected, the user enters the valueset name in "Include Order Types" or "Exclude Order Types" parameter fields. It is the customer's responsibility to create and maintain the valueset to hold the valid order types for Shipment and Booking History collection. Note: ValueSet must be created on source instance and the ValueSet must be "List of values" type.

- **Launch Download.** Each collections Single Request Submission includes a Launch Download parameter. Valid values are 'Yes' and 'No'.

Specifying 'Yes' automatically launches the download of history data into Oracle Demantra as soon as collections have completed. Internally this invokes the Oracle Demantra EP_LOAD procedure to download the data into Oracle Demantra. E-Business Suite Advanced Planning and Scheduling directly populates the staging tables. The Shipment and Booking history collection automatically invokes Download Calendars program if the Launch Download field is set to 'Yes'. See Downloading Calendars, page 14-45.

If Launch Download is set to 'No', then only collections will be done. Advanced Planning and Scheduling collections leave the data in the Oracle Demantra staging tables. Download does not launch automatically. To download the collected items, locations, and history data, manually launch the Download workflow from Oracle Demantra Workflow Manager.

- The Administrator launches the workflow.
- For historical series, all series are loaded.
- An integration profile for each non-history series determines which data are loaded.



4. Click Ok to close the Parameters window.
5. (Optional) On the Collection Utility window, click Schedule to open a window where you can set up the concurrent program to run collections at a future date and time, or periodically. The At these Times field default value causes the program to run "As Soon As Possible" after the program is submitted.
6. Click Submit.

Collecting Legacy Shipment and History Data

The concurrent programs for Legacy Shipment and History collection are:

- **Flat File Loads.** This program loads the shipment and booking history data from a flat file into the Oracle Demantra sales staging table.
- **Sales Data Pre Processor.** This program updates the destination keys from Advanced Supply Chain Planning for all of the levels.
- **Collect Level Type** and **Launch EP_LOAD** are the same programs as those used for the E-Business Suite Shipment and Booking History Data collections.

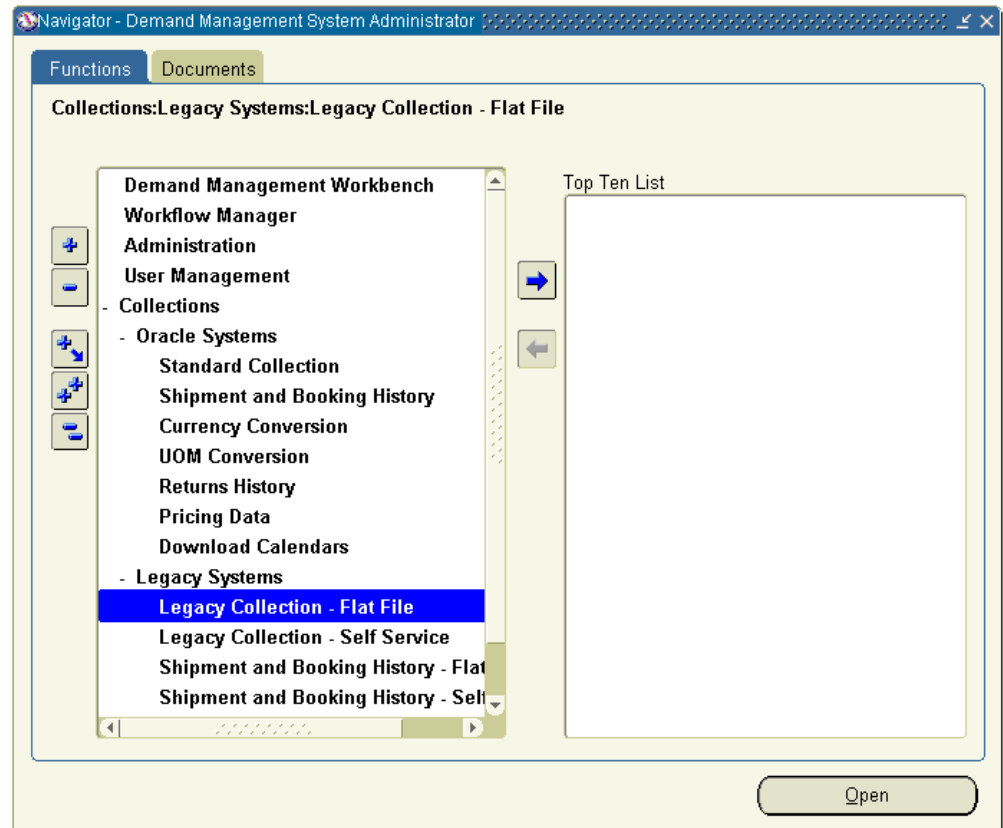
Prerequisites

- ☐ Prior to launching this collection, complete ASCP collections for the legacy instance.

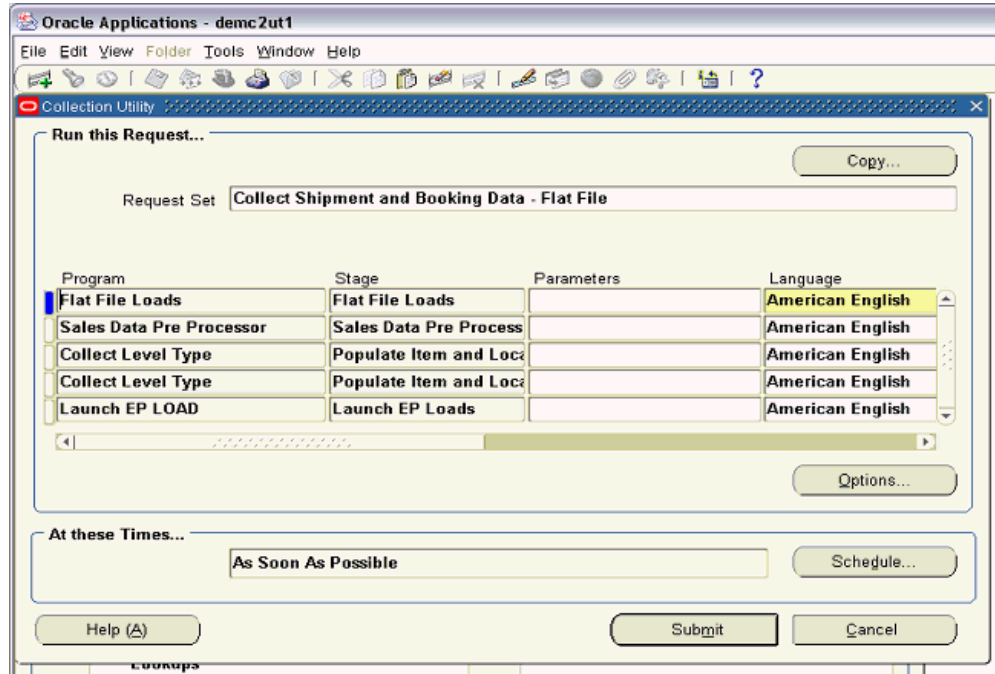
To collect Legacy Shipment and History Data:

1. Navigate to the Legacy Collection Utility.

Collections > Oracle Systems > Collect Shipment and Booking Data - Flat File

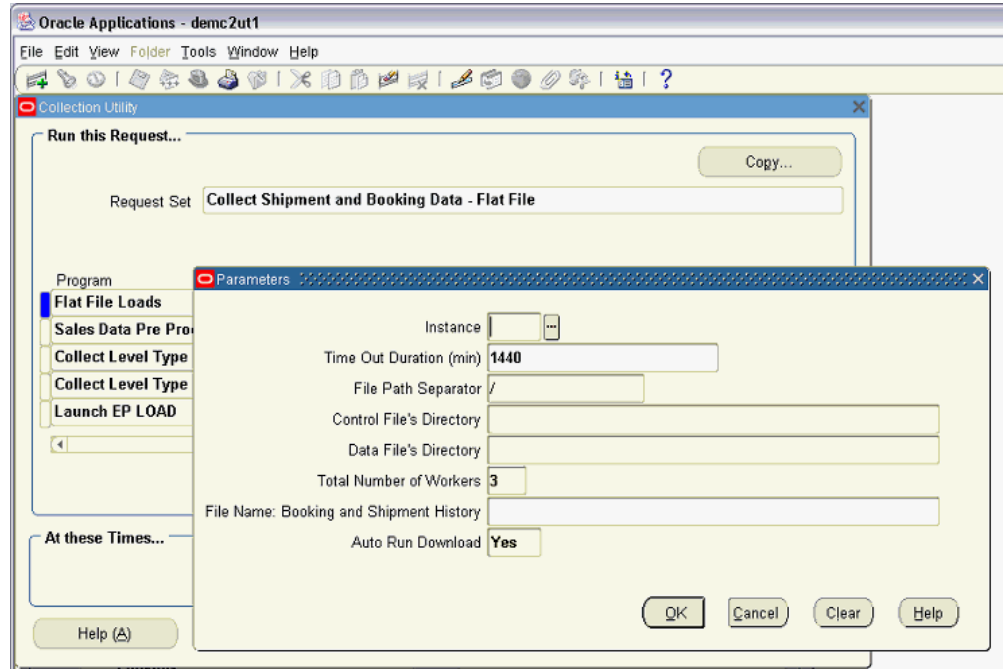


The Collections Utility window appears, listing several collections programs.



2. Highlight the Collect Shipment and Booking Data program.
3. In the row for the Collect Shipment and Booking Data program, click within the Parameters field.

The Parameters window appears. The parameters are described following the image.



The parameters for the Flat File Loads concurrent program are:

- **Instance.** The legacy instance for which data is being loaded.
- **Time Out Duration (min).** This is the maximum duration, in minutes, for which this program is allowed to run.
- **File Path Separator.** Windows and UNIX path separators are different.
- **Control File's Directory.** Specifies the location of the SQL loader control file.
- **Data File's Directory.** Specifies the location of the history data flat file.
- **Total Number of Workers.** Specifies the number of programs that can be run simultaneously.
- **File Name Booking and Shipment History.** Specifies the name of the shipment and booking history flat file.
- **Auto Run Download.** 'Yes' automatically downloads data into Oracle Demantra from the staging tables. 'No' indicates that the download into Oracle Demantra will be accomplished using a separate manual procedure.

Collecting Returns History Data

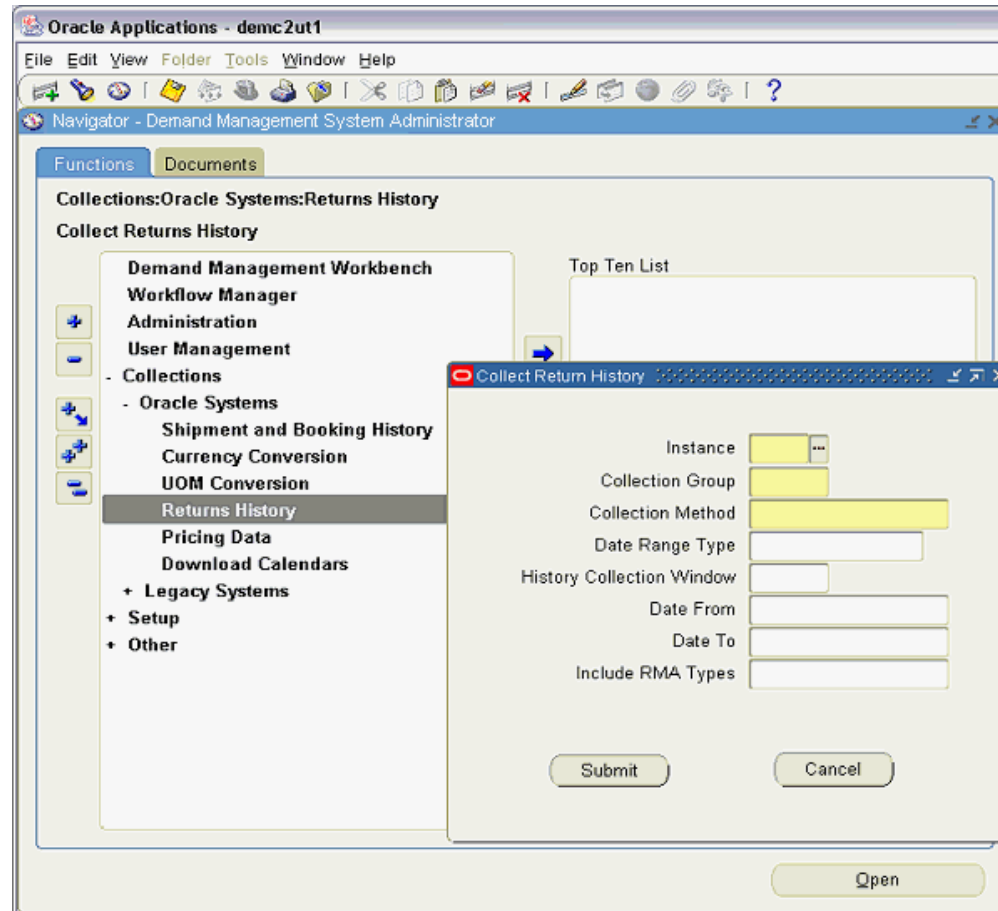
Import Integration Profiles are used to Collect Returns History.

To collect return history data:

1. Navigate to the Collect Return History window.

Collections > Oracle Systems > Returns History

The Collect Return History window appears.



2. Specify Collect Return History parameters.

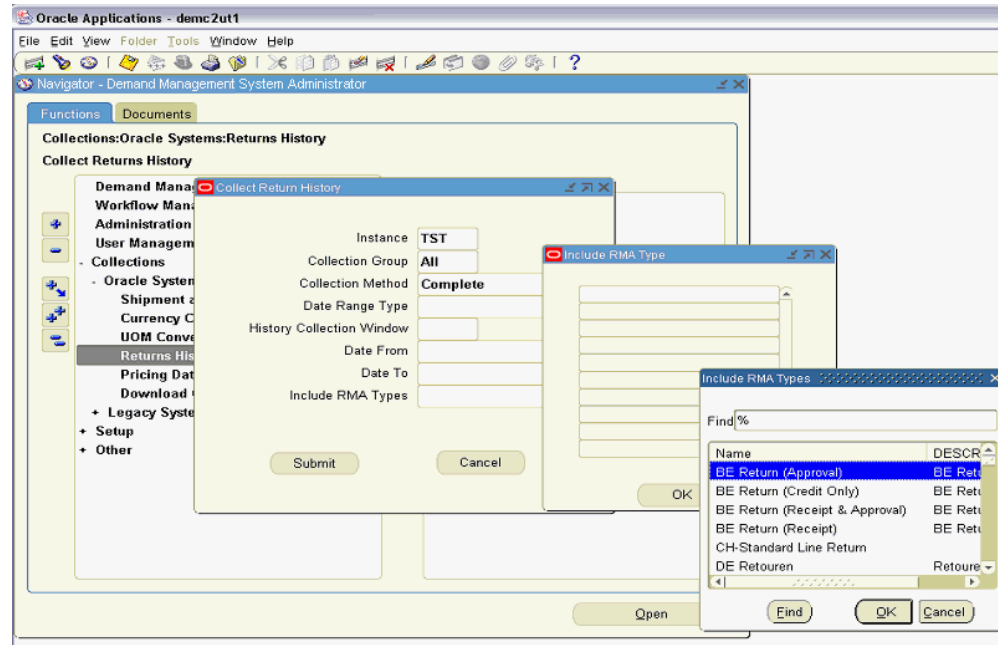
The collection parameters for Returns History are:

- **Instance.** This is the instance code of the source instance, as defined in Instances form.
- **Collection Group.** This group of organizations filters the collected data by organization.
 - The default value is 'All', which implies all Advanced Supply Chain Planning and Demand Planning enabled organizations are available for the specified instance.

- User-defined values can be specified if user-defined collection groups have been created in the Instances form.
 - Only Advanced Supply Chain Planning and Demand Planning enabled organizations can be added to the user-defined collection groups.
 - **Collection Method.** Valid options are: 'Complete' or 'Net-Change'.
 - 'Complete' clears the data in the Return History staging table, collects all available records from the source, and inserts them into the Return History staging table. No date filters are applied for Complete collection. Complete collection is typically used for the first time collection of history data.
 - 'Net-Change' clears the data in the Return History staging table, collects data by applying the specified date filters, and inserts the fetched data into the Return History staging table. Typically, select net change for regular collection of history data, say on a weekly basis.
 - Complete and Net-Change are mutually exclusive.
 - **Date Range Type.** For 'Net-Change' collection, Date Range Type can be either 'Rolling' or 'Absolute'.
 - 'Rolling' implies collection of history data from the system date up to the number of days in the past as specified in the 'History Collection Window Days' field.
 - 'Absolute' requires values in the Date From and Date To fields to specify the date range for which collection is to be done.
 - 'Rolling' and 'Absolute' Date Range Types are mutually exclusive.
 - Date Range Type is not valid for the 'Complete' Collection Method.
 - **History Collection Windows Days.** This field is valid if the 'Rolling' date range type has been chosen. Specify the number of days in the past from the system date for which history data is to be collected.
 - **Date From and Date To.** These fields are valid if the 'Absolute' date range type has been chosen. Specify a date range for the collection of history data.
 - **Include RMA Types.** Null value implies collection of all Return Material Authorization types. Specifying particular RMA types causes only those listed to be collected. Multiple RMA types can be specified.
3. (Optional) To specify particular RMA Types click the cursor within the Include

RMA Types Field.

The Include RMA Type window appears.



4. Select the RMA Type. Click OK.
5. Repeat the previous step as necessary to specify all desired RMA Types.
6. Click OK to close the Include RMA Type window.
7. On the collect Return History window, click Submit to launch the concurrent program to execute Return History collections.

Collecting Currency Conversion Data

Currency conversion rates are dimensioned by time only.

To run the currency conversion collections:

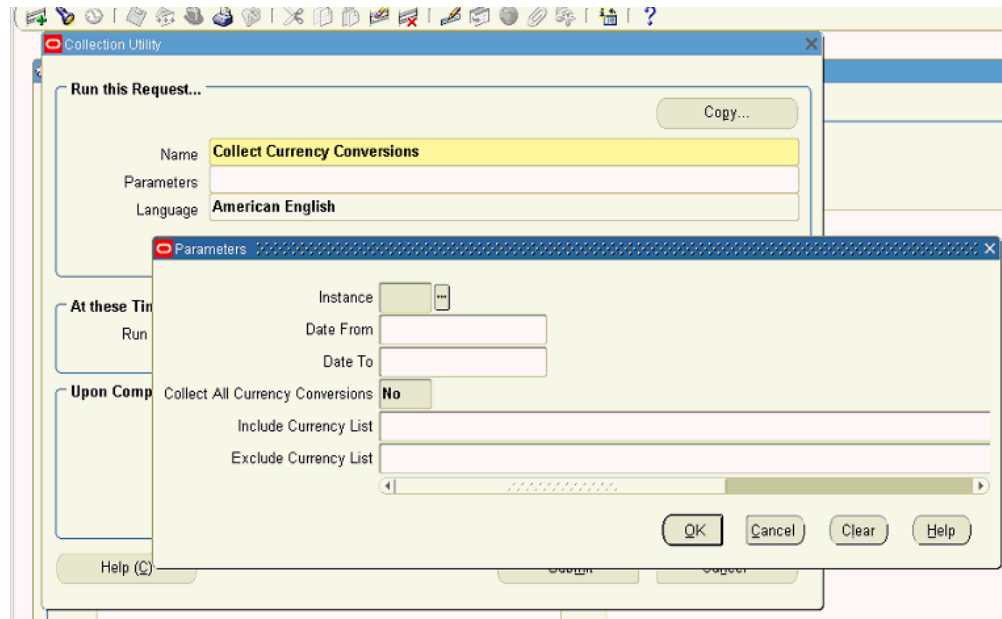
1. Navigate to the Collection Utility

Collections > Oracle Systems > Currency Conversion

The Collections Utility window appears, with the Collect Currency Conversions program name appearing in the Name field.

2. Click within the Parameters field.

The Parameters window appears.



3. Set the program parameters.

The parameter descriptions follow:

- **Instance.** This is the Instance code of the source instance, as defined in the Instances form.
- **Date From and Date To.** Specify a date range for the collection of currency conversion rates. If no dates are specified then all available records available in source are collected without applying any date filters.
- **Collect All Currency Conversions.** Specifying 'Yes' collects Currency conversion rates for all currencies for which conversion rates to the base currency exist in the source. Specifying 'No' enables entry of Currency Codes for which conversion rates can be collected, in the 'Include Currency List' field.
 - **Include Currency List.** If Collect All Currency Conversions is set to 'No', enter Currency Codes one after another with a comma separated delimiter.
 - **Exclude Currency List.** To exclude certain Currency conversion rates from being collected, specify the list of Currency Codes that are not to be collected in the 'Exclude Currency List' field. Enter Currency Codes one after another separated by a comma delimiter
 - Either 'Include Currency List' or 'Exclude Currency List' can be specified but not both.

4. Click OK to close the Parameters window.

5. Click Submit.

Note: The Administrator can run the following script to add more placeholder currencies to Oracle Demantra:

Name: 'Create Seed Entities in Demantra

Script:

```
declare
    retcode number;
begin

    msd_dem_create_dem_seed.create_dem_seed_data(retcode,
    <p_start_no>,
    <p_num_entities>,<p_entity_type>);
end;
/
```

Parameters to be passed to this script:

p_start_no. - starting number of entities to be created (Already units from 100-199 are created)

p_num_entities - number of entities to be created

p_entity_type - 1 (UOM), 2 (CURRENCY), 3 (PRICE LIST), 0 (ALL)

Associating multiple Price Lists with a CTO worksheet

Only one price list is supported for CTO. Usually the first price list collected from EBS is loaded to t_ep_cto_data along with sales_data. So if a user wants to see CTO data in another price list (say PL1), then the following steps should be performed.

1. Find the column in sales_data, where the price list information for PL1 is being stored. This mapping can be found by looking at the definition of the display unit 'PL1'. (ex. Assume price list PL1 has been collected from EBS and mapped to the column EBSPRICELIST112).
2. Create a new series on t_ep_cto_data. The database column name should be set to the column name found in Step 1 above. (ex. New series 'CTO PL1' is created on t_ep_cto_data with database column name EBSPRICELIST112).
3. Create a new import data profile. The series created in Step 2 above should be selected. All other details will be similar to data profile IMPORT_CTO_OPTION_PRICE. (ex. New data profile 'CTO PL1 IMPORT' is created, and the staging table name is BIIO_CTO_PL1_IMPORT).
4. Write a new stored procedure in Demantra to copy THE price list data of 'PL1' from the table MSD_DEM_PRICE_LIST to the new import integration staging table created in Step 3 above. For example, create a new stored procedure to copy data

from msd_dem_price_list to BIIO_CTO_PL1_IMPORT. The following is a sample insert statement for copying data:

```
INSERT INTO BIIO_CTO_PL1_IMPORT ( SDATE, LEVEL1, EBSPRICELIST112 )  
SELECT SDATE, LEVEL1, EBSPRICELIST112 FROM MSD_DEM_PRICE_LIST  
WHERE EBSPRICELIST112 IS NOT NULL )
```

5. Modify the 'EBS Price List Download' workflow.
 - Add a new STORED PROCEDURE STEP as the start step. The step should call the stored procedure created in Step 4 above.
 - At the end, add a TRANSFER STEP for the import data profile created in Step 3 above.
6. After the Price List Collection has been run for PL1 from EBS, the staging table MSD_DEM_PRICE_LIST will have data for the price list PL1. When the workflow 'EBS Price List Download' starts, it:
 - firsts copies the data of 'PL1' from MSD_DEM_PRICE_LIST to BIIO_CTO_PL1_IMPORT
 - loads data for PL1 into sales data
 - loads data for PL1 into t_ep_cto_data.

Collecting Unit of Measure Conversion Data

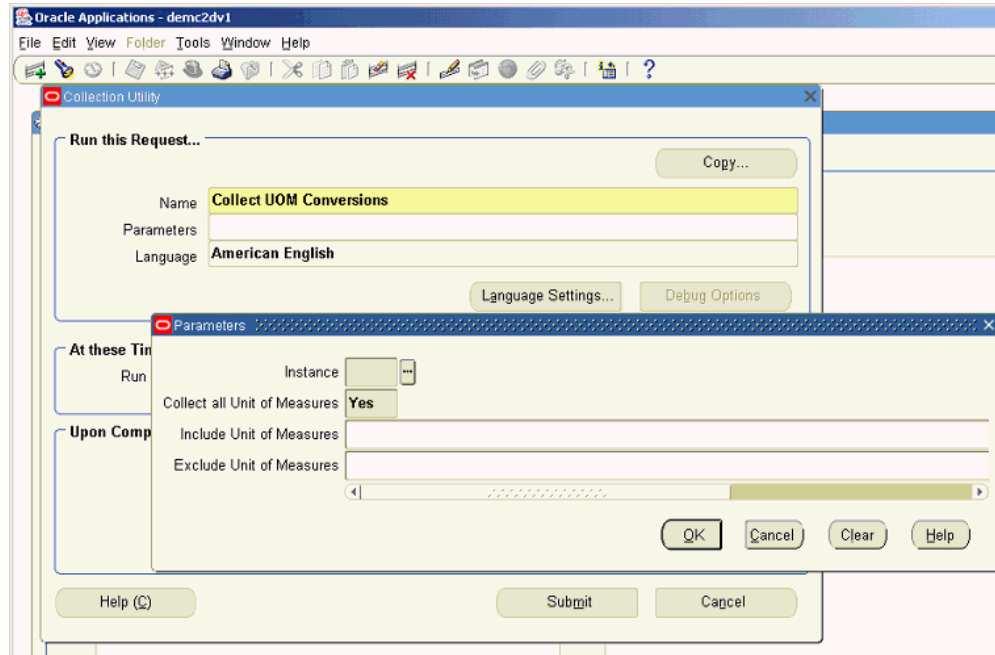
Unit of Measure conversion rates are dimensioned by time only. UOM conversion rates are always calculated with respect to each collected Item's Primary UOM; up to 100 UOM Conversions can be collected.

To run the unit of measure conversion collections:

1. Navigate to the Collection Utility.

Collections > Oracle Systems > UOM Conversion

The Collections Utility window appears, with the Collect UOM Conversions program name appearing in the Name field.



2. Click within the Parameters field.

The Parameters window appears.

3. Set the program parameters.

The parameter descriptions follow:

- **Instance.** This is the instance code of the source instance, as defined in the Instances form.
- **Collect All Units of Measure.**
 Specifying 'Yes' collects UOM conversion rates for all units of measure for which conversion rates exist in the source.
 Specifying 'No' enables entry of UOM Codes for which conversion rates can be collected, in the 'Include Units of Measures' field.
 - **Include Units of Measure.** Enter a list of UOM Codes for which conversion rates can be collected. UOM Codes are entered one after another separated by a comma delimiter
 - **Exclude Units of Measure.** To exclude certain UOM conversion rates from being collected, enter the list of UOM Codes that *are not* to be collected. Enter UOM Codes one after another, separated by a comma delimiter.
 - Either 'Include Units of Measures' or 'Exclude Units of Measures' can be

specified, but not both.

4. Click OK to close the Parameters window.
5. Click Submit.

Note: The Administrator can run the following script to add more placeholder units of measure to Oracle Demantra:

Name: 'Create Seed Entities in Demantra

Script:

```
declare
    retcode number;
begin
    msd_dem_create_dem_seed.create_dem_seed_data(retcode,
    <p_start_no>,
    <p_num_entities>,<p_entity_type>);
end;
/
```

Parameters to be passed to this script:

p_start_no. - starting number of entities to be created (Already units from 100-199 are created)

p_num_entities - number of entities to be created

p_entity_type - 1 (UOM), 2 (CURRENCY), 3 (PRICE LIST), 0 (ALL)

Collecting Price List Data

The Demand Management forecast considers historical demand and causal factors, such as seasonal demand variation, specific promotional events, and price list changes. The price list is dimensioned by item and by time.

Each time you download price list data, the price list data will be cut off at the current forecast horizon end date. As time rolls forward and the corresponding forecast end buckets roll forward, you will not be able to see pricing and revenue data for the new time buckets until you once again download price list data.

Therefore, Oracle recommends that you either:

- download price list data with each forecasting cycle, or
- if you would like to download price list data less frequently, extend the forecast horizon beyond the business standard.

Example

For example, if the business standard forecast horizon is 12 months, extend the forecast

horizon to 15 months. Generate a forecast, then download price lists. The downloaded price list information will then be sufficient to cover the next three forecasting cycles (cycles 2 through 4). A second price list download will only be necessary after forecasting cycle 5.

Oracle also recommends that when you collect pricing data, set the "Date To" parameter to match the current end of the forecasting horizon so that you do not needlessly collect more data than can be downloaded into Oracle Demantra Demand Management.

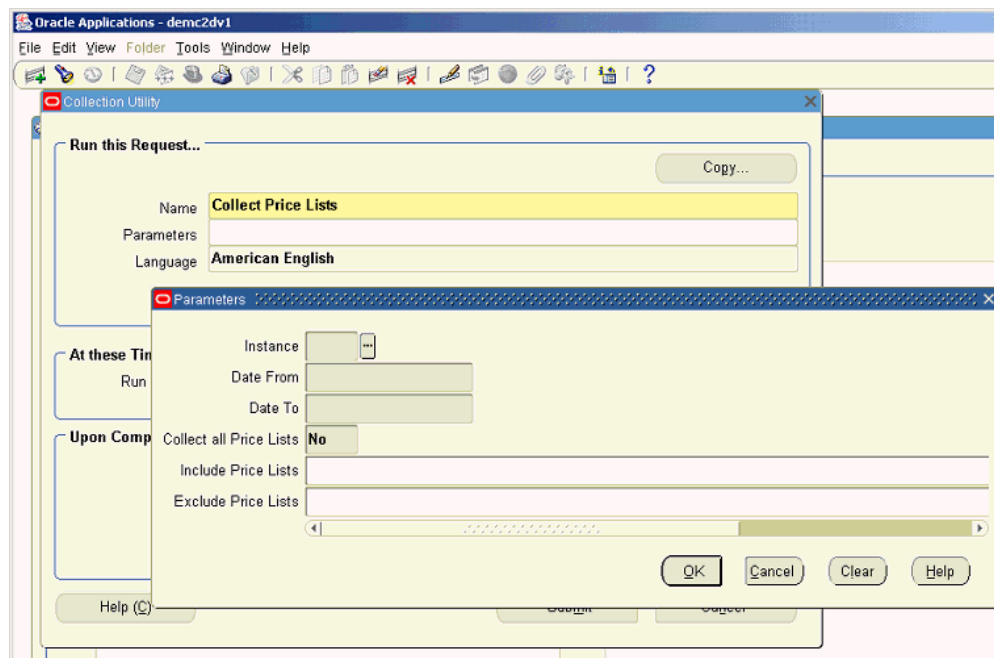
To extend the forecasting horizon:

1. From the Business Modeler navigate Parameters > System Parameters > Engine > Lead.
2. Increase the value of 'Lead'.

To run the price list conversion collections:

1. Navigate to the Collection Utility Collections > Oracle Systems > Price List

The Collections Utility window appears, with the Collect Price List Conversions program name appearing in the Name field.



2. Specify the Collect Price List program Parameters.

The collection parameters are:

- **Instance.** This is the instance code of the source instance, as defined in the Instances form.

- **Date From and Date To.** Used to specify a date range for the collection of Price Lists. These fields are mandatory.

- **Collect All Price Lists.**

Specifying 'Yes' collects price lists up to the number of seeded series available in Oracle Demantra. The program fails to execute if there are more than the default number of price lists.

Specifying 'No' enables entry of specific Price Lists that can be collected, in the 'Include Price Lists' field.

- **Include Price Lists.** Enter Price Lists to be collected, one after another, separated by a comma delimiter.
- **Exclude Price Lists.** To exclude certain Price Lists from being collected, specify the list of Price Lists that *are not* to be collected in the 'Exclude Price Lists' field. Enter Price Lists one after another, with a comma separated delimiter.
- Either 'Include Price Lists' or 'Exclude Price Lists' can be specified, but not both.

Note: The Administrator can run the following script to add more placeholder price lists to Oracle Demantra:

Name: 'Create Seed Entities in Demantra

Script:

```
declare
    retcode number;
begin
    msd_dem_create_dem_seed.create_dem_seed_data(retcode,
    <p_start_no>,
    <p_num_entities>,<p_entity_type>);
end;
/
```

Parameters to be passed to this script:

p_start_no. - starting number of entities to be created (Already units from 100-199 are created)

p_num_entities - number of entities to be created

p_entity_type - 1 (UOM), 2 (CURRENCY), 3 (PRICE LIST), 0 (ALL)

Downloading Calendars

The E-Business Suite contains several types of calendars. Typical calendars include

Manufacturing, Gregorian, and Fiscal. These calendars may be changed in the source system. Such changes need to be reflected and synchronized in the Oracle Demantra system.

The Download Calendars concurrent program downloads collected manufacturing and fiscal calendars, in Advanced Supply Chain Planning collections, to Oracle Demantra. Shipment and Booking history collection automatically invokes this program if the Launch Download field is set to 'Yes'.

The following process to run this program manually can be used if there is a need to download calendars to Oracle Demantra without collecting history.

See Time Units, page 7-3.

To download calendars:

1. Navigate to the Collection Utility user interface.

Collections > Oracle Systems > Download Calendars.

The Collection Utility window with the Download Calendars program name.

The screenshot shows the 'Collection Utility' window in Oracle Applications. The window title is 'Oracle Applications - demc2dv1'. The menu bar includes File, Edit, View, Folder, Tools, Window, and Help. The toolbar contains various icons for file operations. The main area is divided into sections: 'Run this Request...' with fields for Name (Download Calendars), Parameters, and Language (American English); 'At these Times...' with a field for Run the Job (As Soon as Possible); and 'Upon Completion...' with a checked checkbox for Save all Output Files, and fields for Layout, Notify, and Print to (noprint). Buttons for Copy..., Language Settings..., Debug Options, Schedule..., Options..., Help (C), Submit, and Cancel are visible.

2. There are no program parameters. Click Submit.

The calendars specified in the Calendar List are downloaded. See Setting Up the Calendar List, page 14-65.

Purging Data Before Import

Import Integration Profiles bring in forecasts such as manufacturing and sales forecasts, as well as data streams from Advanced Supply Chain Planning. For such data streams, setting the Purge Data Before Import profile option to 'Yes' causes the data in Oracle Demantra to be erased before importing the new version of the data stream. For a

specific combination's data to be purged, at least one occurrence of the combination is required in the incoming data set. Thus the desired behavior is:

Example

Old Sales Forecast in Oracle Demantra

Jan-07	Feb-07	Mar-07
500	600	700

Collected Sales Forecast

Feb-07	Mar-07	Apr-07
800	900	950

Run the Integration Import profile with the Purge Option set to 'Purge All Data.'

New Sales Forecast in Oracle Demantra

Jan-07	Feb-07	Mar-07	Apr-07
	800	900	950

In the example, the new sales forecast starts in February and not in January. Once the download takes place, Oracle Demantra shows a null value in January.

This contrasts with the following behavior, which is obtained by setting the 'Purge Data Before Import' integration profile option to 'No Purge'.

Old Sales Forecast in Oracle Demantra

Jan-07	Feb-07	Mar-07
500	600	700

Collected Sales Forecast

Feb-07	Mar-07	Apr-07
800	900	950

Run the Integration Import profile with the Purge Option set to 'No Purge.'

New Sales Forecast in Oracle Demantra

Jan-07	Feb-07	Mar-07	Apr-07
500	800	900	950

For more information about using the integration interface to set purge and load options for each series, see *Configure Series Load and Purge Options*, page 13-13.

Configure Series Load and Purge Options

The Integration Interface provides the ability to set purge and load options for each series. This controls whether the import integration profile overrides, accumulates, or purges (nulls out) the data for the specified integration profile date range.

For purge details, see the section "Configure Series Load and Purge Options, page 13-13" in the chapter "Demantra Data Tables and Integration Processes".

Download to Oracle Demantra

Collections and Download work together to move the newly collected series, level and reference data into the Oracle Demantra Tables. If the option to automatically launch download is not selected for Shipment and Booking History or non-history data has been collected, EP_LOAD and Import Integration Profiles are run to retrieve data from the staging area into Oracle Demantra Tables.

EP_LOAD

For members and history series, which are downloaded via the EP_LOAD mechanism, the mode of update is:

- If there is a new member, it is added in Oracle Demantra.
- If a member has been deleted in the E-Business Suite source, it stays in Oracle

Demantra along with all series data for combinations that include the member. The administrative user must manually delete the member in Oracle Demantra.

- Series data in the staging area overwrite the series data in Oracle Demantra, for the combinations that are represented in the staging area.
- Series data in Oracle Demantra for combinations that are not in the staging area are left unchanged.
- The staging area is erased after the download.
- All series data in Oracle Demantra, for all combinations, are set to null before the download actions take place.

Import Integration Profile

For all other series, which are downloaded via the Import Integration Profile mechanism, the behavior is the same, except for those series loaded via import integration profiles with the Purge Option set to: Purge all data.

Three EP_LOAD Workflows

The Demand Management System Administrator executes the required download procedures, EP_LOAD and Import Integration Profiles, from the Oracle Demantra Workflow Manager. There are a total of three EP_LOAD workflows, one EP_LOAD workflow for each of the following series:

- Item members
- Location members
- Shipment and Booking History

View according to Schema Groups:

EBS Workflows

New

Modify

Delete

Schema ID	Schema name	Owner	Creation Date	Last Modified	Instances	Status	Action			
631	EBS Full Download	din	12/16/06	02/12/07	0		Edit	Start	Schedule	Delete
632	EBS Return History Download	din	12/16/06	02/12/07	0		Edit	Start	Schedule	Delete
633	EBS Price List Download	din	12/16/06	03/22/07	0		Edit	Start	Schedule	Delete
634	EBS Upload Local Forecast	din	12/16/06	02/12/07	0		Edit	Start	Schedule	Delete
635	EBS Upload Global Zone Forecast	din	12/16/06	02/12/07	0		Edit	Start	Schedule	Delete
636	EBS Upload Local Fcst, Demand Class	din	12/16/06	02/19/07	0		Edit	Start	Schedule	Delete
637	EBS Upload Global Zone Fcst, Demand Class	din	12/16/06	02/19/07	0		Edit	Start	Schedule	Delete

[Process Log](#)

[New Schema](#)

[Refresh](#)

Demand Management Functional Output

Export **Integration Profiles** upload forecast and other relevant data to Oracle Advanced Planning and Scheduling applications, or to a legacy planning system.

Oracle Demantra Demand Management functional outputs:

- Forecast and demand priority for Advanced Supply Chain Planning
- Forecast and forecast accuracy for Inventory Optimization
- Forecast for Strategic Network Optimization

There are four upload workflows:

- EBS Upload Local Forecast
- EBS Upload Global Zone Forecast
- EBS Upload Local Forecast, Demand Class
- EBS Upload Global Zone Forecast, Demand Class
 - The Upload Forecast workflow schema name: APPS is hard coded.

- The Global forecast is output at the "Customer/All Org" hierarchy level.
- Organizations may belong to multiple instances.
- Oracle Demantra calculates functional outputs for 26 weeks when the base time unit is 'Week', or for one year when the base time unit is 'Month'. See Setting and Modifying the Base Time Unit, page 7-5.

Upload from Oracle Demantra

Oracle Demantra Demand Management provides the following redefined export integration profiles:

- EBS Upload Local Forecast
- EBS Upload Global Zone Forecast
- EBS Upload Local Forecast, Demand Class
- EBS Upload Global Zone Forecast, Demand Class

The profile parameters include:

- Horizon range and forecast output levels defined as part of integration profile
- Multiple data series tied to export integration profiles.

The data series internal names must follow the naming convention:

- Forecast: FCST_
- Priority: PRTY_
- Forecast accuracy MAPE: ACRY_MAPE_
- Forecast accuracy MAD: ACRY_MAD_(not Phase 1)
- Destination key: DKEY_

User-created series that users want to expose to downstream applications, such as Oracle Inventory Optimization, Oracle Advanced Supply Chain Planning, and Oracle Strategic Network Optimization, need to respect the above naming conventions and need to be added to user-created export integration profiles.

Note: The Upload Forecast workflow schema name: APPS is hard

coded.

The following integration profiles are manually launched as part of a Workflow after the forecast is approved:

1. Local Forecast
 - Series: Demand Priority, Final Forecast, Mean Absolute Pct Err, Item Destination Key, Organization Destination Key
 - Output Levels: Item, Organization, Week
2. Global Zone Forecast
 - Series: Demand Priority, Final Forecast, Mean Absolute Pct Err, Item Destination Key
 - Output Levels: Item, Zone, Week
3. Local Forecast with Demand Class
 - Series: Demand Priority, Final Forecast, Mean Absolute Pct Err, Item Destination Key, Organization Destination Key
 - Output Levels: Item, Organization, Week, Demand Class
4. Global Zone forecast, Demand Class
 - Series: Demand Priority, Final Forecast, Mean Absolute Pct Err, Item Destination Key
 - Output Levels: Item, Zone, Week, Demand Class

Using Alternate MAPE

In the four pre-configured integration profiles listed above, the in-sample MAPE series Mean Absolute Pct Err series is used. This series is automatically populated by the engine and due to its availability serves as the default MAPE value used. A more accurate out-of-sample MAPE value can be achieved but requires additional steps to be followed.

1. At the end of every planning cycles approved forecast should be archived and frozen, this is done by executing the Archive Forecast workflow
2. As new history becomes available archived forecast results are compared to actual historical results with the output generated as out-of-sample MAPE
3. Out of sample MAPE can be seen in series 1 Month Fcst Lag WMape

4. Results of this series can be saved in the Database using Batch Logic Engine workflow step. The Batch logic engine evaluates series including both server and client expression calculations and saves the resulting values. The Batch Logic Engine should be configured and executed at the level at which the weighted MAPE is most useful.
5. If available a series displaying the out of sample MAPE values can replace the series Mean Absolute Pct Err above

Publish the Forecast to CP

To publish the forecast from Oracle Demantra to Oracle Collaborative Planning:

Note: Before running the 'Publish Forecast to Customer' concurrent program, the forecast from Demantra has to be published using the seeded Demantra workflow 'Upload CP forecast'. This workflow uploads the Demantra forecast to the MSD_DP_SCN_ENTRIES_DENORM table. This workflow uses the 'DM-Forecast-for-CP' export data profile to export the values of the 'Consensus Forecast' series at the Item-Organization-Site levels.

1. Choose Advanced Planning Administrator responsibility.
2. Select Collaboration > Publish Forecast to Customer.
3. Click on Parameters field and specify the following parameters:

- Forecast designator: Enter a name for the forecast data.
- Publish Forecast Type: Select Sales forecast. (Possible values are Sales Forecast/Order Forecast)
- Demand Plan Name: Select the demand plan named 'Demantra'
 - **Note::** Forecasts published from Demantra will be available under a demand plan value named 'Demantra'
- Scenario Name: Select the Demantra export data-profile name. The seeded export data profile name is 'DM-Forecast-for-CP'
 - **Note:** 'DM-Forecast-for-CP' is the seeded export data profile name in Demantra. You can publish the forecast using your own custom data profile name if need be. The data profile used, should export the forecast out of Demantra at the item, organization and site levels
- Organization, planner, item, customer, and customer site (available if customer is selected): optional filters to restrict the forecast information

Upload Forecast

Forecasting by Line of Business (LOB)

Import:

Existing APS and Oracle Demantra capabilities accommodate forecasting by line of

business (LOB).

You can run Collections for a collections group. This restricts the uploaded forecast to the set of organizations that corresponds to a line of business.

You can download for a specific line of business by modifying the data security of the seeded import integration profiles. For data downloaded via EP_LOAD, which does not have a mechanism to control data scope, a business process is provided to stagger the timing of collections for the different lines of business.

Caution: There is a risk that if multiple lines of business run collections very close in time to each other, a single EP_LOAD run may pull in data from multiple lines of business.

See Line Of Business Configuration and Execution, page 14-56.

The Oracle Demantra forecasting engine will run forecasts only for combinations inside of a line of business by invoking the LOB process. Oracle provides user data security for LOB-specific access to data.

A predefined stored procedure called from the LOB Process workflow updates database column do_forecast so that only combinations within the LOB are forecast and run the forecasting engine. The stored procedure also copies existing forecast numbers in the non line-of-business combinations into the new forecast version that is created by the running the forecast engine.

Export:

- Upload data for a line of business only by modifying the data security of seeded export integration profiles.
- An administrator user procedure creates export integration profiles with level value access filtered down to LOB scope.

See Creating a Data Export Profile.

Forecast Tree

The forecast tree organizes data for use by the Analytical Engine. In general, forecasting is most accurate when it can be performed at the lowest possible allowed aggregation level. However, sometimes there is not enough data at that level for all combinations. For those combinations, the Analytical Engine aggregates the data to a higher level and tries to generate a forecast there. See Levels and the Forecast Tree.

- Forecast order for items:
 1. Item
 2. Category

- Forecast order for locations:
 1. Customer Zone
 2. Zone
 3. Organization

Line Of Business Configuration and Execution

When an implementation includes multiple business units, the business units require the batch engine to be executed at different times. Since every batch engine run creates a new forecast for the entire population, this could cause issues if the batch engine runs concurrently for different lines of business. A tailored process supports generating forecasts at different intervals across different parts of the business.

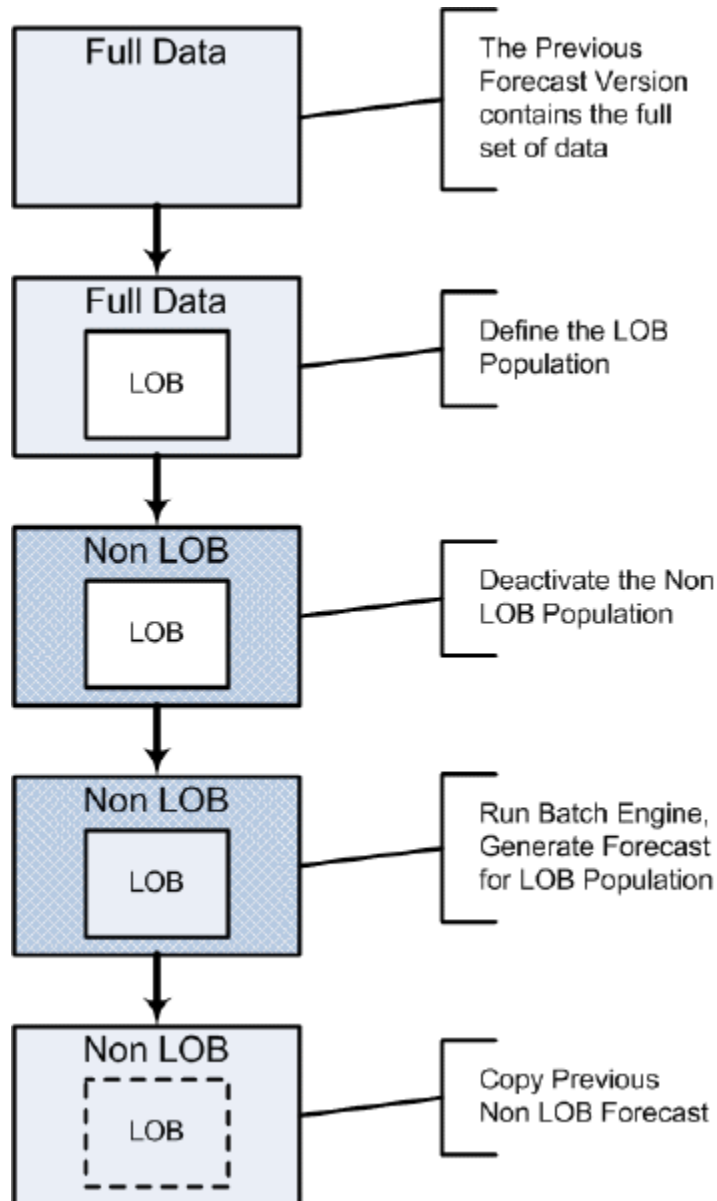
This process should be used when the following conditions apply:

- There is a need for two or more distinct forecast cycles.
- The populations requiring different forecast cycles are easily identifiable.
- The population can be defined either as a member, or as members of one aggregation level, such as Segment, Organization, or Brand.

Important: The LOB engine, while smaller than the batch engine, can still be resource intensive and should not be run when users are in the system. The LOB engine should be activated and scheduled during user downtime and when the simulation engine is not on.

Line of Business Solution Engine Process

1. Define the population participating in the LOB batch run.
2. Exclude all other populations from LOB batch run.
3. Run the batch engine to generate a new forecast for the LOB population. Store the new forecast as a new forecast version.
4. Copy the previous forecast version to the new forecast version for the non LOB population



LOB Procedures

For the initial download, the DM/ SOP System Administrator specifies the organizations to be collected and downloaded during Shipment & Booking History Collection & Download with the Internal Sales Order parameter set to NO. When the processes complete, s/he logs into Business Modeler to add the name(s) of the LOB(s) to the preseeded LOB Level and associate the LOB(s) with the organizations that rollup to the LOB either by using the edit tools with a worksheet or Member Management. After the LOB is setup, Shipment & Booking History Collections & Download are rerun separately for each LOB, according to its planning cycle, with the ISO parameter set to

YES. Users are given access to the LOB or restricted from accessing the LOB via User Security. The DM/SOP Sys Admin modifies the Line of Business - Forecast workflow and the Planning Group workflow for each LOB.

Initial setup for LOBs with ISOs:

1. Set up Instance and include organizations that are part of an LOB
2. Run Standard Collection and set up calendars, price lists and new products
3. Run Shipment and Booking History Collection with the 'Collect Internal Sales Orders' parameter UNCHECKED for the initial Download
 - If the Admin mistakenly checks the 'Collect Internal Sales Orders' parameter for the initial Download, the Order Alignment feature, available in the first release of SOP, can be used to purge the series.
4. After Download, add the LOBs as a member of the LOB level and associate the relevant organizations
5. Assign the LOB to the relevant users via User Security in Business Modeler
6. Ensure the users with access to the LOB are in the same User Group
7. Create a copy of the Planning Group workflow with the following changes
 - Select the User Group given access to the LOB as the recipient for the Group Step
 - Enter the LOB's Final Approver's ID for the Completed/Incomplete steps
8. Rerun Shipment and Booking History Collection with 'Collect Internal Sales Orders' parameter checked
9. Edit the SetLOBParameters' step in the Line of Business - Forecast workflow and set the parameters
10. Modify the Approval step in the Line of Business – Forecast workflow and enter the schema id of the relevant Planning Group workflow.
11. Start the Line of Business – Forecast workflow

The following steps are done by the DM/SOP Sys Admin to setup a component for an LOB and handle ISOs.

DM/SOP System Administrator Detailed LOB Setup

1. Specify the organizations of the LOB to be collected and downloaded when setting up Instances. For ease of use, Collection Groups can be used, but are not required,

to group the LOB's organizations.

Org	Description	Enabled	Collection Group
V1	Vision Operations	<input checked="" type="checkbox"/>	LOB 1
S1	Chicago Subassembly Plant	<input checked="" type="checkbox"/>	OTH
M1	Seattle Manufacturing	<input checked="" type="checkbox"/>	LOB 1
M2	Boston Manufacturing	<input checked="" type="checkbox"/>	SD
D1	Singapore Distribution Center	<input checked="" type="checkbox"/>	SD
D2	Miami Distribution Center	<input checked="" type="checkbox"/>	SD
PM	Vision Project Mfg	<input checked="" type="checkbox"/>	PROJ
P1	Atlanta Manufacturing	<input checked="" type="checkbox"/>	PRJORG
P2	Los Angeles Manufacturing	<input checked="" type="checkbox"/>	PROJ
P3	San Diego Manufacturing	<input checked="" type="checkbox"/>	PRJORG

Selecting Organizations of LOB 1

2. Setup calendar, price lists and new products.
 - Components with multiple LOBs must use the same base time bucket.
3. Run Shipment & Booking History collections for the Organizations specified in Step 1 with the Collect Internal Sales Order parameter set to NO:
 - If the Launch Download parameter is set to YES, proceed to Step 4
 - If Launch Download = NO, run the EBS Full Download workflow

Parameters

Instance: **TST**

Collection Group: **LOB 1** All Enabled Organizations

Collection Method: [Empty]

Date Range Type: [Empty]

History Collection Window Days: [Empty]

Date From: [Empty]

Date To: [Empty]

Booking History - Booked Items - Booked Date: **No**

Booking History - Booked Items - Requested Date: **No**

Booking History - Requested Items - Booked Date: **No**

Booking History - Requested Items - Requested Date: **No**

Shipment History - Shipped Items - Shipped Date: **No**

Shipment History - Shipped Items - Requested Date: **No**

Shipment History - Requested Items - Shipped Date: **Yes**

Shipment History - Requested Items - Requested Date: **No**

OK Cancel Clear Help

LOB Collection & Download

4. Open a worksheet with Line of Business and Organization as the selected Aggregation Levels
 1. Right click Line of Business>New Line of Business
 2. Enter the LOB's name and click Create

Demand Management

File Worksheet Edit View Options Data Help

Organization - Line of Business

AR

DT Sales Area

Game Corp

Default Line of Business

Global Corp

Default

Mega Prod

World Brand

New line of business : LOB1 (Input)

Name: LOB1

New line of business

View line of business

Edit line of business

Delete line of business

Add note

Add New Member to LOB Level

3. For each Organization that is a member of the LOB, right click the Organization>Edit Organization
4. Select the LOB from the dropdown list for Line of Business

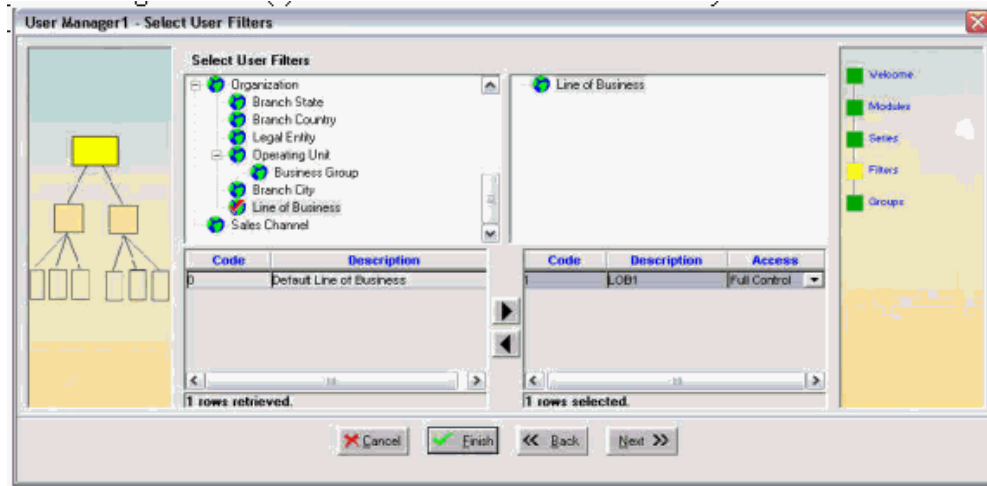
Edit organization : Global Corp (Input)

Name	Global Corp
Branch State	unassociated
Branch Country	unassociated
Branch Category Code 1	unassociated
Branch Category Code 5	unassociated
Branch Category Code 3	unassociated
Branch Category Code 2	unassociated
Branch Category Code 4	unassociated
Branch City	unassociated
Legal Entity	unassociated
Operation Unit	Global Corp
t_ep_job	<div> LOB1 Default Line of Business LOB1 </div>

Cancel Ok

Assigning an Org to an LOB

5. Assign the LOB(s) to the relevant users via User Security in Business Modeler



Assigning LOB to Users

6. Ensure the users with access to the LOB are in the same User Group with the Business Modeler menu Security>Modify Group
7. For each LOB, start Workflow Manager and create a copy of the Planning Group workflow. The recommendation is that the workflow name contains the LOB for which it is used, e.g. LOB1 Planning Group workflow. The following parameters must be changed::
 - The User Group selected as the recipient in the Group Step must be the Group given access to the LOB
 - Final Approver's ID for the Completed/Incomplete steps must be the user that approves the forecast for the LOB
8. For each LOB, using Workflow Manager edit Line of Business – Forecast workflow and change the parameters for step 'SetLOBParameters'. The parameter settings are:

Parameter 1 - should be comma (,) separated list of member descriptions.

Parameter 2 - should be level title..

 - Edit the Approval process step in this workflow and enter the parameters to identify the Planning Group workflow for the specific LOB.
 - The Approval step in the workflow must have the LOB's Planning Group workflow schema number as the schema_id value
9. Run Shipment & Booking History collections again for the Organizations or Collection Groups in Step1 with the Collect Internal Sales Order parameter set to YES:

- If Launch Download = NO, run the EBS Full Download Workflow

Parameter	Value
Date From	
Date To	
Booking History - Booked Items - Booked Date	No
Booking History - Booked Items - Requested Date	No
Booking History - Requested Items - Booked Date	No
Booking History - Requested Items - Requested Date	No
Shipment History - Shipped Items - Shipped Date	No
Shipment History - Shipped Items - Requested Date	No
Shipment History - Requested Items - Shipped Date	Yes
Shipment History - Requested Items - Requested Date	No
Collect Internal Sales Orders	Yes
Collect All Order Types	Yes
Include Order Types	
Exclude Order Types	
Launch Download	Yes

Collect Internal Sales Order Parameter

DM/SOP System Administrator Post Initial Setup Process

1. The Collection & Download process is run for the specific LOB by specifying the Organizations or Collection Group that comprise the LOB in Shipment & Booking History collections
2. Run the Line of Business - Forecast workflow
 - Run LOB Reset workflow if the forecast workflow errors out
3. Upload the forecast using the LOB specific Upload workflow when approval process completes
4. Archive the forecast for all LOBs
 - For LOBs with different planning cycles, the forecast is archived on the more frequent schedule
 - A business process must be put in place to manage the archiving schedule to avoid conflicts for LOBs with different planning cycles

Troubleshooting the Line of Business Forecast Workflow

If problems occur during the LOB Process, the *next* attempt to run the Line of Business Forecast Workflow will not proceed. Due to the LOB process deactivating the non-LOB combinations, it is not desired to have the next LOB Process begin without successfully completing the previous run.

To remedy this, run the LOB Reset workflow. This workflow computes whether combinations should be active, and resets deactivation of the non-LOB population. Since this may take some time on larger environments LOB Reset is not automatically

run as part of the Line of Business Forecast Process.

More information regarding the Line of Business Forecast Process can be seen in the Line of Business Forecast Workflow and in the table `integration_lob_error`.

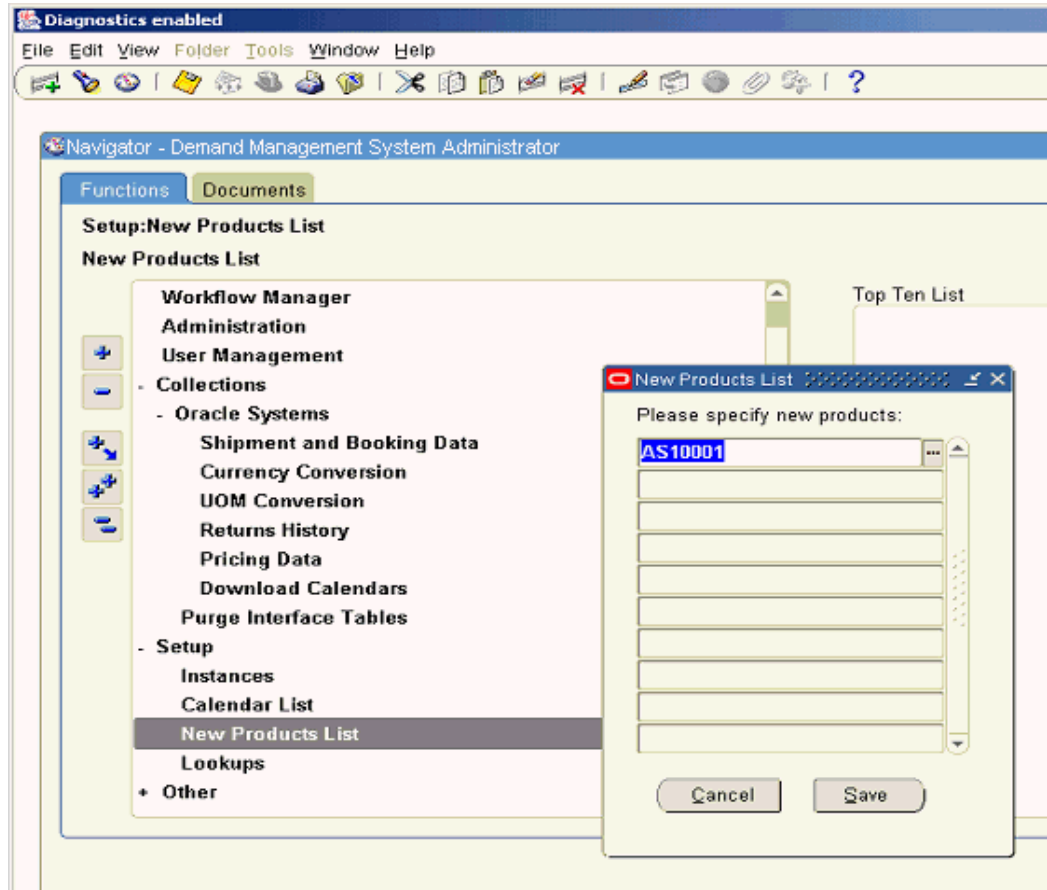
Setting Up the New Products List

This setup specifies certain new items, for which there is no history data, that need to be downloaded into Oracle Demantra.

All items available in Advanced Supply Chain Planning Operational Data Store are displayed in the list of values. Items across multiple instances can be specified. Items that are downloaded into Oracle Demantra are not processed in subsequent collections.

Once a new item is added to the New Product list and downloaded, it remains in the list unless manually deleted. If a new item is downloaded into Oracle Demantra and then subsequently phased out, it remains in Oracle Demantra until manually deleted. If this item needs to be reintroduced in the future, delete the item from the item list in the New Products List user interface, and then add it back to the list. This ensures treatment as a new item that will be downloaded into Oracle Demantra.

Note: All Items added to the New Products list download into Oracle Demantra. No attribute checking occurs.



Setting Up the Calendar List

This setup specifies which manufacturing and fiscal calendars need to be collected and downloaded into Oracle Demantra. A user interface accessed from the E-Business Suite Demand Management menu structure allows the Demand Management System Administrator to list the E-Business Suite calendars to be used for Demand Management analysis.

All calendars available in Advanced Supply Chain Planning Operational Data Store (ASCP ODS) display in the list of values. Calendars across multiple instances can be specified.

Note: If a calendar is removed from this list, it is not deleted from Oracle Demantra automatically. Subsequent collections will not collect and download that calendar.

Example

Take the following example of a Fiscal Calendar collected from the E-Business Suite source. Assume for this example that the Oracle Demantra base time unit is week, with

each week starting on Monday.

Vision Corporate Calendar

		Week	Start Monday	End Sunday
Fiscal Quarter 2007-1	Fiscal Month 2007-1	1	1/1/2007	1/7/2007
		2	1/8/2007	1/14/2007
		3	1/15/2007	1/21/2007
		4	1/22/2007	1/28/2007
	Fiscal Month 2007-2	5	1/29/2007	2/4/2007
		6	2/5/2007	2/11/2007
		7	2/12/2007	2/18/2008
		8	2/19/2007	2/25/2007
	Fiscal Month 2007-3	9	2/26/2007	3/4/2007
		10	3/5/2007	3/11/2007
		11	3/12/2007	3/18/2007
		12	3/19/2007	3/25/2007
		13	3/26/2007	4/1/2007
Fiscal Quarter 2007-2	Fiscal Month 2007-4	14	4/2/2007	4/8/2007
		15	4/9/2007	4/15/2007
		16	4/16/2007	4/22/2007
		17	4/23/2007	4/29/2007

		Week	Start Monday	End Sunday
Fiscal Quarter 2007-3	Fiscal Month 2007-5	18	4/30/2007	5/6/2007
		19	5/7/2007	5/13/2007
		20	5/14/2007	5/20/2007
		21	5/21/2007	5/27/2007
	Fiscal Month 2007-6	22	5/28/2007	6/3/2007
		23	6/4/2007	6/10/2007
		24	6/11/2007	6/17/2007
		25	6/18/2007	6/24/2007
		26	6/25/2007	7/1/2007
	Fiscal Month 2007-7	27	7/2/2007	7/8/2007
		28	7/9/2007	7/15/2007
		29	7/16/2007	7/22/2007
		30	7/23/2007	7/29/2007
	Fiscal Month 2007-8	31	7/30/2007	8/5/2007
		32	8/6/2007	8/12/2007
		33	8/13/2007	8/19/2007
		34	8/20/2007	8/26/2007
	Fiscal Month 2007-9	35	8/27/2007	9/2/2007

		Week	Start Monday	End Sunday
Fiscal Quarter 2007-4	Fiscal Month 2007-10	36	9/3/2007	9/9/2007
		37	9/10/2007	9/16/2007
		38	9/17/2007	9/23/2007
		39	9/24/2007	9/30/2007
	Fiscal Month 2007-11	40	10/1/2007	10/7/2007
		41	10/8/2007	10/14/2007
		42	10/15/2007	10/21/2007
		43	10/22/2007	10/28/2007
	Fiscal Month 2007-12	44	10/29/2007	11/4/2007
		45	11/5/2007	11/11/2007
		46	11/12/2007	11/18/2007
		47	11/19/2007	11/25/2007
	Fiscal Month 2007-12	48	11/26/2007	12/2/2007
		49	12/3/2007	12/9/2007
		50	12/10/2007	12/16/2007
		51	12/17/2007	12/23/2007
		52	12/24/2007	12/30/2007

Run the Download Calendars program with the calendar_id parameter set as "Vision Corporate Calendar" to create the following levels and hierarchy:

- Week (existing level)

- > Fiscal Month (Vision Corporate Calendar) (new level)
- > Fiscal Quarter (Vision Corporate Calendar) (new level)
- > Fiscal Year (Vision Corporate Calendar) (new level)

The level members roll up to parent members (weeks to fiscal months, fiscal months to fiscal quarters, fiscal quarters to fiscal years) according to the calendar definition. The name of the generating calendar (Vision Corporate Calendar, in the above example) in the level names distinguish the newly created levels. In this way, Oracle Demantra accommodates analysis along multiple E-Business Suite calendars. See *Setting and Modifying the Base Time Unit*, page 7-5.

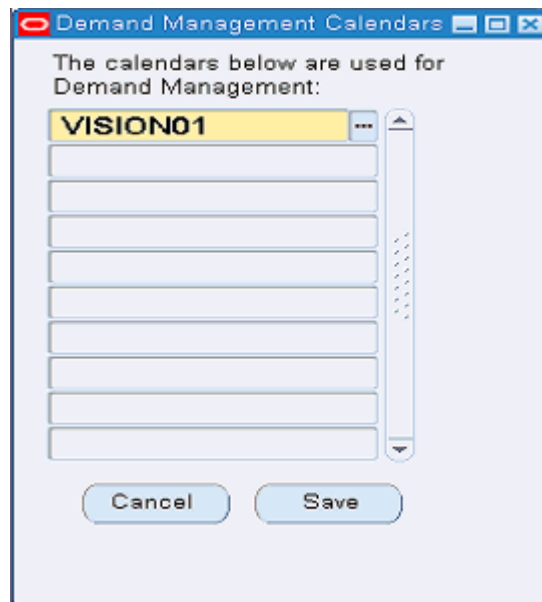
Calendars are always collected using Full Refresh of Shipment and Booking History Collections. Calendars can also be collected by launching Download Calendars from the menu. . In other words, all dates are collected. A check ensures the collected dates fall within the Oracle Demantra default dates. Only levels higher than the lowest model defined time aggregation are loaded. For example, when the base time unit is 'week', only month (period), quarter and year load. For a monthly system, only quarter and year load.

To set up the Calendar List:

1. Navigate to the Demand Management Calendars user interface.

(Setup > Calendar List)

The Demand Management Calendars user interface appears.



This user interface validates that any calendar listed has a time bucket level that matches the Oracle Demantra base time unit. For all calendars defined in the

Calendar List, a stored procedure, invoked via the Oracle Demantra Download Calendars workflow, creates corresponding time hierarchies in Oracle Demantra. If the time hierarchies do not already exist, the procedure adds them to the Demand Management component.

2. To add a calendar to the list, select an empty row.
3. Select a calendar from the drop-down list of values.
4. Click Save.

Base Time Unit

The base time unit is used by the Demand Planner Data Model to aggregate the source data to the specified time bucket size. Allowed settings of the base time unit (time bucket size) are:

- day
- week
- Gregorian month

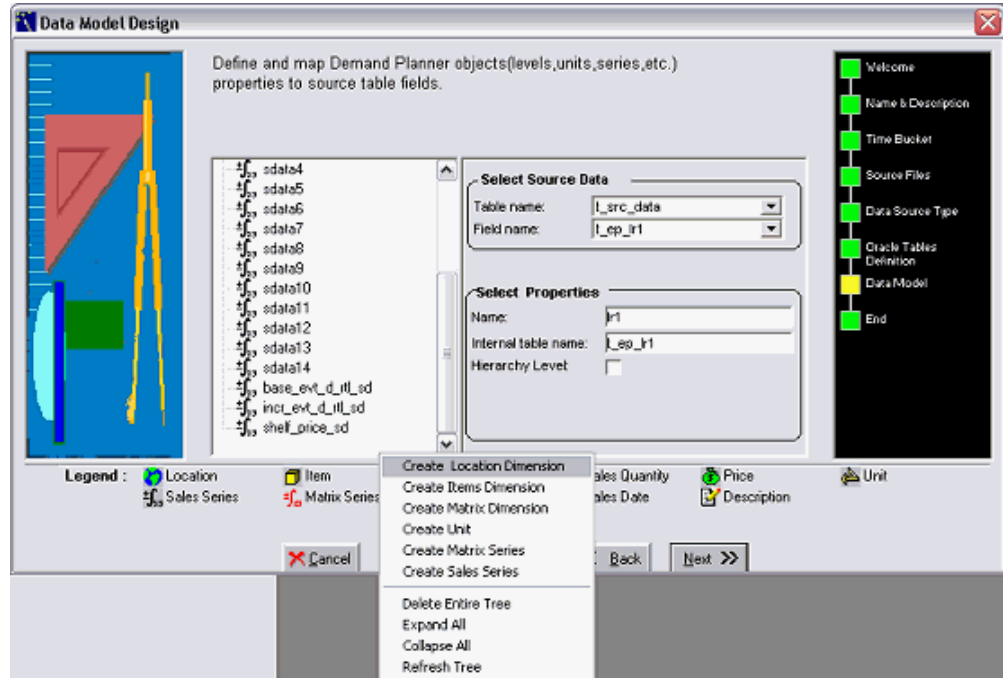
See Setting and Modifying the Base Time Unit, page 7-5.

Creating a New Leaf Level

The Demand Management System Administrator can create a new leaf level for a hierarchy, such as adding a sales representative as a location new leaf level.

1. Navigate to the Open Data Model window.
Business Modeler > Data Model > Open Data Model
2. From the Open Data Model window, navigate to the Data Model Design window. In the left pane, with no node selected, select Create Location Dimension from the right-click menu.

The Data Model Design window appears.



3. Complete the table and field names that will hold level members.
4. Customize the collections code.
5. Run collections.
6. Rebuild the Oracle Demantra model. See Building the Data Model and Manipulating Existing Data Models.

Creating a New Top Level

The Demand Management System Administrator can create a new top level for a hierarchy, such as creating a level for Product Line located above Product Family level.

To create a new top level:

1. Navigate to the Configure Levels window.
Business Modeler > Configuration > Configure Levels
The Configure Levels window opens.
2. Select, and then right-click the current top level node. For example, select product category.
3. From the right-click menu, choose New level.

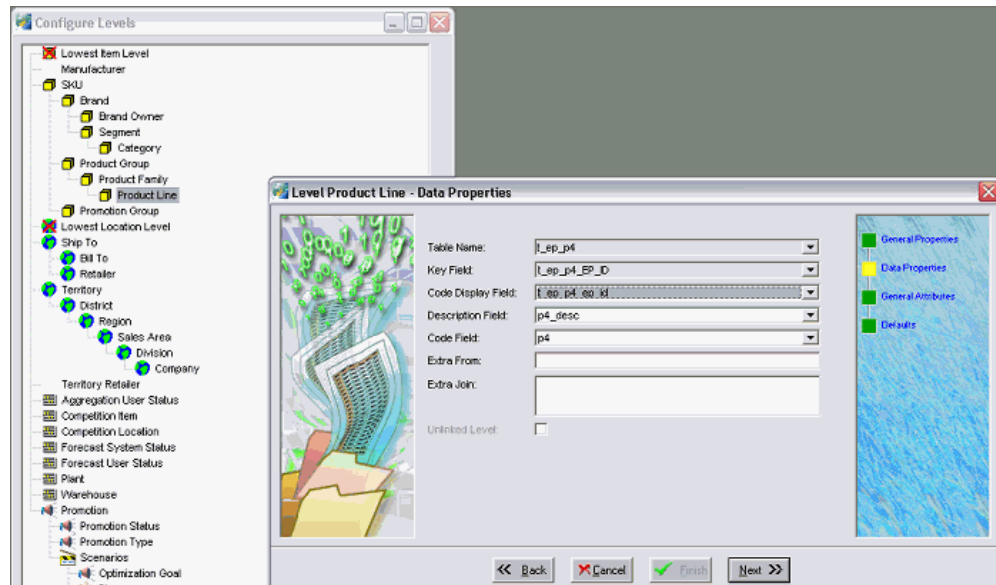
The General Properties window opens.

4. Complete the following General Properties:

- Level Name, such as Super Category
- Type: Product Level
- Status: Enabled.

5. Click Next.

The Level <current node> Data Properties window opens.



6. Define the Table Name and Column that house members of the new level. You are selecting a field in the database table to house members of the new level.

7. Customize the collections code to populate the appropriate table and column with product line member data.

8. Click Next.

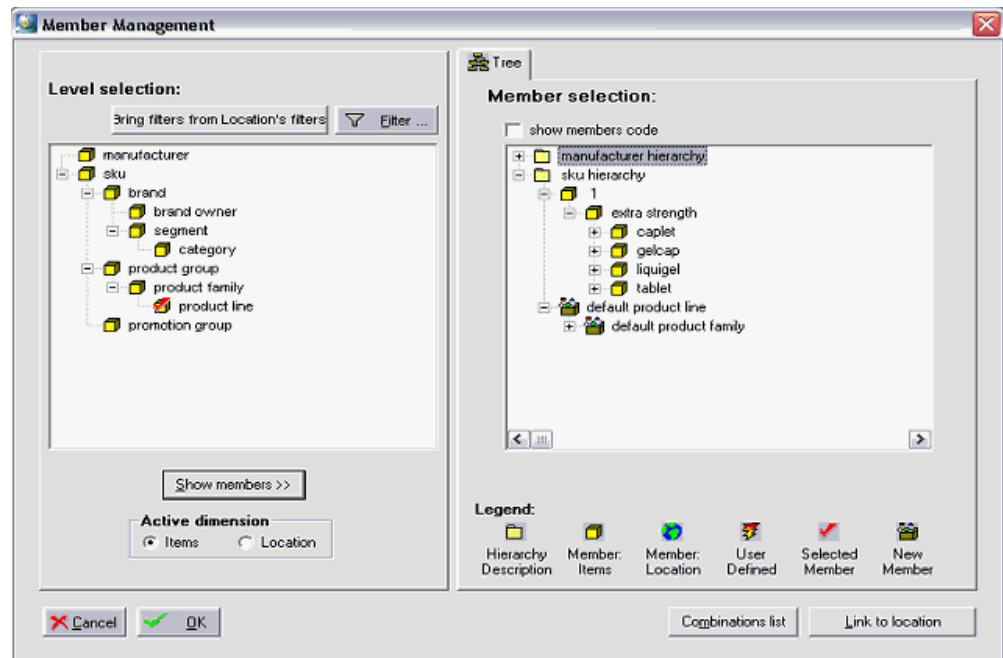
9. Run collections.

10. Next define level value associations of specific parent product lines to child product families. Navigate to Member Management.

Demand Analyst > Tools > Member Management

11. Select the Product Line node.

12. Drag Product Families under the appropriate Product Lines.



13. Click Save.

Note: To Create a New Hierarchy on Top of an Existing Leaf Level, perform multiple iterations of the following procedure for Creating a New Top Level.

Creating a New Intermediate Level

Configure Levels does not directly support the addition of intermediate levels. To accomplish the addition of an intermediate level, the process renames an existing level, and then add a new top level.

For example:

1. Existing level name: 'Category'
 2. Rename 'Category' as 'Subcategory'
 3. Create a new top level named 'Category'.
1. Navigate: Business Modeler > Configuration > Configure Levels
The Configure Levels window opens.

2. Select, and then right-click the current top level node.
3. Rename the current top level to the name of the new intermediate level.
4. Use the Creating a New Top Level procedure to add a new top level above the intermediate level.

Deleting a Level

1. 1. Navigate: Business Modeler > Data Model > Open Data Model
2. Go to Data Model step
3. In the left hand pane, select the level to be deleted.
4. From the right-click menu, select Delete.
5. Rebuild the Oracle Demantra model – See Building the Data Model and Manipulating Existing Data Models.

Approval and Upload Setup Process

During implementation, the Administrator configures the approval process by specifying a Reviewer/Business Owner who is responsible for final approval of the forecast. One Business Owner (Final Approver) has final approval responsibility for the forecast produced by their group of Analysts. Approval is accomplished by use of a Final Approval Series by User Group and by workflow notifications. The Demand Forecast Workflow manages the planning cycle and calls the Planning Group Workflow that is used in the notification process.

Oracle seeds a user group setup with dummy users. The Administrator edits this group to add the names of the Analysts whose forecasts need approval from the Final Approver. The Administrator also edits a seeded Planning Group Workflow to specify the ID of the Final Approver notified when the Analysts' forecasts are ready for review. The seeded workflow can be used as a template if additional group workflows are needed. The Time Property is set to send an alert after four days if any Analysts have not completed their review, and sends an alert and times out after five days if any Analysts still have not completed their review.

This workflow:

1. Notifies the Final Approver and Analysts that the Forecast is available
2. Polls the users to check whether they are 'Done'.
 1. After four days, if any Analysts are not done, a messages is set to the Final

Approver.

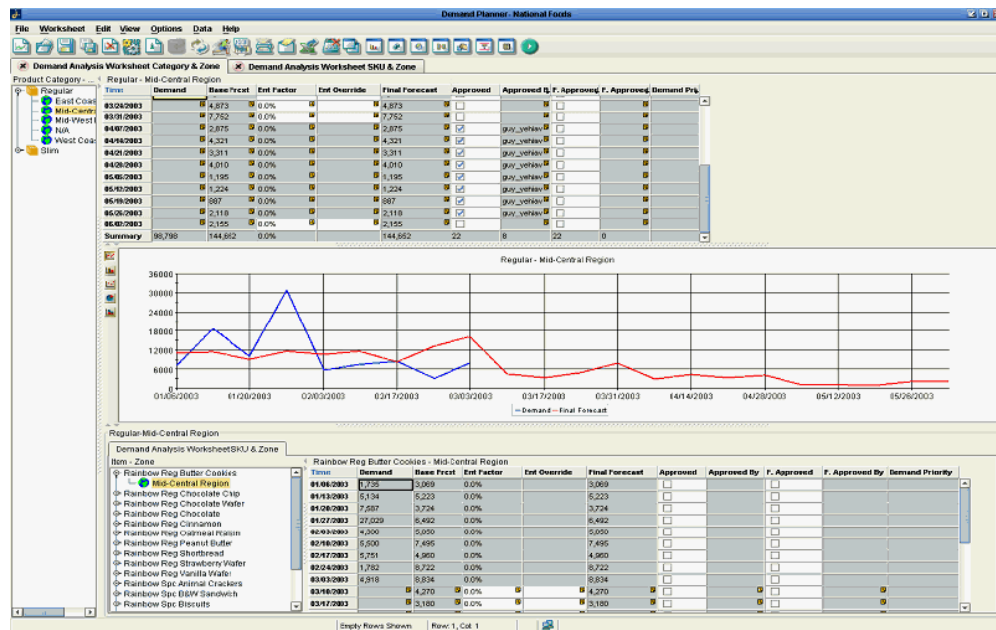
2. After five days, if any Analysts are still not done, a message is again sent to the Final Approver, and the workflow times out. Processing returns to the calling workflow named: 'Demand Forecast'.
3. If all Analysts check 'Done.' before the time out occurs, the Final Approver is notified that the Analysts' forecasts are ready for review. Processing returns to the calling workflow named: 'Demand Forecast'.

Analysts are notified when a forecast is available for review and modification. After their modifications are complete, they select 'Done' in their My Tasks notification. The Workflow polls the user's status, and then notifies the Final Approver when all Analysts are done. If one or more Analysts have not finished their approvals one day prior to the due date, a reminder is sent to the Analysts and Final Approver. On the due date, if any Analyst has not finished their approval process, an exception message is sent to the Final Approver and Analysts to the effect that one or more Analysts have not completed their forecast adjustments.

The Final Approver can lock the forecast by checking the F. Approve check box. After review, the Final Approver either approves or disapproves the forecast. If not approved, the Final Approver contacts the Analysts whose adjustments are in question. This step is not part of the Workflow. If approved, the Final Approval check box is checked and the forecast locked. A notification is sent to the Administrator and the Analysts to the effect that the forecast is available for upload. The Administrator uploads the forecast and other relevant data to the Planning Server.

1. After EP_LOAD completes successfully, the Final Approval and Final Approve By series are set to null, the Forecast is run, and a notification is sent to the Analysts and Final Approver with a Due Date specified.
2. Analysts analyze the forecast, make modifications and, when finished, select 'Done' in MyTasks for the forecast available task.
3. The Workflow polls the status of all Analysts in the User Group. When all Analysts have completed their approvals, a notification is sent to the Final Approver.
 - If all Analysts have not completed their adjustments, on the day before the due date, a reminder message is sent to the Analysts and Final Approver.
 - On the due date, if any Analyst has not completed their approval process, an exception message stating: 'One or more Analysts have not completed their adjustments' is sent to the Final Approver.
4. The Final Approver can lock the forecast at any time by checking the 'Final Approval' column. After review, the Final Approver approves the forecast by selecting 'Done' in MyTasks for the forecast notification. For one level review:

- If the Final Approver approves the forecast, a notification is sent to the Analysts and to the Administrator to initiate the upload.
 - If the Final Approver does not approve the forecast, s/he contacts the Analyst(s) whose adjustments are in question. This is not a Workflow step; it is a business process.
 - If the Administrator does not check the box in My Tasks within the specified time range, the Workflow times out.
5. When the notification from the Final Approver is received, the Administrator uploads the forecast and other relevant information, for example: Demand Priority, Forecast Accuracy.



Profile Options

Source Side Profiles

Profile Name	Description	Default Value
MSC: Configuration	This profile defines whether item descriptions are collected for the appropriate Demantra tables. At the site level, set to 'APS & CP' if you want to collect item descriptions. Set on both the source and destination instances if decentralized.	Set by the Administrator
MSD_DEM: Master Org	This profile defines the Product Family to which an Item rolls up.	Set by the Administrator
MSD_DEM: Category Set Name	This profile defines the Item to Category rollups in each instance.	Inv.Items See Note 1 following this table.
MSD_DEM: Conversion Type	This profile determines which currency conversion rates are collected from the General Ledger tables.	Corporate

Profile Name	Description	Default Value
MSD_DEM: Customer Attribute	<p>This profile option selectively brings the customer names into Oracle Demantra Demand Management to improve system performance. This profile holds the descriptive flexfield column name used to indicate if a customer in the customer table will be used by Demand Management. Only those customers in the Geography dimension that have this flexfield populated will be collected. This profile option value is one of the attribute columns of the RA_CUSTOMERS entity, which indicates whether or not the customer will be used for demand planning purposes.</p> <p>In the customer table, you need to reflect this in a descriptive field. All of the source views for the geography dimension that use the RA_CUSTOMERS entity filter using this attribute column. If the profile option is not provided, then no filtering will occur. If the profile option is provided, then only the entities in the geography dimension that have the attribute in the RA_CUSTOMERS entity specified as Yes will be collected. To set up Key Customers, go to the Customer setup screen in Oracle Applications. Select the relevant customer and set an available flexfield column to Yes. For example, if you use attribute10, then you need to use this information in MSD profile option setup, too. You also need in the source instance setup, the following information for profile option value MSD_DEM_CUSTOMER_ATTRIBUTE: list of values from ATTRIBUTE1 to ATTRIBUTE15.</p>	Set by the Administrator

Note: 1. Set the MSD_DEM: Category Set Name profile to a 'master level' category set. A master level category set does not allow multiple category roll up, such as an item rolling up to multiple categories within the same category set for the same organization.

Destination Side Profiles

Profile Name	Description	Default Value
MSC: Configuration	This profile defines whether item descriptions are collected for the appropriate Demantra tables. At the site level, set to 'APS & CP' if you want to collect item descriptions. Set on both the source and destination instances if decentralized.	Set by the Administrator
MSD_DEM: Currency Code	This profile designates the Demand Management base currency.	US Dollar
MSD_DEM: Two-Level Planning	This profile enables demand forecasts at the Product Family level on the basis of sales histories of member items.	Exclude family members with forecast control 'None'. See Note 2 following this table.
MSD_DEM: Schema	Set this profile value to the database schema name where the Oracle Demantra schema has been installed.	DMTRA_TEMPLATE
MSD_DEM: Data Profile for Price Lists	Set this profile to import integration data profile for price lists.	EBS Price List
MSD_DEM: Maximum seeded units available for price lists	This profile determines the number of slots available for price lists in Oracle Demantra.	30
MSD_DEM: Debug Mode	When set to 'Yes', this profile is used to print debug information to the output file of the concurrent request.	No
MSD_DEM: Host URL	Set this profile to the Oracle Demantra Application Server Host Name, Port Number and Application Name. This profile is used to invoke Oracle Demantra URLs from the E-Business Suite applications.	Set by the Administrator See Note 3 following this table.

Note 2:

You can collect all the product family members and their sales histories regardless of

the forecast control, as long as:

- The product family forecast control is 'Consume' or 'Consume & Derive', and
- The planning method for 'product family' and 'members' is *not* set to 'Not Planned'.

This is achieved by setting the profile value to: Collect all family members and their sales histories.

The MSD_DEM: Two-Level Planning profile default value, Exclude family members with forecast control 'None' enforces the behavior that only 'Consume' or 'Consume & Derive' product family members are collected.

Note 3:

Use the format:

- `http://<host name>:<port number>/<application name>`, or
- `http://<host IP address>:<port number>/<application name>`

For example `http://pc-anwroy-in:8090/Oracle Demantra`

Demantra Demand Management to EBS Service Parts Planning Integration

This chapter describes the integration between Oracle Demantra Demand Management and EBS Service Parts Planning.

This chapter covers the following topics:

- Introduction
- Integration Points
- Business Process
- Service Parts Planning Navigator Menus
- Integration Considerations
- Initial Demantra Setup
- Initial SPP/EBS Setup
- Importing SPP Data into Demantra
- Launching the Import Workflows from the Demantra Workflow Manager
- Service Parts Forecasting in Demantra
- Exporting SPF Data from Demantra to Service Parts Planning in EBS
- Integration Workflows and Integration Profiles
- Service Parts Forecasting Integration Import Profiles
- Service Parts Forecasting Integration Export Profiles

Introduction

Service supply chains often have millions of parts that are planned for, and therefore an even higher number of planned orders. As a result, manually releasing planned orders to execution is often not an option. At the same time, automatically releasing all

planned orders within the release timefence is not a good idea, since the planner may want to be alerted in specific scenarios (for example, when a single order is for a very large value) before the order is released. A planner may want to halt the auto-release of planned orders on an item in case of deviations in the forecast for the item. Some examples would be when the forecast has a very high error or the Leading Indicator forecast is tracking better than the statistical forecast. In such cases, the planner would want to review the forecast, make changes if required, and release planned orders only after incorporating these changes.

Demantra Demand Management is capable of providing the metrics that a planner can use to determine if any unusual scenarios may warrant their attention. SPP is integrated with Demantra Demand Management so that service parts planning information can be sent to Demantra for forecast generation and fine-tuning.

Three Service Parts Forecasting worksheets have been created specifically to support the modification and fine-tuning of the forecast. They include:

1. SPF: Analyze Forecast Organization Latest Revision
2. SPF: Analyze Forecast Latest Revision
3. SPF: Analyze Organization Base Model Spare

After fine-tuning the worksheets, new forecasts can be generated or simulated until a more acceptable forecast is arrived at using engine profiles that specifically support the service parts forecasting function. Then, the planner can return the Demantra service parts forecast and metrics to SPP for planning, analysis or to drive the stopping of the auto-release process. The metrics returned include:

- SPF MAPE (In Sample)
- SPF MAPE (Out of Sample)
- SPF Forecast Volatility
- SPF Average Demand

SPP supports query-based auto-release, wherein, the planner can define queries and release/not release records returned by the query. Examples:

- Planner releases all records returned by a query.
- Planner defines queries to flag alerts. In this case, the planner doesn't want to auto-release orders that trigger these alerts, but is fine releasing all other planned orders within the time fence. These alerts may represent exceptions for excessive forecast error.

The integration between SPP and Demantra Demand Management provides improved information upon which planners can base release decisions, thereby potentially eliminating costly errors.

Integration Points



For more background information about the integration between Demantra Demand Management and the E-Business Suite, please see Demantra Demand Management to EBS Integration, page 14-2.

Business Process

1. Starting in Service Parts Planning (SPP) in the E-Business Suite, planner collects history data for Demantra Demand Planning. The data is loaded into the Demantra staging tables by the import workflows.

If the planner has history data in a legacy application, this can be collected for forecasting in Demantra Demand Planning in the same way.
2. Within Demantra Demand Management, planner can open three SPF worksheets. These include:
 - SPF: Analyze Forecast Organization Latest Revision
 - SPF: Analyze Forecast Latest Revision
 - SPF: Analyze Organization Base Model Spare
3. Planner makes modifications to these worksheets to fine-tune the forecast. Once an acceptable forecast has been arrived at using the engine profiles that support service parts forecasting, the new forecast and metrics can be published to Service Parts Planning on EBS.

4. In SPP, the planner includes the published Demantra forecast as a Demand Schedule in the SPP plan options. This is used for parts planning.

Service Parts Planning Navigator Menus

From the Service Supply Chain Planner, Standard responsibility, the Service Parts Forecasting (Demantra) menu is available for the integration between Service Parts Planning and Demantra. The options are as follows:

Menu	Sub Menu	Description
Service Parts Forecasting (Demantra)	Service Parts Forecasting Demantra Workbench	Launches the Demantra Collaborator Workbench.
	Demantra Workflow Manager	Launches the Demantra Workflow Manager.
	Usage History - Self Service	Launches legacy collections of historical usage data.
	Shipment History - Self Service	Launches legacy collections of the historical shipment data.
	Forecasting Data - EBS	Launches the collection of forecasting data from EBS to Demantra.

Integration Considerations

Determining the Zone of an Install Base Item

The Oracle Install Base has information on the install location where the item-instance is installed. Based on the zipcode of the install base item, the region that it falls under and the zone that it belongs to can be determined. Sourcing rules then specify the organization from which each zone is serviced, and Demantra generates a forecast at that organization. See Determining the Organization to Service an Install Base Item, page 15-5 for more information.

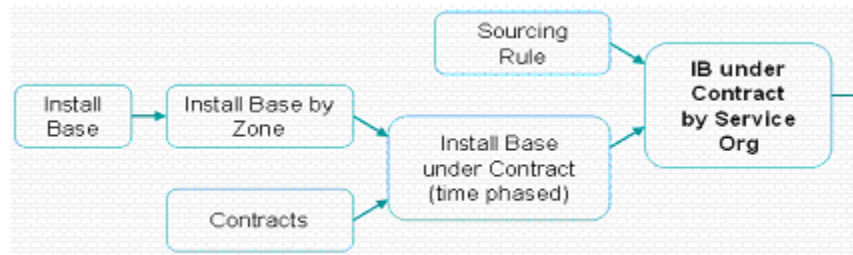
Determining the Install Base Under Contracts

The install base covered by contracts is computed as part of the collection process. The install base and contracts information is read and evaluated. The contracts data is then

overlaid on the install base to determine which of the install base is covered by contracts and pass that information on to Demantra.

Based on the time-phased install base, and the contracts that are effective at various points of time, the install base that is under contract at any point of time is determined and collected.

Determining the Organization to Service an Install Base Item



Typically in the service supply chain, a metropolitan distribution center (DC) services each zone. The planner therefore likes to see the install base (under contract) serviced by a DC to view the forecast for parts required at that DC.

The specification of which DC services which zone is performed through sourcing rules in SPP. This helps determine the organization that each install base item will be serviced from.

This process takes in an assignment set as a parameter. This assignment set has global sourcing rules that specify the organizations from which demands in each zone would source spares. The program inspects the sourcing for each zone and determines the DC that will service the install base under that zone.

Usage History

Usage History collects information regarding the repair work order product that the part is used for.

Note: This collection process is provided for forecasting in Demantra, and is separate from the collection of Usage History for SPP Inline forecasting.

Determining Failure Rates

The product-part combinations (based on the manufacturing BOM, serviceable items' category set and usage history) required to generate a forecast in Demantra is derived from the SPP collection process. If usage history is available, Demantra considers the usage history and install base under contract for items belonging to the category set

specified by the MSC: Category Set for Serviceable Items profile option, and based on these factors, determines the failure rate of each part. The MSC: History Basis for Failure Rate profile option specifies whether the usage history is depot or field service.

It should be noted that the failure rate is at a product part level, across organizations and zones. The failure rate is then aggregated on the usage history and install base under contracts across organizations.

For products that are newly introduced, where there is no usage history, the planner manually inserts the failure rate in the Demantra worksheet.

Initial Demantra Setup

A number of engine and non-engine parameters need to be configured to support Service Parts Forecasting.

Non-Engine Parameters

- SPF_Enable_Worksheet_Calculations
- SPF_History_Periods
- SPF_Default_Profile

See Non-Engine Parameters, page 34-1 for more details.

Engine Parameters

- EngTabDef_Parameters
- EngTabDef_HistoryForecast
- EngTabDef_Matrix
- EngDimDef_ItemorLoc
- EngKeyDefPK
- EngTabDef_Inputs
- EngKeyDef_Supersession
- GLPropSupersession Method

See Engine Parameters, page 58-1 for more details.

Initial SPP/EBS Setup

Some setup is required in SPP to integration Service Parts Planning with Demantra including:

- Configuring SPP Profile Options
- Setting up Instances
- Running Standard Collections

Configuring SPP Profile Options

The following profile options must be set:

- MSC: Category Set for Serviceable Items
- MSC: Collection Time Window for Install Base Under Contracts History
- MSC: History Basis for Failure Rate Computation
- MSD_SPF: History Basis for Failure Rate Computation
- MSD_SPF: Organization Containing Generic BOM for Service Parts Forecasting

For information about setting profile options, see *Oracle E-Business Suite System Administrator's Guide - Maintenance*.

MSC: Category Set for Serviceable Items Profile

This profile specifies the category of items that are enabled for maintenance, to be forecast and planned for. This profile option provides support for legacy collections. This profile specifies the category of items that are enabled for maintenance, to be forecast and planned for. This profile option provides support for legacy collections.

MSC: Collection Time Window for Install Base Under Contracts

This profile specifies the timeframe defined for the install base under contracts history collection. This should not be equal to the default value (12 months).

MSC: History Basis for Failure Rate

This profile option determines which history basis needs to be sent to Demantra. This information is used as the basis for the computation of failure rates. This profile can be set to "Depot Usage History" or "Field Service History."

Demantra considers the usage history for items that are enabled for service maintenance/replacement (belonging to the category set defined in the MSC:Category Set for Serviceable Items profile option) and generates a forecast for them.

MSD_SPF: History Basis for Failure Rate Computation

This profile option specifies which historical data is used to determine the failure rate. The options are Field Service or Depot Repair & Field Service.

MSD_SPF: Organization Containing Generic BOM for Service Parts Forecasting

This profile option specifies the organization from which the BOM for Service Parts Forecasting is derived.

Setting Up Instances

An instance is a database and a set of applications. Setup Instances is run before running Standard Collections to specify the Instances from which Standard Collections obtains data.

Oracle Advanced Planning can plan a single instance or multiple instances. For information about setting up instances, see "Instances" in the Cross-Instance Planning chapter of the *Oracle Service Parts Planning Implementation and User's Guide*.

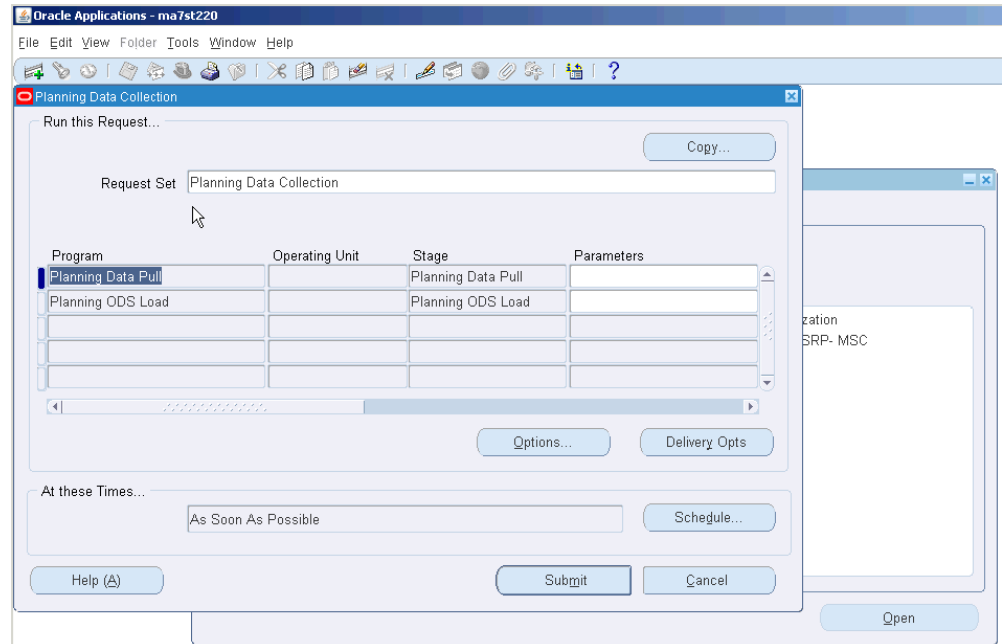
Running Standard Collections

"Standard" Collections refers to the Service Supply Chain Planner concurrent program for collecting new or changed information from the E-Business Suite to the Oracle Data Store (ODS). For information about collections, see "Starting Planning Data Pull and Planning ODS Load" in the "Collecting Source Data" chapter of the *Oracle Service Parts Planning Implementation and User's Guide*.

Important: You must set a variety of profile options before proceeding with the collection. See Configuring SPP Profile Options, page 15-7 for more information.

1. Sign on using the Service Supply Chain Planner.
2. Navigate to the Planning Data Collection window by selecting Collections: Oracle Systems > Standard Collection.

The Planning Data Collection window appears.



3. This window shows that the collections process consists of two sequentially executed concurrent programs. The first program, Planning Data Pull, copies information from the source instance into the APS staging tables on the planning server. The second program, Planning ODS Load, copies information from the APS staging tables into the operation data store on the planning server.
4. To select the Data Pull Parameters to use during Standard Collections, select the Parameters field for the Planning Data Pull program.
The Planning Data Pull Parameters window appears.
5. Complete the Parameters information as follows:

Parameter	Value
Approved Supplier Lists (Supplier Capacities)	Yes, replace all values
ATP Rules	Yes
Bill of Materials/Routings/Resources	Yes
Bills of Resources	Yes

Parameter	Value
Calendars	Yes
Demand Classes	Yes
End Item Substitution	Yes
Forecasts	Yes
Items	Yes
Key Performance Indicator Targets	Yes
Master Demand Schedules	Yes
Master Production Schedules	No
On Hand	Yes
Planning Parameters	Yes
Planners	Yes
PO Receipts	No
Projects/Tasks	Yes
Purchase Orders/Purchase Requisitions	Yes
Reservations	Yes
Resources Availability	Regenerate and collect data
Safety Stock	Yes
Sales Orders	Yes
Sourcing History	No
Sourcing Rules	Yes

Parameter	Value
Subinventories	Yes
Supplier Responses	Yes
Suppliers/Customers/Orgs	Yes
Transportation Details	Yes
Unit Numbers	Yes
Units of Measure	Yes
User Company Association	Create Users and Enable Users
User Supplies and Demands	Yes
Work in Process	Yes
Sales Channel	No
Fiscal Calendar	No
Internal Repair Orders	No
External Repair Orders	No
Payback Demand/Supply	Yes
Currency Conversion	Yes
Delivery Details	No
Install Base under Contracts (months)	Yes -- Install Base under Contract information is collected and used by Demantra to create forecasts.

6. Click OK

7. On the Request form, click Submit.

For more information about each of the parameters, please see the *Service Parts Planning*

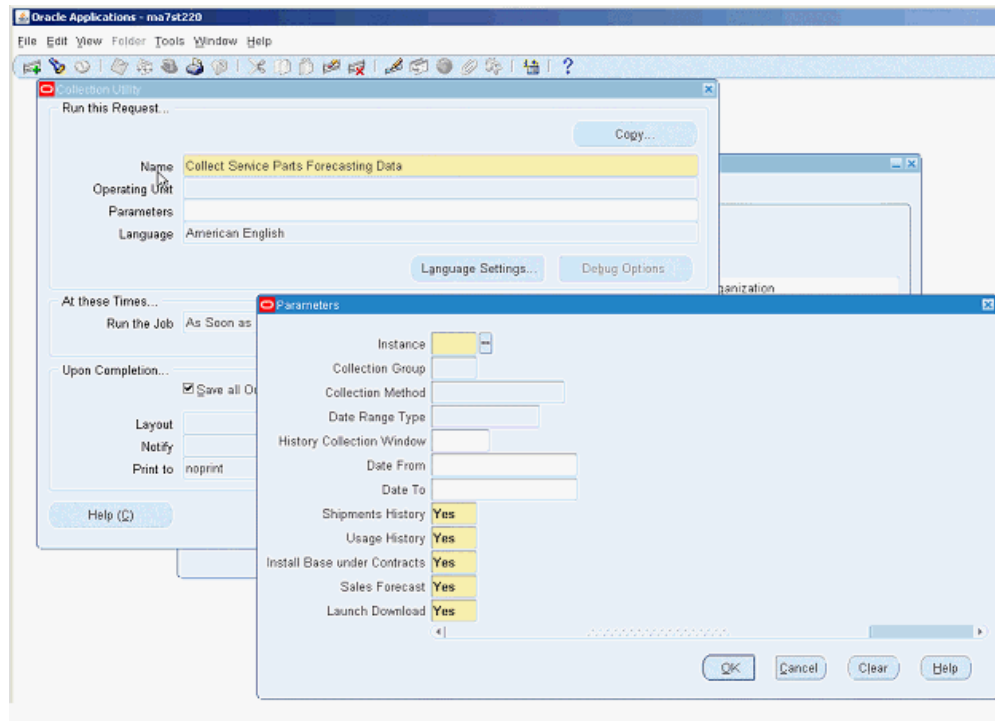
Importing SPP Data into Demantra

There are two sources of data when importing SPP data into Demantra. They are:

- EBS
- Legacy Data

EBS

1. From the Service Supply Chain Planner responsibility, run Forecasting Data-EBS.
The Collection Utility: Parameters form appears.



2. From the Parameters screen, enter the following information:

Parameter	Description
Source Instance	Enter or select the source (target) instance from which the data stream is to be collected.

Parameter	Description
Collection Group	Enter or select the appropriate collection group for the data to be sent to Demantra. Select All to transit data for all enabled organizations You can run collections for a named group of organizations. The default value is all enabled organizations.
Date Range Type	Select Absolute or Rolling from the LOV menu. When this parameter is set to Absolute, the Date From and Date To fields define the time period for which history data is collected. When this parameter is set to Rolling, the History Collection Window field value defines the number of days for which history data is collected.
Collection Window: (optional)	<p>Define the collection window for the data or set the From Date and Used Date fields to.</p> <ul style="list-style-type: none"> From Date (Date From): (optional) Enter the start date of collected data history stream. This field and the Used Date fields are set only when the Collection Window field is not defined. Used Date (Date To): (optional) Enter the end date for the collected data. This field and the From Date fields are set only when the Collection Window field is not defined.
Shipment History	Collects the shipment history data and sends it to Demantra for forecasting.
Usage History	Collects the Usage History with details of the product that the part is being used for, and sends it to Demantra for forecasting.
Install Base under Contracts	Takes the Install Base under contract that has been collected as specified above, and sends it to Demantra for forecasting.

Parameter	Description
Sales Forecast	Although sales forecast information is already available in Demand Management, if selected, this option will transfer the sales forecast from Supply Chain Management to Demantra for use in spare parts forecasting.
Launch Download	Indicates whether the download is launched. Available options: <ul style="list-style-type: none"> • Yes to enable the launch of the SPF Full Download work flow in Demantra • No to disable download launch

3. Once all parameters are set, click OK to start importing SPP data into Demantra.

If you have chosen to launch the download, the SPF Full Download workflow begins within Demantra to:

- Purge any previous download data
- Load Item Master and Location Master into the Demantra staging tables.
- Launch the SPF GL Data Download workflow that loads Item Master, Location Master, Spares BOM, Spares BOM and Spares Usage data into the Demantra data table.

If you chose to not launch the download and want to run the SPF Full Download workflow manually, see *Launching the Import Workflows from the Demantra Workflow Manager*, page 15-15.

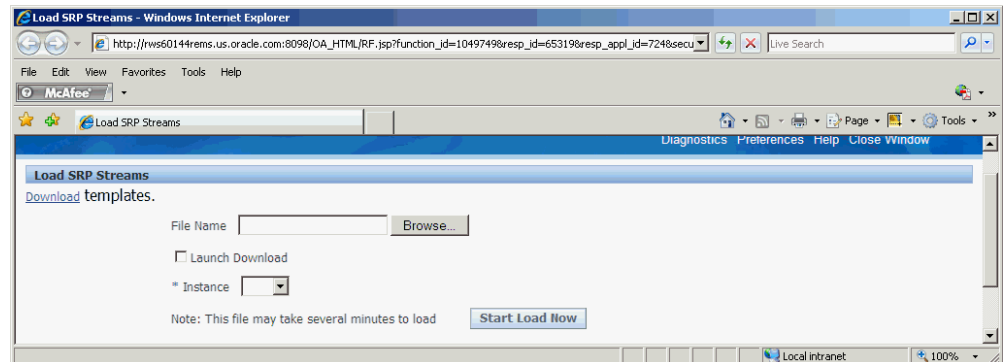
4. Launch the Service Parts Forecasting Demantra Workbench from the Service Supply Chain Planner responsibility, or launch the Demantra Collaborator Workbench independently. When the SPF Full Download workflow is complete, a message appears to indicate that the import was successful in the Collaborator Workbench. Then you can access the three preseeded worksheets to finetune the service parts forecast until you are satisfied with the results.

Legacy Data

Legacy collections are supported for Usage and Shipment History for Demantra. Legacy collections of Install Base under Contracts are not currently supported.

1. From the Service Supply Chain Planner responsibility, run Service Parts Forecasting (Demantra): Usage History – Self Service (or Shipment History – Self Service).

The Load SRP Streams form appears.



2. File Name: Specify the legacy data file. In case of multiple files, the planner can upload a zip file with the individual .dat files.

Note: You can download templates for individual .dat files from this page.

3. Launch: Check this option if you would like to begin the download. If you are collecting usage history, the SPF Full Download workflow is launched. If you are collecting shipment history, the GL Data Download workflow is launched.

If you chose to not launch the import workflows from this form, you can launch them manually. See Launching the Import Workflows from the Demantra Workflow Manager, page 15-15 for more information.

4. Instance: Choose from the available instances.
5. Click Start Load Now to load the legacy data file(s) now.

Launching the Import Workflows from the Demantra Workflow Manager

If you did not choose to launch the download through the Parameter form, you can start the import workflows manually.

To launch the import workflows from the Demantra Workflow Manager:

1. From the Service Supply Chain Planner responsibility, run Demantra Workflow Manager. The Demantra Workflow Management form appears.
2. From the View according to Schema Groups field, choose Spare Forecasting. All the workflows associated with the SPP to Demantra integration are displayed.

The screenshot shows the Oracle Demantra Workflow Management interface in a Windows Internet Explorer browser. The page title is 'Oracle Demantra Workflow Management - Windows Internet Explorer'. The address bar shows 'http://dempr1app.us.oracle.com:8009/demantra/workflow/ProcessManager.jsp'. The page has a 'Welcome dm' message and a 'Workflow Management' section. Below this, there is a dropdown menu for 'View according to Schema Groups' set to 'Spare Forecasting'. There are buttons for 'New', 'Modify', and 'Delete'. The main content is a table with columns: Schema ID, Schema name, Owner, Creation Date, Last Modified, Instances, Status, and Action. The table lists 12 workflows, including 'SPF Upload Data', 'SPF Incremental Upload', 'Calculate Spares Forecast', 'Export Consensus Forecast for SPF', 'SPF GL Data Download', 'SPF Full Download', 'Import SPF Child', 'Import SPF Base Model', 'Import SPF Latest Revision', and 'Import SPF'.

Schema ID	Schema name	Owner	Creation Date	Last Modified	Instances	Status	Action
1373	SPF Upload Data	dm	6/15/10	7/15/10	0	●	Edit Start Schedule Delete
1374	SPF Incremental Upload		6/15/10	7/2/10	0	●	Edit Start Schedule Delete
1375	Calculate Spares Forecast	dm	6/15/10	9/17/10	0	●	Edit Start Schedule Delete
1376	Calculate Spares Forecast for LOB	dm	6/16/10	6/16/10	0	●	Edit Start Schedule Delete
1393	Export Consensus Forecast for SPF	dm	7/6/10	7/12/10	0	●	Edit Start Schedule Delete
1394	SPF GL Data Download	dm	7/6/10	7/28/10	0	●	Edit Start Schedule Delete
1395	SPF Full Download	dm	7/6/10	9/27/10	0	●	Edit Start Schedule Delete
1413	Import SPF Child	dm	7/13/10	7/13/10	0	●	Edit Start Schedule Delete
1414	Import SPF Base Model	dm	7/13/10	7/13/10	0	●	Edit Start Schedule Delete
1415	Import SPF Latest Revision	dm	7/13/10	7/13/10	0	●	Edit Start Schedule Delete
1416	Import SPF	dm	7/13/10	7/13/10	0	●	Edit Start Schedule Delete

There are two import workflows associated with the SPP to Demantra integration:

- SPF Full Download – if data to be collected includes item, location or install base information.
 - SPF GL Data Download – if data to be collected includes only spares BOM, spare usage and shipment history.
3. Select the applicable download workflow and click the Start button to import the SPP data into Demantra. A message appears in the Collaborator Workbench when the import is complete.

Service Parts Forecasting in Demantra

Use the SPF worksheets provided in the Demantra Collaborator Workbench to review and fine-tune the service parts forecast.

Service parts forecasts can be run either in batch mode or simulated. The following engine profiles are specifically configured for service parts forecasting:

Batch Engine Profile	Simulation Engine Profile	Description
Forecast Install Base	Simulation Install Base	This engine profile supports the forecasting of install base under contract.
Forecast Spares Demand	Simulation Spares Demand	This engine profile supports forecasting of spares at an organization.

In batch mode, the engine profile can be specified through the Engine Administrator application, workflow, or URL. When running a simulation in a worksheet, the engine profile can be specified in the Simulation dialog box.

For more information about SPF Forecasting, see the *Demantra Demand Management User Guide*.

Exporting SPF Data from Demantra to Service Parts Planning in EBS

Exporting SPF data from Demantra can be accomplished either by running a full upload or exporting only changes that planners make in Demantra worksheets.

Running Full Upload

The Demantra SPF data can be fully uploaded to SPP, typically once per planning cycle. The SPF Upload Data workflow, which uploads forecast and metrics to SPP, can be scheduled or run directly from the Workflow Manager.

Running Interim Uploads

After you have fine-tuned and simulated a service parts forecast and you are satisfied with the results, you can export the changes to the forecast and metrics to SPP.

1. From each of the preseeded SPF worksheets, you can run the Publish SPF Changes method. Access the method from each SPF worksheet by right-clicking on a level (SPF, SPF Base Model, SPF Child, SPF Latest Revision or SPF Parent Item), then choosing Methods. This method checks the last export date and compares it to the last update date. It then does the following:
 - If it is the first time data is been exported, the method then runs the SPF Upload Data.
 - If data has been updated since the last export, the method runs SPF Incremental Upload, exporting only modified data.

These workflows upload the forecast and metrics to the Demantra staging tables.

2. In SPP, the planner includes the published Demantra forecast as a Demand Schedule in the SPP plan options.

Integration Workflows and Integration Profiles

Multiple workflows and integration profiles facilitate the integration between SPP and Demantra Demand Management.

Workflows can be launched in multiple ways. Workflows can be started by a method, scheduled, started by another workflow, or launched directly from the Demantra Workflow Manager.

For the integration between SPP and Demantra Demand Management, workflows include transfer steps that execute integration interface profiles. These integration interface profiles define the data, levels and intermediary tables involved with the integrations. Some workflows include multiple transfer steps that are run in a specific order. Although these workflows and integration interface profiles are preconfigured, both the workflows and integration interfaces can be modified to better suit your integration requirements.

SPF Workflows

The following workflows are used with SPP to Demantra Demand Management integration:

Workflow	Description
Calculate Spares Forecast	Recalculates Failure Rate % for each Spare across all Base Models and Orgs.
Calculate Spares Forecast for LOB	Recalculates Failure Rate % for each Spare across all Base Models and Orgs in a line of business.
Export Consensus Forecast for SPF	Exports forecast and metrics to Demantra staging tables. This workflow is used to export the sales forecast to SPP to help predict the future install base.
Import SPF Base Model	Imports the level IMPORT_SPF_BASE_MODEL. This workflow is called by the SPF GL Data Download workflow.

Workflow	Description
Import SPF Child	Imports the level IMPORT_SPF_CHILD. This workflow is called by the SPF GL Data Download workflow.
Import SPF Data	Imports the usage information and then configures and runs proport. This workflow is called by the SPF GL Data Download workflow.
Import SPF Latest Revision	Imports the level IMPORT_SPF_LATEST_REVISION. This workflow is called by the SPF GL Data Download workflow.
Import SPF Level	Imports the level IMPORT_SPF_LEVEL. This workflow is called by the SPF GL Data download workflow.
Run Forecast Install Base	Checks whether the engine is configured to run on Linux or Windows and executes the engine appropriately with the Forecast Install Base engine profile.
Run Forecast Spares Demand	Checks whether the engine is configured to run on Linux or Windows and executes the engine appropriately with the Forecast Spares Demand engine profile.

Workflow	Description
SPF Calc Forecast Accuracy	<p>The seeded workflow is used to calculate the error and variability associated with Service Parts Forecasting by running the APPPROC_CALC_ACCURACY procedure. This workflow can be called ad-hoc when accuracy measures should be generated. The seeded workflow is configured to aggregate information at levels Organization and Latest revision for the last four periods of history. The first three series pairs generate an accuracy measure for the final, analytical and calculated forecast streams by comparing the latest archived forecast with actual usage values. The last series pairs compare the last two archived versions of the final forecast with each other to determine forecast variability. If additional error calculation processes are required, it is recommended that additional steps be added to this workflow or separate workflows be created to call the APPPROC_CALC_ACCURACY stored procedure.</p>
SPF Full Download	<p>This workflow first purges the previous download data, then loads Item Master, Location Master, configures and runs proposit, launches the SPF GL Data Download workflow. It then collects data from staging tables, and loads Item Master, Location Master, Spares BOM, Spares Usage at the Base Model/Spares/Org level into the Demantra data tables.</p>

Workflow	Description
SPF Generate Forecast	<p>This workflow does the following:</p> <ul style="list-style-type: none"> Generates analytical forecast for future install base. Triggers the recalculation of forecasts based on revised install base values. Generates an analytical forecast for spares, and reviews analytical forecast values. <p>It launches the following workflows in order: Run Forecast Install Base, Calculate Spares Forecast, Run Forecast Spares Demand.</p>
SPF GL Data Download	<p>This workflow imports the following levels into Demantra from SPP: SPF Base Model, SPF Child, SPF Latest Revision, SPF Level, SPF Usage Data. It then configures and runs the propt, imports SPF Shipment History and SPF Sales Forecast Data.</p>
SPF Incremental Upload	<p>This workflow exports forecast and metrics to external tables for combinations that have changed since the last full or incremental upload.</p>
SPF Upload Data	<p>This workflow exports forecast and metrics to external tables for all forecasted combinations.</p>
SPF-Archive Forecast	<p>This workflow archives the various forecast series and makes them available for MAPE and Bias metrics calculations.</p>

Service Parts Forecasting Integration Import Profiles

The following import integration profiles support Service Parts Forecasting:

- SPF
- Purge SPF History Data

Tip: The integration interface dimensional levels can be modified. See *Creating or Modifying an Integration Interface*, page 29-2 for more details.

SPF

Description: Retrieves SPP data and loads it into the Demantra tables.

Workflow: SPF GL Data Download

Data Profiles

Data Profiles	Series/Levels
IMPORT_SPF_DATA	Series: <ul style="list-style-type: none">SPF Engineering Estimated FailureSPF Item ShipmentsSPF Item Usage Output Levels: SPF/Item/Demand Class/Organization/Site/Sales Channel
IMPORT_SPF_SALES_FORECAST_DATA	Series: SPF Consensus Forecast Output Levels: Item
IMPORT_SPF_SHIPMENT_HISTORY	Series: SPF Item Shipments Output Levels: Item/Organization

Level Profiles

Level Profiles	Series/Levels
IMPORT_SPF_BASE MODEL	Level Profile Parameter: SPF Base Model
IMPORT_SPF_CHILD	Level Profile Parameter: SPF Child
IMPORT_SPF_LATEST_REVISION	Level Profile Parameter: SPF Latest Revision

Level Profiles	Series/Levels
IMPORT_SPF_LEVEL	Level Profile Parameter: SPF

Purge SPF Hlstory Data

Description: Purges SPF historical data.

Workflow: SPF-Archive Forecast

Data Profiles	Series/Levels/Filters
Purge Install Base History	Series: SPF Install Base Under Contract Output Level: Organization
Purge Spare History Data	Series:
Purges spare usage and shipment history data	<ul style="list-style-type: none"> SPF Item Shipments SPF Item Usage Output Level: SPF Base Model Filtered by Organization: Default Organization

Service Parts Forecasting Integration Export Profiles

The following integration export profiles support Service Parts Forecasting:

SPF Upload Forecast Data & Metrics

Description: Uploads SPF final forecast at item, org levels.

Workflow: SPF Upload Data

Data Profiles	Series/Levels/Filters
SPF Upload Final Forecast	Series: SPF Final Forecast Output Levels: Item/Org Filtered by Organization Type: Spares Forecasting
SPF Upload Metrics	Series: <ul style="list-style-type: none"> • SPF Average Demand • SPF Forecast MAPE (In Sample) • SPF Forecast MAPE (Out of Sample) • SPF Forecast Volatility Output levels: Item/Org Filtered by Organization Type: Spares Forecasting

Demantra Sales and Operations Planning to EBS Integration

This chapter describes the integration between Sales and Operations Planning and EBS products, including Strategic Network Optimization (SNO), Advanced Supply Chain Planning (ASCP), and Rapid Planning (RP). It also describes the handling of SCI data.

This chapter covers the following topics:

- Introduction
- Integration Points
- Sales and Operations Planning Navigator Menus
- General Configuration
- Integration Workflows and Interface Profiles
- Demantra Sales and Operations Planning to Strategic Network Optimization Integration
- Demantra Sales and Operations Planning to Advanced Supply Chain Planning Integration
- Demantra Sales and Operations Planning to Rapid Planning Integration
- SCI Data

Introduction

This chapter explains the integration processes that synchronize or move data between Oracle Demantra Sales and Operations Planning and EBS applications.

Oracle Demantra Sales and Operations Planning provides access to your historical sales data (both shipments and booking history), returns, and other reference data organized into multiple hierarchies that reflect the needs of your organization.

The Sales and Operations Planning integration with EBS leverages and extends the existing integration between EBS and Oracle Demantra Demand Management. To

integrate Oracle Demantra Sales and Operations Planning with EBS, certain modifications are required to both your EBS setup and the Sales and Operations Planning application. The following sections outline the required changes.

Integration Points

The following information is transferred between EBS and Demantra S&OP during the integration process. The EBS applications supported include ASCP, SNO and Rapid Planning.



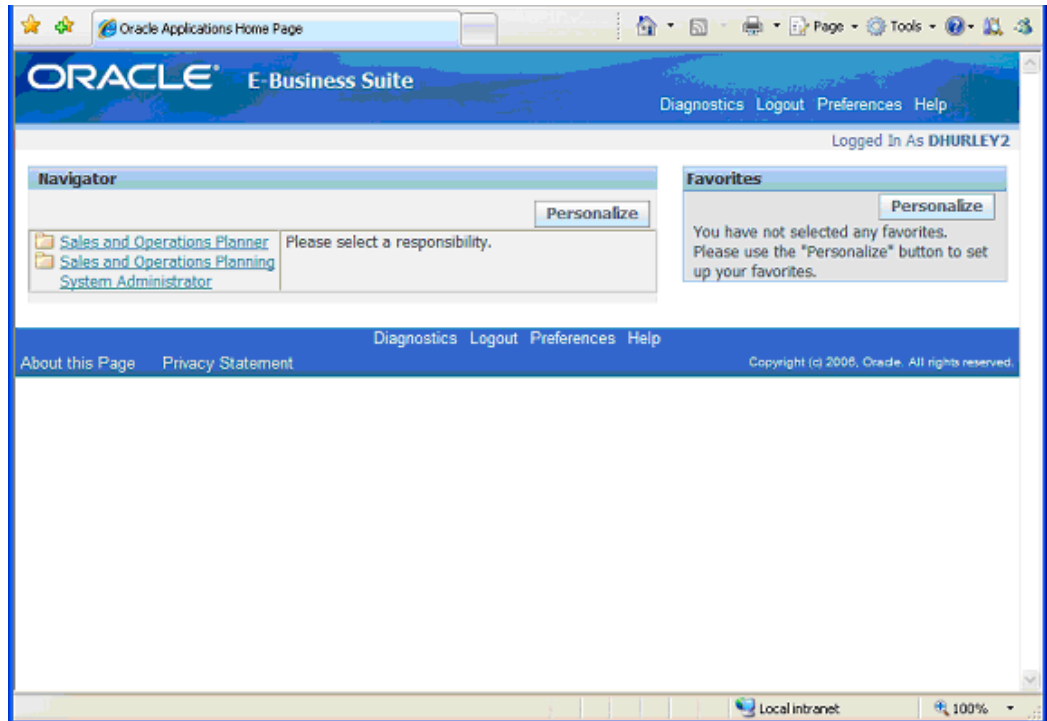
Note: In the Demantra 7.3.0.1 release, safety stock is not supported for Rapid Planning.

Sales and Operations Planning Navigator Menus

The Oracle E-Business Suite Navigator provides the following two responsibilities:

- Sales and Operations Planner
- Sales and Operations Planning System Administrator

Note: EBS users can have both Demand Management and Sales and Operations Planning responsibilities, and therefore access both Demantra Demand Management and Sales and Operations Planning.



Login to E-Business Suite and there is a Sales and Operations Planning System Administrator responsibility in the Navigator. The following options appear for this responsibility:

- Sales and Operations Planning Workbench: provides access to the Oracle Demantra Collaborator Workbench.
- Workflow Manager: provides access to the Oracle Demantra Workflow Manager.
- Administration: provides access to the Collaborator Workbench Administration page in Demantra. Use the Collaborator Workbench Administrator to control access to menu items.
- User Management: provides access to the Oracle Demantra User Management Console. Use this tool to log out users with hanged sessions.
- Collections: Oracle and legacy systems allow the collection of various entities from EBS and ASCP ODS with an option to download data into Demantra. Some available entities include Shipment and Booking History, Currency Conversion, UOM Conversion, Returns History and Calendars.
- Setup – Instances: allows setting up multiple instances from where collections can be done.
- Setup – Calendar List: Permits setup of calendars for download into Oracle

Demantra.

- Setup - New Products List: Permits setup of new products for download into Oracle Demantra.
- Setup - Price List: permits setup of price lists for download into Demantra.

The Sales and Operations Planner responsibility only has access to the Oracle Demantra Collaborator Workbench.

General Configuration

Levels and Hierarchies

The levels and hierarchies that are seeded in Oracle Demantra Sales and Operations planning are a super set of those seeded in Demantra Demand Management. The following table lists the unique hierarchies seeded in Sales and Operations Planning.

Levels	Hierarchy
Plan Scenario	Plan Scenario > Scenario Status
Resource	Scenario Resource > Resource
Supplier	Site > Supplier

Time Scaling for the EBS Integration Interface Data Profiles

The integration interfaces supporting the integration between ASCP/SNO/Rapid Planning to Demantra S&OP are designed for a Demantra weekly time scale. If your Demantra model requires a time scale other than weekly, you can change to monthly or daily. This change would need to be made to all the import and export integration interface data profiles you plan to use.

Caution: Data from ASCP and SNO are imported into Demantra in telescopic fashion like a few weeks in days, then aggregated in weeks, followed by months. If the time scale is changed to a day level, this could result in performance scalability issues for medium to large customer models.

See *Creating or Modifying an Integration Interface*, page 29-2 for details about changing the integration interface data profile time scale.

Approved Supplier List

The Approved Supplier List is used to specify supplier capacities and this information flows down to supply planning applications such as SNO/ASCP. To properly configure to support S&OP, the Supplier Site must be setup in the Approved Supplier List.

Integration Workflows and Interface Profiles

Multiple workflows and integration interface profiles facilitate the integration between EBS and Demantra S&OP.

Workflows can be launched in multiple ways. In some cases, workflows are started by a method. In other cases, workflows are started by another workflow. Lastly, workflows can be started directly from the Demantra Workflow Manager.

For the integration between EBS and Demantra S&OP, workflows include transfer steps that execute integration interface profiles. These integration interface profiles define the data, levels and intermediary tables involved with the integration. Some workflows include multiple transfer steps that are run in a specific order.

Although these workflows and integration interface profiles are preconfigured, both the workflows and integration interface profiles can be modified to better suit your integration requirements.

Workflow Overview

The following workflows are used with EBS to Demantra S&OP integration:

- **Download Plan Scenarios:** Updates the Demantra S&OP Supply Review Dashboard with the plans available from EBS.
- **Download Plan Scenario Data:** Populates the Demantra database with ASCP, SNO and Rapid Planning data. This workflow is launched with the Load Scenario Data method. The workflow includes transfer steps that execute the following integration import interfaces in this order: Plan Scenarios, Resources, Purge Plan Data, and then Plan Data.
- **Download SCI Data:** Downloads SCI data from EBS to Demantra S&OP. The workflow includes transfer steps that execute the following integration import interfaces in this order: Purge Backlog Data, then SCI.
- **Download Unit Cost:** Downloads the unit cost from ODS to Demantra S&OP.
- **Upload Consensus Forecast -- Org. Week:** Uploads the Consensus Forecast from Demantra S&OP to EBS using the Organization and Week Dimensions.
- **Upload Consensus Forecast -- Org. Period:** Uploads the Consensus Forecast from Demantra S&OP to EBS using the Organization and Fiscal Period Dimensions.

- Upload Consensus Forecast -- Zone. Week: Uploads the Consensus Forecast from Demantra S&OP to EBS using the Zone and Week Dimensions.
- Upload Consensus Forecast -- Zone. Period: Uploads the Consensus Forecast from Demantra S&OP to EBS using the Zone and Fiscal Period Dimensions.
- Upload Consensus Forecast -- Org. Demand Class.Week: Uploads the Consensus Forecast from Demantra S&OP to EBS using the Organization, Demand Class and Week Dimensions.
- Upload Consensus Forecast -- Zone. Demand Class. Week: Uploads the Consensus Forecast from Demantra S&OP to EBS using the Zone, Demand Class and Week Dimensions.

Supply Plan Integration Import Profiles

The following integration interfaces are used for EBS integration:

- Plan Data: Loads data from SNO/ASCP/Rapid Planning plans to S&OP.
- Plan Scenarios: Loads ASCP/SNO/Rapid Planning plans that have been associated with S&OP into the Supply Review Dashboard as new plans.
- Purge Backlog Data: Purges SCI data before downloading new SCI data from EBS.
- Purge Plan Data: Purges data from previous plan data imports prior to loading more recent plan data.
- Resources: Loads the resource level in members in Demantra from EBS supply plans.
- SCI: Loads SCI data from EBS to S&OP.
- Unit Cost: Loads item costs from ODS.

Profile Name	Launched by Workflow	Series	Output Levels
Plan Data There are two data profiles: <ul style="list-style-type: none"> Resource Capacity Other Plan Data 	Download Plan Scenario Data Note: The Load Scenario Data method launches this workflow.	Resource Capacity Data Profile: <ul style="list-style-type: none"> Available Premium Capacity Available Standard Capacity Required Capacity Other Plan Data Profile: <ul style="list-style-type: none"> Available Premium Supplier Capacity Available Standard Supplier Capacity Beginning On-Hand Constrained Forecast Dependent Demand Inventory Plan Planned Shipments Production Plan Required 	Resource Capacity Data Profile: Scenario Resource, Organization Other Data Profile: Plan Scenario, Item, Organization, Site, Demand Class, Sales Channel

Profile Name	Launched by Workflow	Series	Output Levels
		Supplier Capacity	
		<ul style="list-style-type: none"> Safety Stock 	
		<p>Note: The Inventory Plan series is only applicable for SNO models only.</p> <p>Note: In Demantra 7.3.0.1, safety stock is not supported for the Rapid Planning integration.</p>	
Plan Scenarios	Download Plan Scenarios		
	Download Plan Scenario Data		
Purge Backlog Data	Download SCI Data	Past Due Backlog Total Backlog	

Profile Name	Launched by Workflow	Series	Output Levels
Purge Plan Data There are two data profiles: <ul style="list-style-type: none"> • Purge Plan Data • Purge Resource Data 	Download Plan Scenario Data	Purge Plan Data Profile: <ul style="list-style-type: none"> • Available Premium Supplier Capacity • Available Standard Supplier Capacity • Beginning On-Hand • Constrained Forecast • Dependent Demand • Inventory Plan • Planned Shipments • Production Plan • Required Supplier Capacity • Safety Stock Purge Resource Data Profile: <ul style="list-style-type: none"> • Available Premium Capacity • Available Standard 	Purge Plan Data Profile: Plan Scenario Purge Resource Data Profile: Linked Scenario

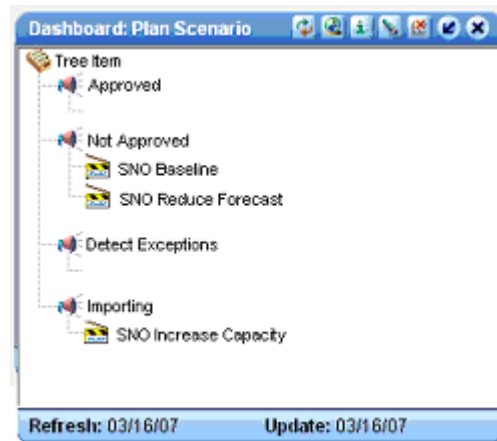
Profile Name	Launched by Workflow	Series	Output Levels
		Capacity	
		<ul style="list-style-type: none"> Required Capacity 	
Resources	Download Plan Scenario Data		Resources Level Profile: Resources
There are two level profiles:			Scenario Resources Level Profile: Scenario Resources
<ul style="list-style-type: none"> Resources 			
<ul style="list-style-type: none"> Scenario-Resources 			
SCI	Download SCI Data	SCI: Backlog Data Profile:	SCI: Backlog Data Profile: Item, Organization, Customer
There are two data profiles:		<ul style="list-style-type: none"> Backlog Data Customer Source 	SCI: Other Data Profile: Item: Organization
<ul style="list-style-type: none"> SCI: Backlog 		<ul style="list-style-type: none"> Past Due Backlog 	
<ul style="list-style-type: none"> SCI: Other 		<ul style="list-style-type: none"> Total Backlog 	
		SCI: Other Data Profile:	
		<ul style="list-style-type: none"> Actual Onhand no edit 	
		<ul style="list-style-type: none"> Actual Production 	
Unit Cost	Download Unit Cost	Unit Cost	Item, Organization

Tip: The integration interface dimensional levels can be modified. See Creating or Modifying an Integration Interface, page 29-2 for more information.

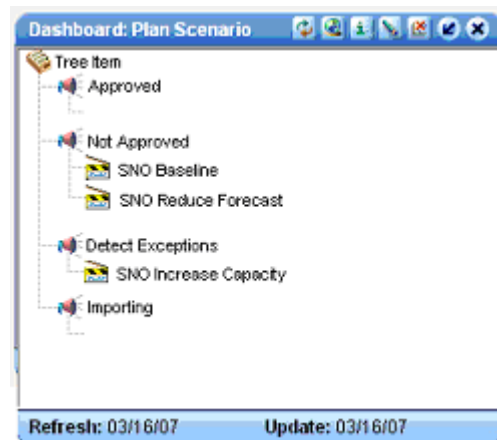
Import Status Messages

The following status messages are used during the plan import:

- New - When loading supply plan members and the member does not already exist in S&OP, the Supply Plan name is added and the status set to "New".
- Importing - While a Supply Plan scenario is importing, its status is set to Importing. For example, the S&OP Manager logs into Collaborator workbench, opens a worksheet and there is no data for a particular supply plan scenario. Opening the scenario properties, the scenario status appears as "Importing" and it is understood why there is no data.

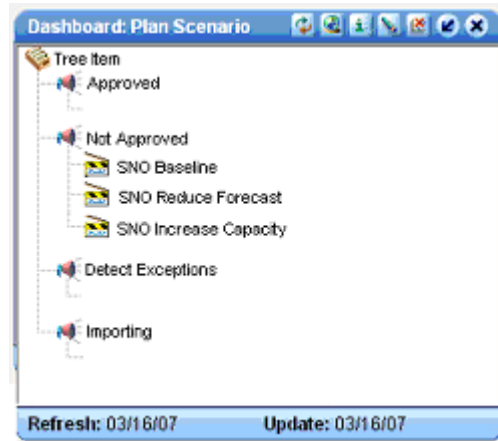


- Detect Exceptions - After importing a supply plan scenario, exception detection workflow runs automatically and scenario status changes to "Detect Exceptions". Any scenarios with this status will be included in the exception worksheets.

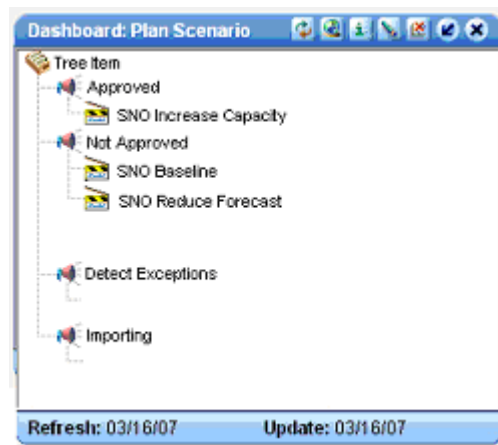


- Not Approved: The Complete Exception Review level method is run manually and changes the supply plan status from "Detect Exception" to "Not Approved". It is

also possible for the S&OP Manager to "unapprove" a scenario using the Complete Exception Review level method.



- Approved: S&OP Manager runs level method to approve plan scenario. Scenario Status changes from Not Approved to Approved.



Consensus Forecast Integration Export Profiles

The purpose of the export integration profiles are to expose the S&OP consensus forecast to EBS applications.

A major requirement for sales and operations planning is that data from the Supply Plan maps back to the Demand Plan hierarchy. For the Supply Review and Executive Review meeting, it is necessary to compare the supply plans to the demand plans. The demand plan has a hierarchy that allows you to view the data at any level. To compare the supply plan to the demand plan, a hierarchy must exist that allows you to view the supply plan at different levels as well.

Note: It is important to synchronize the planning periods (or time buckets) between Demand Planning and Supply Planning. For example, if Sales and Operations Planning periods are 4-4-5, then Strategic Network Optimization must have the same period sizes. EBS to ASCP integration also supports weekly buckets therefore export integration profiles must exist to support weeks. Demantra aggregates data into the required bucket size.

Series: Consensus Fcst

Output Levels: (Item, Org, Week) and (Item, Org, Period) where Period is fiscal month. SNO does not support (Gregorian) calendar months from EBS.

Export Integration Interface Profiles:

Profile Name	Launched by Workflow	Series	Output Levels
Consensus Fcst(Org, Week)	Upload Consensus Forecast - Org, Week	Consensus Fcst	Item, Org, Weeks
Consensus Fcst(Org,Fiscal)	Upload Consensus Forecast - Org, Period	Consensus Fcst	Item, Org, Fiscal 445
Consensus Fcst(Zone, Week)	Upload Consensus Forecast - Zone, Week	Consensus Fcst	Item, Zone, Weeks
Consensus Fcst(Zone, Fiscal)	Upload Consensus Forecast - Zone, Period	Consensus Fcst	Item, Zone, Fiscal 445
Local Consensus Fcst(Item, Org, DC, Week)	Upload Consensus Forecast - Org, Demand Class, Week	Consensus Fcst	Item, Org, Demand Class, Weeks
Global Zone Consensus Fcst(Item, Zone, DC, Week)	Upload Consensus Forecast - Zone, Demand Class, Week	Consensus Fcst	Item, Zone, Demand Class, Week

Note: The Demand Priority and Mean Absolute Pct Err (MAPE) from the Demantra Demand Management component must be included in the S&OP component. They are not used in any worksheet or pre-seeded export integration profile, but they must be available if you want to include them. Demand Management includes Demand Priority

in the Demand Analysis worksheets and both Demand Priority and Mean Absolute Pct Err in export integration profiles.

Tip: The integration interface dimensional levels can be modified so you can generate Consensus Forecast exports and Plan Data imports that best suit your organization. See *Creating or Modifying an Integration Interface*, page 29-2 for more information.

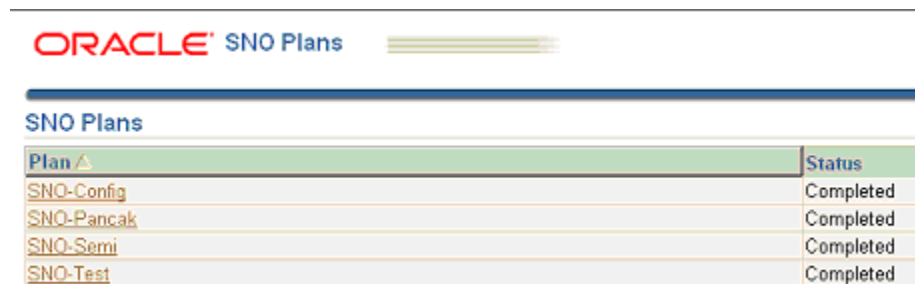
Demantra Sales and Operations Planning to Strategic Network Optimization Integration

Oracle Strategic Network Optimization enables you to design your entire supply network with agility in mind. In addition, SNO allows you to consider hedge strategies in your design and be able to simulate the outcome of unpredictable events on your supply network flow.

Strategic Network Optimization is integrated with the Oracle E-Business Suite. SNO plans can be used in Demantra Sales and Operations Planning, allowing you to develop demand consensus forecasts. These consensus forecasts can be uploaded to SNO for additional simulation and creation of a feasible, resource-constrained and cost/profit optimal supply plan.

Prerequisite: Configuration of SNO Plan Options

SNO uses the "Use for Sales and Operations Planning" plan option to indicate that a plan should be included when creating import integration profiles. The available SNO Plans are available in the workbench:



ORACLE SNO Plans	
SNO Plans	
Plan	Status
SNO-Config	Completed
SNO-Pancak	Completed
SNO-Semi	Completed
SNO-Test	Completed

One can designate a SNO plan for use in Sales and Operations Planning by selecting the "Use for Sales and Operations Planning" check box when defining the SNO plan. This serves to generate the import integration profiles as specified in the next section.

Strategic Planning Options (TST.M1)

Schedule: SNOPLAN.BA Description: sno plan for testing

☒ Use for Sales and Operations Planning

Main Organizations

Buckets: 10 Weeks Planned Resources: All Resources

Plan Start Date: 01-JAN-2007 Bottleneck Resource Group:

Plan End Date: 11-MAR-2007 Assignment Set (Input): JC_ASG_SET

Assignment Set (Output): jc-output

Items

Include Items belonging to the following categories:

Category Set: JC-ITEMS Test Set:

Category	Description
ASSEMBLY.SUBASSY	Computer Systems
COMPONENT.ELECTR	Accessories
COMPONENT.MECHAN	Hardware
COMPUTER.HANDHELD	Handheld Computers and PDAs

Note: If a plan member has already been collected in S&OP using the Download Plan Scenarios workflow and the "Use for Sales and Operations Planning" check box is then deselected in Plan Options, you must delete the plan manually from within S&OP. In other words, the Download Plan Scenarios workflow adds new members but does not delete existing plan members.

Note: The profile option "MSC: SNO Decimal Precision" has a default value of 2. Increasing this to 4 or more will help ensure data consistency between the SNO plan and the S&OP scenario.

Integration Steps

1. Arrive at a proposed consensus forecast in S&OP.
2. Export the Consensus Forecast demand plan to EBS. From the Demantra Workflow Manager, run the applicable Upload Consensus Forecast workflow. This workflow loads the consensus forecast to an intermediate Demantra file (msd_dp_scn_entries_denorm table) which is also used when publishing from Demand Management.
3. In EBS, define the supply plan that will consume the Consensus Forecast. Configure the plan options to "Use for Sales and Operations Planning".

4. Open the SNO plan. SNO creates a feasible, resource-constrained and cost/profit optimal supply plan using the demand plan as input. Often several supply plan simulations are created to explore the implications of the many options in satisfying the demand plan. The horizon is typically two years into the future and the plans are usually created in fiscal monthly buckets.
5. Publish a SNO plan to S&OP. SNO plans designated for use in S&OP include a new publish profile:
 - Name: Release to Sales & Operations Planning
 - Description: Releases Constrained Forecast, Production, Inventory and Capacity Plans to S&OP
6. Export the SNO supply plan to Demantra. In the Demantra Workflow Manager, start the Download Plan Scenarios workflow.
7. In the Demantra Collaborator Workbench, the Supply Review Dashboard lists the new plans from EBS including those from SNO.

Note: When scenarios are first downloaded from SNO, they appear under "New" in the Supply Review dashboard. The other statuses that a scenario could be under are "Importing" (when data is being collected into S&OP), "Detect Exceptions", "Not Approved", and "Approved"..

8. Right-click on the SNO plan you want to download. Choose Methods, then Load Scenario Data. This launches the Download Plan Scenario Data workflow that populates the Demantra database with SNO data. All the data from the SNO plan is available in all the seeded supply review worksheets in S&OP.
9. Click Yes to confirm that you want to load scenario data.
10. When the Download Plan Scenario Data workflow completes, the SNO plan appears in the Supply Review Dashboard Detect Exceptions list.
11. Right-click on the SNO plan and click Open, then select the type of S&OP worksheet you want to use to view the SNO plan data. Alternatively, you can open a preseeded S&OP worksheet directly from the Collaborator Workbench. Review and compare supply plan scenarios and exceptions in seeded S&OP worksheets such as Constrained Forecast Product Category, Resource Rough Cut Capacity, Consolidated Plan Product Category, Financial Summary Product Category and KPI Scenario Comparison Product Category worksheets.

Planning Integration

Oracle Advanced Supply Chain Planning enables you to perform simultaneous material and capacity planning across multiple distribution and manufacturing facilities and time horizons in a single planning run, while at the same time accounting for the latest consensus forecasts, sales orders, production status, purchase orders, and inventory policy recommendations. Demantra S&OP can be used with ASCP to develop consensus forecasts for ASCP plans. The S&OP consensus forecast can be uploaded into ASCP for consideration when ASCP plans are run. The Demantra S&OP to ASCP integration supports all three ASCP plan types: Master Plan, Production Plan, and Manufacturing Plan.

Prerequisite: Configuration of ASCP Plan Options

You can indicate which ASCP plans you want to make available to S&OP with the "Use for Sales and Operations Planning" plan option. Demantra creates an import integration profile for all ASCP plans with the Use for ASCP option enabled.

The Download Plan Scenarios workflow collects these S&OP eligible ASCP plans and then populates the plan IDs in S&OP. To pull actual plan data, the Load Scenario Data method must be run from within S&OP for a specific plan ID.

The screenshot shows the 'Plan Options (TST.M1)' window. At the top, the 'Plan' tab is set to 'Constrained' and the 'Plan Type' is 'Manufacturing Plan'. The 'Main' tab is selected, showing various configuration options:

- Planned Items:** Demand schedule ...
- Material Scheduling Method:** Order Start Date
- End Item Substitution Set:** (empty)
- Schedule By:** Schedule Ship Date
- Assignment Set:** TST:ASET:ASCP
- Demand Priority Rule:** (empty)
- Overwrite:** All
- Demand Class:** (empty)
- ☐ Demand Time Fence Control
- ☐ Planning Time Fence Control
- ☒ Display Key Performance Indicators
- ☐ Include Critical Components
- ☒ Append Planned Orders
- ☐ Move Jobs to PIP
- ☒ Lot for Lot
- ☒ Use for Sales and Operations Planning

Forecast Allocation and Consumption

- ☒ Do Not Spread Forecast
- ☐ Spread Forecast Evenly
- ☐ Consume by Forecast Bucket
- ☐ Explode Forecast
- Backward Days:** (empty)
- Forward Days:** (empty)

☒ Enable Pegging

- ☒ Peg Supplies by Demand Priority
- Reservation Level:** None
- Hard Pegging Level:** None

Note: If a plan member has already been collected in S&OP using the Download Plan Scenarios workflow and the "Use for Sales and Operations Planning" check box is then deselected in Plan Options, you must delete the plan manually from within S&OP. In other words, the Download Plan Scenarios workflow adds new members but does not delete existing plan members.

Integration Steps

1. Arrive at a proposed consensus forecast in S&OP.
2. Upload the Consensus Forecast to ASCP. From the Demantra Workflow Manager, run the applicable Upload Consensus Forecast workflow. This workflow loads the consensus forecast to an intermediate Demantra file (msd_dp_scn_entries_denorm table) which is also used when publishing from Demand Management.
3. In EBS, define the supply plan that consumes the forecast. Configure the plan options as "Use for Sales and Operations Planning".
4. In EBS, run the ASCP plan.
5. Download the ASCP plan data to Demantra S&OP. In the Demantra Workflow Manager, start the Download Plan Scenarios workflow.

6. In the Demantra Collaborator Workbench, the Supply Review Dashboard lists the new plans from EBS including those from ASCP.

Note: When scenarios are first downloaded from ASCP, they appear under "New" in the Supply Review dashboard. The other statuses that a scenario could be under are "Importing" (when data is being collected into S&OP), "Detect Exceptions", "Not Approved", and "Approved"..

7. Right-click on the ASCP plan you want to download. Choose Methods, then Load Scenario Data. This launches the Download Plan Scenario Data workflow that populates the Demantra database with ASCP data. All the data from the ASCP plan is available in all the seeded supply review worksheets in S&OP.
8. Click Yes to confirm that you want to load scenario data.
9. When the Download Plan Scenario Data workflow completes, the ASCP plan appears in the Supply Review Dashboard Detect Exceptions list.
10. Right-click on the ASCP plan and click Open, then select the type of S&OP worksheet you want to use to view the ASCP plan data. Alternatively, you can open a preseeded S&OP worksheet directly from the Collaborator Workbench.

Demantra Sales and Operations Planning to Rapid Planning Integration

Oracle Rapid Planning allows you to quickly react to unexpected events such as a new "hot demand" from an important customer, a product quality issue, a supplier yield bust, or a sudden product line breakdown. It delivers a fast, incremental planning engine combined with easy mass editing of data and a powerful user experience to instantly assess the impacts of changes without having to wait for daily tactical planning runs to complete and your ERP to catch up. Included are embedded analytics, robust exception management, and a spreadsheet-style user interface to provide predictive and actionable insight to identify the most profitable decision.

Rapid Planning and Demantra Sales and Operations Planning can work in tandem to create the best plans for your organizations. You can export the consensus forecast from S&OP, evaluate with supply constraints in Rapid Planning, and import the constrained forecast into S&OP for operational review. Additional revised forecasts from S&OP and simulations in Rapid Planning can provide alternative supply plans for review. The types of decisions that this integration would work well with include:

- Analyzing the supply impact of creating a promotion (which increases forecast for an item)
- Checking whether it is feasible from a supply standpoint to introduce a new product at a specific point in time

- Replanning based on a new set of forecast numbers at the conclusion of a forecasting cycle

Prerequisites: Configuration of Rapid Planning Plan Options

To enable the integration between Rapid Planning and S&OP, you must select the "Use for Sales and Operations Planning" option in the Rapid Planning plan options.

The Download Plan Scenarios workflow collects these eligible S&OP Rapid Planning plans and then populates the plan IDs in S&OP. To pull actual plan data, the Load Scenario Data method must be run from within S&OP for a specific plan ID.

The screenshot shows the 'Plan Options' window with the following details:

- Plan Name:** DMP1
- Plan Description:** demo plan
- Define Plan Attributes:**
 - Planned Item Type: MRP
 - Unconstrained Plan: ☐
 - End Item Substitution Set:
 - Demand Priority Rule: Schedule Date
 - Planning Time Fence Control: ☐
 - Demand Time Fence Control: ☐
 - Explode Forecast: ☐
 - ATP Enabled: ☐
 - Organization: SLC:M1
 - Planned Items: Demand Schedule Items Only
 - Assignment Set: SLC:SOP-AS1
 - Simulation Set:
 - Override: All ☐ None ☐
 - Category Set: Inv Items
 - Use for Sales and Operations Planning: ☒
 - 24 x 7: ☐
- Define Plan Attributes (continued):**
 - Number of Planning Buckets (Daily): 90
 - Plan Start Date: 1/7/2011
 - Plan End Date: 4/6/2011
- Forecast Allocation and Consumption:**
 - Forecast Spreading: Do Not Spread Forecast
 - Consume by Forecast Bucket: ☒
 - Forward Days:
 - Backward Days:

Note: If a plan member has already been collected in S&OP using the Download Plan Scenarios workflow and the "Use for Sales and Operations Planning" check box is then deselected in Plan Options, you must delete the plan manually from within S&OP. The Download Plan Scenarios workflow adds new members but does not delete existing plan members.

Note: Additional supply simulations using the same forecast can be run in Rapid Planning by creating copies of the Rapid Planning plan. Therefore, when downloading supply plan scenarios in S&OP, there might be multiple Rapid Planning plans. Ensure that copied plans also have the "Use for Sales and Operations Planning" option selected in the plan options.

Integration Steps

1. Arrive at a proposed consensus forecast in S&OP.

2. Export the consensus forecast to Rapid Planning. At this point, the forecast and demand priority are exported for Rapid Planning. From the Demantra Workflow Manager, run the applicable Upload Consensus Forecast workflow. This workflow loads the consensus forecast to an intermediate Demantra file (msd_dp_scn_entries_denorm table) which is also used when publishing from Demand Management.
3. Launch the Rapid Planning plan. The plan is simulated with the consensus forecast as it loads.
4. In EBS, define the Rapid Planning plan that consumes the forecast by configuring the plan options to "Use for Sales and Operations Planning".
5. Load the Rapid Planning plan.
6. Save the changes to the Rapid Planning plan to commit the memory-resident plan data to tables.

Important: Please do not omit this step or the plan data from Rapid Planning is not saved to tables and not available for Demantra S&OP.

7. Import the new plan into S&OP to see the impact on the consensus forecast. In the Demantra Workflow Manager, start the Download Plan Scenarios workflow.

In the Demantra Collaborator Workbench, the Supply Review Dashboard lists the new plans from EBS including those from Rapid Planning.

Note: When scenarios are first downloaded from Rapid Planning, they appear under "New" in the Supply Review dashboard. The other statuses that a scenario could be under are "Importing" (when data is being collected into S&OP), "Detect Exceptions", "Not Approved", and "Approved"..

8. In the Supply Review Dashboard, right-click on the Rapid Planning plan you want to download. Choose Methods, then Load Scenario Data. This launches the Download Plan Scenario Data workflow that populates the Demantra database with Rapid Planning data. All the data from the Rapid Planning plan is available in all the seeded supply review worksheets in S&OP.
9. Click Yes to confirm that you want to load scenario data.
10. When the Download Plan Scenario Data workflow completes, the Rapid Planning plan appears in the Supply Review Dashboard Detect Exceptions list.

11. Right-click on the Rapid Planning plan and click Open, then select the type of S&OP worksheet you want to use to view the Rapid Planning plan data. Alternatively, you can open a preseeded S&OP worksheet directly from the Collaborator Workbench.
12. Repeat the above steps until all hot demand and supply issues have been adequately handled both in the S&OP forecast and the Rapid Planning plan.

SCI Data

Collections of SCI Data

Collections use existing Demand Management collections as the basis of integration. The only difference between the Demand Management and Sales and Operations Planning administrator responsibilities is a new SCI Data menu option for S&OP. The collection parameters are:

- Instance: Instance code of source instance as defined in the Instances form.
- Collection Group: This is a group of organizations (that is, for which subset of orgs to collect data). Used to filter the collected data by organizations.
 - Default value is 'All', which implies all 'ASCP' and 'DM' enabled organizations available for the specified instance.
 - User-defined values can be specified if user defined collection groups have been created in the Instances form.
 - Only 'ASCP' and 'DM' enabled organizations can be added to the user-defined collection groups.
- Collection Method: 'Complete' or 'Net-Change'
 - Complete will clear the data in Demantra sales staging table and collect all available records from source and insert them into the Demantra sales staging table. No date filters are applied for a Complete collection. Typically used for first time collection of history data.
 - Net-Change will clear the data in Demantra sales staging table and collect data by applying the specified date filters and insert the fetched data into Demantra sales staging table. Typically used for regular collection of history data, say on a weekly basis.

Note: These are mutually exclusive.

- Date Range Type: For 'Net-Change' collection this can be 'Rolling' or 'Absolute'.
 - Not valid for 'Complete' collections.
 - If it is 'Rolling' it implies the history data will be collected from the system date up to the number of days in past as specified in the 'History Collection Window Days' field.
 - If it is 'Absolute' it implies user must enter values in the Date From and Date To fields to specify the date range for which collection is to be done.

Note: These are mutually exclusive.

- History Collection Windows Days: Valid if 'Rolling' date range type has been chosen. Used to specify the number of days in past from system date for which history data is to be collected.
- Date From and Date To: Used to specify a date range for the collection of SCI data. If no dates are specified then all available records available in source are collected without applying any date filters.

SCI Input Series

The following input series are collected from SCI:

Series	Loaded Level
Total Backlog	PRD: Item; TIM: Day; ORG: Org; GEO: Customer; DCS: All Demand Class
Past Due Backlog	PRD: Item; TIM: Day; ORG: Org; GEO: Customer; DCS: All Demand Class
Actual Onhand	PRD: Item; TIM: Day; ORG: Org; GEO: All Geo; DCS: All Demand Class
Actual Production	PRD: Item; TIM: Day; ORG: Org; GEO: All Geo; DCS: All Demand Class

Total Backlog (that is, orders that have been booked but not fulfilled) is used in conjunction with the Shipments Forecast (from Demand Management) and the Bookings Forecast (in S&OP) to calculate Projected Backlog.

Past Due Backlog is imported but not currently used in any worksheet. This series can

be used to understand how much of the Total Backlog is overdue (that is, booked orders that are still open beyond the scheduled ship date). You can compare Past Due Backlog with Total Backlog to understand the percentage of past due orders.

Actual Inventory is used to calculate Inventory Adherence %. Demantra compares the planned inventory with the actual inventory for the percentage (Actual / Planned).

Actual Production is used to calculate Production Adherence %. Demantra compares the planned production with the actual production for the percentage (Actual/Planned).

Demantra Demand Management to EnterpriseOne Integration

This chapter describes how Demantra Demand Management (DM) integrates with EnterpriseOne.

This chapter covers the following topics:

- Overview
- Architectural Process
- Integration Points Overview
- Business Process
- Mapping

Overview

Important: The content in this chapter refers to the new integration available between J.D. Edwards EnterpriseOne and the Value Chain Planning product suite (which includes Oracle Demantra) using the Process Integration Pack (PIP) *Oracle Value Chain Planning Integration to J.D. Edwards EnterpriseOne*.

It does not refer to the older flat-file based integration between JD Edwards EnterpriseOne and Demantra.

Please contact Oracle Support Services for the certification status of this PIP-based integration with Oracle Demantra version 7.3.

This chapter explains the integration processes that synchronize or move data between the Oracle Demantra and EnterpriseOne applications.

Oracle Demantra Demand Management provides access to EnterpriseOne historical sales data, returns, and other reference data organized into multiple hierarchies that

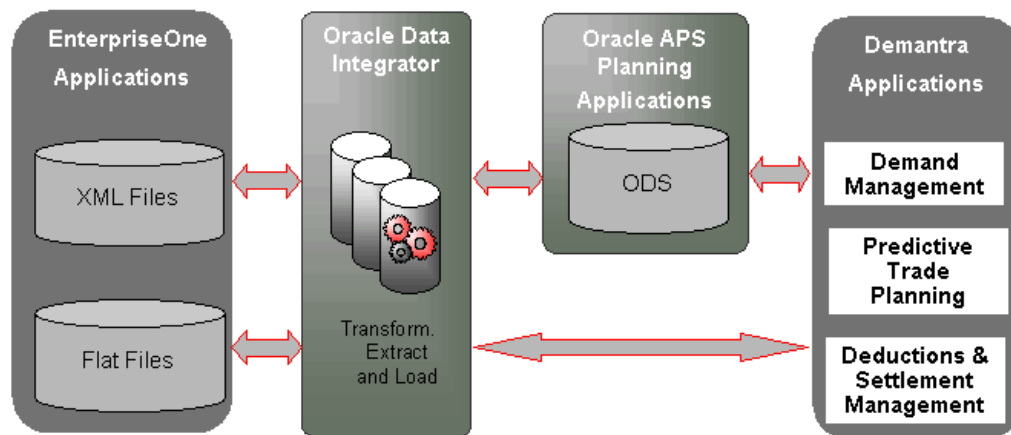
reflect the needs of your organization.

Integration between JD Edwards EnterpriseOne and Oracle Demantra Demand Management is supported by a combination of xml and flat-files that are transformed by the Oracle Data Integrator (ODI) to meet the requirements of Oracle APS Planning and Demantra Demand Management into an intermediate file structure. The intermediate file is then imported into the Oracle Demantra Demand Management data model using workflows.

Forecasts are generated and then approved within Demand Management. This process may be iterative in nature, and allows for manual intervention before finalizing the forecast. At this point, the forecast is extracted into the intermediate file structure, and in turn imported into the EnterpriseOne data model via ODI in the existing forecast table. Optionally, customers using other VCP applications like Advanced Supply Chain Planning can publish the generated forecast to the VCP tables using the standard Demantra workflows.

Architectural Process

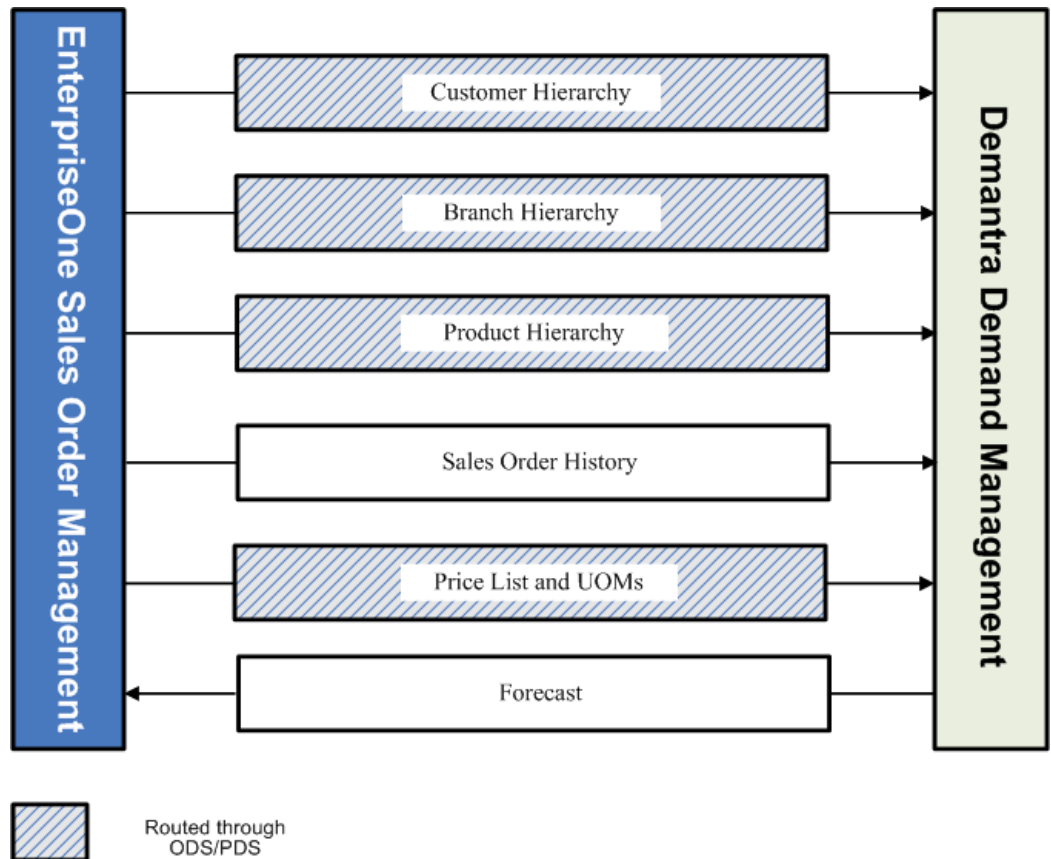
Oracle EnterpriseOne and Oracle Demantra exchange information through the use of the Oracle Data Integrator (ODI) Adapter for Value Chain Planning, the Oracle Data Integrator (ODI) to transform the data, and Oracle Demantra integration interfaces and workflows as shown in the following diagram:



The integration processes can be run when required from Oracle EBS Advanced Planning.

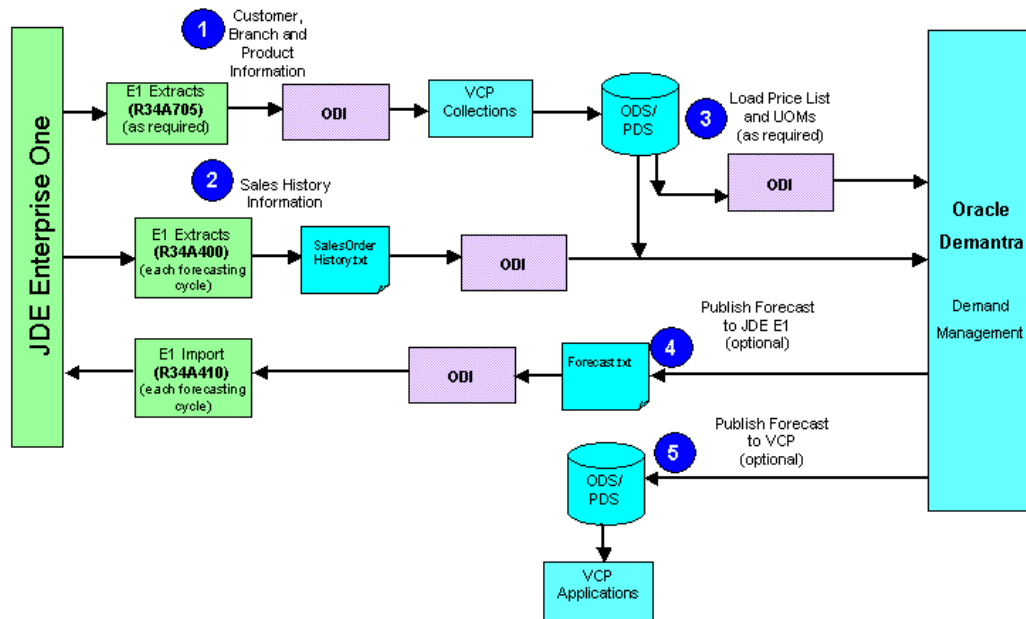
Integration Points Overview

The following integration points are part of the integration between the Oracle Demantra Demand Management module and EnterpriseOne applications.



Business Process

The following diagram shows the flow of data between EnterpriseOne and Demantra Demand Management:



Note: The relevant EnterpriseOne extracts need to be run prior to launching the collection programs in APS.

The process is as follows:

1. From APS, run Collect Planning Data. This loads ODS with planning information from EnterpriseOne including customer, branch and product information.
2. From APS, run Collect Sales History. This program processes the SalesOrderHistory.txt file extracted from EnterpriseOne, transforms it and loads it into Demantra.
3. From APS, run Collect Price List and UOM. This loads the price list and UOM information into Demantra.
4. Within Demantra, create a forecast based on the sales history gathered from EnterpriseOne.
5. From APS, run Publish Forecast to Source System. This generates a flat file named forecast.txt and loads it into EnterpriseOne.
6. (Optional) The forecast can also be published for use by other VCP applications using the Demantra workflows "EBS Upload Local Forecast" and "EBS Upload Local Fcst, Demand Class".

For more details about how to implement EnterpriseOne integration with Demantra Demand Management, see:

- *Oracle Value Chain Planning Integration to JD Edwards EnterpriseOne Installation Guide*
- *Oracle Value Chain Planning Integration to JD Edwards EnterpriseOne Implementation Guide*

Mapping

There are two files transferred between Oracle Demantra Demand Management and EnterpriseOne after the Collect Planning Data procedure has been run from APS Planning (which loads Customer, Branch and Product information into ODS/PDS, as well as other planning data). They are:

- Sales Order History (EnterpriseOne to Demantra)
- Forecast (Demantra to EnterpriseOne)

Sales Order History

The purpose of the Sales Order History extract is for EnterpriseOne to provide either full or incremental sales order history to Demantra Demand Management. Details:

EnterpriseOne Extract Name: SalesOrderHistory.txt

Transformed File Name: DemHistory.dat

ODI Package Name: LoadE1SalesOrderHistoryDataToDMPkg

Demantra Workflow: EBS Full Download

See the *Oracle Value Chain Planning Integration to JD Edwards EnterpriseOne Implementation Guide* for more information about how to change the download of the sales order history extract from full to incremental.

Sample layout of the SalesOrderHistory.txt file from EnterpriseOne:

EnterpriseOne Field	Example
Order Number	3363
Order Type	SO
Order Company	200
Order Line No	1
Business Unit	TORONTO

EnterpriseOne Field	Example
Customer No Sold To	1744
Customer No Ship To	1744
Short Item Number	9797760
2nd Item Number	OCLV_110_DROP_OUT
3rd Item Number	OCLV_110_DROP_OUT
Actual Ship Date	
Invoice Date	
Quantity	54
Requested Date	5/8/2006
Scheduled Pick Date	5/8/2006
Promised Shipment Date	5/8/2006
Promised Delivery Date	5/8/2006
Price Per Unit	105.62
Company	200

When you launch the "Collect Sales History" program from the "Advanced Supply Chain Planner" or the "Demand Management System Administrator" responsibility in Oracle EBS, the SalesOrderHistory.txt file is transformed by ODI into the format required by Demantra.

The format required by Demantra follows the format of the DemHistory.dat file that is used to load sales history information into Demantra using the legacy collections framework as follows:

DemHistory.dat Fields	Example	Mapping to SalesOrderHistory.txt
ACTUAL_QTY	54	<p>Quantity field from SalesOrderHistory.txt.</p> <p>Note: For a given combination of item, business unit (branch in EnterpriseOne, organization in Oracle), customer ship to and requested date, if there are multiple records in SalesOrderHistory.txt then the quantities are aggregated and a single record is created in DemHistory.dat.</p>
SALES_DATE	8-May-06	Requested Date from SalesOrderHistory.txt
DM_ITEM_CODE	9797760	Short Item Number field from SalesOrderHistory.txt
DM_ORG_CODE	TORO1	<p>Business Unit field from SalesOrderHistory.txt</p> <p>Note: As there is a restriction in the field size for the organization code in the Value Chain Planning ODS tables, the business unit name of "TORONTO" in this example is translated into "TORO1" table used in the new integration. Refer to "Creating User Maintained Data" in <i>Oracle Value Chain Planning Integration to JD Edwards EnterpriseOne Implementation Guide</i></p>

DemHistory.dat Fields	Example	Mapping to SalesOrderHistory.txt
DM_SITE_CODE	Olympia Sports Inc:3150:3150+Olympia Sports Inc:OU1	<p>Customer.xml:customer:parentName+Parameter.Demantra_Field_Delimiter+Customer.xml:customer:parentAddressNumber+Parameter.Demantra_Field_Delimiter customer.xml:customer:customerCode+Parameter.Field_Delimiter+Customer.xml:customer.name+Parameter.Demantra_Field_Delimiter+OPERATING_UNIT</p> <ul style="list-style-type: none"> Customer.xml is one of the main extract files from EnterpriseOne that contains Customer information. Refer to <i>Oracle Value Chain Planning Integration to JD Edwards EnterpriseOne Implementation Guide</i> for more details on Customer.xml Data from Customer.xml is found using customerCode (in Customer.xml:customer) = "SalesOrderHistory.txt:7.Customer No Ship To" If parentAddressNumber is null in Customer.xml from EnterpriseOne, then use the customer:customerCode instead, and if the parentName is null, in Customer.xml, then use the customer.name. <p>For more details on the below fields in User Defined Integration Data.xml, please refer to "Creating User Maintained Data" in <i>Oracle Value Chain Planning Integration to JD Edwards EnterpriseOne</i></p>

DemHistory.dat Fields	Example	Mapping to SalesOrderHistory.txt
<p><i>Implementation Guide :</i></p> <ul style="list-style-type: none"> Operating Unit is the above customers' category code pointed to by the parameter "Operating_Unit_Category" in the User Defined Integration Data.xls. Parameter.Demantra_Field_Delimiter = Value defined for parameter called "Demantra_Field_Delimiter" in the Parameters table in User Defined Integration Data.xls. Default value is ';' (semicolon). Parameter.Field_Delimiter = Value defined for parameter "Field_Delimiter" in the Parameters table in User Defined Integration Data.xls. Default value is '+'. 		
EBS_SALES_CHANNEL_CODE	DRT	Value of the customer category code in Customer.xml pointed to by the parameter Sales_Channel_Category in the Parameters table in User Defined Integration Data.xls, where Customer.xml:customer:customerCode = "Customer No Ship To" in SalesOrderHistory.txt.
EBS_DEMAND_CLASS_CODE	EST	Value of the customer category code in Customer.xml pointed to by the parameter Demand_Class_Category in the Parameters table in User Defined Integration Data.xls, where Customer.xml:customer:customerCode = "Customer No Ship To" in SalesOrderHistory.txt.

DemHistory.dat Fields	Example	Mapping to SalesOrderHistory.txt
EBS_BOOK_HIST_BOOK_QTY_ _BD	54	Quantity field from SalesOrderHistory.txt.
EBS_BOOK_HIST_REQ_QTY_ BD		<null>
EBS_BOOK_HIST_BOOK_QTY_ _RD		<null>
EBS_BOOK_HIST_REQ_QTY_ RD		<null>
EBS_SHIP_HIST_SHIP_QTY_S D		<null>
EBS_SHIP_HIST_SHIP_QTY_R D		<null>
EBS_SHIP_HIST_REQ_QTY_R D		<null>
EBS_PARENT_ITEM_CODE		<null>
EBS_BASE_MODEL_CODE		<null>

Forecast

When you run the "Publish forecast to source system" program from the Advanced Supply Chain Planner or the Demand System Administrator responsibility in APS, the forecast from Demantra is exported out into a file called forecast.txt and is transformed to the format required by EnterpriseOne. Details:

Demantra Extract Name: forecast.txt

Note: ODS is case-sensitive to file names; ensure that forecast.txt is all lower case.

ODI Package Name: LoadDMForecast

Demantra Integration Interface Name: AIA-E1 Upload

Demantra Integration Data Profile Name: AIA-Forecast data

Demantra Workflow: AIA-Forecast_Export

The forecast.txt file is transformed by ODI and then loaded into EnterpriseOne. The layout of the forecast.txt file from Demantra is as shown:

Field Name	Sample Data
Short Item Number	9797760
Branch/Plant	TORONTO
<Blank Field>	
Sales Date	4/7/2008 0:00
Customer No Sold To	1744
Final Forecast	80

The forecast is exported out of Demantra into forecast.txt at an Item, Organization and Customer site level. The customer site value exported out of Demantra follows the concatenated format that is used in EBS-Demantra collections. See "Sales Order History" for more information about the format of the DM_SITE_CODE field in DemHistory.dat mapping table. ODI transforms the value of the customer site to the EnterpriseOne format.

Demantra Predictive Trade Planning to EnterpriseOne Integration

This chapter describes how Demantra Predictive Trade Planning (PTP) integrates with EnterpriseOne.

This chapter covers the following topics:

- Overview
- Architectural Process
- Integration Points Overview
- Business Process
- Promotion Method Configuration
- Modeling Considerations
- Mapping

Overview

Important: The content in this chapter refers to the new integration available between J.D. Edwards EnterpriseOne and the Value Chain Planning product suite (which includes Oracle Demantra) using the Process Integration Pack (PIP) *Oracle Value Chain Planning Integration to J.D. Edwards EnterpriseOne*.

It does not refer to the older flat-file based integration between JD Edwards EnterpriseOne and Demantra.

Please contact Oracle Support Services for the certification status of this PIP-based integration with Oracle Demantra version 7.3.

The Oracle Demantra Predictive Trade Planning (PTP) solution provides industry leading capabilities for companies that are primarily in the Consumer Goods industry,

to plan, manage and optimize their trade promotion spending.

PTP Integration Builds On Demand Management Integration

PTP integration builds on existing Oracle Demantra Demand Management integration. Please read the chapter "Demantra Demand Management to EnterpriseOne Integration" first. All of the information there applies to integrating Predictive Trade Planning, except where specifically noted in this chapter. In particular, the following collections must be run prior to running the PTP collections:

1. Collect Planning Data
2. Collect Sales History
3. Collect Price List and UOM

For more information about Oracle Demantra integration with the JD Edwards EnterpriseOne application suite, see the guides titled:

- *Oracle Value Chain Planning Integration to JD Edwards EnterpriseOne Installation Guide*
- *Oracle Value Chain Planning Integration to JD Edwards EnterpriseOne Implementation Guide*

Batch-based Approach Using Flat Files

The integration between JD Edwards EnterpriseOne and Oracle Demantra is a batch-based approach utilizing flat files to exchange information. It is accomplished through the use of Oracle EBS collection procedures, Oracle Data Integrator, Oracle Demantra workflows and JD Edwards EnterpriseOne RunUBEXml procedures.

JD Edwards EnterpriseOne to Oracle Demantra

The APS collection procedure "Collect PTP Data" performs the following steps:

1. Collects the EnterpriseOne extracts generated by the runubexml program.
2. Passes the flat file through the Oracle Data Integrator where the data is transformed.
3. Populates the Demantra interface tables.
4. Runs an Oracle Demantra workflow that moves the information from the interface tables to the production tables.

Oracle Demantra to JD Edwards EnterpriseOne

The APS collection procedure "Publish PTP Results" performs the following steps:

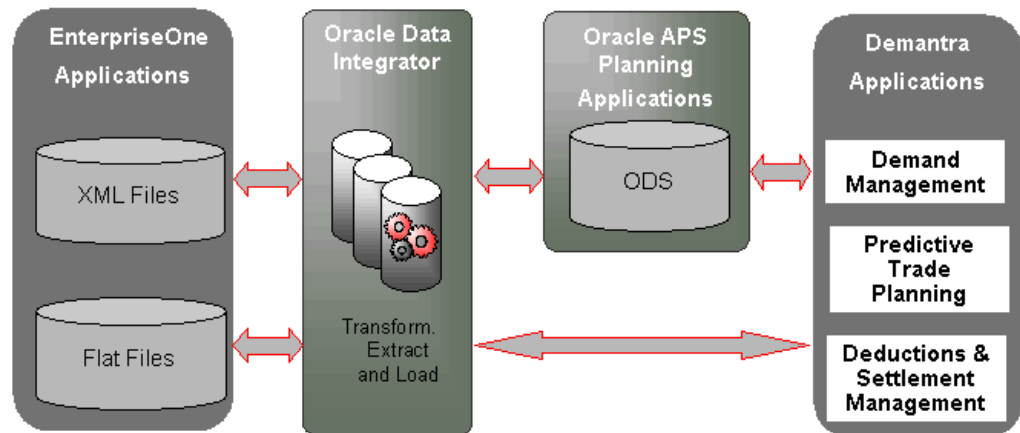
1. Runs a Demantra workflow to extract the required information from Demantra and

send the data to a flat file.

2. Passes the flat file through the Oracle Data Integrator where the data is transformed.

Architectural Process

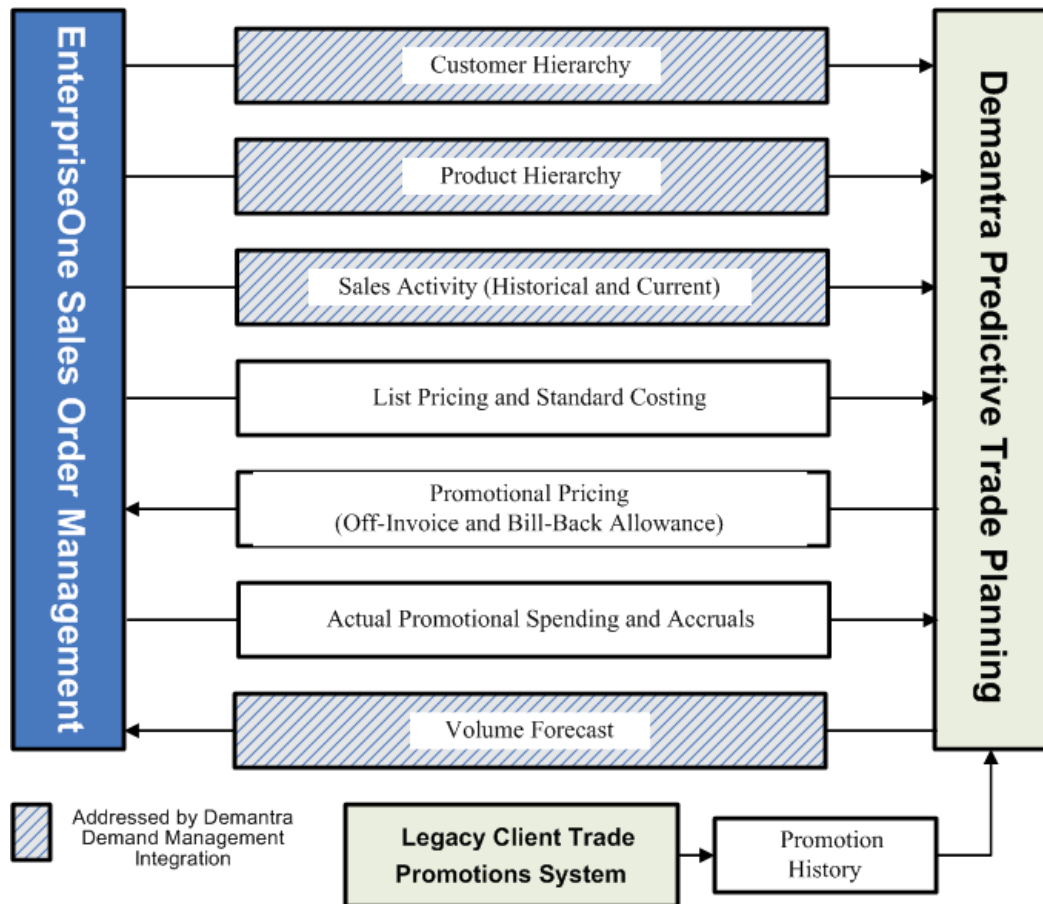
Oracle EnterpriseOne and Oracle Demantra exchange information through the use of the Oracle Data Integrator (ODI) Adapter for Value Chain Planning, the Oracle Data Integrator (ODI) to transform the data, and Oracle Demantra integration interfaces and workflows as shown in the following diagram:



The integration processes can be run when required from APS Planning. After EnterpriseOne to Demantra Demand Management integration loads Demantra with sales history, price list and units of measure data, PTP data can be collected (list price, item cost and price history). The PTP data flows directly from EnterpriseOne to Demantra via ODI, where it is transformed.

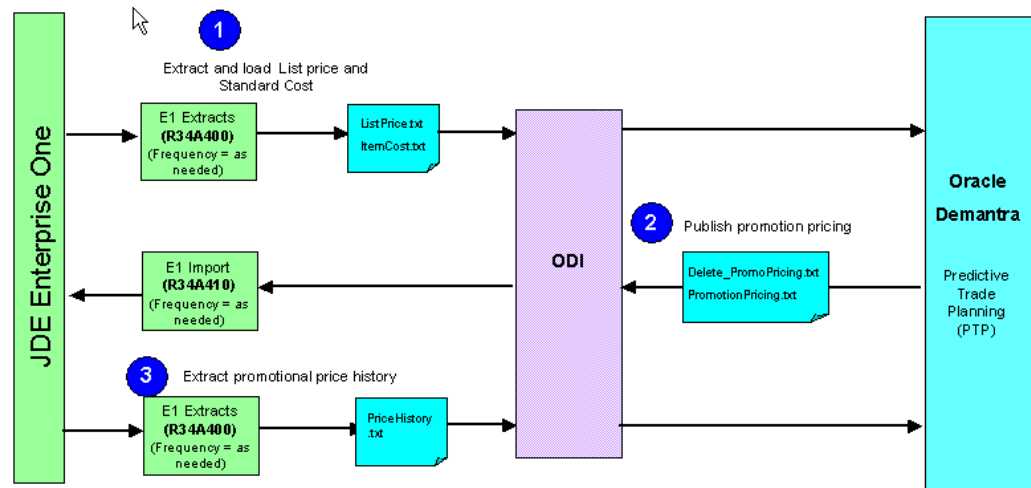
Integration Points Overview

The following integration points are part of the integration between the Oracle Demantra Predictive Trade Planning (PTP) module and JD Edwards EnterpriseOne applications.



Business Process

The following diagram provides more details about the business process and data that flows between EnterpriseOne and Oracle Demantra PTP:



Note: The relevant EnterpriseOne extracts need to be run prior to launching the collection programs in APS.

The business process is as follows:

1. In APS, run "Collect Planning Data" to collect master data to Oracle APS Planning (ODS).

- **Customer Hierarchy** – Customer Master information is interfaced from J D Edwards EnterpriseOne to Oracle APS Planning (ODS), then to Oracle Demantra where it populates the Demantra Site Hierarchy. This process is part of the Demantra Demand Management to EnterpriseOne integration.
- **Branch Hierarchy** – Branch master information is interfaced from J.D. Edwards EnterpriseOne to Oracle APS Planning (ODS), then to Oracle Demantra where it populates the Demantra Organization Hierarchy. This process is part of the Demantra Demand Management to EnterpriseOne integration.
- **Product Hierarchy** – Product Master information is interfaced from J D Edwards EnterpriseOne to Oracle APS Planning (ODS), then to Oracle Demantra where it populates the Demantra Item Hierarchy. This process is part of the Demantra Demand Management to EnterpriseOne integration.

2. In APS, run "Collect Sales History" to collect sales history from EnterpriseOne to Demantra.

Sales Activity (Historical and Current) – Shipment information interfaced from EnterpriseOne to Demantra populates the Demantra sales data structure. This interface supports loading historical sales data as well as ongoing updates of current sales data. This process is part of the Demantra Demand Management to EnterpriseOne integration.

3. **In APS, run "Collect Price List and UOM" to collect pricing and units of measure from Oracle APS Planning (ODS).** This procedure adds pricing and units of measure to the sales activity data already transferred from EnterpriseOne.
4. **In APS, run "Collect PTP Data" to collect list price, item cost and price history from EnterpriseOne.**
 - **List Price and Standard Cost** – Future List Price and Standard Cost information interfaced from EnterpriseOne to Demantra populates the Demantra sales data structure. This information is used when planning future promotions.
 - **Promotion History** – Historical promotional information must be loaded into Demantra at the beginning of an implementation for the Demantra promotion modeling engine to predict the impact of future promotions. This information is expected to come from a legacy Trade Promotions Management system. The existing Demantra Integration Interface is used to load this information.
5. **In Demantra PTP, run the forecasting engine**

The Predictive Trade Planning forecasting engine generates baseline forecasts for the customer hierarchy and product hierarchy based on the sales activity and promotion history.
6. **In Demantra PTP, plan promotions.**

Practitioners use the Predictive Trade Planning module and process to plan future promotions.
7. **In APS, run "Publish PTP Results" to load future promotional pricing from Demantra to EnterpriseOne.**

Promotional Pricing (Off-Invoice and Bill-Back Allowances) – Promotional Pricing information interfaced from Demantra to the EnterpriseOne Advanced Pricing module applies correct price discounts during order management.
8. **In EnterpriseOne, process orders against new promotions.**
9. **Extract Price History information from EnterpriseOne (PriceHistory.txt).**
10. **In APS, run "Collect PTP Data" to load price history into Demantra, ensuring that the Price History parameter is set to "Yes"**

Actual Promotional Spending or Accruals – The actual amount spent (for off-invoice promotions) or accrued (for bill-back promotions) interfaced from EnterpriseOne to Demantra captures the actual cost of promotions.
11. **In APS, run "Publish Forecast to Source System" to load the Volume Forecast from Demantra to EnterpriseOne.**

Volume Forecast – Oracle Demantra generates a projected volume forecast and

provides this information to JD Edwards EnterpriseOne. This process is part of the Demantra Demand Management to EnterpriseOne integration.

Promotion Method Configuration

Pre-seeded Workflows are provided for flagging promotions that need to be integrated to EnterpriseOne. During implementation the standard promotion methods must be updated to use these Workflows instead of the standard ones. The following configuration must be done in order to ensure that promotions are properly integrated to EnterpriseOne.

In Business Modeler, select "Configuration" and then "Configure Methods". Select "Promotion" in the drop-down and then edit each of the following methods (double-click on the method in the list to edit it).

- New Promotion - change Workflow from "Create Member" to "New Promotion"
- Edit Promotion – change Workflow from "Edit Member" to "Edit Promotion"
- Delete Promotion - change Workflow from "Delete Member" to "Delete Promotion"

Modeling Considerations

Note: PTP integration builds on existing Oracle Demantra Demand Management integration. For information about Oracle Demantra Demand Management integration, see "Demantra Demand Management to EnterpriseOne Integration".

Levels

The following Oracle Demantra level supports EnterpriseOne to Demantra PTP integration:

Integration Status

Description	Integration Status is a parent to the Promotion level. It is an indicator that is used to determine whether or not a promotion needs to be exported for EnterpriseOne.
Demantra Hierarchy	Integration Status - Promotion

Level Type	General Level
Demantra Database Table	promo_integ_status

Attributes

The following Oracle Demantra attribute supports EnterpriseOne to Demantra PTP integration:

promo_integ_status

Description	This is a lookup level to the new Integration Status level.
Demantra Level	Promotion
Demantra Database Table	Promotion
Demantra Database Column	promo_integ_status_id

Series

The following Oracle Demantra series support EnterpriseOne to Demantra PTP integration:

BB Amt

Description	BB Amt is the actual Bill-Back Accrual amount as provided by EnterpriseOne
Data Table	Promotion
Update Field	bb_amt
Data Type	Numeric

OI Amt

Description	OI Amt is the actual Off-Invoice Spending amount as provided by EnterpriseOne
Data Table	Promotion
Update Field	oi_amt
Data Type	Numeric

Promotion Desc

Description	Promotion Desc is a series that points to the existing description field on the Promotion level. Required for exporting Promotion Description using a Data Profile.
Data Table	<level> promotion
Update Field	N/A
Data Type	String

Promotion Pay Indicator

Description	The Promotion Pay Indicator shows whether the Promotion Rate Amount is Off-Invoice ("O") or Bill-Back ("B"). Used in the Promotional Pricing export for EnterpriseOne.
Data Table	Promotion
Update Field	promo_pay_indicator
Data Type	String

Promotion Rate Amt

Description	This is the Promotion Rate Amount that is exported for EnterpriseOne. This equals the existing Buy-Down series multiplied by negative one. (EnterpriseOne requires promotional discounts to be represented as negative numbers.)
Data Table	Promotion
Update Field	promo_rate_amt
Data Type	Numeric

Promotion Integration Status

Description	Promotion Integration Status is a series that points to the new promo_integ_status attribute on the Promotion level.
Data Table	<Level> promotion
Update Field	promo_integ_status_id
Data Type	Numeric

Promotion History Load

During the initial implementation of Demantra PTP, it is recommended that two years of promotional history is loaded. It is unlikely that this information will reside in EnterpriseOne. As such, no automated feed of historical promotions from EnterpriseOne to Demantra is provided. Historical promotions should be loaded from the applicable source using the standard Demantra promotional load process.

Customer Hierarchy Load

When the customer hierarchy data is loaded into Demantra during Demand Management integration, the Retailer and Bill To levels in Demantra are populated with relevant data from EnterpriseOne. The Retailer level is populated with the Parent Address Number (into the code field of the level) and Parent Address Name (into the code description field of the level). If the Parent Address Number and Parent Address Name are null in EnterpriseOne, the Customer Code and Customer Name are

populated instead.

For a PTP implementation, it is recommended that the Parent Address Number and Parent Address Name fields in EnterpriseOne be populated. Bill To level is populated with the Bill To Number (into the code field of the level) and the Bill To Name (into the description field of the level).

Mapping

Note: PTP integration builds on existing Oracle Demantra Demand Management integration. For information about Oracle Demantra Demand Management integration, see *Demantra Demand Management to EnterpriseOne Integration*.

There are four files transferred between Oracle Demantra PTP and EnterpriseOne:

- Price History (EnterpriseOne to Demantra)
- List Price (EnterpriseOne to Demantra)
- Standard Cost (EnterpriseOne to Demantra)
- Promotion Pricing (Demantra to EnterpriseOne)

Note: You may need to modify the default values of time filters defined in the integration interface data profiles to suit your implementation needs.

Pricing History

The purpose of the Price History extract is for EnterpriseOne to provide the actual amount spent (for off-invoice promotions) or accrued (for bill-back promotions). This captures the actual cost of promotions in Demantra. Details:

EnterpriseOne Extract Name: PriceHistory.txt

ODI Package Name: LoadE1PriceHistoryDataToDMPkg

Demantra Integration Interface Names:

- AIA-LoadPriceHistoryBB (bill-back promotions)
- AIA-LoadPriceHistoryOI (off-invoice promotions)

Demantra Integration Data Profile Names:

- AIA-LoadPriceHistoryBB

- AIA-LoadPriceHistoryOI

Demantra Workflow: AIA-E1ToPTP_PriceHistory_Download

To prevent loading the same amount multiple times, the load process checks for duplicate records in prior loads. If a record has already been loaded, it is bypassed during the current load. This allows overlapping EnterpriseOne extract date ranges to be used without overstating the amounts in PTP. Duplicate records are determined by an exact match on the following eleven fields in the EnterpriseOne extract file:

- Document (Order No, Invoice)
- Order Type
- Order Company (Order Number)
- Line Number
- Order Suffix
- Price History Alternate Key
- Price History Alternate Key Source
- Sequence Number
- Sub-Sequence Number
- Tier
- Target Application

Caution: The Price History load requires that the promotion, for which the actual off-invoice spending or bill-back accrual amount applies, already exists in Demantra. In the case of historical information, if the corresponding promotions have not been loaded into Demantra, this load will generate errors.

Historical promotional spending is not a requirement for Demantra. Therefore no recovery is required if the historical promotional spending load does not complete successfully.

The Price History extract fields are transformed by ODI to match the format of the fields in the new integration and then the file is loaded into Demantra. The layout of the Price History extract from EnterpriseOne is as shown:

EnterpriseOne Extract Field Name	Field Type and (Value)	Demantra Mapping
Document (Order Number, Invoice)	Number (8)	Used to check for duplicates
Order Type	Varchar2 (2)	Used to check for duplicates
Order Company (Order Number)	Varchar2 (5)	Used to check for duplicates
Line Number	Number (6,3)	Used to check for duplicates
Short Item Number	Number (10)	Maps to the Item level
Customer No Ship To	Number (10)	Maps to the Site level
Business Unit	Varchar2 (200)	Maps to the Organization level
Actual Ship Date	Date	Maps to the Time Period
Promotion ID	Varchar2 (12)	Maps to the Promotion level
Cost Type	Char	"O" for Off-Invoice "B" for Bill-Back
Amount - Extended Price	Number (15,15)	The actual extended amount. Maps to "OI Amt" if the Cost Type equals "O". Maps to "BB Amt" if the Cost Type equals "B".
Order Suffix	Varchar2 (3)	Used to check for duplicates
Price History Alternate Key	Number (15)	Used to check for duplicates
Price History Alternate Key So	Varchar2 (2)	Used to check for duplicates
Sequence Number	Number (4)	Used to check for duplicates

EnterpriseOne Extract Field Name	Field Type and (Value)	Demantra Mapping
Sub-Sequence Number	Number (4)	Used to check for duplicates
Tier	Number (2)	Used to check for duplicates
Target Application	Char (1)	Used to check for duplicates

List Price

The purpose of this interface is to load Manufacturer's List Prices for future time periods. List Price is used to calculate profitability when planning future promotions. Details:

EnterpriseOne Extract Name: ListPrice.txt

ODI Package Name: LoadE1ListPriceDataToDMPkg

Demantra Integration Interface Names:

- LoadPriceGlobal - Global list prices that apply to all customers; imports list price information into the "List Price SD" series by Item, Organization levels.
- AIA-LoadPriceSpecific - Customer-specific list price information; imports list price information into the "List Price SD" series by Item, Organization, and Site levels.

Demantra Integration Data Profile Names:

- PriceGlobal
- AIA-PriceSpecific

Demantra Workflow: AIA-E1ToPTP_PromoPrice_Download

Date Range

EnterpriseOne provides this information with an Effective Date and an Expiration Date. During load processing, this date range is converted into individual time periods to align with the Demantra data structure. For example, using a weekly model a date range of January 1, 2008 through December 31, 2008 would be converted into 52 individual week entries.

EnterpriseOne has the capability to define customer specific prices or global prices that apply to all customers. If the Customer Number field is null, then the List Price applies to all customers.

1. The load processing will first load all global prices where the Customer Number

field is null. In Demantra this will be stored for all active Site / Item / Organization combinations for each of the time periods within the date range.

2. The load processing then loads any customer specific prices for all active Item / Organization combinations for the customer for each of the time periods within the date range.

This two-step process will overlay the global prices initially loaded with any customer specific prices.

The List Price extract fields are transformed by ODI to match the format of the fields in the new integration and then the file is loaded into Demantra. The layout of the List Price extract from EnterpriseOne is as shown:

EnterpriseOne Extract Field Name	Field Type and (Value)	Demantra Mapping
Short Item Number	Number (8)	Maps to the Item level.
2nd Item Number	Varchar2 (25)	Not Used
3rd Item Number	Varchar2 (25)	Not Used
Branch/Plant	Varchar2 (12)	Maps to the Organization level
Customer Number	Number (8)	Maps to the Site level. If null, then List Price applies to all customers.
Expired Date	Date	The date through which the List Price is valid.
Effective Date	Date	The date the List Price goes into effect.
Amount - List Price	Calculated	The List Price.

Standard Cost

The purpose of this interface is to load Manufacturer's Cost Of Goods Sold (COGS) values for future time periods. Details:

EnterpriseOne Extract Name: ItemCost.txt

ODI Package Name: LoadE1ItemCostDataToDMPkg

Demantra Integration Interface Name: LoadCostGlobal

Demantra Integration Data Profile Name: CostGlobal

Demantra Workflow: AIA-E1ToPTP_PromoCost_Download

COGS is used to calculate profitability when planning future promotions. COGS is stored in the existing "COGS SD" series which resides on the Sales Data table in Demantra. This table contains an entry for each unique Site / Item / Organization / Time Period combination. COGS information from EnterpriseOne does not include Effective or Expiration Dates. When received it is assumed to take effect immediately and stay in effect through the last future date for which data is stored in the application.

EnterpriseOne does not have the capability to define customer specific COGS values. The COGS value applies to all customers. In Demantra, data are stored for all active Site / Item / Organization combinations for each of the future time periods

The COGS extract fields are transformed by ODI to match the format of the fields in the new integration and then the file is loaded into Demantra. The layout of the COGS extract from EnterpriseOne is as shown:

EnterpriseOne Extract Field Name	Field Type and (Value)	Demantra Mapping
Short Item Number	Number (8)	Maps to the Item level.
Branch/Plant	Varchar2 (12)	Maps to the Organization level
2nd Item Number	Varchar2 (25)	Not Used
3rd Item Number	Varchar2 (25)	Not Used
Item Cost	Calculated	The COGS value
Cost Method	Number (8)	Not Used
Cost Method Description	Varchar2 (30)	Not Used

Promotional Pricing (Off-Invoice and Bill-Back Allowances)

The purpose of this interface is to provide promotional pricing information to EnterpriseOne to be used in the Advanced Pricing module. Details:

Extract Names:

- PromotionPricing.txt
- Delete_PromoPricing.txt

ODI Package Name: LoadDMPromotionPricingDataToE1Pkg

Demantra Integration Interface Name: AIA-E1UploadPromoPrices

Demantra Integration Data Profile Name: AIA-ExtractPromoPrices

Demantra Workflow: AIA-PTPTOE1_UploadPromotionPrices

Promotions are exported from Demantra if they meet the following criteria:

- A promotion has an off-invoice or bill-back allowance. This is based on the Payment Type. Promotions with only a Fixed Cost or with a Scan-Down allowance are not sent to EnterpriseOne since they do not impact order pricing.
- A promotion has one of the following statuses, based on the Promotion Status level. Promotions in an Unplanned or Planned status are not sent to EnterpriseOne.
 - c. Approved
 - d. Committed
 - e. Partial Paid
 - f. Paid
 - g. Closed
- It is a new promotion, or an existing promotion that has been edited. This is based on the new Integration Status level.

The following table defines the layout of the Promotional Pricing file and maps the fields to the corresponding fields in Demantra:

EnterpriseOne Field Name	EnterpriseOne Field Type	Demantra Type	Demantra Level or Series	Comments
Promotion Row ID	Number (15)	Level	Promotion	Demantra Promotion ID
Promotion Name	String (50)	Series	Promotion Desc	Demantra Promotion Description
Customer Number	Number (10)	Level	Site	EnterpriseOne Ship-To Customer Number

EnterpriseOne Field Name	EnterpriseOne Field Type	Demantra Type	Demantra Level or Series	Comments
Customer Number Description	String (30)		Null Value	Blank
Short Item Number	Number (10)	Level	Item	EnterpriseOne Item Number
Item Number Description	String (30)	Level	Item Description	EnterpriseOne Item Description
Promotion Effective Date	Date	Series	Start Ship	Starting Ship Date of Promotion
Promotion Expiration Date	Date	Series	End Ship	Ending Ship Date of Promotion
Promotion Amount	Number (15)	Series	Promotion Rate Amt	The Promotion Buy-Down Rate multiplied by -1 (EnterpriseOne requires that price discounts are negative values)
Cost Type	Char	Series	Promotion Pay Indicator	Payment Type: "O" for Off-Invoice / "B" for Bill-Back. This indicates whether the Promotion Amount is off-invoice or bill-back.

When new promotions are sent to EnterpriseOne, this file contains a record for every Account and Item combination on the promotion. Regardless of the customer or product level that the promotion is created at in Demantra, the information will always be sent to EnterpriseOne at the Account location level and the Item product level. This lowest level is typically the Ship-To Customer / SKU level. If a promotion is deleted in Demantra after it has been sent to EnterpriseOne, the following entries are sent to EnterpriseOne, which will cause the promotion expire in EnterpriseOne:

- Promotion Row ID = Demantra Promotion ID
- Promotion Name = Demantra Promotion Description

- Customer Number = Null
- Customer Number Description = Null
- Short Item Number = Null
- Item Number Description = Null
- Promotion Effective Date = Yesterday (must be prior date)
- Promotion Expiration Date= Yesterday (must be prior date)
- Promotion Amount = 0
- Cost Type = one entry with a value of "O" and one entry with a value of "B"

If a promotion is modified in Demantra after it has been sent to EnterpriseOne, a set of delete entries (as described previously) will first be sent to EnterpriseOne followed by a set of new promotion entries. This will cause the promotion to be replaced in EnterpriseOne.

Note: Promotional changes are handled by completely replacing the promotion in EnterpriseOne. There is no logic to provide delta changes to EnterpriseOne.

Promotional Pricing Workflow

The AIA-PTPTOE1_UploadPromotionPrices workflow creates and processes two versions of the Promotional Pricing file. The first pass contains only delete entries. These are for promotions that were deleted, or the delete portion for promotions that were modified. The second pass contains only new entries. These are for new promotions, or the new (replacement) portion for promotions that were modified.

Interfaces

Database stored procedures are used to generate the delete promotions version of the Promotional Pricing file. The new promotions version of the Promotional Pricing file is generated using the Integration Interface AIA-E1UploadPromoPrices. It uses the AIA-ExtractPromoPrices Data Profile to export the information based on the criteria defined previously.

Two versions of the promotional pricing export with EnterpriseOne are supported to identify modified and deleted promotions. The details are as follows:

1. The modified and deleted promotions are identified using the "Integration Status" level, which has the following members: New Promotion, Edited Promotion and No Changes to Promotion.
2. A stored procedure step at the end of each of the following existing workflows:

- **New Promotion** – sets the promotion to belonging to the Parent Level "New Promotion" (also sets the last_update_date for the promotion level to sysdate)
 - **Edit Promotion** – stores the Promotion ID, Code and Description of the edited promotion in a database table , as well as sets the promotion to belonging to the Parent Level "Edited Promotion" (also sets the last_update_date for the promotion level to sysdate)
 - **Delete Promotion** – stores the Promotion ID, Code and Description of the deleted promotion in a database table
3. A series called "Promotion Integration Status" captures changes to any of the following series which impact the EnterpriseOne integration:
- Start Ship Date
 - End Ship Date
 - Buy-down Rate unit
 - Payment Type

This series has a client expression with an "is modified" expression used to capture changes to the 4 series above, if they are changed a the "Integration Status" Promotion Level is set to "Edited Promotion".

4. At the end of the workflow, after all export processes have run, the "Integration Status" is set to "No Changes to Promotion" for all promotions.

Demantra Deductions and Settlement Management to EnterpriseOne Integration

This chapter describes how Demantra Deductions and Settlement Management (DSM) integrates with EnterpriseOne.

This chapter covers the following topics:

- Overview
- Architectural Process
- Integration Points
- Business Process
- Modelling Considerations
- Mapping

Overview

Important: The content in this chapter refers to the new integration available between J.D. Edwards EnterpriseOne and the Value Chain Planning product suite (which includes Oracle Demantra) using the Process Integration Pack (PIP) *Oracle Value Chain Planning Integration to J.D. Edwards EnterpriseOne*.

It does not refer to the older flat-file based integration between JD Edwards EnterpriseOne and Demantra.

Please contact Oracle Support Services for the certification status of this PIP-based integration with Oracle Demantra version 7.3.

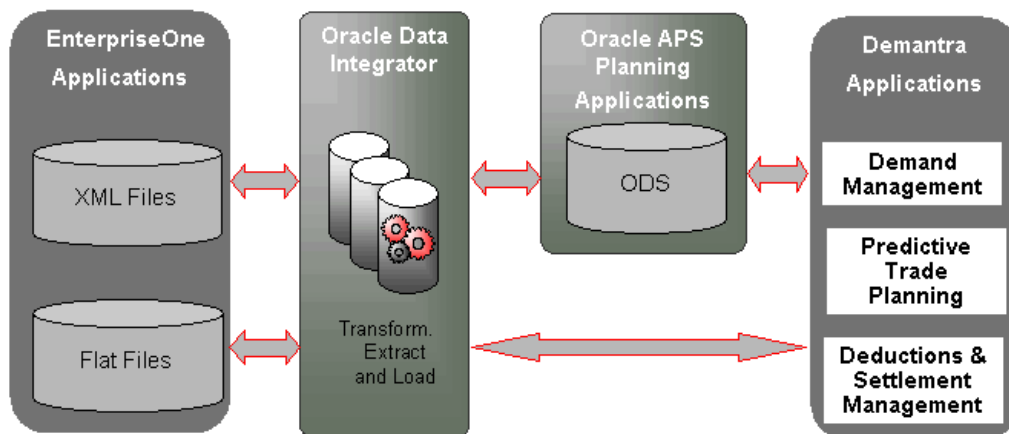
The Oracle Demantra Deductions and Settlement Management (DSM) product requires implementation of Oracle Demantra Demand Management and Predictive Trade Planning (PTP) as prerequisites. This chapter describes integrating DSM to the JD

Edwards EnterpriseOne application suite, given DM and PTP have already been integrated. For more information, see:

- "Demantra Demand Management to EnterpriseOne Integration"
- "Demantra Predictive Trade Planning to EnterpriseOne Integration"
- *Oracle Value Chain Planning Integration to JD Edwards EnterpriseOne Installation Guide*
- *Oracle Value Chain Planning Integration to JD Edwards EnterpriseOne Implementation Guide*

Architectural Process

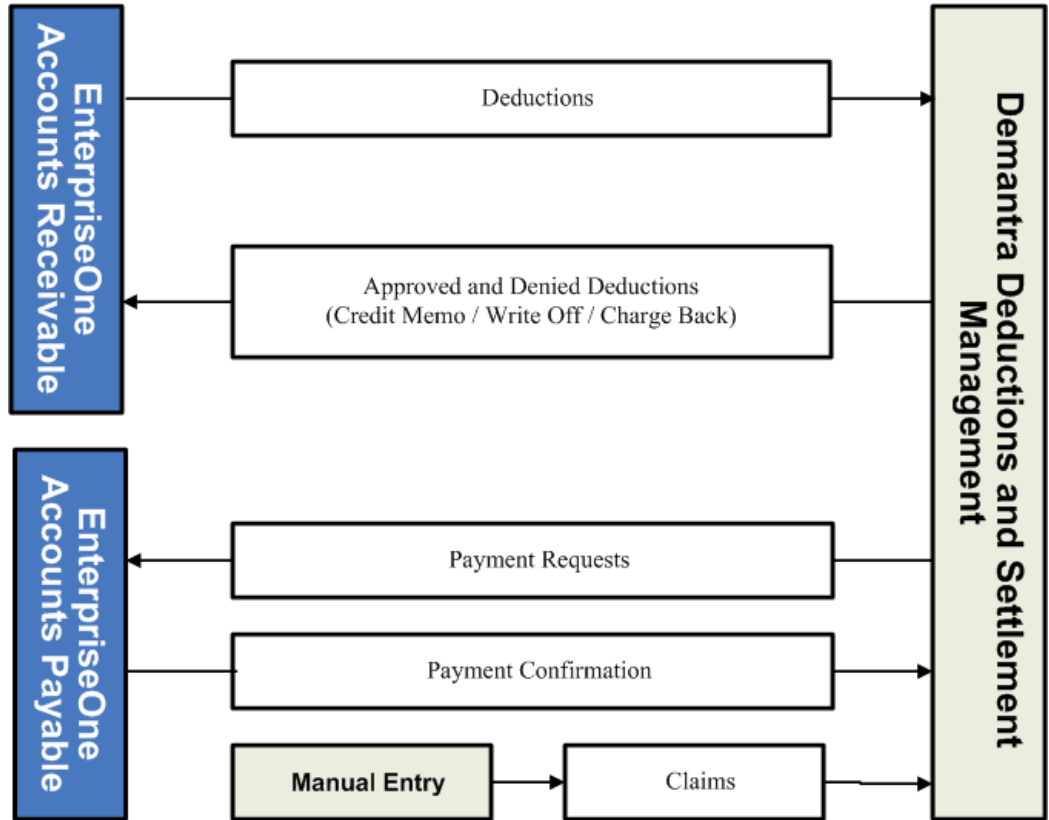
Oracle EnterpriseOne and Oracle Demantra exchange information through the use of the Oracle Data Integrator (ODI) Adapter for Value Chain Planning, the Oracle Data Integrator (ODI) to transform the data, and Oracle Demantra integration interfaces and workflows as shown in the following diagram:



The integration processes can be run when required from APS Planning. After EnterpriseOne to Demantra Demand Management integration loads Demantra with sales history, price list and units of measure data, DSM data can be collected (deductions, claims). The DSM data flows directly from EnterpriseOne to Demantra via ODI, where it is transformed.

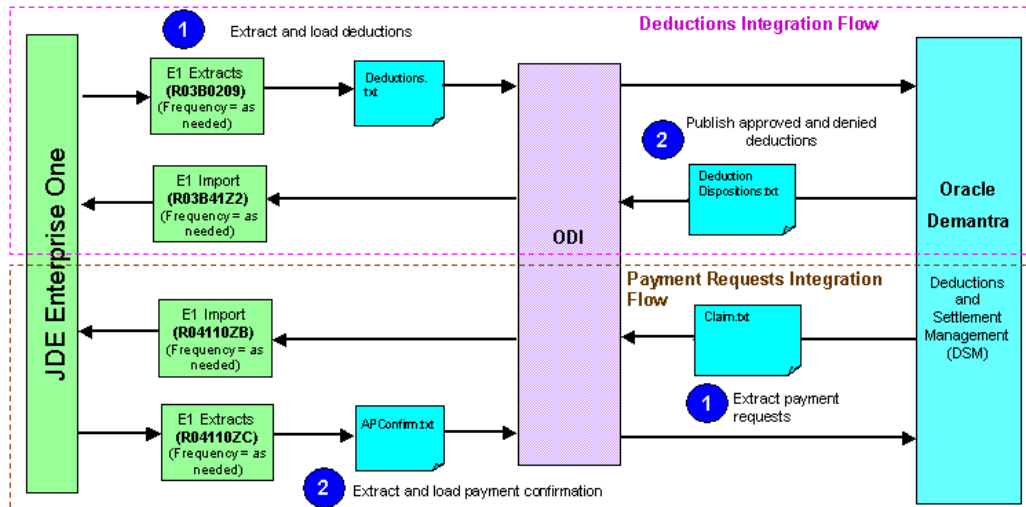
Integration Points

The following integration points are part of the integration between the Oracle Demantra Deductions and Settlement Management (DSM) module and JD Edwards EnterpriseOne applications.



Business Process

The following diagram provides more details about the business process and data that flows between EnterpriseOne and Oracle Demantra DSM:



Note: Extracting information from EnterpriseOne needs to be done completely by the user prior to running any of the collection programs that bring data into DSM (such as Deductions and Payment Confirmations). Similarly, importing information back into EnterpriseOne needs to be done after running any publish programs that export data out of DSM (such as Approved/Denied Deductions and Payment Requests).

The business process is as follows:

1. From APS, run Collect DSM Data with the parameter "Load Deductions" = Yes.

Deductions – Deduction information extracted from the EnterpriseOne Accounts Receivable (A/R) module is loaded into DSM via ODI. When new deductions are created in EnterpriseOne, they are sent from EnterpriseOne to DSM via ODI and loaded as settlements of type "Deduction".

2. Review deductions in DSM, match them to promotions, and approve or deny the deductions.

3. From APS, run Publish DSM Results with the parameter "Publish Deduction Dispositions" = Yes.

Approved and Denied Deductions – When deductions are cleared in DSM, the resulting information is transformed by ODI and available for import back into the EnterpriseOne Accounts Receivable (A/R) module. You will need to import the Deduction Dispositions file into EnterpriseOne.

Note: A single deduction received from EnterpriseOne can be split into multiple deductions in DSM. Each of the split deductions

interfaces back to EnterpriseOne separately. Only deductions that have reached a status of Approved or Denied interface to EnterpriseOne.

4. **Enter claims in DSM, match them to promotions and approve the claims.**

5. **From APS, run Publish DSM Results with parameter "Publish Claims" = Yes.**

Payment Requests – Requests for Payment, such as Check Requests, are exported from DSM and available for import into the EnterpriseOne Accounts Payable (A/P) module. You will need to import the Claims into EnterpriseOne.

6. **Review the payment request in EnterpriseOne and issue payments against the payment requests from DSM and extract the payment requests from EnterpriseOne.**

7. **From APS, run Collect DSM Data with parameter "Load Payment Confirmations" = Yes.**

Payment Confirmation – The payment confirmations extracted from EnterpriseOne are transformed by ODI and loaded into DSM where they can be viewed against the claims entered in Step 4.

The following additional interface to Oracle Demantra is provided:

Claims Entry – Claims for trade payment must be manually entered into the Demantra DSM module.

Modelling Considerations

Setup

The following must be manually setup in preparation for the integration:

- The "GL Code" level in the Settlement hierarchy must have an entry with a "GL Code Desc" value of "Default."
- The "Status" level in the Settlement hierarchy must have an entry with a "Settlement Status Desc" value of "New."
- The "Type" level in the Settlement hierarchy must have an entry with a "Settlement Type Desc" value of "Deduction."
- The "Invoice" level in the Settlement hierarchy must have an entry with an "Invoice Desc" value of "DEFAULT SETTLEMENT Invoice."

- The "Linked Promotion" level in the Settlement hierarchy must have an entry with a "Promotion Desc" value of "Default Promotion."
- The "GL Code" level in the Settlement hierarchy must be populated with valid EnterpriseOne Reason Codes.

The table containing valid EnterpriseOne Currency Codes is called "curr_code".. This table will need to be populated with valid EnterpriseOne currency codes. Initial values are:

CODE_ID	CURR_CODE	CODE_DESC
0	USD	US Dollar
1	CAD	Canadian Dollar
2	EUR	European Euro

Series

The following Oracle Demantra series support the integration:

- The "Closed Settlement Amount" references the existing settlement_amount field on the Settlement table for Settlements with a Status of:
 - Approved
 - Duplicate
 - Denied, or
 - Write Off
- "Settlement Account ID" that references the existing t_ep_lr2_EP_ID field on the Settlement table.

Attributes

The following attributes and series support the DSM integration. The "Corresponding Series" column indicates the series that corresponds to the attribute. Series are required to show attributes in worksheets and include them in the integration interface data profiles:

Note: For every row in this table, the Demantra Level is: "Settlement"

and the Demantra Database Table is "SETTLEMENT".

EnterpriseOne Field Name	EnterpriseOne Field Type	Demantra Level Attribute	Demantra Database Table Column	Corresponding Series	Comments
Settlement Amount (Transaction Currency)	Number (15,2)	Settlement Amount Native Curr	SETTLEMENT _AMOUNT_ NATIVE_CURRENCY	Settlement Amount Native Curr	Settlement amount in the native currency. Applies to both Deduction and Claims.
Transaction Currency Code	String (3)	Native Curr	NATIVE_CURRENCY_ID	Native Curr	Native currency code. Applies to both Deduction and Claims
Display Decimals	Integer (1)	Native Display Decimals	NATIVE_DISPLAY_DECIMALS	Native Display Decimals	Number of decimal positions that the native currency amount should be rounded to during conversions and splits
Document – Original	Number (8)	EnterpriseOne Document	E1_DOCUMENT_NT	E1 Document	EnterpriseOne reference. Not used in DSM
Document Type – Original	String (2)	EnterpriseOne Document Type	E1_DOCUMENT_NT_TYPE	E1 Document Type	EnterpriseOne reference. Not used in DSM
Document Pay Item – Orig	String (3)	EnterpriseOne Document Pay Item	E1_DOCUMENT_NT_PAY_ITEM	E1 Document Pay Item	EnterpriseOne reference. Not used in DSM

EnterpriseOne Field Name	EnterpriseOne Field Type	Demantra Level Attribute	Demantra Database Table Column	Corresponding Series	Comments
Document Company (Original)	String (5)	EnterpriseOne Document Company	E1_DOCUMENT_COMPANY	E1 Document Company	EnterpriseOne reference. Not used in DSM
Invoice Date	Date	EnterpriseOne Invoice Date	E1_INVOICE_DATE	E1 Invoice Date	Invoice Date for invoice that deduction occurred on. Since we won't be loading Invoices from EnterpriseOne, it is probably best to not try to populate the existing Invoice Date on the invoice file.
Check Amount (Base Currency)	Number (15,2)	Cust Check Amount	CUST_CHECK_AMOUNT	Cust Check Amount	Amount of the customer's check in the base currency that paid the invoice from which the deduction was created
Check Amount (Transaction Currency)	Number (15,2)	Cust Check Amount Native Curr	CUST_CHECK_AMOUNT_NATIVE_CURR	Cust Check Amount Native Curr	Amount of the customer's check in the native currency that paid the invoice from which the deduction was created
Currency Code	String(3)	Cust Check Native Curr Code	CUST_CHECK_NATIVE_CURR_CODE	Cust Check Native Curr Code	Currency Code for the customer's check
Display Decimals	Integer(1)	Cust Check Display Decimals	CUST_CHECK_DISPLAY_DECIMALS	Cust Check Display Decimals	Display decimals for the customer's check amount in native currency

EnterpriseOne Field Name	EnterpriseOne Field Type	Demantra Level Attribute	Demantra Database Table Column	Corresponding Series	Comments
Claim Invoice #	Char	Claim Invoice Num	CLAIM_INV OICE_NUM	Claim Invoice Num	The Invoice Number from the claim paperwork. Manually entered when entering Claim.
Claim Invoice Date	Date	Claim Invoice Date	CLAIM_INV OICE_DATE	Claim Invoice Date	The Invoice Date from the claim paperwork. Manually entered when entering claim.
Settlement Type	String (2)	Check Type	CHECK_TYPE	Check Type	Indicates the type of payment issued in EnterpriseOne to satisfy the check request
Check Amount	Number (15,2)	Check Amount	CHECK_AMOUNT	Check Amount	The amount of the payment generated from EnterpriseOne for the claim
Currency	String (3)	Check Currency	CHECK_CURRENCY	Check Currency	The currency code for the payment issued from EnterpriseOne
Multiple Payments	String (10)	Multiple Payment Indicator	MULTIPLE_PAYMENT_INDICATOR	Multiple Payment Indicator	Indicates if multiple payments were issued in EnterpriseOne to satisfy the claim

Customer Load Enhancements

Bill-To to Customer Hierarchy Load – DSM activities are performed at a bill-to customer level. The deduction and claim information exchanged with EnterpriseOne is at the Bill-To level. The integration between J.D. Edwards EnterpriseOne and Demantra Demand Management loads the bill-to information from EnterpriseOne into the Bill To level in Demantra. A Bill To level is preconfigured in the Demantra site hierarchy and DSM contains a preconfigured level called Invoiced Bill To that is mapped to the Bill To level in the site hierarchy.

The integration loads Bill-To Address Number from EnterpriseOne into the code field

of the Bill To level and it loads the Bill-To Address Name from EnterpriseOne into the description field of the Bill To level.

Multi-Currency Enhancements

Multi-Currency Support – EnterpriseOne supports multi-currency. Therefore, deductions can be created in different currencies in EnterpriseOne. When deductions are sent from EnterpriseOne to DSM both the native, or transaction, currency amount and a base, or common, currency amount are included. DSM is enhanced to store the native currency amount, and to recalculate this amount when a deduction is split. When approved or denied deductions are sent from DSM to EnterpriseOne, the amount in the native currency is included.

Mapping

There are four files transferred between Oracle Demantra DSM and JD Edwards EnterpriseOne. They are:

- New Deductions (EnterpriseOne to Demantra DSM)
- Approved and Denied Deductions (Demantra DSM to EnterpriseOne)
- Payment Requests (Demantra DSM to EnterpriseOne)
- Payment Confirmation (EnterpriseOne to Demantra DSM)

Lastly, there is another interface, Claims Entry, which is a manual method for users to enter claims into Demantra DSM.

Note: You may need to modify the default values of the time filters defined in the integration interface data profiles to suit your implementation needs.

New Deductions

The purpose of this interface is to extract new Deductions from EnterpriseOne and load them into DSM as Settlements of Type "Deduction". Details:

EnterpriseOne Extract Name: Deductions.txt

ODI Package Name: LoadE1DeductionsDataToDMPkg

Demantra Integration Interface Names:

- Settlement Invoice Integration
- SETTLEMENT LEVEL import

Demantra Integration Data Profile Name:

- Invoice Import Integration
- SETTLEMENT Import

Demantra Workflow: AIA-E1ToDSM_New Deduction_Download

Note: This Workflow contains a step to create an Invoice for every Bill-To that exists in the Bill-To level (parent of Site). The Invoice Code is set to the Bill-To Code. When a Settlement is loaded, it is related to the Invoice that corresponds to the Bill-To on the Settlement.

The following table maps the fields in the flat file exported from EnterpriseOne to the corresponding fields in Demantra after being transformed by ODI:

Note: For every row in this table, the Level is: "Settlement" and the Table is "SETTLEMENT".

EnterpriseOne	Field Type	Settlement Level	Column	New
Extract Field Name	(Value)	Attribute		
E1 Deduction Id	Number (15)	N/A	SETTLEMENT_CODE	
Settlement Date of Origin	Date	Date Posted	DATE_POSTED	
Settlement Amount (Base Currency)	Number (15,2)	Settlement Amount	SETTLEMENT_AMOUNT	
Settlement Amount (Transaction Currency)	Number (15,2)	Settlement Amount Native Curr	SETTLEMENT_AMOUNT_NATIVE_CURR	Y

EnterpriseOne	Field Type	Settlement Level	Column	New
Extract Field Name	(Value)	Attribute		
Transaction Currency Code	String (3)	Native Curr Code	NATIVE_CURR_CODE	Y
Display Decimals	Integer (1)	Native Display Decimals	NATIVE_DISPLAY_DECIMALS	Y
Document – Original	Number (8)	EnterpriseOne Document	E1_DOCUMENT	Y
Document Type – Original	String (2)	EnterpriseOne Document Type	E1_DOCUMENT_TYPE	Y
Document Pay Item – Orig	String (3)	EnterpriseOne Document Pay Item	E1_DOCUMENT_PAY_ITEM	Y
Document Company (Original)	String (5)	EnterpriseOne Document Company	E1_DOCUMENT_COMPANY	Y
Invoice Date	Date	EnterpriseOne Invoice Date	E1_INVOICE_DATE	Y
Check #	String (25)	Customer Check #	CUSTOMER_CHECK_NUM	

EnterpriseOne Extract Field Name	Field Type (Value)	Settlement Level Attribute	Column	New
Check Date	Date	Cust Check Date	CUSTOMER_CHECK_DATE	
Check Amount (Base Currency)	Number (15,2)	Cust Check Amount	CUST_CHECK_AMOUN	Y
Check Amount (Transaction Currency)	Number (15,2)	Cust Check Amount Native Curr	CUST_CHECK_AMOUNT_NATIVE_CURR	Y
Currency Code	String (3)	Cust Check Native Curr Code	CUST_CHECK_NATIVE_CURR_CODE	Y
Display Decimals	Integer (1)	Cust Check Display Decimals	CUST_CHECK_DISPLAY_DECIMALS	Y
G/L or Reason code	Default	GL Code	GL_CODE_ID	
Status	New	Status	SETTLEMENT_STATUS_ID	
Settlement Type	Deduction	Settlement Type	SETTLEMENT_TYPE_ID	
Company	String (5)	NA	NA	
Customer Number	Number (8)	Account	T_EP_LR2_EP_ID	

Note: Customer Number is the Customer Bill-to Number.

The following table maps the required fields for a Settlement Level import:

Demantra Field Name	EnterpriseOne Extract Field Name	Comments
SETTLEMENT_CODE	E1 Deduction ID	
SETTLEMENT_DESC	E1 Deduction ID	
INVOICE_CODE	Bill-To Customer	Invoice is a required field and is used to identify the Bill-To Customer for the Settlement. Since actual invoices are not loaded from E1 to DSM, an Invoice is automatically created for each Bill-To. The Invoice Code will equal the Bill-To Code, which is populated with the EnterpriseOne Bill-To Number.
GL_CODE_CODE	G/L or Reason code	A GL Code in the Settlement hierarchy with a value of "Default" must exist
SETTLEMENT_STATUS_CODE	Status	A Status in the Settlement hierarchy with a value of "New" must exist
SETTLEMENT_TYPE_CODE	Settlement Type	A Type in the Settlement hierarchy with a value of "Deduction" must exist
PROMOTION_CODE	Hard-coded value "Default Promotion"	A Linked Promotion in the Settlement hierarchy with this value must exist.
DATE_POSTED	Settlement Date of Origin	

Approved and Denied Deductions

The purpose of this interface is to extract approved and denied Deductions from DSM and load them into EnterpriseOne. Details:

Demantra Extract Name: DeductionDispositions.txt

ODI Package Name: LoadDMDedDispositionsDataToE1Pkg

Demantra Integration Interface Name: E1 Deduction Export

Demantra Integration Data Profile Name: E1DeductionExport

Demantra Workflow: AIA-DSMtoE1_Deduction_Export

The following table defines the layout of the Approved and Denied Deduction file and maps the fields to the corresponding fields in Demantra:

Note: For every row in this table, the Demantra Level is: "Settlement" and the Demantra Database Table is "SETTLEMENT".

EnterpriseOne Field Name	EnterpriseOne Field Type	Demantra Series	Demantra SETTLEMENT Database Table Column	Comments
E1 Deduction Id	Number (15)	N/A	SETTLEMENT_ CODE	Settlement Level Code
Settlement Amount (Base Currency)	Number (15,2)	Closed Settlement Amount	SETTLEMENT_ AMOUNT	Not used by EnterpriseOne..
Settlement Amount (Native Currency)	Number (15,2)	Settlement Amount Native Curr	SETTLEMENT_ AMOUNT_NATIV E_CURR	
Deduction Reason Code	String (2)	GL Code	GL_CODE_ID	Based on codes manually provided during implementation.
Status (Approved / Denied)	N/A	Settlement Status	SETTLEMENT_ST ATUS_ID	Not used by EnterpriseOne.

Claims Entry

The purpose of this interface is to provide a method for users to manually enter claims into Demantra DSM. The "New Settlement" method is used to enter claims into Demantra DSM.

Attribute	Editable	Mandatory	Comments
Settlement Type	Y	Y	Select value "Claim" from drop-down list.
Settlement Invoice	Y	Y	Select from a drop-down list of invoices which have been automatically populated with an invoice for each Bill-To Customer.
E1 Company Code	Y	Y	Enter the E1 Company Code that applies to the claim. The default value is 00000. This value is mandatory for the claim to be successfully imported into EnterpriseOne.
Settlement Amount	Y	Y	Enter claim amount in base currency
Settlement Amount (Native Currency)	Y	Y	Enter claim amount in native/transaction currency
Native Curr	Y		Select claim native/transaction currency from a list of currency codes
Native Display Decimals	Y		
Date Posted	Y	Y	Enter date
Claim Invoice Num	Y		
Claim Invoice Date	Y		
Settlement Status	Y		
GL Code	Y		Default value of "DEFAULT GL Code"
Settlement Owner	Y		

Payment Requests

The purpose of this interface is to extract check requests for approved claims from DSM and load them into EnterpriseOne. Details:

Demantra Extract Name: Claims.txt

ODI Package Name: LoadDMClaimDataToE1Pkg

Demantra Integration Interface Name: E1 Claim Export

Demantra Integration Data Profile Name: E1_ClaimExport

Note: This data profile does an incremental export of settlements with a Type of "Claim" or "Claim resulting FROM an original claim split" and a Status of "Approved".

Demantra Workflow: AIA-DSMToE1_Claim_Export

The following table defines the layout of the Approved Check Requests file and maps the fields to the corresponding fields in Demantra:

Note: For every row in this table, the Demantra Level is: "Settlement" and the Demantra Database Table is "SETTLEMENT".

EnterpriseOne	EnterpriseOne	Demantra Series	Demantra SETTLEMENT	Comments
Field Name	Field Type	Level Attribute	Database Table Column	
Check Request ID	String (22)	N/A	SETTLEMENT_CODE	Code field for Settlement that the Check Request is for. Link to SETTLEMENT using SETTLEMENT_ID from CHECK_REQUEST
Customer / Account	String (14)	Settlement Account ID	T_EP_LR2_EP_ID	

EnterpriseOne	EnterpriseOne	Demantra Series	Demantra SETTLEMENT	Comments
Field Name	Field Type	Level Attribute	Database Table Column	
E1 Company Code	String (5)	E1 Company Code	E1_COMPANY_CODE	The Customer Number (T_EP_LR2_EP_ID) and E1 Company Code fields are concatenated together with a space when the workflow AIA-DSMToE1_Claim_Export exports the payment requests into claim.txt. For example, if Customer Number=4663 and E1 Company Code=00200, then the resulting second field in claim.txt is "4663 00200".
Claim Invoice #	String (25)	Claim Invoice Num	CLAIM_INVOICE_NUM	
Claim Invoice Date	Date	Claim Invoice Date	CLAIM_INVOICE_DATE	
Settlement Date of Origin	Date	Settlement Date Posted	DATE_POSTED	
Settlement Amount (Base Currency)	Number (15,2)	Closed Settlement Amount	SETTLEMENT_AMOUNT	
Settlement Amount (Native Currency)	Number (15,2)	Settlement Amount Native Curr	SETTLEMENT_AMOUNT_NATIVE_CURR	

EnterpriseOne	EnterpriseOne	Demantra Series	Demantra SETTLEMENT	Comments
Field Name	Field Type	Level Attribute	Database Table Column	
Transaction Currency Code	String (3)	Native Curr	NATIVE_CURR_CODE	

Payment Confirmation

The purpose of this interface is to extract processed payments from EnterpriseOne and update the corresponding Check Request in DSM to indicate that the payment has been issued. Details:

EnterpriseOne Extract Name: APConfirm.txt

ODI Package Name: LoadE1APConfirmDataToDMPkg

Demantra Integration Interface Name: E1 Payment Confirmation Import

Demantra Integration Data Profile Name: E1_Payment_Import

Demantra Workflow: AIA-E1ToPTP_APConfirm_Import

The following table maps the fields in the flat file exported from EnterpriseOne to the corresponding fields in Demantra:

Note: For every row in this table, the Demantra level is: "Settlement" and the Demantra database table is "SETTLEMENT".

EnterpriseOne	EnterpriseOne	Demantra Settlement	Demantra SETTLEMENT
Field Name	Field Type	Level Attribute	Database Table Column
Check Request ID	String (30)	N/A	SETTLEMENT_CODE
Check #	Number (8)	Supplier Check #	SUPPLIER_CHECK_NUM
Settlement Type	String (2)	Check Type	CHECK_TYPE
Check Amount	Number (15,2)	Check Amount	CHECK_AMOUNT
Currency	String(3)	Check Curr Code	CHECK_CURRENCY

EnterpriseOne	EnterpriseOne	Demantra Settlement	Demantra SETTLEMENT
Field Name	Field Type	Level Attribute	Database Table Column
Check Date	Date	Supplier Check Date	SUPPLIER_CHECK_DATE
Multiple Payments	String(10)	Multiple Payment Indicator	MULTIPLE_PAYMENT_INDICATOR

Demantra Predictive Trade Planning to Siebel Integration

This chapter provides detailed information about how Demantra Predictive Trade Planning (PTP) integrates with Siebel Trade Promotion Management (TPM).

This chapter covers the following topics:

- Introduction
- Architectural Process
- Integration Point Overview
- Business Process
- Setup Overview
- Siebel Setup
- ODI Setup
- Demantra Setup
- Main Batch File Configurations
- Running Integration
- Siebel to Demantra ODI Packages and Batch Files
- Demantra to Siebel ODI Packages and Batch Files

Introduction

To date, most consumer goods companies have focused on gaining control over spending and achieving administrative efficiency in promotion execution—what is referred to as Foundational Trade Promotion Management (TPM). This is a critical first step, but only the beginning of achieving the full potential benefits of automating the TPM process. This foundation provides the transactional data to learn from historical information and apply this learning to improve future sales and operations

performance.

Oracle Siebel TPM provides a robust, mature solution to support Foundational TPM. This includes functionality related to planning and executing trade promotions, managing funds, planning volumes, and managing deductions.

The next level, TPM Intelligence, refers to a set of analytic, predictive modeling and optimization capabilities that enable consumer goods companies to plan more profitable promotions resulting in improved sales performance.

Oracle Demantra Predictive Trade Planning (PTP) provides the TPM Intelligence. It includes a promotion simulation capability to predict the lift, incremental volume, and profitability of promotions before they are executed. The advanced promotional modeling engine determines the true net lift achieved by decomposing the promotional lift into cannibalization, halo effect, pre and post activity, and brand switching. The optional Promotion Optimization add-on provides the ability to automatically identify the best promotion to run based on identified goals and constraints.

By integrating the Oracle Siebel TPM application with Demantra PTP, the Siebel TPM solution is enhanced with volume intelligence, promotion intelligence, advanced account planning, and promotion optimization.

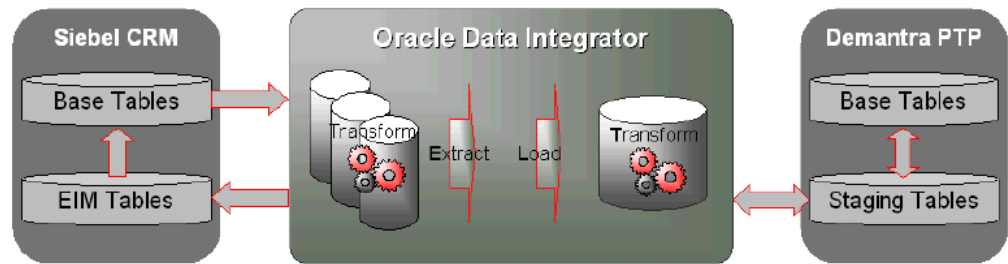
There are two scenarios for using Siebel TPM and Demantra PTP together. They are:

- Promotion planning done in Siebel with Demantra running in the background. The Demantra baseline forecast and lift coefficients are transferred into Siebel on a regular basis so that this information is available during the promotion planning process.
- Promotion planning done in either Siebel or Demantra with the results synchronized every 30 minutes. The user has the ability to add, modify, and delete promotions in either application and have the changes synchronized to the other application in a near real-time fashion.

The Demantra-Siebel integration includes a set of interfaces, pre-seeded workflows, related hierarchy levels and data series.

Architectural Process

The integration between Oracle Siebel and Oracle Demantra is a batch-based approach utilizing database staging tables to exchange information. It is accomplished through the use of Siebel's Enterprise Integration Manager, the Oracle Data Integrator (ODI) to transform the data, and Oracle Demantra integration interfaces and workflows in the following diagram:



The integration processes can be scheduled on a nightly or weekly basis for high-volume integration points such as accounts, products, and shipment data. On-going integration processes can provide near real-time synchronization of promotional plans. For example, you can specify promotions that you want transferred to Demantra every 30 minutes for near real-time integration.

Integration Point Overview

The following integration points are part of the integration between the Oracle Demantra PTP module and Oracle Siebel applications.



The following table details the frequency and recommended load for each integration point is detailed below:

Interface	Flow	Initial Load	On-going Load	On-going Frequency
Account Planning Hierarchy	Siebel to Demantra	Full Load	Full Load	Nightly
Product Planning Hierarchy	Siebel to Demantra	Full Load	Full Load	Nightly
Promotion Tactics Master List	Siebel to Demantra	Full Load	N/A	N/A
Promotion Status Master	Siebel to Demantra	Full Load	N/A	N/A
Price and Cost	Siebel to Demantra	Future Load (18 months)	Full Load	Nightly
Sales Activity	Siebel to Demantra	Historical Load (2 years)	Incremental Load	Nightly
Plan Master List	Siebel to Demantra	Historical Load (2 years)	Incremental Load	Nightly
Promotions	Siebel to Demantra	Historical Load (2 years)	Incremental Load	Nightly
Promotions	Siebel to Demantra	Historical Load (2 years)	Selected Promotions	Every 30 Minutes
Promotions	Demantra to Siebel	N/A	Selected Promotions	Every 30 Minutes
Baseline Volume Forecast	Demantra to Siebel	Future Load (18 months)	Full Load	Weekly
Promotional Lift Coefficients	Demantra to Siebel	Future Load (18 months)	Full Load	Weekly

Business Process

There are two scenarios for using Siebel TPM and Demantra PTP together. They are:

- Siebel Trade Planning/Demantra Intelligence (Regular Synchronization)
- Demantra PTP/Siebel TM (Near Real-Time and Regular Synchronization)

Siebel Trade Planning/Demantra Intelligence Business Process (Regular Synchronization)

All promotion planning is done in Siebel and the Demantra user interface is not accessed. Regularly scheduled processes generate a baseline forecast and lift coefficients in Demantra and transfer this information to Siebel for use during the promotion planning process.

1. The Sales Account Manager logs into Siebel and creates a new promotion. He specifies the account and the associated products/promoted product groups. He also enters all the relevant details associated with the promotion such as tactics and cost components.
2. The Sales Account Manager initiates the "Baseline" process in Siebel to populate the baseline volume for the promotion. Note that the Demantra generated baseline, which has already been loaded into Siebel, is used.
3. The Sales Account Manager initiates the "Simulate" process in Siebel to retrieve and apply lift coefficients for the promotion. Note that the Demantra generated lift coefficients, which have already been loaded into Siebel, are used.
4. The Sales Account Manager finalizes the promotion and submits it for approval.

Nightly Extract Transfers

The following interfaces are required for synchronization:

- Account Planning Hierarchy (Siebel to Demantra)
- Product Planning Hierarchy (Siebel to Demantra)
- Sales Activity (Siebel to Demantra)
- Price and Cost (Siebel to Demantra)
- Plan Master List (Siebel to Demantra)
- Promotions (Siebel to Demantra)

Weekly Extract Transfers

The following interfaces are required for synchronization:

- Baseline Volume Forecast (Demantra to Siebel)
- Promotional Lift Coefficients (Demantra to Siebel)

Demantra Predictive Trade Planning/Siebel Trade Management (Near Real-time and Regular Synchronization)

Promotion planning is done using both Siebel and Demantra. The user has the ability to add and modify promotions in either application and to automatically have the changes synchronized to the other application in a near real-time fashion.

There are two options:

- Siebel is the primary promotion planning application.
- Demantra is the primary promotion planning application.

Siebel Primary Promotion Planning Application

The following is the business process when Siebel is preferred as the primary promotion planning application:

1. The Sales Account Manager logs into Siebel and selects an account plan. He adds new promotions to the plan and modifies existing promotions.
2. For any promotions that the Sales Account Manager wants to work on in Demantra, he changes the promotion status to "Send to Demantra". At this point the promotions are automatically locked for editing in Siebel.
3. An automated scheduled process is running on a pre-set interval (for example, every 30 minutes) that exports all promotions in the "Send to Demantra" status from Siebel and sends them to Demantra for processing. During this process, the following occurs:
 - The status of the promotions are changed from "Send to Demantra" to "Locked by Demantra" in Siebel (the promotions remain locked for editing in Siebel).
 - If the promotion does not already exist in Demantra then it is created in Demantra.
 - If the promotion already exists in Demantra then it is updated in Demantra based on the information in Siebel.
 - The promotion transfer status in Demantra is set to "Unlocked" for each of these

promotions allowing them to be edited in Demantra.

4. Once the transfer is complete, the Sales Account Manager logs into Demantra and runs simulations, optimizations, and lift decompositions on the transferred set of promotions. The Sales Manager can also modify the promotions (for example, change timing, tactics, price points) and add new promotions.
5. When finished, the Sales Account Manager changes the promotion transfer status in Demantra to "Transfer to Siebel". At this point the promotions are automatically locked for editing in Demantra.
6. An automated scheduled process is running on a pre-set interval (for example, every 30 minutes) which exports all promotions where the promotion transfer status is "Transfer to Siebel" from Demantra and sends them to Siebel for processing:
 - The promotion transfer status of the promotions is changed from "Transfer to Siebel" to "Locked" in Demantra (the promotions remain locked for editing in Demantra).
 - If the promotion does not already exist in Siebel then it is created in Siebel.
 - If the promotion already exists in Siebel then it is updated in Siebel based on the information in Demantra.
 - All of these promotions are unlocked for editing in Siebel.
7. The Sales Account Manager reviews the promotions in Siebel and modifies the funding information if needed.
8. The Sales Account Manager finalizes the promotions in Siebel and submits for approval.

Demantra Primary Promotion Planning Application

The following is the business process when Demantra is preferred as the primary promotion planning application:

1. The Sales Account Manager logs into Demantra and creates a new set of promotions.
2. The Sales Account Manager runs simulations, optimizations, and lift decompositions on the set of promotions.
3. When finished, the Sales Account Manager changes the promotion transfer status in Demantra to "Transfer to Siebel". At this point the promotions are automatically locked for editing in Demantra.

4. An automated scheduled process is running on a pre-set interval (for example, every 30 minutes) which exports all promotions where the promotion transfer status is "Transfer to Siebel" from Demantra and sends them to Siebel for processing:
 - The promotion transfer status of the promotions is changed from "Transfer to Siebel" to "Locked" in Demantra (the promotions remain locked for editing in Demantra).
 - If the promotion does not already exist in Siebel then it is created in Siebel.
 - If the promotion already exists in Siebel then it is updated in Siebel based on the information in Demantra.
 - All of these promotions are unlocked for editing in Siebel.
5. The Sales Account Manager reviews the promotions in Siebel and modifies the funding information if needed.
6. The Sales Manager finalizes the promotions in Siebel and submits for approval.

Nightly Extract Transfers

The following interfaces are required for synchronization:

- Customer Account Hierarchy (Siebel to Demantra)
- Product Hierarchy (Siebel to Demantra)
- Sales (Siebel to Demantra)
- Pricing and Cost (Siebel to Demantra)
- Promotional Plan Master (Siebel to Demantra)
- Promotions (Siebel to Demantra)

Weekly Extract Transfers

The following interfaces are required for synchronization:

- Future Baseline Volume Forecast (Demantra to Siebel)
- Future Lift Coefficients (Demantra to Siebel)

Near Real-Time Extract Transfers

The following interfaces are required for near real-time integration:

- Promotions (Siebel to Demantra) – every 30 minutes
- Promotions (Demantra to Siebel) – every 30 minutes

Setup Overview

This chapter provides an overview of the installation checklist and discusses how to install and configure the software components necessary for the integration of Oracle Siebel with Oracle Demantra.

Prerequisites

Verify your software versions before installing and configuring Oracle Siebel to integrate with:

- Oracle Demantra version 7.2.1 or later
- Siebel version 7.5 or later
- Oracle Data Integrator version 10.1.3.4.0 or later
- JAVA JDK version 1.5 or later

Setup Steps

The following steps are required to set up the integration between Oracle Demantra and Oracle Siebel.

Step	Instruction
1	Install Demantra version 7.2.1 or later.
2	Install Siebel version 7.5 or later.
3	Install Oracle Data Integrator version 10.1.3.4.0 or later.
4	Install Oracle Demantra Integration Pack for Siebel CRM Consumer Goods 1.0. See the <i>Oracle Demantra Integration Pack for Siebel CRM Consumer Goods 1.0 Installation Guide</i> for more information.
5	Configure Siebel application. See Siebel Setup, page 20-10 in this guide.
6	Configure Oracle Data Integrator. See ODI Setup, page 20-13 in this guide.

Step	Instruction
7	Configure Demantra. See Demantra Setup, page 20-24 in this guide.
8	Align time units and week start day between Siebel and Demantra. The base time unit is week starting Mondays by default. It is recommended that you use weekly time units and align the week start day to match the associated Siebel implementation.
9	If users will be using both Siebel and Demantra to plan promotions, add new users to both Demantra and Siebel and assign permissions.
10	Load the following extracts: Account Planning Hierarchy (Siebel to Demantra) Product Planning Hierarchy (Siebel to Demantra) Promotional Tactics Master List (Siebel to Demantra) Promotion Status (Siebel to Demantra) Pricing and Cost (Siebel to Demantra) Sales Activity (Siebel to Demantra) Plan Master List (Siebel to Demantra) Promotions (Siebel to Demantra)
11.	Run the Demantra Forecasting Engine to generate the baseline forecast and lift coefficients before running the Demantra-Siebel extracts.
12.	Run the following extracts: Baseline Volume Forecast (Demantra to Siebel) Promotional Lift Coefficients (Demantra to Siebel) Promotions (Demantra to Siebel)
13	Set scheduler to run nightly, weekly and near real-time integration extracts as desired.

Siebel Setup

The following configurations need to be made in Siebel to support integration with Demantra:

- Configuring Campaign_State values
- Configuring State models
- Record locking requirements
- Other data required for integration

Configuring Campaign_State Values

Lists of values (LOVs) are sets of values that populate static pick lists from which a user can select when using Siebel CRM. The values in the LOV are stored as records in the database (S_LST_OF_VAL). They are grouped together using the Type field. For example, the values of LOV that appear in the Status field of the Account Entry Applet all have a Type value of ACCOUNT_STATUS. (List of configurations necessary)

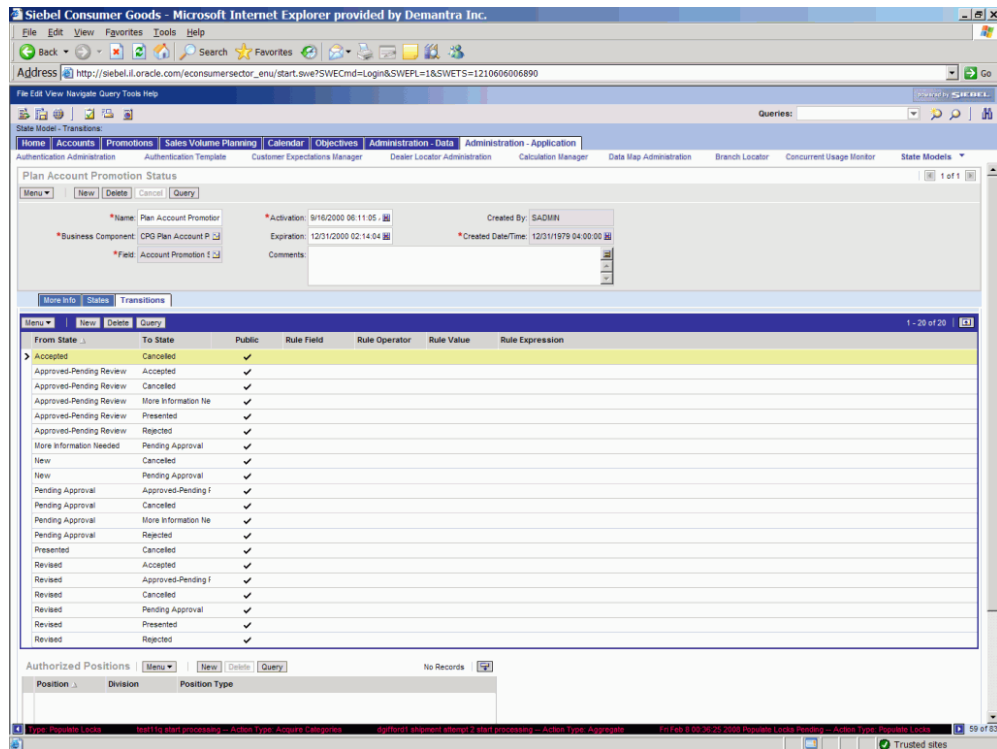
To configure Campaign_State values:

1. Log in to Siebel as an administrator.
2. Click Sitemap (Globe icon on toolbar).
3. Select Administration - Data.
4. Select List of Values. A list of all the value types appears.
5. Select New and create the following value:
 - Type: CAMPAIGN_STATUS
 - Display Value: Locked by Demantra.
6. Select New and create the following value:
 - Type: CAMPAIGN_STATUS
 - Display Value: Send to Demantra
7. From the Menu drop-down list, select Save Record.
8. Click Clear Cache to refresh.

Configuring State Models

If you use a State Model to control the Promotion Status transitions, then you will need to add the new Campaign State values to this State Value. The "Send to Demantra" and "Locked by Demantra" should be added as Transitions to the appropriate State Model (for example, "Plan Account Promotion Status"). To do this, follow the steps below:

1. Log in to Siebel as an administrator.
2. Click Sitemap (the Globe icon on the toolbar).
3. Select Administration - Application.
4. Select State Models.
5. Query for Plan Account Promotion Status.



6. Select the Transitions tab and create new entries for the following statuses that users will use to send promotions to Demantra:

From State	To State
(Existing Status)	Send to Demantra
Send to Demantra	Locked by Demantra

Record Locking

Implementors are responsible for creating a routine that locks records whose

CAMPAIGN_STATUS="Locked by Demantra". This routine must be created for each implementation.

Other Data Required for the Integration

The following reference data must be created in Siebel before running the integration with Demantra:

- Account types
- Product types
- Promotion status
- Promotional tactics

ODI Setup

The following sections describe the setup that needs to be made in ODI to support Demantra-Siebel integration:

- Overview of the ODI Applications
- Understanding Packages and Scenarios
- Configuring the Demantra Database Connection
- Configuring the Demantra Agent
- Configuring ODI Variables

Overview of the ODI Applications

Oracle Data Integrator (ODI) streamlines the high performance movement and transformation of data between heterogeneous systems in batch, real time, synchronous, and asynchronous modes; it dramatically enhances user productivity with an innovative, modularized design approach and built-in connectivity to all major databases, data warehouse appliances, analytic applications and SOA suites.

Oracle Data Integrator includes graphical modules and software agents that enable you to:

- Reverse engineer application models.
- Check data consistency.
- Design, test, operate, and maintain interfaces between applications.

- Check the data flows processed by the interfaces, with error isolation and recycling.
- Identify missing data input.

The following ODI modules are used with the Demantra-Siebel integration:

Agent: The Agent is a Java service that can be placed as listener on a TCP/IP port. This service allows:

- On demand execution of sessions (model reverses, packages, scenarios, interfaces, and so on) from Oracle Data Integrator modules. For this, you must launch a listener agent.
- Execution of scheduled scenarios, in addition to on demand executions. The physical agent contains an optional scheduler that allows scenarios to be launched automatically according to predefined schedules. For this, you must launch a scheduler agent. A physical agent can delegate executions to other physical agents. This process enables load balancing between agents.

Designer: Through the Designer module, you can handle:

- Models: Descriptions of the data and applications structures
- Projects: The developments made with Designer

The Designer module stores this information in a work repository, while using the topology and the security information defined in the master repository.

Operator: Through the Operator module, you can manage:

- Interface executions in the sessions.
- Scenarios in production.

The Operator module stores this information in a work repository, while using the topology defined in the master repository.

Topology Manager: Through the Topology Manager module, you can manage:

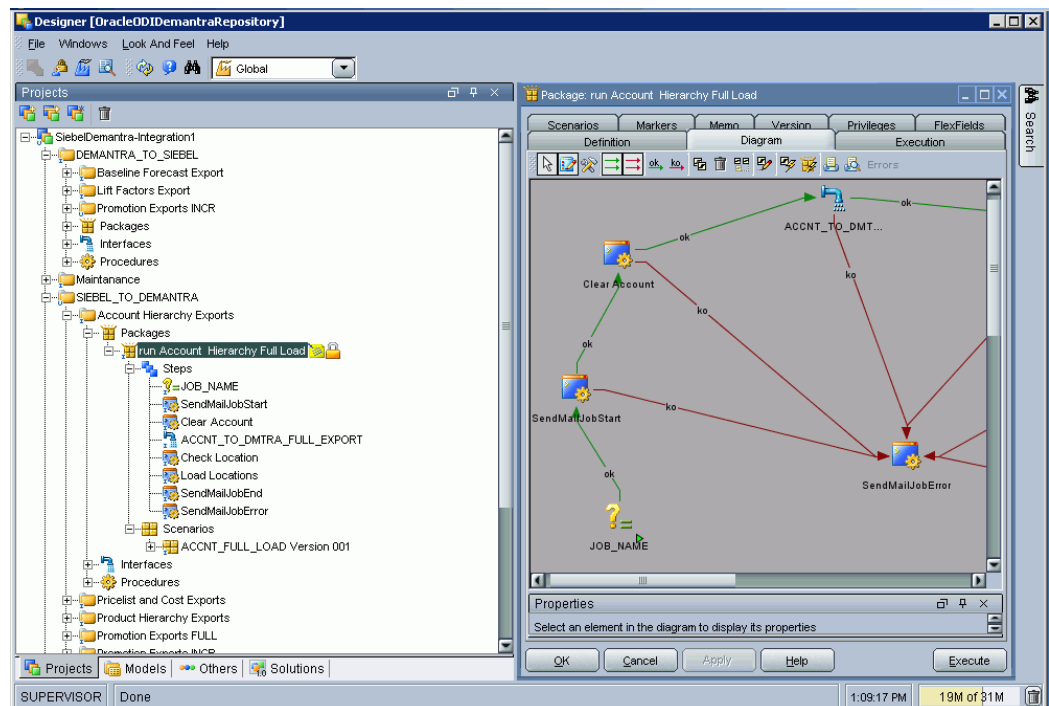
- Topology of your information system
- Technologies and their data types
- Data servers linked to these technologies and the schemas they contain
- Contexts, the languages and the agents
- Repositories

The Topology module stores this information in a master repository. This information can be used by all the other modules.

Understanding Packages and Scenarios

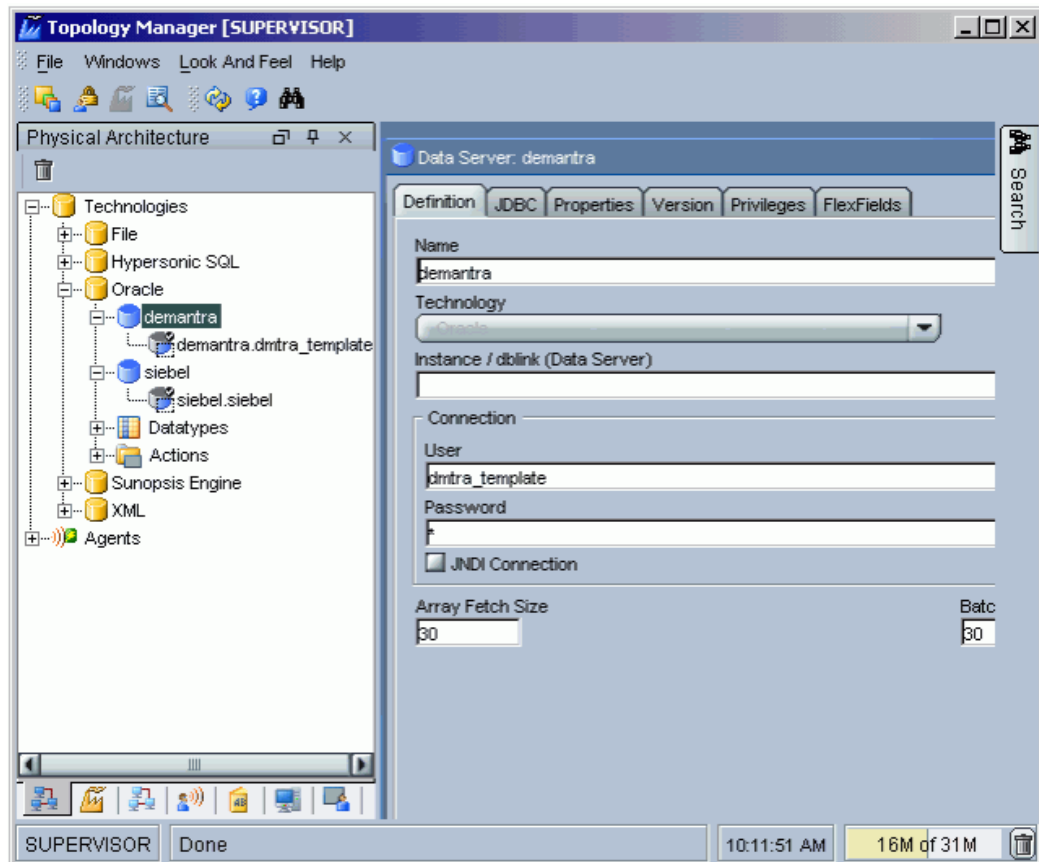
The Demantra-Siebel integration extracts can be viewed and managed by the ODI Designer in two folders DEMANTRA_TO_SIEBEL and SIEBEL_TO_DEMANTRA. Within each folder, the associated extracts are shown as sub-folders such as Account Hierarchy Exports, Pricelist and Cost Exports, Product Hierarchy Exports and so on. Within each sub-folder, a variety of packages may be displayed for that sub-folder. For example, the Pricelist and Cost Exports show two packages: full and incremental.

Lastly, within each package, the ODI Designer displays the steps involved with the extraction and transformation process. The following illustration shows the run Account Hierarchy Full Load scenario steps:



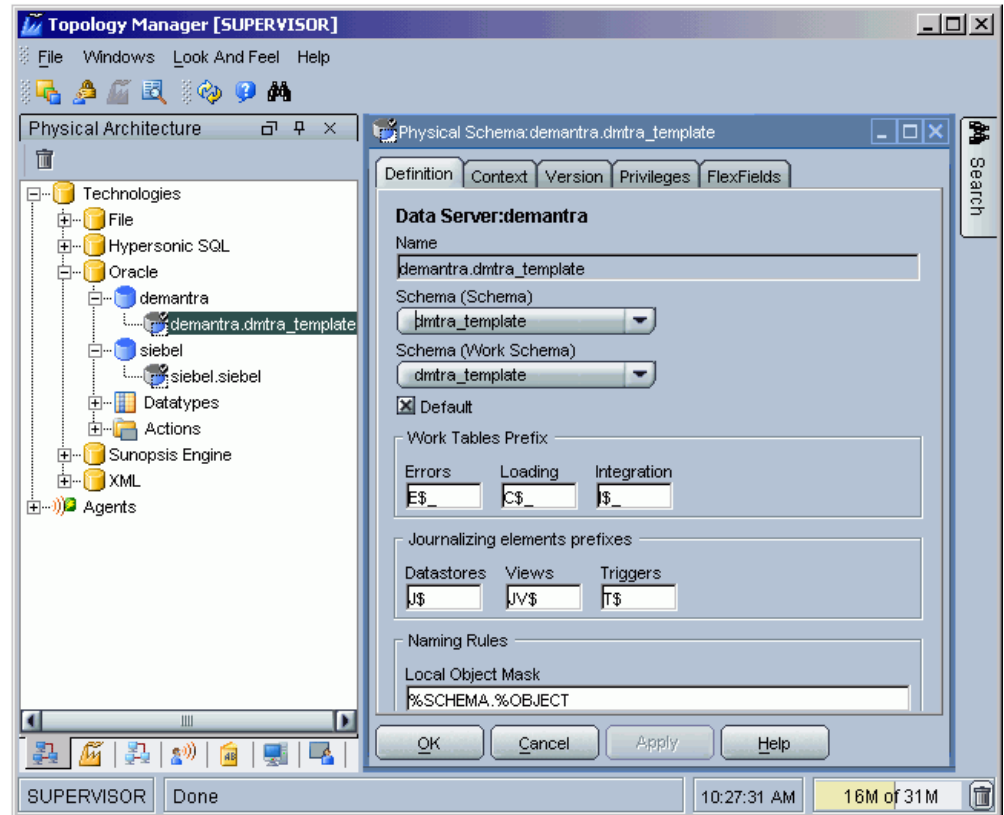
Configuring the Demantra Database Connection

Using the Topology Manager, you can configure your database connection information, Java agent location, as well as your ODI work repository information.

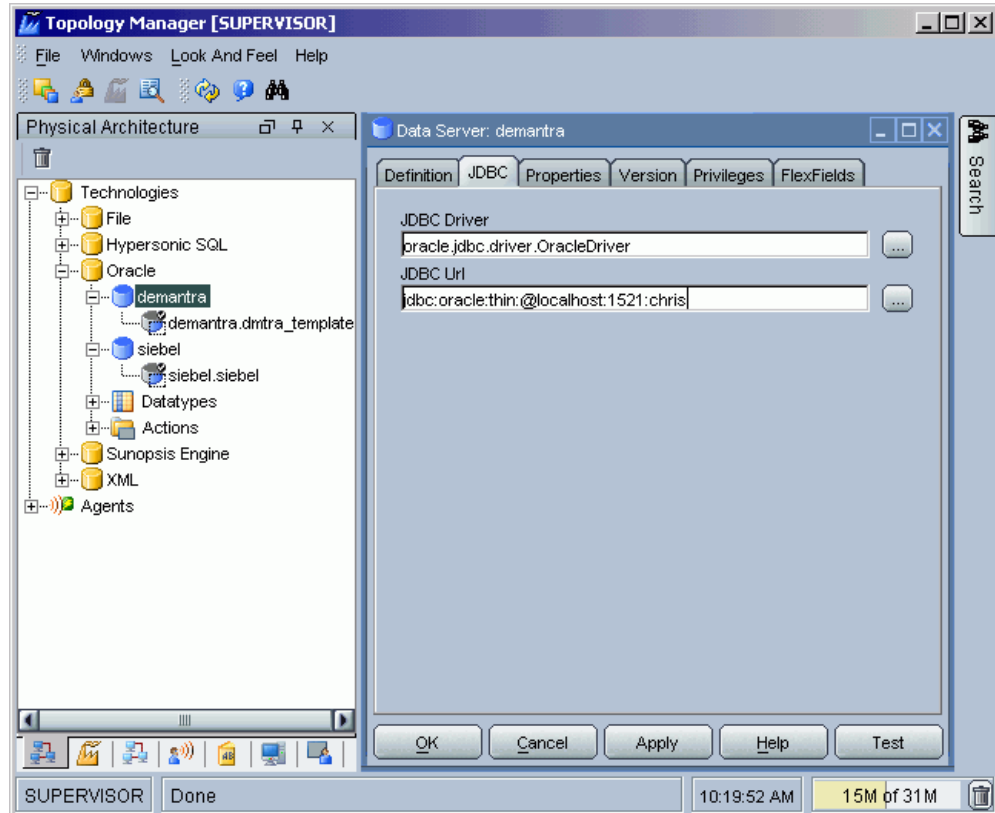


To configure the Demantra database connection:

1. From the ODI Topology Manager, expand the Technologies folder.
2. Expand the Oracle folder.
3. Click on demantra.
4. On the Definition tab, set the following:
 - User: Schema user being used as the repository for Demantra data.
 - Password: Password for the Demantra schema.
5. If you are using a work schema, click the physical schema as shown:



6. On the Definition tab, set the following:
 - Schema (Work Schema): The schema user being used as the work repository for Demantra data.
7. Click Apply.
8. On the JDBC tab, set the following:
 - JDBC URL: The URL allowing you to connect to the data server. The JDBC URL format is `jdbc:oracle:thin:@<host>:<port>:<sid>`



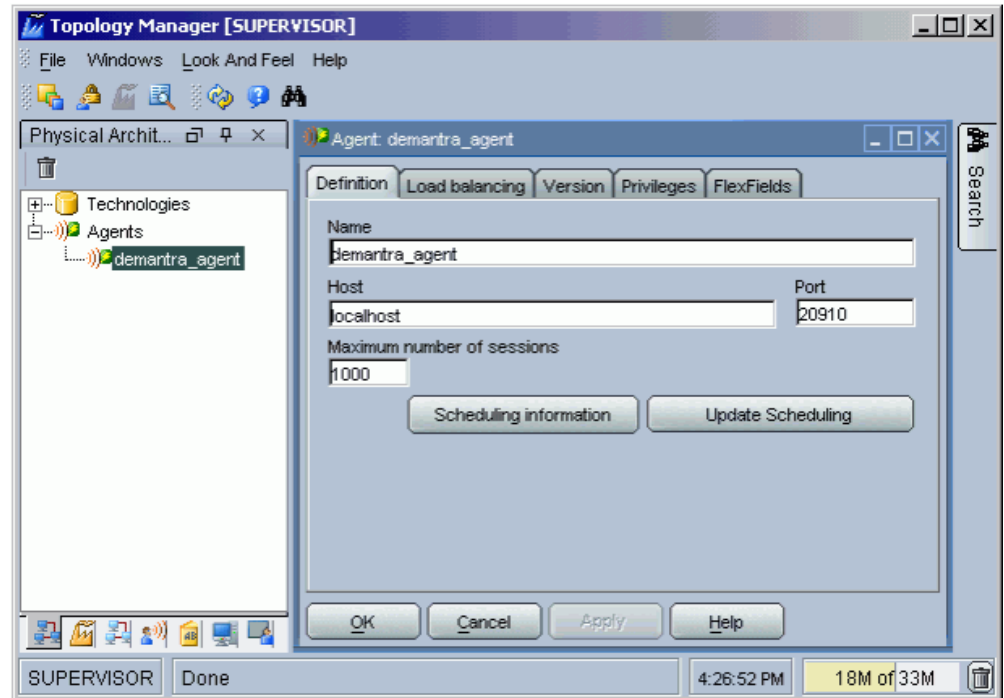
9. Click Test to test the connection.
10. Click Apply.

Configuring the Demantra Agent

The Demantra Agent accepts and controls various job requests from the distributed ODI components. The agent is configured by the ODI Topology Manager. By default, the Demantra-Siebel Integration configures the agent on the Demantra application server machine, listening on port 20910. The agent can be moved to any machine that has ODI installed.

To configure the Demantra Agent:

1. From the ODI Topology Manager, expand the Agents folder.
2. Click the demantra_agent.



3. On the Definition tab, set the following:
 - Host: Network name or IP address of the machine the agent has been launched on.
 - Port: Listening port used by the agent. By default, this port is the 20910.
4. If the Agent is launched, click Test. A successful connection window should appear

Configuring ODI Variables

A number of variables must be set within ODI to support the Demantra-Siebel integration. Fortunately, most of the variables are set when the Oracle Demantra Integration Pack for Siebel CRM Consumer Goods 1.0 is installed. However, you may want to change some of the global ODI variables periodically. The following is a summary of the ODI variables used during Demantra-Siebel integration:

Variable Name	Description
APP_SERVER_URL	Used internally. Do not change.

Variable Name	Description
DEFAULT_SCENARIOS_ID	Scenario ID with which Siebel promotions interfaced to Demantra are associated. Default is 142, the Scenario ID for the pre-seeded "Sandbox" scenario.
DEFAULT_TRANSFER_STAT_CODE	Code of the promotion transfer status with which promotions created in Siebel and interfaced to Demantra are associated. The default is "Default".
DEM_SBL_HOME	The directory where the "Demantra Integration Pack for Siebel CRM Consumer Goods" is installed. This is set by the Demantra Integration Pack for Siebel CRM Consumer Goods Installer. For example: D:\ODI_DEM_SBL_HOME.
DEM_WORKFLOW_PASS	Password of user who maintains the Demantra-Siebel integration workflows. Set by the Demantra Integration Pack for Siebel CRM Consumer Goods installer. For example, ptp.
DEM_WORKFLOW_SCHEMA	Used internally. Do not change.
DEM_WORKFLOW_USER	User who maintains the Demantra-Siebel PTP integration workflows. Set by the Demantra Integration Pack for Siebel CRM Consumer Goods installer. For example, ptp.
DMTRA_PROMO_TO_SBL_STG	Used internally. Do not change.
EIM_BIN_DIR	The location of the Siebel Server executable file (srvrmgr.exe). For example, d:\sia8\client\bin.
EIM_ENTERPRISE	Siebel Enterprise server name. For example: SIEB78.
EIM_EXECUTABLE	The name of the Siebel Server executable file. The default is srvrmgr.exe.

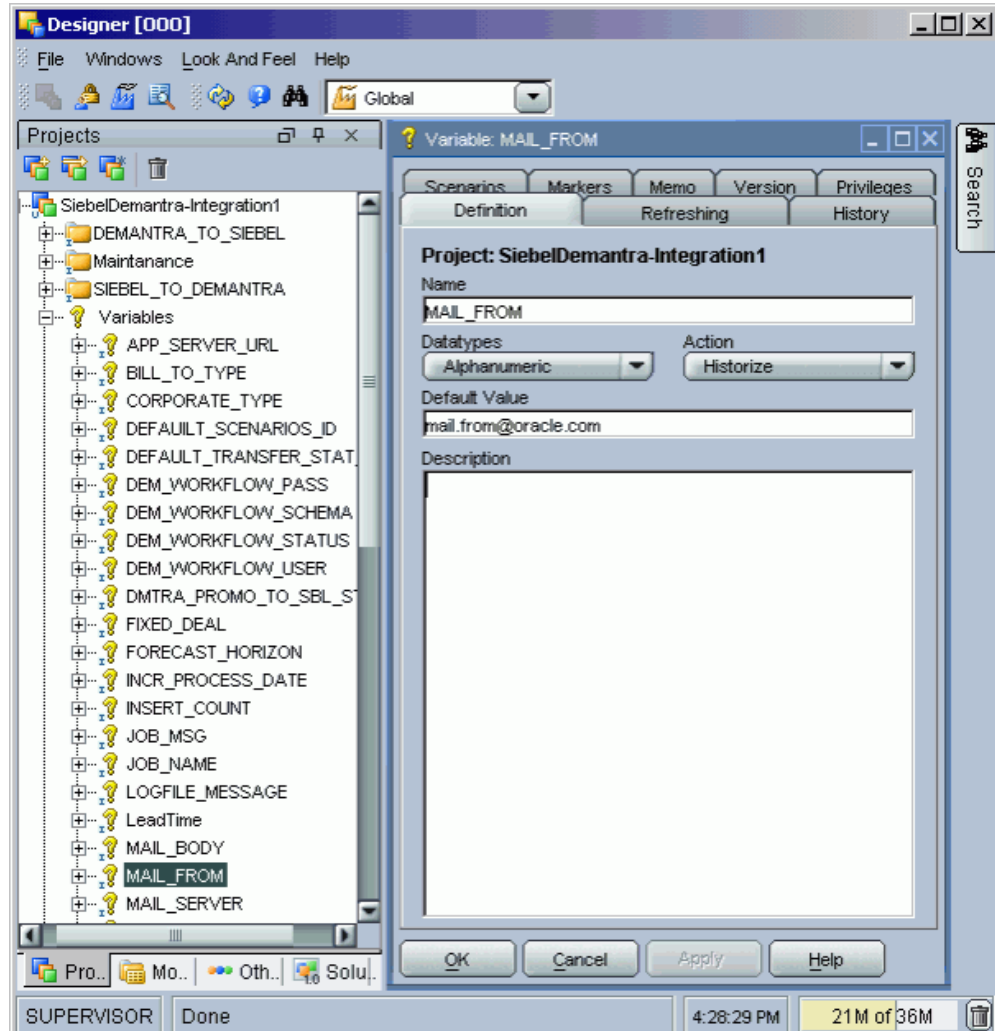
Variable Name	Description
EIM_GATEWAY	Siebel Gateway Name Server name. For example, wa2048.oracle.com.
EIM_IFB	Used internally. Do not change.
EIM_LOG_LEVEL	Sets logging of EIM jobs. For example, Traceflags=1, sqlflags=4, errorflags=3.
EIM_PASSWORD	Siebel password.
EIM_SERVER	Siebel server name.
EIM_SHELL	Command to start the shell. For Windows use cmd.exe /c. For Linux/UNIX, use sh -c.
EIM_USER	Siebel username.
INCR_PROCESS_DATE	The number of days prior to the current system date that the incremental promotion interface from Siebel uses to select promotions. Any promotion created or modified within this window is sent to Demantra. This variable is used by the incremental interfaces for shipments and price/cost. For example, 4.
INSERT_COUNT	Used internally. Do not change.
JOB_NAME	Used internally. Do not change.
LeadTime	Used internally. Do not change.
MAIL_FROM	Email address for notification result queries. Set by the Demantra Integration Pack for Siebel CRM Consumer Goods installer. For example, someone@yahoo.com.
MAIL_SERVER	Host or IP address that manages your email services. Set by the Demantra Integration Pack for Siebel CRM Consumer Goods installer. For example, mail@oracle.com.

Variable Name	Description
MAIL_TO	Email addresses for those notified of integration results. Separate multiple email addresses with a comma. Set by the Demantra Integration Pack for Siebel CRM Consumer Goods installer. For example, someone@yahoo.com.
MAX_FORE_SALES_DATE	Used internally. Do not change.
MIN_FORE_SALES_DATE	Used internally. Do not change.
NA	The value to be used for loading into Demantra when a value is not available from Siebel. The default is "N/A".
PLAN_SUB_TYPE	Variable used to identify plans in the Siebel S_SRC table. For example, PLAN_ACCOUNT.
PROMO_CATG_SUB_TYPE	Variable used to identify promotions by category in S_SRC table. For example, PLAN_ACCOUNT_PROMOTION_CATEGORY.
PROMO_LOCKED_BY_DMTRA	The promotion status used to identify promotions that have been locked by Demantra. Default is "Locked by Demantra". This must match the CAMPAIGN_STATE List of Values entry.
PROMO_PROD_SUB_TYPE	Variable used to identify promotions by product in Siebel S_SRC table. For example, PLAN_ACCOUNT_PROMOTION_PRODUCT.
PROMO_SEND_TO_DMTRA	The promotion status used to identify promotions that have been flagged to be sent to Demantra. The default is "Send to Demantra". This must match the CAMPAIGN_STATE List of Values entry.
PROMO_STATUS_TYPE	Promotion status type used for Demantra-Siebel integration. For example, CAMPAIGN_STATUS.

Variable Name	Description
PROMO_SUB_TYPE	Variable used to identify promotions in Siebel S_SRC table. For example, PLAN_ACCOUNT_PROMOTION.
TIME_PERIOD	Time bucket used for planning purposes. Needs to be aligned in both Siebel and Demantra. For example, Week.
TYPE_DIVISION	Siebel account type used for division level. For example, Division.
TYPE_PG	Siebel product type used for promoted product group level. For example, Promoted Group.
TYPE_SHIPTO	Siebel account type used for ship to level. For example, Ship To.
V_ACCOUNT	Used internally. Do not change.
V_PRODUCT	Used internally. Do not change.
LOGFILE	Name of Demantra-Siebel integration log file. This file is located on the ODI server. For example, Siebel_demantra_integration.log.
LOGPATH	Path where Demantra-Siebel integration log file located. This file is located on the ODI server. For example, Integration\logs.

To configure the ODI variables:

1. From the ODI Designer, view the Projects.
2. Expand the Variables group.



3. Select a variable from the group and change the definition displayed on the right.
4. Click Apply.

Demantra Setup

This is typically a one-time setup that can be done with a database tool such as SqlPlus, Benthic, or Toad.

The following shows the account field name mappings between Siebel and Demantra. The Demantra Reference refers to the Level in the Site Hierarchy. The Siebel Reference refers to the Account Type.

Map Code	Demantra Reference	Siebel Reference
1	Site	Ship To
2	Bill To	Bill To
3	Retailer	Division
4	Corporate	Corporate

The following shows the product field name mappings between Siebel and Demantra. The Demantra Reference refers to the Level in the Item Hierarchy. The Siebel Reference refers to the Product Type or Catalog Hierarchy.

Map Code	Demantra Reference	Siebel Reference
1	Item	Product
2	Promoted Group	Promoted Group
3	Brand	Brand
4	Product SubGroup	Product Sub Category
5	Product Group	Product Category
6	Product Line	Catalog
7	Product Status	Product Status

The PROMOTION_TYPES table is shown below with sample data. Promotional Tactics that are transferred from an external source must have their TACTIC ID and DISCOUNT fields manually configured. The TACTIC ID associates a pre-seeded set of Tactic Groups with the detailed Promotional Tactics, by referencing the TACTICS_GROUPS table shown further down. The DISCOUNT field contains the percentage discount of the promotion and should correspond to the label in the PROMOTION TYPE CODE field.

PROMOTION TYPE ID	PROMOTION TYPE CODE	PROMOTION TYPE DESC	LAST UPDATE DATE	TACTIC ID	DISCOUNT
0	Default	Default Promotion Type	---	1	0
1	None	No Promotion Type	3/12/2008 13:19	1	0
208	88-27XAW	FEATURE 5%	2/19/2008 21:09	3	5
204	88-27XAX	FEATURE 10%	2/19/2008 21:09	3	10
205	88-27XAY	FEATURE 15%	2/19/2008 21:09	3	15
206	88-27XAZ	FEATURE 20%	2/19/2008 21:09	3	20
207	88-27XB0	FEATURE 25%	2/19/2008 21:09	3	25
103	88-27XBD	DISPLAY 10%	2/19/2008 21:09	2	10
104	88-27XBE	DISPLAY 15%	2/19/2008 21:09	2	15
307	88-27XB4	F\&\D 5%	2/19/2008 21:09	4	5
302	88-27XB5	F\&\D 10%	2/19/2008 21:09	4	10
303	88-27XB6	F\&\D 15%	2/19/2008 21:09	4	15
304	88-27XB7	F\&\D 20%	2/19/2008 21:09	4	20

PROMOTION TYPE ID	PROMOTION TYPE CODE	PROMOTION TYPE DESC	LAST UPDATE DATE	TACTIC ID	DISCOUNT
305	88-27XB8	F\&\D 25%	2/19/2008 21:09	4	25
306	88-27XB9	F\&\D 35%	2/19/2008 21:09	4	35
405	0V-FHYD	TPR 5%	2/19/2008 21:09	5	5
402	0V-FHYF	TPR 10%	2/19/2008 21:09	5	10
403	0V-FHYH	TPR 15%	2/19/2008 21:09	5	15
404	88-27XAT	TPR 20%	2/19/2008 21:09	5	20

The TACTICS_GROUPS table is shown below. It provides a pre-seeded list of Promotional Tactics Groups. Tactics groups 6-12 are placeholder tactics which can be used if additional tactic groups are desired beyond the preconfigured set.

TACTIC ID	TACTIC DESC
0	Default Tactics
1	No Tactics Defined
2	Display Only
3	Feature Only
4	Feature \&\ Display
5	TPR
6	Tactic 6

TACTIC ID	TACTIC DESC
7	Tactic 7
8	Tactic 8
9	Tactic 9
10	Tactic 10
11	Tactic 11
12	Tactic 12

The Analytical Engine is configured to process promotions with an associated positive discount. The default discount for a promotion should be entered in the DISCOUNT field of PROMOTION_TYPE table. Promotion specific discount overrides can be done by entering both Shelf Price and Promoted Price for the promotion. A promotion without default or specific positive discount will be ignored by the engine.

Main Batch File Configurations

All the Demantra-Siebel integration extracts can be run from the Windows command shell. The batch routines are installed by the Oracle Demantra Integration Pack for Siebel CRM Consumer Goods 1.0 Installation process. By default, these batch routines are located in the `<root>\ODI_DEM_SBL_Home\integration` directory.

There are two key files that need to be customized for your environment:

- `Runscenario.bat`: Defines the variables used for running the ODI application and scenarios. All ODI package batch routines reference this batch routine.
- `Runsvrmgr.bat`: Defines the variables used for launching the Siebel application and running the Enterprise Integration Manager. All the `EIM_<package name>` batch files reference this batch routine.

These batch routines must be defined before the integration extracts can be run successfully from the command shell.

Within the `<root>\ODI_DEM_SBL_Home\integration` directory, the extract batch routines are organized into two directories specifying the direction of the integration:

Directory	Extract Subdirectories
Demantra_Siebel	Baseline_Load
	Liftfactor_Load
	Promotion_Load
Siebel_Demantra	AccountLoad
	PricelistLoad
	ProductLoad
	PromotionDataLoad
	<ul style="list-style-type: none"> • Full • Incremental
	PromotionLevelLoad
	PromotionLock
	SalesHistoryLoad

Within each extract directory is a main batch file suffixed with _MAIN. This batch file has been designed to run all the extract packages in the correct order.

At the start and end of each extraction package procedure, email messages are sent to the system administrator. This information is also available in the log file and from within the ODI Operator module.

Configuring odiparams.bat

In order to run the integration batch files, you will need to edit the odiparams.bat file to add the following variable setting code. This file is located in the <ODI_HOME>\bin directory.

The following variables need to be set:

Variable Name	Description
ODI_SECU_DRIVER	Java Oracle Driver. For example, oracle.jdbc.driver.OracleDriver.

Variable Name	Description
ODI_SECU_URL	JDBC URL set in the following format: jdbc:oracle:thin:@<host name>:<port>:<SID> For example, jdbc:oracle:thin@localhost:1521:chris
ODI_SECU_USER	ODI master schema. For example: ODI_DEM_SBL.
ODI_SECU_ENCODED_PASS	Encoded password for the ODI master schema. You can find the encoded password by running the following: <ODI_HOME> \bin\agent.bat encode <password>. For example, g.yHxWAOP2QcswgMddNepx9ot.
ODI_SECU_WORK_REP	Work repository schema. If no work repository is being used, it is the same as the master schema. For example, ODI_DEM_SBL.
ODI_USER	Encoded password for the ODI master repository. SUNOPSIS is the default password for the ODI master repository. You can find the encoded password by running the following: <ODI_HOME>\bin\agent.bat encode <password>. For example, 1234.

Configuring the runscenario.bat

From the <root>\ODI_DEM_SBL_Home\integration directory, configure the following variable:

Variable Name	Description
ODI_BIN_DIR	Directory where the ODI applications reside.

Configuring the runsvrmgr.bat

From the <root>\ODI_DEM_SBL_Home\integration directory, configure the following variables:

Variable Name	Description
EIM_G	Siebel host location. For example, demsb15.us.oracle.com.
EIM_E	Siebel Enterprise server name. For example, Sieb78.
EIM_S	Siebel server name. For example, demsbl2.
EIM_U	Username to access the Siebel Enterprise Integration Manager. Setting this variable will set the username for all the IFB files.
EIM_P	Password to access the Siebel Enterprise Integration Manager. Setting this variable will set the password for all the IFB files.

Running Integration

Once the Demantra-Siebel integration components have been configured and tested, the ODI scenarios can be run and scheduled. The process is as follows:

1. Start the Agent.
2. Run the scenarios using scheduled batch routines. Alternatively, scenarios can be scheduled using the ODI Operator.
3. View the progress of the scenarios, if desired.
4. Review the log file results if errors.

Launching the Listener Agent

The Agent is a Java TCP/IP service that can be placed as listener on a TCP/IP port. This agent works as follows:

- When it starts, it runs as a listener on its port.
- When an execution query arrives on the agent, it executes the tasks associated with

the query then returns as a listener on the port.

Note: A listener agent has no scheduled executions, and does not handle scheduled scenarios.

To launch the listener agent:

From the <ODI_HOME>\bin directory, run the agent.bat file (or agent.sh in UNIX). The agent is then launched as listener.

See *Oracle Data Integrator User's Guide* for more information.

Stopping an Agent

You can stop a listener, scheduler or web agent that is listening on a TCP/IP port through the agentstop command.

To stop an agent:

From the <ODI_HOME>\bin\ directory, run the agentstop.bat file (or agentstop.sh in UNIX).

The listening agent is stopped.

Important: For security reasons, it is only possible to stop an agent from the same machine as where the agent's process was started. It is not possible to stop a remote agent.

See *Oracle Data Integrator User's Guide* for more information.

Launching a Scheduler Agent

The Agent is a Java TCP/IP service that can be used to execute scenarios using predefined schedules or on demand. This agent works as follows:

- When it starts, it connects to the work repository to get its scheduled executions (all the scenario schedules planned for it), then runs as listener on its port.
- When an execution query arrives on this agent, it executes the tasks associated with the query.
- When a scenario must be launched in accordance to the scheduled executions in memory, the agent executes it, then returns as listener on the port.
- The agent's scheduled executions can be updated at any time by clicking the scheduling update button in the physical agent window.

Note: A scheduler agent has all the functionality of a listener agent.

To launch a scheduler agent:

From the <ODI_HOME>\bin\ directory, run the agentscheduler.bat file (or agentscheduler.sh in UNIX).

The agent starts up and goes to retrieve its scheduled executions on the work repository.

See *Oracle Data Integrator User's Guide* for more information.

Displaying Scheduling Information

The scheduling information enables you to visualize the agents' scheduled tasks.

Important: The scheduling information is retrieved from the Agent's schedule. The Agent must be started and its schedule refreshed in order to display accurate schedule information.

To display the scheduling information from the Operator:

1. Open the ODI Operator
2. Click Scheduling Information.

Status of Scenarios

When scenarios are being run, you can view their status in the ODI Operator at the session, step or task level.

There are six possible status values:

- Done: The session, step or task was executed successfully.
- Error: The session, step or task has terminated due to an error.
- Running: The session, step or task is being executed.
- Waiting: The session, step or task is waiting to be executed.
- Warning (Tasks only): The task has terminated in error, but because errors are allowed on this task, this did not stop the session.
- Queued (Sessions only) : The session is waiting for an agent to be available for its execution.

When terminated, a session takes the status of the last executed step (Done or Error).
When terminated, the step, takes the status of the last executed task (Except if the task

returned a Warning. In this case, the step takes the status Done).

Handling Errors

To analyze an error:

1. In the ODI Operator tree view, identify the session, the step and the task in error.
2. Double-click the Error marker on the task. The task window opens:
 - In the Execution tab, the return code and message give the error that stopped the session.
 - In the Definition tab, the failed order is displayed.

Logging

An email message is sent to the system administrator confirming the success of each scenario run. When there are errors, log files are available for review. The Demantra-Siebel integration log files are located in the `<root>\ODI_DEM_SBL_Home\integration\logs` directory.

Siebel to Demantra ODI Packages and Batch Files

This section provides information about the Siebel to Demantra extract packages and associated batch files. They include:

- Account Planning Hierarchy
- Plan Master List
- Pricing and Cost
- Product Planning Hierarchy
- Promotions
- Promotion Status
- Promotional Tactics Master List
- Sales Activity

Account Hierarchy Planning

Siebel is the system of record for customer and account information. The Account Planning Hierarchy as defined in Siebel is sent to Demantra to populate the Customer

Location Hierarchy in Demantra. This data is used in Demantra for planning promotions, creating baselines, and aggregating data.

Transformation Details

Customer types can differ for any given implementation based on the client requirements. Therefore, the SIEBEL_ACCOUNT_MAPPING table is designed to map the different Siebel customer types to Demantra entities. The table is preseeded as follows:

Required Data Element	Demantra Reference	Siebel Reference
Ship to Location	Site	Ship-to
Billed Location	Bill-to	Bill-to
Aggregate Customer	Retailer	Division
Corporate	Corporate	Corporate

See Demantra Setup, page 20-24 for more information on configuring the SIEBEL_ACCOUNT_MAPPING table.

Mapping

Demantra Field	Type	Siebel Source Table	Siebel Source Field	Comments
Site Number	VARCHAR2(30)	S_ORG_EXT	OU_NUM	Data at the Ship – To Level
Site Name	VARCHAR2(100)	S_ORG_EXT	NAME	Data at the Ship – To Level
Site Description	VARCHAR2(255)	S_ORG_EXT	DESC_TEXT	Data at the Ship – To Level
Status	VARCHAR2(30)	S_ORG_EXT	CUST_STAT_CD	Data at the Ship – To Level
Bill to Number	VARCHAR2(30)	S_ORG_EXT	OU_NUM	Data at the Sold – To Level

Demantra Field	Type	Siebel Source Table	Siebel Source Field	Comments
Bill to Name	VARCHAR2(100))	S_ORG_EXT	NAME	Data at the Sold – To Level
Bill to Description	VARCHAR2(255))	S_ORG_EXT	DESC_TEXT	Data at the Sold – To Level
Retailer Number	VARCHAR2(30)	S_ORG_EXT	OU_NUM	Data at the Division level
Retailer Name	VARCHAR2(100))	S_ORG_EXT	NAME	Data at the Division level
Retailer Description	VARCHAR2(255))	S_ORG_EXT	DESC_TEXT	Data at the Division level
Corporate Number	VARCHAR2(30)	S_ORG_EXT	OU_NUM	Data at the Corporation level
Corporate Name	VARCHAR2(100))	S_ORG_EXT	NAME	Data at the Corporation level
Corporate Description	VARCHAR2(255))	S_ORG_EXT	DESC_TEXT	Data at the Corporation level
City	VARCHAR2(50)	S_ADDR_PER	CITY	Data at the Ship – To : Address Level
State	VARCHAR2(10)	S_ADDR_PER	STATE	Data at the Ship – To : Address Level
Country	VARCHAR2(30)	S_ADDR_PER	COUNTRY	Data at the Ship – To : Address Level
Zip Code	VARCHAR2(30)	S_ADDR_PER	ZIPCODE	Data at the Ship – To : Address Level

Business Rules

1. It is recommended that this process be scheduled on a regular interval (daily).
2. Customer Account Hierarchy is always a full load.
3. Customer and parent hierarchy details for customers with the site level type in Demantra are transferred from Siebel to Demantra. In the example above, only customers with the type Ship-to are moved from Siebel to Demantra.
4. When the Name or Description fields are not populated in Siebel, they are populated with the Number or Name fields of the corresponding levels.
5. If any of the corresponding levels are not available in Siebel, the administrator or implementation consultant needs to populate the parent hierarchy to the same hierarchy. For example, let us assume that Division doesn't exist for a particular Siebel Implementation. The SIEBEL_ACCOUNT_MAPPING table needs to be mapped as follows:

Required Data Element	Demantra Reference	Siebel Reference
Ship to Location	Site	Ship-to
Billed Location	Bill-to	Bill-to
Aggregate Customer	Retailer	Corporate
Corporate	Corporate	Corporate

ODI Package and Batch File Details

ODI Package Name: run Account Hierarchy Full Load

ODI Scenario: ACCNT_FULL_LOAD

Recommended Scheduling Frequency: Daily

Directory of batch files: <root>

\ODI_DEM_SBL_Home\integration\Siebel-Demantra\AccountLoad

Package Name	Description of Functionality
ACCNT_FULL_LOAD.cmd	Extracts the Customer Account Hierarchy data from Siebel and transfers to the Demantra staging table

Plan Master List

Siebel is the system of record for plan information. Demantra needs Siebel plan information so that promotions created in Demantra can be assigned to specific plans. Typically, a year's worth of promotions for a particular category or brand are grouped into plans for ease of navigation and reporting.

Transformation Details

In Siebel, plans are specific to the accounts for which the promotions have been planned. Currently in Demantra, plans are associated with promotions irrespective of the account at which the promotion is created.

Mapping

Demantra Field	Type	Siebel Source Table	Siebel Source Field	Comments
Item Number	VARCHAR2(50)	S_PROD_INT	PART_NUM	Product / SKU / Item Number
Item Name	VARCHAR2(100)	S_PROD_INT	NAME	Product / SKU / Item Name
Item Description	VARCHAR2(255)	S_PROD_INT	DESC_TEXT	Product / SKU / Item Desc Empty if no data
Promoted Group Number	VARCHAR2(50)	S_PROD_INT	PART_NUM	Empty if not implemented in a project
Promoted Group Name	VARCHAR2(100)	S_PROD_INT	NAME	Empty if not implemented in a project

Demantra Field	Type	Siebel Source Table	Siebel Source Field	Comments
Promoted Group Description	VARCHAR2(255))	S_PROD_INT	DESC_TEXT	Empty if no data or not implemented in a project
Product Sub Category #	VARCHAR2(100))	S_CTLG_CAT	NAME	Category Number will be substituted - if no data or not implemented in a project
Product Sub Category Name	VARCHAR2(100))	S_CTLG_CAT	DISPLAY_NAME	Category Name will be substituted - if no data or not implemented in a project
Product Sub Category Description	VARCHAR2(250))	S_CTLG_CAT	DESC_TEXT	Category Desc will be substituted - if no data or not implemented in a project
Product Category #	VARCHAR2(100))	S_CTLG_CAT	NAME	---
Product Category Name	VARCHAR2(100))	S_CTLG_CAT	DISPLAY_NAME	---
Product Category Description	VARCHAR2(250))	S_CTLG_CAT	DESC_TEXT	Empty if no data or not implemented in a project
Brand	VARCHAR2(100))	S_CTLG_CAT	NAME	Empty if no data or not implemented in a project
Catalog Number	VARCHAR2(75)	S_CTLG	NAME	---

Demantra Field	Type	Siebel Source Table	Siebel Source Field	Comments
Catalog Name	VARCHAR2(250)	S_CTLG	DESC_TEXT	---
Primary Unit of Measure	VARCHAR2(30)	S_PROD_INT	UOM_CD	---
Item Status	VARCHAR2(30)	S_PROD_INT	STATUS_CD	---

Business Rules

1. It is recommended that this process be scheduled on a regular basis (daily).
2. Plan information is always a full load.
3. Plan information needs to be loaded after Account Hierarchy Planning.

ODI Package and Batch File Details

ODI Package Name: run Promotion Plan Exp

ODI Scenario: PROMO_PLAN_EXPORT

Recommended Scheduling Frequency: Full export daily

Directory of batch files: <root>

\ODI_DEM_SBL_Home\integration\Siebel-Demantra\PromotionLeveLoad\

Package Name	Description of Functionality
PROMO_PLAN_EXPORT.cmd	<ul style="list-style-type: none"> • Extracts all promotion plans from Siebel • Launches the Demantra web server • Starts the IMPORT: SIEBEL_PLAN workflow in Demantra to populate Demantra's data tables.

Pricing and Cost

Siebel is the system of record for manufacturer list price and manufacturer cost of goods sold. The manufacturer list price and manufacturer cost as stored in Siebel is sent to

Demantra to populate cost and price information in Demantra. List price and cost information is used in Demantra for promotion evaluation such as promotion profitability.

Transformation Details

In Siebel, the price and cost information are stored as price lists. Even though Siebel allows a product to have more than one price for a given account at any given period, it is uncommon to have more than one product pricelist associated with an account for a specific time period.

In Demantra, prices are stored in the transaction table at the lowest level (site, item and week). Therefore, Siebel data based on effective dates is converted into multiple records in Demantra. For example, Siebel records product pricelist information for a date range as follows:

Product	Ship-To	Start Date	End Date	List Price	Manufacturer Cost
Fizz Cola 20 oz	ABCMart	1-1-2008	1-31-2008	1.00	0.85

This record is translated as follows in Demantra, assuming a weekly system with Sunday start date:

Product	Ship-To	Sales Date	List Price	Manufacturer Cost
Fizz Cola 20 oz	ABCMart	1-6-2008	1.00	0.85
Fizz Cola 20 oz	ABCMart	1-13-2008	1.00	0.85
Fizz Cola 20 oz	ABCMart	1-20-2008	1.00	0.85
Fizz Cola 20 oz	ABCMart	1-27-2008	1.00	0.85

Mapping

Demantra Field	Type	Siebel Source Table	Siebel Source Field
Product Name	VARCHAR2(100)	S_PROD_INT	PROD_NAME

Demantra Field	Type	Siebel Source Table	Siebel Source Field
Product Number	VARCHAR2(50)	S_PROD_INT	PART_NUM
Site Name	VARCHAR2(30)	S_ORG_EXT	NAME
Site Number	VARCHAR2(100)	S_ORG_EXT	OU_NUM
Shipment Volume	VARCHAR2(22)	S_PROD_SHIPMENT	ACTL_CASES
Unit of Measure	VARCHAR2(30)	S_PROD_SHIPMENT	PREF_UOM_CD
Start Date	DTYPE_DATE	S_PERIOD	START_DT
End Date	DTYPE_DATE	S_PERIOD	END_DT

Business Rules

1. It is recommended to schedule this package on a regular interval (daily).
2. Account, Product and Sales extracts need to be loaded before the Pricing and Cost extract.
3. Price and cost information are always loaded for the future 18 months. Each subsequent load overwrites previously loaded data
4. After the translation, if there are multiple records for a given product, ship-to customer and sales date, the record is consolidated into one record by taking the minimum of list price and maximum of manufacturer cost. The following records are consolidated into one record as shown:

Product	Ship-To	Sales Date	List Price	Manufacturer Cost
Fizz Cola 20 oz	ABCMart	1-6-2008	1.00	0.95
Fizz Cola 20 oz	ABCMart	1-6-2008	1.00	0.85
Fizz Cola 20 oz	ABCMart	1-6-2008	1.15	0.95

Consolidated Record:

Product	Ship-To	Sales Date	List Price	Manufacturer Cost
Fizz Cola 20 oz	ABCMart	1-6-2008	1.00	0.95

ODI Package and Batch File Details

ODI Package Names:

- run PriclistCost Incr Export
- run PriclistCost Full Export

ODI Scenarios:

- PRICELISTCOST_INCR_EXPORT
- PRICELISTCOST_FULL_EXPORT

Recommended Scheduling Frequency: Full export during setup; daily incremental export thereafter. Account, Product and Sales extracts need to be loaded before the Pricing and Cost extract.

Directory of batch files: <root>

\ODI_DEM_SBL_Home\integration\Siebel-Demantra\PricelistLoad

Package Name	Description of Functionality
PRICLISTCOST_FULL_EXPORT.cmd	<ul style="list-style-type: none"> • Extracts the full manufacturer list Price and manufacturer cost of goods sold data from Siebel. • Launches the Demantra web server. • Starts the IMPORT: LIST_PRICE_COST workflow in Demantra to populate Demantra's data tables.

Package Name	Description of Functionality
PRICLISTCOST_FULL_EXPORT.cmd	<ul style="list-style-type: none"> Extracts the full manufacturer list Price and manufacturer cost of goods sold data from Siebel. Launches the Demantra web server. Starts the IMPORT: LIST_PRICE_COST workflow in Demantra to populate Demantra's data tables.
PRICLISTCOST_FULL_EXPORT.cmd	<ul style="list-style-type: none"> Extracts the full Manufacturer List Price and Manufacturer Cost of Goods Sold data from Siebel Launches the Demantra web server Starts the IMPORT: LIST_PRICE_COST workflow in Demantra to populate Demantra's data tables.
PRICLISTCOST_INCR_EXPORT.cmd	<ul style="list-style-type: none"> Extracts any new or changed Siebel Manufacturer List Price and Manufacturer Cost of Goods Sold data since the last extraction. Launches the Demantra web server Starts the IMPORT: LIST_PRICE_COST workflow

Product Planning Hierarchy

Siebel is the system of record for product information. The product hierarchy as defined in Siebel is sent to Demantra to populate the product item hierarchy in Demantra. This data is used in Demantra for planning promotions, creating baselines, and aggregating data.

Transformation Details

The product types may vary for any given implementation based on the client requirements. A SIEBEL_PRODUCT_MAPPING table is maintained in Demantra which

maps all the product types necessary for integration. It is preseeded as follows:

Required Data Element	Demantra Reference	Siebel Reference
Product – SKU	Item	Product
Promoted Product Group	Promoted Group	Promoted Group
Brand	Brand	Brand
Product Sub Category	Product SubGroup	Product Sub Category
Catalog	Product Line	Catalog
Product Status	Product Status	Product Status

See Demantra Setup, page 20-24 for more information about the SIEBEL_PRODUCT_MAPPING table.

Mapping

Demantra Field	Type	Siebel Source Table	Siebel Source Field	Comments
Item Number	VARCHAR2(50)	S_PROD_INT	PART_NUM	Product / SKU / Item Number
Item Name	VARCHAR2(100)	S_PROD_INT	NAME	Product / SKU / Item Name
Item Description	VARCHAR2(255)	S_PROD_INT	DESC_TEXT	Product / SKU / Item Desc Empty if no data
Promoted Group Number	VARCHAR2(50)	S_PROD_INT	PART_NUM	Empty if not implemented in a project
Promoted Group Name	VARCHAR2(100)	S_PROD_INT	NAME	Empty if not implemented in a project

Demantra Field	Type	Siebel Source Table	Siebel Source Field	Comments
Promoted Group Description	VARCHAR2(255))	S_PROD_INT	DESC_TEXT	Empty if no data or not implemented in a project
Product Sub Category #	VARCHAR2(100))	S_CTLG_CAT	NAME	Category Number will be substituted - if no data or not implemented in a project
Product Sub Category Name	VARCHAR2(100))	S_CTLG_CAT	DISPLAY_NAME	Category Name will be substituted - if no data or not implemented in a project
Product Sub Category Description	VARCHAR2(250))	S_CTLG_CAT	DESC_TEXT	Category Desc will be substituted - if no data or not implemented in a project
Product Category #	VARCHAR2(100))	S_CTLG_CAT	NAME	---
Product Category Name	VARCHAR2(100))	S_CTLG_CAT	DISPLAY_NAME	---
Product Category Description	VARCHAR2(250))	S_CTLG_CAT	DESC_TEXT	Empty if no data or not implemented in a project
Brand	VARCHAR2(100))	S_CTLG_CAT	NAME	Empty if no data or not implemented in a project
Catalog Number	VARCHAR2(75)	S_CTLG	NAME	---

Demantra Field	Type	Siebel Source Table	Siebel Source Field	Comments
Catalog Name	VARCHAR2(250)	S_CTLG	DESC_TEXT	---
Primary Unit of Measure	VARCHAR2(30)	S_PROD_INT	UOM_CD	---
Item Status	VARCHAR2(30)	S_PROD_INT	STATUS_CD	---

Business Rules

1. It is recommended that this package be scheduled on a regular interval (daily).
2. Product Hierarchy is always a full load.
3. When the Name or Description fields are not populated in Siebel, they are populated with the Number or Name fields of the corresponding levels.
4. Only the products with the type corresponding to the item level in Demantra and associated parent hierarchy elements are selected to be sent to Demantra. In the example above, only products with the type Product will be moved from Siebel to Demantra. But each product record will have the hierarchy details based on the reference table.
5. If any of the corresponding levels are not available in Siebel, then during the mapping process, the administrator or implementation consultant needs to populate the parent hierarchy to the same hierarchy.

ODI Package and Batch File Details

ODI Package Name: run Product Hierarchy Full Load

ODI Scenario: PROD_FULL_LOAD

Recommended Scheduling Frequency: Daily

Directory of batch files: <root>

\ODI_DEM_SBL_Home\integration\Siebel-Demantra\ProductLoad

Package Name	Description of Functionality
PROD_FULL_LOAD.cmd	Extracts the product data from Siebel and transfers it to the Demantra staging tables.

Promotion Status

Promotional status defines the status of the Promotion. The preconfigured values in Demantra are:

- a. Unplanned
- b. Planned
- c. Approved
- d. Committed
- e. Partially Paid
- f. Paid
- g. Closed

Siebel is the system of record for the master listing of promotional status values. Promotion status is a parent level to the promotion level in Demantra. The master listing of promotional statuses as defined in Siebel is sent to Demantra to populate the promotion status level in Demantra. All the pre-seeded Demantra statuses have been retained and the Siebel statuses added.

Note that Demantra provides the ability to take a "snapshot" of key promotional series to be used for planned vs. actual comparison. This snapshot functionality is hard-coded to the promotion status with the ID value of 4 (this is "d. Committed" in the pre-seeded values). During implementation, whatever status the snapshot should be taken at should be set to ID value 4.

As this definition doesn't change often, this is typically a one time process that happens once per implementation. If changes are made to the promotional status master list in Siebel, then this interface can be run on-demand to update Demantra.

Business Rules

1. It is recommended that this process be run once and whenever the promotion status values change in Siebel.
2. Promotional status information is always a full load. It must be loaded before the

Promotions extract.

Mapping

The Promotional Status data will be transformed from the Siebel table's structures into the structure and format of the Demantra Staging/Interface tables. The required data elements are:

Required Data Element	Demantra Reference	Siebel Reference
Promotional Status ID	Promotional Status ID	Promotional Status ID
Promotional Status Description	Promotional Status Description	Promotional Status Description
Promotional Status Code	Promotional Status Code	Promotional Status Code

ODI Package and Batch File Details

Recommended Scheduling Frequency: Once or when needed.

ODI Package Name:run Promotion Status Export

Directory of batch files: <root>\

ODI_DEM_SBL_Home\integration\Siebel-Demantra\PromoLevelLoad

Package Name	Description of Functionality
PROMO_STATUS_EXPORT.cmd	Extracts the promotion status values from Siebel to Demantra.

Promotions

Promotion information is sent from Siebel to Demantra for three separate instances as follows:

1. **Historical Load:** Historical promotion information should be loaded into Demantra for the modeling engine to predict future promotions. Promotional history is typically for the past two years. If a subset of customers/products is configured, to be sent to Demantra, only the promotion history for this subset is sent to Demantra. Promotion history is usually loaded once for each implementation.
2. **On-Going Nightly Update:** On a nightly basis current promotional information can be loaded into Demantra to ensure that Demantra has current information for the

calculation of baseline volume and lift coefficients. This extract is incremental and includes only promotions that have been created, modified, or deleted since the last incremental extract was drawn.

3. Near Real-Time Promotional Planning: Users can specify promotions to transfer from Siebel to Demantra by setting the Status of each promotion to "Send to Demantra". It is recommended that a background process be scheduled to run every 30 minutes to load these promotions into Demantra. Only promotions in the "Send to Demantra" status are included. After the promotions have been transferred, the status is changed to "Locked by Demantra" and then the record is locked until it returns from Demantra.

Mapping

Demantra Field	Siebel Source Table	Siebel Source Field	Comments
Promotion Name	S_SRC	NAME	---
Promotion Number	S_SRC	PROMO_NUM	---
Promotion ID	S_SRC	ROW_ID	---
Promotion name	S_SRC	NAME	---
Promotion Status	S_SRC	STATUS_CD	---
Promotion Type	S_SRC	SRC_CD	---
Promotion St Dt	S_SRC	PROG_END_DT	---
Promotion End Dt	S_SRC	PROG_END_DT	---
Account Name	S_ORG_EXT	NAME	Join PR_ACCNT_ID of S_SRC with PAR_ROW_ID of S_ORG_EXT
Account Type	S_ORG_EXT	OU_TYPE_CD	Join PR_ACCNT_ID of S_SRC with PAR_ROW_ID of S_ORG_EXT

Demantra Field	Siebel Source Table	Siebel Source Field	Comments
Plan	S_SRC	NAME	Join PAR_SRC_ID OF PROMOTION ROD_ID
Shipment Start Date	S_SRC	SHIP_START_DT	---
Shipment End Date	S_SRC	SHIP_END_DT	---
Product Name	S_PROD_INT	NAME	Join PROD_ID of S_SRC with ROW_ID of S_PROD_INT
Product Number	S_PROD_INT	PART_NUM	Join PROD_ID of S_SRC with ROW_ID of S_PROD_INT
Product Type	S_PROD_INT	TYPE	Join PROD_ID of S_SRC with ROW_ID of S_PROD_INT
Tactics	S_CG_PROMTACTIC	TACTICS_CD	Join SRC_ID of S_CG_PROMTACTIC with S_SRC ROW_ID
ACV %	S_SRC_CHNL	PROMO_ACV_PCT	Join ROW_ID of S_SRC with PAR_ROW_ID of S_SRC_CHNL
Off Invoice Allowance	S_SRC_CHNL	CMMT_XINV_CASE RATE	---
Bill back Allowance	S_SRC_CHNL	CMMT_BILL_BACK	---
Scan Down	S_SRC_CHNL	CMMT_VAL_SCAN_RED	---
Fixed Cost	S_SRC_CHNL	CMMT_FIXED_COST	---

Business Rules

1. It is recommended that this process be run as follows:

- On-demand for the initial history load including two years worth of promotional data.
 - Nightly as an automated scheduled process for an incremental load of all promotions created or modified in Siebel since the last update.
 - On-going during the day (every 30 minutes) for an incremental load of all promotions with the "Send to Demantra" status.
2. Each time the process for promotion data is run, a snapshot of the information for all promotion data elements is included.
 3. The assumption is that the all of the data elements loaded using the promotion data interface are the same for all location members, if the promotion is associated to more than one location (Site, Retailer, Division).
 4. The promotion transfer status in Demantra is set as follows for all promotions loaded from Siebel (including both new and modified promotions):
 - For the full load, the promotion transfer status is set to "Locked".
 - For the nightly incremental load, the promotion transfer status is set to "Locked".
 - For the near real-time load, the promotion transfer status is set to "Unlocked".
 5. For new promotions created in Siebel and sent to Demantra, any promotion attributes that are not specifically included in the feed from Siebel, should be set to the configured default value for the attribute. If no default value is specified, then 0 (zero) should be used. This includes the following promotion attributes:
 - Promotion Type
 - Integration Status
 - Scenario
 - Plan

ODI Package/Extract Overview

The Promotion Exports extract has been designed to support two product types and four account types. Implementations run only the one (or two) combinations that align to the way your promotions are planned in Siebel.

The product types are Product (SKU) and Promoted Product Group (a grouping of products typically promoted together). The account types are Ship-To Customer, Bill-To Customer, Customer Division, and Corporate Customer. Separate versions of the

promotion extracts are provided for each possible combination as listed below:

- Product AND Bill-To Customer
- Product AND Corporate Customer
- Product AND Customer Division
- Product AND Ship-To Customer
- Promoted Product Group AND Bill-To Customer
- Promoted Product Group AND Corporate Customer
- Promoted Product Group AND Customer Division
- Promoted Product Group AND Ship-To Customer

There are three ODI Package groups:

- Promotion Exports Full
- Promotion Exports Incr
- Promotion Export NRT

Within each group, there are two packages:

- Promotion Level (which represents the header)
- Promotion Data (which represents the details)

Finally, the near real-time extraction process ends with the Siebel_Promotion_Lock procedure which changes the status of transferred promotions in Siebel to "Locked by Demantra".

Important: After the status has been changed to "Locked by Demantra" in Siebel, it isn't actually locked. The administrator or implementation consultant needs to create a procedure to truly lock these records in the database.

Recommended scheduling frequency:

- Full: On-Demand for the initial history load including two years worth of promotional data.
- Incremental: Nightly as an automated scheduled process for an incremental load of all promotions created or modified in Siebel since the last update.

- Near Real-time: Every 30 minutes suggested to transfer all promotions in the "Send to Demantra" status to Demantra.

Mapping

ODI Package and Batch File Details: Full Load

Full load packages:

Promotion Level	Promotion Data
Run PLevel Full Extract	Run PData Full Extract PRD-SHIP
Run PLevel Full Extract	Run PData Full Extract PRD-BILL
Run PLevel Full Extract	Run PData Full Extract PRD-DIV
Run PLevel Full Extract	Run PData Full Extract PRD-CORP
Run PLevel Full Extract	Run PData Full Extract PG-SHIP
Run PLevel Full Extract	Run PData Full Extract PG-BILL
Run PLevel Full Extract	Run PData Full Extract PG-DIV
Run PLevel Full Extract	Run PData Full Extract PG-CORP

Batch Details:

Directory of full promotion header batch file: <root>\
 ODI_DEM_SBL_Home\integration\Siebel-Demantra\PromotionLevelLoad

Package Name	Description of Functionality
PROMO_LEVEL_FULL_EXTRACT.cmd	<ul style="list-style-type: none"> • Extracts the full set of promotion headers from Siebel • Launches the Demantra web server • Starts the IMPORT: SIEBEL_PROMOTION workflow

Directory of full promotion detail batch files: <root>

Package Name	Description of Functionality
PDATA_FULL_EXTRACT_PG_BILL.cmd	<ul style="list-style-type: none">• Extracts the full set of promotion group data sorted by billing account from Siebel.• Launches the Demantra web server.• Starts the IMPORT:
PDATA_FULL_EXTRACT_PG_CORP.cmd	<ul style="list-style-type: none">• Extracts the full set of promotion group data sorted by corporate ID from Siebel.• Launches the Demantra web server.• Starts the IMPORT: SIEBEL_PDATA_PG_CORP workflow
PDATA_FULL_EXTRACT_PG_DIV.cmd	<ul style="list-style-type: none">• Extracts the full set of promotion group data sorted by division from Siebel.• Launches the Demantra web server.• Starts the IMPORT: SIEBEL_PDATA_PG_DIV workflow
PDATA_FULL_EXTRACT_PG_SHIP.cmd	<ul style="list-style-type: none">• Extracts the full set of promotions group data sorted by shipping account from Siebel.• Launches the Demantra web server.• Starts the IMPORT: SIEBEL_PDATA_PG_SHIP workflow

Package Name	Description of Functionality
PDATA_FULL_EXTRACT_PRD_BILL.cmd	<ul style="list-style-type: none"> Extracts the full set of promotions data from Siebel organized by product and billing account. Launches the Demantra web server. Starts the IMPORT: SIEBEL_PDATA_PROD_BILL workflow
PDATA_FULL_EXTRACT_PRD_CORP.cmd	<ul style="list-style-type: none"> Extracts the full set of promotions data from Siebel organized by product and corporate account. Launches the Demantra web server. Starts the IMPORT: SIEBEL_PDATA_PROD_CORP workflow
PDATA_FULL_EXTRACT_PRD_DIV.cmd	<ul style="list-style-type: none"> Extracts the full set of promotions data from Siebel organized by product and division. Launches the Demantra web server. Starts the IMPORT: SIEBEL_PDATA_PROD_DIV workflow
PDATA_FULL_EXTRACT_PRD_SHIP.cmd	<ul style="list-style-type: none"> Extracts the full set of promotions data from Siebel organized by product and shipping account. Launches the Demantra web server. Starts the IMPORT: SIEBEL_PDATA_PROD_SHIP workflow

ODI Package and Batch File Details: Incremental Load

Incremental load packages

Promotion Level	Promotion Data
Run Promotion Level Incr Extract	Run PData Incr Extract PRD-SHIP
Run Promotion Level Incr Extract	Run PData Incr Extract PRD-BILL
Run Promotion Level Incr Extract	Run PData Incr Extract PRD-DIV
Run Promotion Level Incr Extract	Run PData Incr Extract PRD-CORP
Run Promotion Level Incr Extract	Run PData Incr Extract PG-SHIP
Run Promotion Level Incr Extract	Run PData Incr Extract PG-BILL
Run Promotion Level Incr Extract	Run PData Incr Extract PG-DIV
Run Promotion Level Incr Extract	Run PData Incr Extract PG-CORP
Run Promotion Level Incr Extract	Run PData Incr Extract PRD-SHIP

Incremental batch files:

Directory of incremental promotion header batch file: <root>\
 ODI_DEM_SBL_Home\integration\Siebel-Demantra\PromotionLevelLoad

Package Name	Description of Functionality
PROMO_LEVEL_INCR_EXTRACT.cmd	<ul style="list-style-type: none"> Extracts the incremental set of promotion headers from Siebel Launches the Demantra web server Starts the IMPORT: SIEBEL_PROMOTION workflow

Directory of incremental promotion details batch files: <root>\
 ODI_DEM_SBL_Home\integration\Siebel-Demantra\PromotionDataLoad\INCR

Package Name	Description of Functionality
PDATA_INCR_EXTRACT_PG_BILL.cmd	<ul style="list-style-type: none"> Extracts an incremental set of promotion group data sorted by billing account from Siebel Launches the Demantra web server Starts the IMPORT: SIEBEL_PDATA_PG_BILL workflow
PDATA_INCR_EXTRACT_PG_CORP.cmd	<ul style="list-style-type: none"> Extracts an incremental set of promotion group data sorted by corporate ID from Siebel Launches the Demantra web server Starts the IMPORT: SIEBEL_PDATA_PG_CORP workflow
PDATA_INCR_EXTRACT_PG_DIV.cmd	<ul style="list-style-type: none"> Extracts an incremental set of promotion group data sorted by division from Siebel Launches the Demantra web server Starts the IMPORT: SIEBEL_PDATA_PG_DIV workflow
PDATA_INCR_EXTRACT_PG_SHIP.cmd	<ul style="list-style-type: none"> Extracts an incremental set of promotion group data sorted by shipping account from Siebel Launches the Demantra web server Starts the IMPORT: SIEBEL_PDATA_PG_SHIP workflow

Package Name	Description of Functionality
PDATA_INCR_EXTRACT_PRD_BILL.cmd	<ul style="list-style-type: none"> Extracts an incremental set of promotions data from Siebel organized by product and billing account Launches the Demantra web server Starts the IMPORT: SIEBEL_PDATA_PROD_BILL workflow
PDATA_INCR_EXTRACT_PRD_CORP.cmd	<ul style="list-style-type: none"> Extracts an incremental set of promotions data from Siebel organized by product and corporate account Launches the Demantra web server Starts the IMPORT: SIEBEL_PDATA_PROD_CORP workflow
PDATA_INCR_EXTRACT_PRD_DIV.c	<ul style="list-style-type: none"> Extracts an incremental set of promotions data from Siebel organized by product and division Launches the Demantra web server Starts the IMPORT: SIEBEL_PDATA_PROD_DIV workflow
PDATA_INCR_EXTRACT_PRD_SHIP.cmd	<ul style="list-style-type: none"> Extracts an incremental set of promotions data from Siebel organized by product and shipping account Launches the Demantra web server Starts the IMPORT: SIEBEL_PDATA_PROD_SHIP workflow

ODI Package and Batch File Details: Near Real-Time

Near Real-time Packages:

Promotion Level	Promotion Data
Run Promotion Level NRT Extract	run PData Incr Extract PRD-SHIP
Run Promotion Level NRT Extract	run PData Incr Extract PRD-BILL
Run Promotion Level NRT Extract	run PData Incr Extract PRD-DIV
Run Promotion Level NRT Extract	run PData Incr Extract PRD-CORP
Run Promotion Level NRT Extract	run PData Incr Extract PG-SHIP
Run Promotion Level NRT Extract	run PData Incr Extract PG-BILL
Run Promotion Level NRT Extract	run PData Incr Extract PG-DIV
Run Promotion Level NRT Extract	run PData Incr Extract PG-CORP
Run Promotion Level NRT Extract	run PData Incr Extract PRD-SHIP

Near real-time batch files:

Directory of incremental promotion header batch files: *<root>*\
 ODI_DEM_SBL_Home\integration\Siebel-Demantra\PromotionLevelLoad

Package Name	Description of Functionality
PROMO_LEVEL_NRT_EXTRACT.cmd	<ul style="list-style-type: none"> Extracts the promotions with the status "Send to Demantra" from Siebel Launches the Demantra web server Starts the IMPORT: SIEBEL_PROMOTION workflow

Directory of incremental promotion details batch files: *<root>*\
 ODI_DEM_SBL_Home\integration\Siebel-Demantra\PromotionDataLoad\INCR

Package Name	Description of Functionality
PDATA_INCR_EXTRACT_PG_BILL.cmd	<ul style="list-style-type: none"> Extracts an incremental set of promotion group data sorted by billing account from Siebel Launches the Demantra web server Starts the IMPORT: SIEBEL_PDATA_PG_BILL workflow
PDATA_INCR_EXTRACT_PG_CORP.cmd	<ul style="list-style-type: none"> Extracts an incremental set of promotion group data sorted by corporate ID from Siebel Launches the Demantra web server Starts the IMPORT: SIEBEL_PDATA_PG_CORP workflow
PDATA_INCR_EXTRACT_PG_DIV.cmd	<ul style="list-style-type: none"> Extracts an incremental set of promotion group data sorted by division from Siebel Launches the Demantra web server Starts the IMPORT: SIEBEL_PDATA_PG_DIV workflow
PDATA_INCR_EXTRACT_PG_SHIP.cmd	<ul style="list-style-type: none"> Extracts an incremental set of promotion group data sorted by shipping account from Siebel Launches the Demantra web server Starts the IMPORT: SIEBEL_PDATA_PG_SHIP workflow

Package Name	Description of Functionality
PDATA_INCR_EXTRACT_PRD_BILL.cmd	<ul style="list-style-type: none"> Extracts an incremental set of promotions data from Siebel organized by product and billing account Launches the Demantra web server Starts the IMPORT: SIEBEL_PDATA_PROD_BILL workflow
PDATA_INCR_EXTRACT_PRD_CORP.cmd	<ul style="list-style-type: none"> Extracts an incremental set of promotions data from Siebel organized by product and corporate account Launches the Demantra web server Starts the IMPORT: SIEBEL_PDATA_PROD_CORP workflow
PDATA_INCR_EXTRACT_PRD_DIV.cmd	<ul style="list-style-type: none"> Extracts an incremental set of promotions data from Siebel organized by product and division Launches the Demantra web server Starts the IMPORT: SIEBEL_PDATA_PROD_CORP workflow
PDATA_INCR_EXTRACT_PRD_DIV.cmd	<ul style="list-style-type: none"> Extracts an incremental set of promotions data from Siebel organized by product and division Launches the Demantra web server Starts the IMPORT: SIEBEL_PDATA_PROD_DIV workflow

Package Name	Description of Functionality
PDATA_INCR_EXTRACT_PRD_SHIP.cmd	<ul style="list-style-type: none"> Extracts an incremental set of promotions data from Siebel organized by product and shipping account Launches the Demantra web server Starts the IMPORT: SIEBEL_PDATA_PROD_SHIP workflow

Directory of locking batch file: <root>

\ODI_DEM_SBL_HOME\integration\Siebel-Demantra\PromotionLock

Package Name	Description of Functionality	Description of Functionality
SIEBEL_PROMOTIONL_LOCK_MAIN.cmd	Runs:	<ul style="list-style-type: none"> ODI_SIEBEL_PROMOTION_LOCK.cmd EIM_PROMO_STATUS.BAT
ODI_SIEBEL_PROMOTION_LOCK.cmd		Selects the promotions that need their status changed from "Send to Demantra" to "Locked by Demantra".
EIM_PROMO_STATUS.BAT		Changes the promotion status to "Locked by Demantra" for the selected promotions.

Promotional Tactics

Promotional tactics define the type of promotional activity. Typical tactics are:

- Feature Only: Referring to a feature or ad in a retailer's circular without any display activity.
- Display Only: Referring to an in-store display such as an end-aisle display or a display unit in the aisle without feature activity.
- Feature & Display: Referring to both feature and display activity.
- TPR: Meaning Temporary Price Reduction, which refers to a reduction in retail price without any associated feature or display activity.

Siebel is the system of record for the master listing of promotional tactics.

The existing "Veh Type" series in Demantra stores the promotional tactic. This series resides on Promotion Data and is supported by a lookup table containing the valid promotional tactics.

The master listing of promotional tactics as defined in Siebel is sent to Demantra to populate the lookup table for the promotional tactic series. The lookup table stores the tactic type and discount percentage for each promotional tactic. These fields are manually maintained and are needed to support the causal factors for the Forecasting Engine. For example, if the promotional tactic from Siebel was "Feature with a 10% discount" the tactic type would be set to "Feature" and the discount percentage would be set to "10." See the Demantra Setup, page 20-24 section for more information about modifying this mapping table.

As this definition doesn't change often, this is typically a one-time process that happens once per implementation. If changes are made to the promotional tactics master list in Siebel, then this interface can be run on-demand to update Demantra.

Business Rules

1. It is recommended that this process be loaded once, or if the promotional tactics change in Siebel.
2. The promotional tactic information is always a full load.

Mapping

The Promotional Tactics data will be transformed from the Siebel table's structures into the structure and format of the Demantra Staging/Interface tables. The required data elements are:

Required Data Element	Demantra Reference	Siebel Reference
Promotional Tactic ID	Promotional Tactic ID	Promotional Tactic ID
Promotional Tactic Description	Promotional Tactic Description	Promotional Tactic Description

ODI Package and Batch File Details

Recommended Scheduling Frequency: Once or when needed.

ODI Package Name: run Promotion Hierarchy Exp

Directory of batch files: <root>\

ODI_DEM_SBL_Home\integration\Siebel-Demantra\PromoLevelLoad

Package Name	Description of Functionality
PROMO_TYPE_EXPORT.cmd	Extracts the promotional tactics from Siebel to Demantra.

Sales Activity

Historical shipment data must be loaded into Demantra for the modeling engine to predict the baseline forecast. Shipment data is used to derive the baseline forecast for all the products in different accounts. It is also used to derive the proportion logic used to aggregate and disaggregate the data at different hierarchy levels. The integration does not include UOM conversion capability and assumes that all volume data in both applications is stored in a common unit of measure.

Business Rules

1. You must perform a one-time load of two years of shipment history.
2. It is recommended that this process be scheduled on a regular interval (weekly). This on-going load is always incremental, including only those records that got changed from the previous load.

Mapping

Demantra Field	Type	Siebel Source Table	Siebel Source Field
Product Name	VARCHAR2(100)	S_PROD_INT	PROD_NAME
Product Number	VARCHAR2(50)	S_PROD_INT	PART_NUM
Site Name	VARCHAR2(30)	S_ORG_EXT	NAME
Site Number	VARCHAR2(100)	S_ORG_EXT	OU_NUM
Shipment Volume	VARCHAR2(22)	S_PROD_SHIPMENT	ACTL_CASES
Unit of Measure	VARCHAR2(30)	S_PROD_SHIPMENT	PREF_UOM_CD
Start Date	DTYPE_DATE	S_PERIOD	START_DT
End Date	DTYPE_DATE	S_PERIOD	END_DT

ODI Package and Batch File Details

Recommended Scheduling Frequency:

- Full: On-demand for the initial history load including two years worth of historical shipment data.
- Incremental: Nightly as an automated scheduled process for an incremental load of all shipments created or modified in Siebel since the last update.

ODI Package Names:

- run Sales History Incr Load
- run Sales History Full Load

Directory of batch files: <root>\

ODI_DEM_SBL_Home\integration\Siebel-Demantra\SalesHistoryLoad

Package Name	Description of Functionality
SALES_HISTORY_INCR_LOAD.cmd	Extracts the incremental shipping data from Siebel to Demantra.
SALES_HISTORY_FULL_LOAD.cmd	Extracts two years of historical shipping data from Siebel to Demantra.

Demantra to Siebel ODI Packages and Batch Files

There are IFB files within each Demantra-Siebel extract directory. These files provide the Siebel Enterprise Integration Manager with configuration information such as the user name and password. The IFB files reference the user name and password in the runsrvrmgr.bat file, so there is no need to configure each IFB file.

Baseline Volume Forecast

The baseline volume forecast is a key element in promotion planning to understand the effect of the promotion compared to the incremental volume. In addition, baseline volume is used to evaluate promotion effectiveness.

The ongoing load is always a full load for 18 months in the future. Baseline records for products that have been deleted or inactivated in Siebel are filtered out. The integration does not include UOM conversion capability and assumes that all volume data in both applications is stored in a common unit of measure.

Recommended Scheduling Frequency: Weekly

Directory: <root>\ ODI_DEM_SBL_Home\integration\Demantra-Siebel\Baseline_Load

Package Name	Description of Functionality
BASLINE_TO_SIEBEL_LOAD_MAIN.cmd	<p>Runs the following batch files in sequence:</p> <ol style="list-style-type: none">1. ODI_APPPROC_BLE_VOLUME_BASE_TTL.cmd2. ODI_BASLINE_FORECAST_EXPORT.cmd3. EIM_BASLINE_LOAD.bat <p>If you have another routine that runs the APPPROC_BLE_VOLUME_BASE_FUTURE procedure, you can remove the reference to the ODI_APPPROC_BLE_VOLUME_BASE_TTC.cmd in this batch file.</p>
ODI_APPPROC_BLE_VOLUME_BASE_TTL.cmd	<p>Updates sales_data.vol_base_ttl for dates in the past This procedure sets the field equal to the value of sales_data.sdata5, the syndicated base volume. This procedure must be run before the Baseline Forecast can be exported. If your organization uses a routine that already runs the APPPROC_BLE_VOLUME_BASE_FUTURE.cmd procedure, this batch routine does not need to be run.</p>
ODI_BASLINE_FORECAST_EXPORT.cmd	<ul style="list-style-type: none">• Launches the Demantra webserver• Starts the EXPORT:BASELINE_VOLUME_FORECAST workflow to extract data to the Demantra temporary staging table.
EIM_BASLINE_LOAD.bat	<p>Launches the Enterprise Integration Manager and loads the extract data according to the configuration settings in the Baseline_Import.ifb file.</p>

Mapping

Demantra Name	Siebel Target Table	Siebel Target Field	Type
Product Name	S_PROD_BASELINE	PROD_NAME	VARCHAR(100)
Product Number	S_PROD_INT	PART_NUM	VARCHAR(50)
Site Name	S_ORG_EXT	NAME	VARCHAR(30)
Site Number	S_ORG_EXT	OU_NUM	VARCHAR(100)
Start Date	S_PERIOD	START_DT	DTYPE_DATE
End Date	S_PERIOD	END_DT	DTYPE_DATE
Shipment Volume	S_PROD_BASELINE	BASE_SALES	Number

Promotional Lift Coefficients

Demantra generates lift coefficients for predicting the incremental volume associated with a planned promotion based on the parameters of the promotion such as promotion tactics.

Lift coefficients are generated at the product (SKU) or promoted product group level by customer account (retailer). Category level lift coefficients are not included.

Lift coefficients are generated for a combined promotional tactic and price discount percentage. The following is a list of the most likely sets of tactics and price discounts and varies by implementation:

- Feature with no price discount
- Feature with up to a 5% price discount
- Feature with up to a 10% price discount
- Feature with up to a 15% price discount
- Feature with up to a 20% price discount
- Feature with up to a 25%
- Feature with more than a 25% price discount
- Display with no price discount

- Display with up to a 5% price discount
- Display with up to a 10% price discount
- Display with up to a 15% price discount
- Display with up to a 20% price discount
- Display with up to a 25%
- Display with more than a 25% price discount
- Feature & Display with no price discount
- Feature & Display with up to a 5% price discount
- Feature & Display with up to a 10% price discount
- Feature & Display with up to a 15% price discount
- Feature & Display with up to a 20% price discount
- Feature & Display with up to a 25%
- Feature & Display with more than a 25% price discount
- TPR of up to a 5% price discount
- TPR of up to a 10% price discount
- TPR of up to a 15% price discount
- TPR of up to a 20% price discount
- TPR of up to a 25%
- TPR of more than a 25% price discount

Business Rules

1. Lift coefficients are generated for the time period of one year. The ongoing load is always a full load for 18 months in the future.
2. Lift coefficient records for products that have been deleted or inactivated in Siebel will be filtered out in the interface.

Mapping

Name	Demantra Source Field	Siebel Target Table	Siebel Target Field
Promotion Name	PROMO_NAME	S_SRC	NAME
Promotion Number	PROMO_NUM	S_SRC	SRC_NUM
Promotion Status	PROMO_STATUS	S_SRC	STATUS_CD
Plan Number	PLAN_NUM	S_SRC	SRC_NUM
Plan Name	PLAN_NAME	S_SRC	NAME
Promotion St Dt	PROMO_ST_DT	S_SRC	PROG_END_DT
Promotion End Dt	PROMO_END_DT	S_SRC	PROG_END_DT
Account Name	ACCNT_NAME	S_ORG_EXT	NAME
Account Number	ACCNT_NUM	S_ORG_EXT	OU_NUM
Shipment Start Date	SHIP_ST_DT	S_SRC	SHIP_END_DT
Shipment End Date	SHIP_END_DT	S_SRC	SHIP_END_DT

ODI Package and Batch File Details

Recommended Scheduling Frequency: Weekly or Monthly

Directory: <root>\

ODI_DEM_SBL_Home\integration\Demantra-Siebel\Lifftfactor_Load

Package Name	Description of Functionality
LIFT_FACTOR_EXPORT_MAIN.cmd	Runs the following batch files in sequence: <ul style="list-style-type: none">• ODI_LIFT_FACTOR_EXPORT.cmd• EIM_LFT_FCTR.bat

Package Name	Description of Functionality
ODI_LIFT_FACTOR_EXPORT.cmd	Extracts promotional lift coefficients from Demantra internal tables.
EIM_BASELINE_LOAD.bat	Launches the Enterprise Integration Manager and loads the extract data according to the configuration settings in the LIFT_COEFFICIENT.ifb file.

Promotions

Users specify promotions to transfer from Demantra to Siebel by setting the promotion transfer status of each promotion to "Transfer to Siebel". It is recommended that a background process run every 30 minutes to load these promotions into Siebel. Once these promotions have been transferred, the promotion transfer status is changed to "Locked" so that the promotion is not transferred the next time the process is run.

Promotions can be exported from Demantra at the correct customer account and product level to align with the account type and product type in Siebel. For promotions created in Siebel and sent to Demantra, the account type and product type for the promotion in Siebel must be retained. If these promotions are modified in Demantra and transferred back to Siebel, they must be sent at the appropriate levels of the Demantra site and item hierarchies that correspond to the Siebel account type and product type.

Business Rules

1. This process is designed to run on an on-going basis during the day (every 30 minutes) for an incremental load of all promotions where the promotion transfer status is "Transfer to Siebel".
2. As part of the process, the promotion transfer status is changed from "Transfer to Siebel" to "Locked" for each processed promotion.
3. It is assumed that all of the data elements loaded using the promotion data interface are the same for all location members, if the promotion is associated to more than one location (Site, Retailer, Division).
4. New promotions created in Demantra are assigned a Demantra promotion ID. Siebel accepts and uses this promotion ID and does not assign its own.
5. Funds are not be assigned to promotions in Demantra. New promotions created in Demantra are sent to Siebel with a fund default of "DemantraFund". The user will be required to maintain the promotion in Siebel and assign the appropriate fund(s).

It is recommended that Siebel be configured to not allow promotions to be approved until a valid fund(s) has been assigned.

6. When new promotions are created in Demantra, they are automatically assigned a status of "Unplanned". Promotions in this unplanned status are used for planning and testing purposes and are not considered valid promotions. While in the unplanned state, these promotions cannot be transferred to Siebel.

Mapping

Demantra Name	Demantra Source Field	Siebel Target Table	Siebel Target Field
Promotion Name	PROMO_NAME	S_SRC	NAME
Promotion Number	PROMO_NUM	S_SRC	SRC_NUM
Promotion Status	PROMO_STATUS	S_SRC	STATUS_CD
Plan Number	PLAN_NUM	S_SRC	SRC_NUM
Plan Name	PLAN_NAME	S_SRC	NAME
Promotion St Dt	PROMO_ST_DT	S_SRC	PROG_END_DT
Promotion End Dt	PROMO_END_DT	S_SRC	PROG_END_DT
Account Name	ACCNT_NAME	S_ORG_EXT	NAME
Account Number	ACCNT_NUM	S_ORG_EXT	OU_NUM
Shipment Start Date	SHIP_ST_DT	S_SRC	SHIP_END_DT
Shipment End Date	SHIP_END_DT	S_SRC	SHIP_END_DT
Demantra Name	Demantra Source Field	Siebel Target Table	Siebel Target Field
Category Name	PROD_NAME	S_CTLG_CAT	---
Category Number	PROD_NUM	S_CTLG_CAT	---

Demantra Name	Demantra Source Field	Siebel Target Table	Siebel Target Field
Product Name	PROD_NAME	S_PROD_INT	NAME
Product Number	PROD_NUM	S_PROD_INT	PART_NUM
Product Type	PROD_TYPE	S_PROD_INT	TYPE

Demantra Name	Demantra Source Field	Siebel Target Table	Siebel Target field
ACV %	ACV	S_SRC_CHNL	PROMO_ACV_PCT
Off Invoice Allowance	OI_RATE	S_SRC_CHNL	CMMT_XINV_CASE RATE
Bill back Allowance	BB_RATE	S_SRC_CHNL	CMMT_BILL_BACK
Scan Down Allowance	SD_RATE	S_SRC_CHNL	CMMT_VAL_SCAN_RED
Fixed Cost	FXD_AMT	S_SRC_CHNL	CMMT_FIXED_COST
Promoted Price	PROMOTED_PRICE	S_SRC_CHNL	PROMO_PRI_PT
Tactics	TACTIC	S_CG_PROMTACTIC	TACTICS_CD

ODI Package/Extract Overview

The Promotion Exports extract has been designed to support two product types and four account types. Implementations generally run only the one (or two) combinations that align to the way the promotions are planned in Siebel.

The product types are product (SKU) and promoted product group (a grouping of products typically promoted together). The account types are ship-to customer, bill-to customer, customer division, and corporate customer. Separate versions of the promotion extracts are provided for each possible combination as listed below:

- Product AND Bill-To Customer
- Product AND Corporate Customer

- Product AND Customer Division
- Product AND Ship-To Customer
- Promoted Product Group AND Bill-To Customer
- Promoted Product Group AND Corporate Customer
- Promoted Product Group AND Customer Division
- Promoted Product Group AND Ship-To Customer

Depending on the combinations preferred, a corresponding main batch file runs the subroutines that extract the correct promotion information. The subroutines are organized in the following order:

- Step 1a: Extraction of promotion header with the appropriate combination. (This is the only step that varies in the main batch files.)
- Step 1b: Transfer of promotion header to Siebel Enterprise Integration Manager.
- Step 2a: Extraction of promotion category data.
- Step 2b: Transfer of promotion category data to Siebel Enterprise Integration Manager.
- Step 3a: Extraction of promotion by product data.
- Step 3b: Transfer of promotion product data to Siebel Enterprise Integration Manager.
- Step 4a: Extraction of deleted promotion tactics.
- Step 4b: Transfer of deleted promotion tactics to Siebel Enterprise Integration Manager.
- Step 5a: Extraction of promotion details.
- Step 5b: Transfer of promotion details to Siebel Enterprise Integration Manager.

The extraction process ends with the ODI_Demantra_PROMOTION_LOCK.cmd to lock the promotion transfer status so that it isn't transferred when the extracts are run the next time.

Note: The runsrvrmgr.bat sets the username and password for all the ifb files.

ODI Package and Batch File Details

Recommended Scheduling Frequency: Every 30 minutes

Directory of batch files: *<root>*\

ODI_DEM_SBL_Home\integration\Demantra-Siebel\PROMOTION_LOAD

Directory of IFB files: *<root>*

\ODI_DEM_SBL_Home\integration\Demantra-Siebel\PROMOTION_LOAD\IFB

Package Name	Description of Functionality
PROMOTION_TO_SIEBEL_PD_BILL_MAIN.cmd	<p data-bbox="873 306 1325 333">Runs the following batch files in sequence:</p> <ol style="list-style-type: none"> <li data-bbox="873 359 1365 575"> <p data-bbox="873 359 1224 386">1. Loading Promotions Header</p> <ul style="list-style-type: none"> <li data-bbox="922 415 1365 474">• ODI_STEP_1A_HEADER_PD_BILL.cmd. <li data-bbox="922 518 1365 575">• EIM_STEP_1B_PROMOTION_HDR_STEP.BAT <li data-bbox="873 627 1365 848"> <p data-bbox="873 627 1276 655">2. Loading Promotions By Category</p> <ul style="list-style-type: none"> <li data-bbox="922 684 1365 743">• ODI_STEP_2A_PROMO_CATAGORY_EXP.cmd <li data-bbox="922 787 1365 848">• EIM_STEP_2B_PROMOTION_CATG_STEP.BAT <li data-bbox="873 900 1365 1121"> <p data-bbox="873 900 1263 928">3. Loading Promotions by Product</p> <ul style="list-style-type: none"> <li data-bbox="922 957 1365 1016">• ODI_STEP_3A_PROM_BY_PRODUCT_EXP.cmd <li data-bbox="922 1060 1365 1121">• EIM_STEP_3B_PROMOTION_PROD_STEP.BAT <li data-bbox="873 1173 1365 1394"> <p data-bbox="873 1173 1195 1201">4. Loading Promotions Data</p> <ul style="list-style-type: none"> <li data-bbox="922 1230 1365 1289">• ODI_STEP_4A_PROMO_DETAILS_EXP.cmd <li data-bbox="922 1333 1365 1394">• EIM_STEP_4B_PROMOTION_PROD_DTL_STEP.BAT <li data-bbox="873 1446 1365 1667"> <p data-bbox="873 1446 1308 1474">5. Loading Promotions Promoted Price</p> <ul style="list-style-type: none"> <li data-bbox="922 1503 1365 1562">• ODI_STEP_5A_PROMO_PRICE_EXP.cmd <li data-bbox="922 1606 1365 1667">• EIM_STEP_5B_Promoted_Price_STEP.BAT

Package Name	Description of Functionality
PROMOTION_TO_SIEBEL_PD_CORP_MAIN.cmd	<p>Same as PROMOTION_TO_SIEBEL_PD_BILL_MAIN.cmd except for the following:</p> <ol style="list-style-type: none"> Loading Promotions Header <ul style="list-style-type: none"> ODI_STEP_1A_HEADER_PD_CORP.cmd EIM_STEP_1B_PROMOTION_HDR_STEP.BAT
PROMOTION_TO_SIEBEL_PD_DIV_MAIN.cmd	<p>Same as PROMOTION_TO_SIEBEL_PD_BILL_MAIN.cmd except for the following:</p> <ol style="list-style-type: none"> Loading Promotions Header <ul style="list-style-type: none"> ODI_STEP_1A_HEADER_PD_DIV.cmd EIM_STEP_1B_PROMOTION_HDR_STEP.BAT
PROMOTION_TO_SIEBEL_PD_SHIP_MAIN.cmd	<p>Same as PROMOTION_TO_SIEBEL_PD_BILL_MAIN.cmd except for the following:</p> <ol style="list-style-type: none"> Loading Promotions Header <ul style="list-style-type: none"> ODI_STEP_1A_HEADER_PD_SHIP.cmd EIM_STEP_1B_PROMOTION_HDR_STEP.BAT

Package Name	Description of Functionality
PROMOTION_TO_SIEBEL_PG_BILL_MAIN.cmd	<p>Same as PROMOTION_TO_SIEBEL_PD_BILL_MAIN.cmd except for the following:</p> <ol style="list-style-type: none"> Loading Promotions Header <ul style="list-style-type: none"> ODI_STEP_1A_HEADER_PG_BILL.cmd EIM_STEP_1B_PROMOTION_HDR_STEP.BAT
PROMOTION_TO_SIEBEL_PG_CORP_MAIN.cmd	<p>Same as PROMOTION_TO_SIEBEL_PD_BILL_MAIN.cmd except for the following:</p> <ol style="list-style-type: none"> Loading Promotions Header <ul style="list-style-type: none"> ODI_STEP_1A_HEADER_PG_CORP.cmd EIM_STEP_1B_PROMOTION_HDR_STEP.BAT
PROMOTION_TO_SIEBEL_PG_DIV_MAIN.cmd	<p>Same as PROMOTION_TO_SIEBEL_PD_BILL_MAIN.cmd except for the following:</p> <ol style="list-style-type: none"> Loading Promotions Header <ul style="list-style-type: none"> ODI_STEP_1A_HEADER_PG_DIV.cmd EIM_STEP_1B_PROMOTION_HDR_STEP.BAT

Package Name	Description of Functionality
PROMOTION_TO_SIEBEL_PG_SHIP_MAIN.cmd	<p>Same as PROMOTION_TO_SIEBEL_PD_BILL_MAIN.cmd except for the following:</p> <ol style="list-style-type: none"> Loading Promotions Header <ul style="list-style-type: none"> ODI_STEP_1A_HEADER_PG_SHIP.cmd EIM_STEP_1B_PROMOTION_HDR_STEP.BAT
ODI_STEP_1A_HEADER_PD_BILL.cmd	Launches the Demantra web server and starts the EXPORT: SIEBEL_PROMOTIONS_PD_BILL workflow.
ODI_STEP_1A_HEADER_PD_BILL.cmd	Launches the Demantra web server and starts the EXPORT: SIEBEL_PROMOTIONS_PD_BILL workflow.
ODI_STEP_1A_HEADER_PD_CORP.cmd	Launches the Demantra web server and starts the EXPORT: SIEBEL_PROMOTIONS_PD_CORP workflow.
ODI_STEP_1A_HEADER_PD_DIV.cmd	Launches the Demantra web server and starts the EXPORT: SIEBEL_PROMOTIONS_PD_DIV workflow.
ODI_STEP_1A_HEADER_PD_SHIP.cmd	Launches the Demantra web server and starts the EXPORT: SIEBEL_PROMOTIONS_PD_SHIP workflow.
ODI_STEP_1A_HEADER_PG_BILL.cmd	Launches the Demantra web server and starts the EXPORT: SIEBEL_PROMOTIONS_PG_BILL workflow.
ODI_STEP_1A_HEADER_PG_CORP.cmd	Launches the Demantra web server and starts the EXPORT: SIEBEL_PROMOTIONS_PG_CORP workflow.

Package Name	Description of Functionality
ODI_STEP_1A_HEADER_PG_DIV.cmd	Launches the Demantra web server and starts the EXPORT: SIEBEL_PROMOTIONS_PG_DIV workflow.
ODI_STEP_1A_HEADER_PG_SHIP.cmd	Launches the Demantra web server and starts the EXPORT: SIEBEL_PROMOTIONS_PG_SHIP workflow.
ODI_STEP_2A_PROMO_CATAGORY_EXP.cmd	Extracts the Promotions By Category data from Demantra.
ODI_STEP_3A_PROM_BY_PRODUCT_EXP.cmd	Extracts the Promotions By Product data from Demantra.
ODI_STEP_4A_DELETE_PROMO_TACTICS.cmd	Extracts the Promotions data from Demantra
ODI_STEP_5A_PROMO_DETAILS_EXP.cmd	Extracts the Promotions Promoted Price data from Demantra.
EIM_STEP_1B_PROMOTION_HDR_STEP.BAT	Launches the Enterprise Integration Manager and loads the promotion header according to the configuration settings in the PROMOTION_HDR.ifb file.
EIM_STEP_2B_PROMOTION_CATG_STEP.BAT	Launches the Enterprise Integration Manager and loads the promotion category data according to the configuration settings in the PROMOTION_CATG.ifb file.
EIM_STEP_3B_PROMOTION_PROD_STEP.BAT	Launches the Enterprise Integration Manager and loads the promotion product data according to the configuration settings in the PROMOTION_PROD.ifb file.
EIM_STEP_4B_DELETE_PROMO_TACTICS.BAT	Launches the Enterprise Integration Manager and loads the deleted promotion tactics data according to the configuration settings in the PROMOTION_TACTICS_DEL.ifb file.

Package Name	Description of Functionality
EIM_STEP_5B_PROMOTION_PROD_DTL_S TEP.BAT	Launches the Enterprise Integration Manager and loads the promotions promoted price data according to the configuration settings in the PROMOTION_PROD_DTL.ifb file.
ODI_DEMANTRA_PROMOTION_LOCK.cmd	Changes the status of the promotions that have the transfer status "Transfer to Siebel" to "Locked". This batch file is run after the main batch routine.

Demantra Sales and Operations Planning to Hyperion Integration

This chapter describes how Demantra Sales and Operations Planning integrates with Hyperion.

This chapter covers the following topics:

- Demantra Sales and Operations Planning - Hyperion Planning Integration Overview
- Architectural Process
- Integration Points Overview
- Business Processes

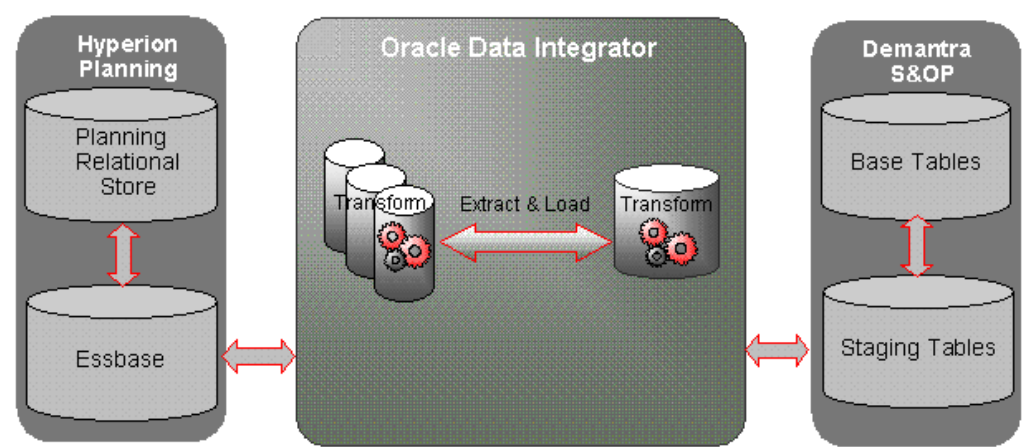
Demantra Sales and Operations Planning - Hyperion Planning Integration Overview

Accurately predicting revenue and operating performance is a daunting challenge facing many enterprises today. Despite awareness of the adverse impact of missed forecasts on their business plans, the most common solution for budgeting and planning is still a disconnected spreadsheet that makes the planning process unreliable and inefficient. The resulting long budget cycles and forecasting inaccuracies prevent responsiveness to change, causing companies to miss business opportunities while wasting money and resources on declining business segments. By integrating Oracle Hyperion Planning and Oracle Demantra, top-down planning and bottom-up planning can be linked to improve the accuracy and reliability of annual plans, budgets and financial forecasts.

The Demantra S&OP - Hyperion Planning integration includes a set of interfaces, pre-seeded worksheets, workflows, methods and data series.

Architectural Process

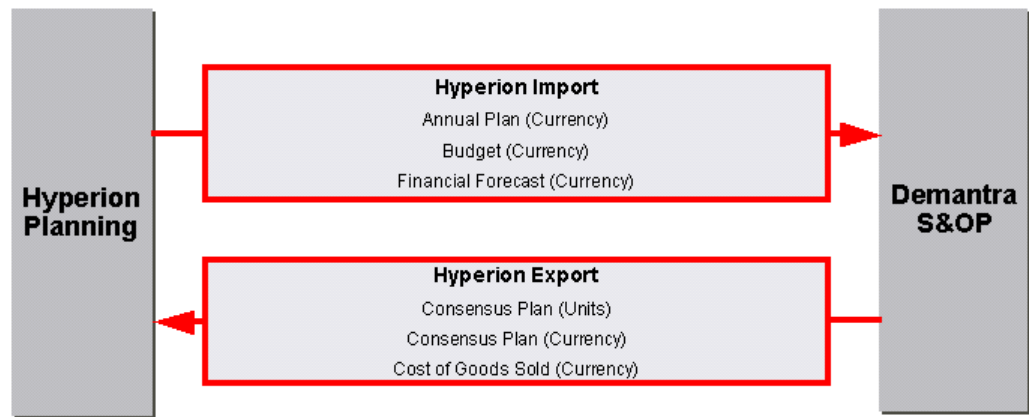
Oracle Hyperion Planning and Oracle Demantra exchange information through the use of the Oracle Data Integrator (ODI) Adapter for Hyperion Planning, the Oracle Data Integrator (ODI) to transform the data, and Oracle Demantra integration interfaces and workflows as shown in the following diagram:



The integration processes can be run when required from Demantra worksheets, workflows or ODI.

Integration Points Overview

The following integration points are part of the integration between the Oracle Demantra Sales and Operations Planning module and Hyperion Planning applications.



The following table details the frequency and recommended load for each integration point is detailed below:

Interface	Flow	Suggested Frequency
Hyperion Import	Hyperion to Demantra	Quarterly or monthly
Hyperion Export	Demantra to Hyperion	Weekly or monthly

Business Processes

1. The Chief Financial Officer develops the annual plan using a bottom-up planning process followed by top-down budget. The financial forecast is a refinement of the budget and includes actuals for historical periods.
2. The annual plan, budget and financial forecast (all currency) are exported and used in Demantra S&OP as financial performance targets to compare with the consensus forecast (in currency).
3. The S&OP Manager compares the financial forecast with the Demantra baseline forecast to identify gaps between the revenue predicted by the Demantra's bottom-up forecast and Hyperion's top-down financial plan.
4. A consensus forecast is developed collaboratively in Demantra S&OP
5. To increase demand and close revenue gaps, promotions are adjusted or created in Demantra's Promotion Trade Planning. A demand plan and a supply plan are generated.
6. Value Chain Planning is used to balance constrained supply and unconstrained demand by evaluating costs and constraints. A feasible solution is returned to Demantra S&OP
7. Additional capacity planning is performed in Demantra S&OP The resulting consensus forecast (currency and units) and cost of goods sold (currency) are exported from Demantra to Hyperion weekly (or another regular frequency).
8. Revenue and profitability targets are monitored in Hyperion.

For more information about the S&OP - Hyperion Planning Integration, please refer to the following documents:

- *Oracle Application Integration Architecture 2.4: Installation and Upgrade Guide*
- *Oracle Demantra Sales and Operations Planning Integration to Hyperion Planning 2.4 – Implementation Guide*

Integration with Demand Signal Repository

This chapter describes how Demantra integrates with Demand Signal Repository.

This chapter covers the following topics:

- Solution Overview
- Importing DSR Data into Demantra
- Running Multiple Forecasts

Solution Overview

The integration with DSR requires a number of new Levels, Series, Workflows, Profiles, and Worksheets (a full listing of these objects is provided in the section *Objects for DSR Integration* below).

Business Process

The DSR integration provides the following benefits:

- Synchronizes DSR data for item and geography.
- Provides a data extract from the DSR at the required level using Demantra's internal identifiers, e.g. Demantra's item and geography members at Demantra's lowest level.
- Provides a data extract from the DSR at the required level using manufacturer's internal identifiers, e.g. manufacturer's master data for item and geography at the lowest level.
- Provides a Demantra import integration profile and corresponding workflow for loading DSR output data into Demantra's internal tables.

Overview of Procedures

Here is a summary of the DSR procedures available within Demantra:

1. Import DSR data from the staging tables by running the *DSR Sales Data Download* workflow.
2. Create multiple forecasts by running the *DSR POS Forecast Calculation* workflow.

Objects for DSR Integration

The following objects are utilized in the Demantra integration with Demand Signal Repository. Detailed specifications for these objects are given at the end of this chapter.

Series

- Retailer POS Sales by Site
- Retailer POS Forecast
- Retailer DC/WHSE Forecast
- Retailer Stores OnHand Total by Site
- Retailer Store Orders Total by Site
- Retailer DC/WHSE OnHand
- Retailer DC/WHSE Withdrawals
- Retailer DC/WHSE Intransit
- POS Override
- Baseline POS Forecast
- POS Forecast Override
- Final POS Forecast
- POS Simulation

Series Groups

- DSR

Workflows

- DSR Sales Data Download
- DSR POS Forecast Calculation

Workflow Groups

- DSR

Engine Profiles

- DSR POS Forecast
- DSR POS Simulation
- DSR POS Forecast Calculation

Integration Profiles

- DSR Sales Import

Importing DSR Data into Demantra

Prerequisites

- ☐ Before running the *DSR Sales Data Download* workflow, data must already be exported from DSR into the Demantra staging tables. See the *DSR Integration Guide* for that procedure.

Running the DSR Sales Data Download workflow:

For running a workflow see the *Starting Workflow Instances* section of the *Managing Workflows* chapter of the *Demantra Implementation Guide*.

1. Log into the Workflow Manager.
2. Locate the DSR Sales Data Download workflow in the table and click the corresponding Start button on that row. This starts a new workflow instance that transfers exported DSR data from the staging area into the Demantra production area of the database.

Running Multiple Forecasts

The *DSR POS Forecast Calculation* workflow runs the forecast engine using the *DSR POS Forecast* engine profile to generate multiple forecasts.

Running the DSR POS Forecast Calculation workflow:

For running a workflow see the *Starting Workflow Instances* section of the *Managing Workflows* chapter of the *Demantra Implementation Guide*.

1. Log into the Workflow Manager.
2. Locate the DSR POS Forecast Calculation workflow in the table and click the

corresponding Start button on that row. This starts a new workflow instance that generates multiple forecasts.

Part 3

Basic Configuration

Getting Started with the Configuration Tools

This chapter introduces the primary tools you use to configure Demantra, namely, Business Modeler and Workflow Manager.

This chapter covers the following topics:

- About Demantra Configuration Tools
- Illegal Characters in Demantra
- Desktop and Business Modeler Automatic Install
- Logging Onto the Business Modeler
- Refreshing the Data in the Business Modeler
- Working with Lists
- Configuring Parameters
- Making Changes Available to Users
- Quitting Business Modeler
- Logging into the Workflow Manager
- Quitting the Workflow Manager
- Setting Browser Locale

About Demantra Configuration Tools

In an implementation, you typically perform the following tasks.

Task	Tool used...	For details, see...
Define the basic levels and create a load script to load the data for them.*	Business Modeler	"Using the Data Model Wizard"
Add more levels if needed.	Business Modeler	"Configuring Levels"
Define series and series groups.	Business Modeler	"Configuring Series and Series Groups"
Configure units of measure, financial indexes, and conversion rates for use in series and worksheets.	Business Modeler	"Configuring Units, Indexes, and Update-Lock Expressions"
Define integration.	Business Modeler	"Series and Level Integration"
Load supplementary data.	Business Modeler	"Importing Supplementary"
Create workflows.	Workflow Manager	"Creating Workflows"
Define methods.	Business Modeler	"Configuring Methods"
Define components to subdivide the data.	Business Modeler (In previous releases, you used a separate utility, the Business Application Modeler.)	"Creating or Modifying a Component"
Configure the engine: <ul style="list-style-type: none"> • Set up causal factors. • Set up the forecast tree. • For Promotion Effectiveness: Configure the influence groups and influence ranges that affect how the engine works. • Tune the Analytical Engine. 	Business Modeler	"Configuring and Running the Analytical Engine"

Task	Tool used...	For details, see...
Run the Analytical Engine and check the results.	Engine Administrator	
Write database procedures to maintain data as needed.	Text editor	Outside the scope of this documentation
Create additional users for the components, as needed.	Business Modeler (In previous releases, you used a separate utility, the Security Manager.)	"Managing Security"
Create user groups for collaboration.		
Define security for menu options.	Collaborator Workbench Administrator	
Define worksheets.	Worksheet designer	Oracle Demantra Demand Management User's Guide or other manual
Optionally customize Collaborator Workbench.	Text editor, graphics editor	"Customizing Demantra Web Pages"
<p>* In some cases, you use database setup scripts instead of the Business Modeler. See the following chapters:</p> <ul style="list-style-type: none"> • "Configuring Promotion Effectiveness" • "Configuring DSM" • "Configuring Promotion Optimization for PTP" 		

Illegal Characters in Demantra

Within Demantra, do not use the following special characters:

- Single quote (')
- Double quote (")
- Ampersand (&)

If you use these characters, unexpected results may occur.

Desktop and Business Modeler Automatic Install

When a user accesses the Power Builder Desktop and Business Modeler tools for the first time, or when the version has changed, the Application Initiator installs or updates these tools automatically from a common location provided by the Demantra Application server.

This topic describes:

- Automatic (also called silent) install of the Power Builder Desktop (Desktop) and the Business Modeler
- Automatic login to the Desktop and the Business Modeler

Automated Process

1. The User logs into the Collaborator Workbench, and then selects Business Modeler, Member Management, or Chaining Management from the Tools and Applications Menu.



2. The Application Initiator checks the user roll. If the user roll matches one of the following system rolls, then the process moves on to the next step.
 - **System Manager**
 - **Supervisor**

If the user roll does not match one of the listed system rolls, the user is notified with the message: "You don't have the correct permissions to execute this program." The process terminates.

3. The Application Initiator compares the Application Server Version with the version of the tools installed on the client. If needed, based on the user roll, the Application Initiator downloads and installs the appropriate files:
 - **System Manager**
Desktop, Business Modeler, required DLL's, and Oracle Instant Client.
 - **Supervisor**

Desktop, required DLL's, and Oracle Instant Client.

4. The Application Initiator stores the installed tools version in the client's environment to be used again whenever needed for version validation.
5. Depending on the selection made in step 1, the Application Initiator executes the Desktop or Business Modeler with the URL to the application server:
 - Business Modeler. The Business Modeler application is started and the user is logged in automatically.
 - Member Management. The Desktop starts and the user is logged in automatically. The Member Management window opens.
 - Chaining Management. The Desktop starts and the user is logged in automatically. The Chaining Management window opens.

Automatic or Silent Install Requirements

The automatic or silent install installs the following tools:

- Business Modeler and Supporting DLL's
- Desktop for Chaining and Members Management
- Security Management (for DS.INI and DLL's)
- Oracle 10g Instant Client
- Dependencies, such as DLL's and environment variables

All the above applications should be packed into an Install Anywhere package, and then placed on the server. The install package should reside in a folder called "tools" under the context of the Demantra Web-Application. This location should be accessible using a URL:

`http://{host}:{port}/{context}/tools/desktopInstaller`

This URL should be used for the automatic install, and be available to users who need to install the tools manually.

TNS Configuration

A TNS configuration is required for the proper function of web initiation of Desktop applications (Business Modeler, Chaining, Members Management). This configuration involves creating an appropriate TNS entry in the TNSNAMES.ORA file. This file usually resides under the default HOME of an "oracle" installation folder (for example C:\oracle\ora9\network\ADMIN).

The new entry should be like this sample:

```
MY.SERVER.NAME =  
(DESCRIPTION=(ADDRESS=( PROTOCOL=TCP ) ( HOST= MY.SERVER.NAME ) (   
PORT=1234 ))  
  ( CONNECT_DATA=  
    ( SID=orcl10 )  
  )  
)
```

MY.SERVER.NAME should be the same as the corresponding value of "ServerName" in the APS_PARAMS table.

Troubleshooting tip:

If, after install, the desktop applications do not initiate when the menu items are selected, please add to the PATH parameter (My Computer -> properties (right click menu) -> Advanced (tab) -> Environment Variables -> PATH) the path to the folder where Oracle Demantra applications are installed (C:\Program Files\Oracle Demantra Spectrum\Desktop).

Logging Onto the Business Modeler

Prerequisites

- ☐ Before starting the Business Modeler, make sure that the database is running.

To log onto the Business Modeler:

1. On the Start menu, click Programs.
2. Click Demantra > Demantra Spectrum release > Business Modeler.
A login window appears.
3. Enter your user name and password.
4. Click Login.

Access to Business Modeler functions:

Depending on your user name, you may not have access to all the functions of the Business Modeler.

Refreshing the Data in the Business Modeler

To refresh the display of data in the Business Modeler:

1. Click the Refresh button in the tool bar. Or click File > Refresh.

Working with Lists

Within the Business Modeler, you use screens that present two lists of elements, where you specify your selections. To make selections, you move elements from the left list to the right list.

To move elements from one list to the other:

1. To move a single element, double-click it. Or click it and drag it to the other list.
2. To move all elements, right-click and then choose Select All Rows. Then hold down the Shift key and drag from one list to the other.
3. To move several adjacent elements, click the first element, press Shift and click the last element. Then hold down the Shift key and drag from one list to the other.
4. To move several elements that are not adjacent, press Ctrl and click each element you want. Then hold down the Shift key and drag from one list to the other.

Configuring Parameters

During the implementation process, you often have to set values for parameters. You use the Business Modeler to configure almost all parameters, and the changes are recorded in the audit trail.

See also: Non-Engine Parameters.

To view and edit parameters in the Business Modeler:

1. Log onto the Business Modeler as described in "Logging onto the Business Modeler."
2. Click Parameters > System Parameters.

The System Parameters dialog box appears. This dialog box includes the following tabs:

Tab	Typical parameters on this tab
Worksheet	Maximum number of members on which a user can work at the same time Flag that switches on debug mode
System	Base time unit

Tab	Typical parameters on this tab
Database	Database version Initial sizes of tablespaces
Engine	Maximum number of forecasts that are kept Parameters that control the proportion mechanism
Application Server	Date format Server name

- Find the parameter of interest. The dialog box provides find, sort, and filter capabilities to help you with this.
- To change the value of the parameter, click the Value field for that parameter.
- Type the new value or select an allowed value from the drop-down menu.
- Click Save to save your changes.
- Click Close.

See also

Making Changes Available to Users

When you make changes in the Business Modeler, those changes are not necessarily available to users immediately.

To make changes available to users:

- Save your changes within Business Modeler. To do so, click File > Save.
- Make sure that the changes are included in the components in which the users work. See "Creating or Modifying a Component".
- Load the changes into the system. How you do this depends on which user interfaces the users are working with:
 - For the Web-based products, stop and restart the Web server. Information on this is beyond the scope of the Oracle documentation.

- For Demand Planner or Demand Replenisher, either restart the user interface or use the System menu to reload the configuration.
4. If you have created a new series, make sure this series is included in the appropriate worksheets. See the Oracle Demantra Demand Management User's Guide or other user guides.

Note: These steps are necessary when you make changes to series, levels, level attributes, units, indexes, level methods, integration interfaces, components, or users.

Quitting Business Modeler

To quit Business Modeler:

1. Click File > Exit. Or click the Exit button.

Logging into the Workflow Manager

A user must belong to a group that is flagged as a 'Collaboration Group' in order to access the Workflow Manager. For more information on User Groups see: Creating or Modifying a User Group, page 61-16

To log into the Workflow Manager:

1. Before logging onto the Workflow Manager, make sure that the database and the Web server are both running.

Quitting the Workflow Manager

To Quit the Workflow Manager:

1. Click Logout in the upper right of the screen.

Setting Browser Locale

If your users use Microsoft Internet Explorer, Oracle requires that you or they set the browser locale. The applet locale is based on the browser locale.

Having browser locale set, Oracle Demantra can manage both the portal and applet. With client machine locale alone, Oracle Demantra can manage applets, but cannot determine the locale for portal parts.

If an applet starts as standalone, Oracle Demantra uses the client machine locale.

The instructions for setting the browser locale are in Oracle Demantra User's Guide > First Time Login.

Database Tools

The Business Modeler provides a simple user interface for creating and modifying database tables that is useful during implementation.

This chapter covers the following topics:

- Creating a Table
- Modifying a Table
- Recompiling the Database Functions and Procedures
- Viewing the Procedure Error Log
- Wrapping Database Procedures
- Cleaning Up Temporary Tables
- Monitoring Oracle Sessions
- Index Adviser

Creating a Table

To create a table:

1. Log onto the Business Modeler as described in "Logging onto the Business Modeler".
2. Click Tools > Database > Create Table.
3. In the Table Name field, type a name for the new table.
4. In the Column Name field, enter the name of a column in the new table.
5. In the Column Type field, enter the column type (one of the standard data types supported by the database).

6. If the selected column type contains a string of characters, select a width in the Width field.
7. In the Decimal field (if field is numeric), specify the number of positions that should appear after the decimal point.
8. In the Nulls field, specify whether this field is allowed to contain a null value.
9. If this field should be a primary key for this table, select the Primary Key check box.
10. When you have completed creating the field, click Add or press Tab to move to the next field. Repeat this process for any field you want to create for the table.

Note: To delete a field, select it and then click Delete. The field cannot be deleted after the table has been created.

11. When you have completed creating all the fields for the new table, click Create to create the table, and then click Close.

Modifying a Table

To modify a table:

1. Click Tools > Database > Alter Table.
The Select Table dialog box appears
2. Double-click the name of the table you want to modify.
The Alter Table dialog box appears. Here, each row displays information about a field (column) of the table.
You can modify the white columns: Width; Dec (number of decimal points), and Primary Key.
3. To add a column, click Add and enter the required information, as described in "To create a table".
4. To save modifications to the table, click Alter.
5. To close the dialog box, click Close.

To delete a column:

1. Select the column.
2. Click Delete.

Note: A column cannot be deleted if you have modified it.

Recompiling the Database Functions and Procedures

To recompile the database functions and procedures:

1. Log onto the Business Modeler as described in "Logging onto the Business Modeler."
2. Click Tools > Recompile.
The database procedures and functions compilation dialog box appears.
3. Click Compile to compile the SQL procedures.
4. (Optional) Click Show Error to view possible expression errors. Click Print to print the errors.
5. Click Close to close the dialog box.

Viewing the Procedure Error Log

The procedure error log enables you to see error messages produced by procedures. For each error message, it displays the date and time of the message, the procedure name, and the text content of the message.

To display the procedure error log:

1. Log onto the Business Modeler as described in "Logging onto the Business Modeler."
2. Click Tools > Procedure Error Log.
The Procedure Error Log screen appears. The most recent errors are displayed first.

To display an error message:

1. Select an error from the procedure name and date list.
The error message appears in the error message pane.

To sort the list:

1. Right-click the list and then select Sort.

The sort screen appears.

2. To determine how the items will be sorted, click one or more required boxes in Columns Available for Sorting, and drag them to Sort Columns, or double-click the required boxes in Columns Available for Sorting.

To reverse this process drag a box in the opposite direction, or double-click a box in Sort Columns.

3. To specify an ascending sort order, make sure Ascending is checked. For a descending order, clear the box.
4. Click OK.

To find an item in the list:

1. Right-click the list and then select Find.
2. In the Find where box, select the type of element to find.
3. In the Find what box, type the text you want to find.
4. In the Search box, select Up or Down to specify the direction of the search.
5. (Optional) Select one or more of the check boxes:
 - Whole Word: Search for the exact match of a word.
 - Match Case: Search for the exact match of a word (case sensitive).
 - On Line Search: For immediate search results (even if only part of a word has been entered in the Find what box).
6. Click Find Next to begin (or continue) searching.

To filter the list:

1. Right-click the list and then select Filter.

The filter screen appears.

2. Select the required Column, Operator and Value.
3. (Optional) To add additional criteria, select the required Logical operator and click Add.
4. Click OK.

To copy the error message to the clipboard:

1. Click Copy.

To clear (truncate) the list:

1. Click Truncate.

A warning message appears.

Caution: The truncate action cannot be reversed.

2. Click Yes to confirm the action.

To refresh the list from the database:

1. Click Refresh.

To close the Procedure Error Log screen:

1. Click Close.

Wrapping Database Procedures

So that you can more easily locate and resolve problems, the database procedures are provided in unwrapped form. When you go live, it is required that you wrap the procedures and compile them in wrap mode into the database.

To wrap the database procedures:

1. Search the Demantra installation directory for a file called wrap30.bat. The location depends on which database you are using.
2. Create a directory, for example, wrap.
3. Inside this folder, create two subdirectories called, for example, source and target.
4. Copy this file into the folder:
5. Copy the database procedures into the source folder.
6. From the command line, execute the following command:

`wrap30 source_dir target_dir`

For example: wrap30 source target

After the execution is finished, the wrapped file will be in the target folder.

7. In the database, replace the unwrapped procedures with the new wrapped ones.

Cleaning Up Temporary Tables

Demantra provides an option for deleting its temporary tables.

To clean up tables:

1. Log onto the Business Modeler as described in "Logging onto the Business Modeler."
 2. Click Tools > Maintain > Tables Cleanup.
The Tables Clean Up dialog box appears, listing the Demantra temporary tables.
 3. Click the appropriate check boxes for the tables you want to delete, and then click Delete.

Monitoring Oracle Sessions

The Oracle Sessions Monitor enables you to monitor Oracle sessions and their status.

To monitor the Oracle sessions:

1. Log into the Business Modeler.
2. Click Tools > Oracle Sessions Monitor.
The Oracle Sessions Monitor window appears.
3. Here, do any of the following:
 - To sort or filter sessions, right-click and then select the appropriate command from the pop-up menu.
 - To select all sessions, right-click and then click Select All.
 - To delete a session, right-click the session and then click Delete.
 - To terminate a session, select a session, and then click Kill a Session.

Index Adviser

The Index Adviser is a tool that automatically analyzes a Demantra database schema

and recommends what indexes should be added in order to optimize the performance of the Worksheets Run and Filter Executions (in the Worksheet wizard).

The following procedure shows an example of how the tool could be used.

1. Access the program in your browser from the following url:

`http://localhost:8081/demantra/admin/indexAdvisor.jsp`

2. Log in (e.g. as dm/dm)
3. The Index Adviser screen appears:

The screenshot shows the 'Index Adviser' web application. It has a title bar 'Index Adviser' in blue. Below it, there are three main sections: 'Worksheets', 'Users', and 'Output Directory'. Each section has a checkbox for 'All' (e.g., 'All Worksheets'), a text input field with a red asterisk indicating it's mandatory, and a 'Get List' button. The 'Worksheets' section has a hint: 'A comma-separated list of worksheets ID-s.' The 'Users' section has a hint: 'A comma-separated list of users ID-s.' The 'Output Directory' section has a 'Directory:' label, a text input field containing 'http://localhost:8081/demantra/indexAdvisor', and a 'Browse...' button. At the bottom, there is a red asterisk indicating mandatory fields and an 'Execute' button.

4. Click 'Get List' in the 'Worksheets' section to display all worksheets.

The screenshot shows a table with three columns: 'Selected', 'Worksheet ID', and 'Worksheet Name'. There are five rows of data. Above the table, there are three buttons: 'Select All', 'Unselect All', and 'Invert'. The table data is as follows:

Selected	Worksheet ID	Worksheet Name
<input type="checkbox"/>	12183	Baseline: Brand (Monthly)
<input type="checkbox"/>	12181	Baseline: Brand (Weekly)
<input type="checkbox"/>	12405	Baseline: Detail
<input type="checkbox"/>	12365	Baseline: Overview
<input type="checkbox"/>	12178	Baseline: SRTT (Monthly)

5. From the Index Adviser screen, click 'Get List' in the 'Users' section to display all users below the Index Adviser screen.

1

Selected	User ID	User Name
<input type="checkbox"/>	11	rtsop
<input type="checkbox"/>	387	dm
<input type="checkbox"/>	389	ptp1
<input type="checkbox"/>	391	rp
<input type="checkbox"/>	399	tsop1

6. Select one or more users and worksheets.
7. 7. After selecting a set of Users and Worksheets, their corresponding UserIDs and WorksheetIDs appear in the Index Advisor screen, as shown below.

Index Advisor

Worksheets

☐ All Worksheets

Worksheets*:

A comma-separated list of worksheets ID-s.

Users

☐ All Users

Users*:

A comma-separated list of users ID-s.

Output Directory

Directory:

The output directory on the server where files be stored at.

* Mandatory fields.

8. Click Execute. The schema is analyzed and the following Results page is displayed. This page shows Filter Indexes and Worksheet Indexes.

Back Create Script

Index Advisor Results

Select All Unselect All Invert

Filter Indexes				
Selected	Name	Usability	Affected Users	Affected Works
<input type="checkbox"/>	Index #1	4	sop, Sop1, Strategic1, Marketing1	"Consensus Waterfall Analysis Pro
<input type="checkbox"/>	Index #2	1	rt sop	"Baseline: Brand (Weekly)"
<input type="checkbox"/>	Index #3	1	rt sop	"Baseline: Overview"

Worksheet Indexes				
Selected	Name	Usability	Affected Users	Affected Works
<input type="checkbox"/>	Index #1	4	sop, Sop1, Strategic1, Marketing1	"Consolidated Plan Product Catego
<input type="checkbox"/>	Index #2	4	sop, Sop1, Strategic1, Marketing1	"Consolidated Plan Product Catego
<input type="checkbox"/>	Index #3	4	sop, Sop1, Strategic1, Marketing1	"Consensus Waterfall Analysis Pro
<input type="checkbox"/>	Index #4	4	sop, Sop1, Strategic1, Marketing1	"Consolidated Plan Product Catego
<input type="checkbox"/>	Index #5	4	sop, Sop1, Strategic1, Marketing1	"Consolidated Plan Product Catego
			sop, Sop1, Strategic1, Marketing1	"Consolidated Plan Product Catego

- Check at least one index from the Filters (Users) or Worksheet Indexes, and click Create Script. An SQL script is generated for adding indexes to the schema. From this page click Open/Download to save the result to a file or to open an editor for viewing and modifying the script.

Back Open/Download

Filter/WS Index's Scripts

```
-- The script(s) for the creation of Filter Index #1
CREATE UNIQUE INDEX [FILTER_INDEX_1]_IDX ON ITEMS
(IS_LINKED, ITEM_ID, T_EP_EBS_PROD_CAT_EP_ID)
TABLESPACE [TABLESPACE]
COMPUTE STATISTICS;

CREATE INDEX [FILTER_INDEX_1]_IDX ON MDP_MATRIX
(T_EP_EBS_PROD_CAT_EP_ID, T_EP_ORGANIZATION_EP_ID)
TABLESPACE [TABLESPACE]
COMPUTE STATISTICS;

-- The script(s) for the creation of Filter Index #2
CREATE UNIQUE INDEX [FILTER_INDEX_2]_IDX ON LOCATION
(IS_LINKED, LOCATION_ID, T_EP_LS6_EP_ID)
TABLESPACE [TABLESPACE]
COMPUTE STATISTICS;

-- The script(s) for the creation of Filter Index #3
CREATE UNIQUE INDEX [FILTER_INDEX_3]_IDX ON ITEMS
(IS_LINKED, ITEM_ID, T_EP_I_ATT_1_EP_ID)
TABLESPACE [TABLESPACE]
COMPUTE STATISTICS;

-- The syntax for the creation of WS Index #4
CREATE UNIQUE INDEX [WS_INDEX_4]_IDX ON SUPPLY_PLAN_MATRIX
(ITEM_ID, LOCATION_ID, SUPPLY_PLAN_ID, FROM_DATE, UNTIL_DATE, LAST_UPDATE_DATE, SUPPLY_PLAN_LUD)
```

The DBA can refer to the report to manually create all or a sub-set of the recommended indexes in the database.

Using the Data Model Wizard

This chapter describes how to use the Data Model Wizard.

This chapter covers the following topics:

- About the Data Model Wizard
- Before Using the Data Model Wizard
- Getting Started with the Data Model Wizard
- Describing the Staging Tables
- Specifying the Structure of the Staging Tables
- Joining Multiple Source Files or Tables
- Defining the Minimal Required Data Model
- Declaring the Sales Date
- Declaring the Sales Quantity
- Defining an Item Level and a Location Level
- Saving the Data Model
- Defining Additional Elements
- Declaring the Unit Price
- Defining a Unit of Measure
- Adding Higher Levels
- Adding Level Attributes
- Defining Other Values
- Impacts of Changing Levels, Series, and Units
- Navigating the Data Model
- Building the Data Model
- Loading the Data into the Data Model

- Manipulating Existing Data Models

About the Data Model Wizard

The Data Model Wizard helps you perform the following related tasks:

- Describe the lowest item level and lowest location level in the system
- Specify which data fields to use as the sales date and quantity
- Specify the base time bucket in the system
- Define additional levels of type item, location, or combination
- Define series
- Define units of measure and other elements of the data model
- Create the EP_LOAD_MAIN procedure, which loads data into the data model from staging tables or from files, according to your choice.

Note: Some of these tasks can be performed elsewhere in the Business Modeler. The Data Model Wizard, however, is the only tool that lets you specify certain basics such as the sales date, and quantity, and base time bucket.

Before Using the Data Model Wizard

Before you use the Data Model Wizard, be sure to do the following:

- Read the "Levels" Chapter and make sure you understand the basic data requirements of Demantra.
- Obtain some sample data for items, locations, and sales. Make sure that this data contains all the codes needed to define the desired item and location levels. This data can be in the form of either text files or database tables:
 - If you use text files, the files must be either comma-delimited or tab-delimited.
 - If you use database tables, create these tables before you start the wizard. These tables must be in the same database user as the Oracle database.
- Carefully plan the levels you will configure.

Caution: Because the Data Model Wizard automatically removes

any levels that are not defined within the wizard, you generally use the wizard only at the start of the implementation process.

In some cases, you use database setup scripts instead of the Business Modeler. See "Configuring Promotion Effectiveness", "Configuring DSM", and "Configuring Promotion Optimization for PTP".

Getting Started with the Data Model Wizard

A new data model can be based on an existing template or on an empty template. Note that a data model cannot be converted into a template.

To start to build the data model:

1. Click Data Model > New Data Model.

The Create Data Model/Template screen appears.

2. Click the Data Model or Template button radio button.

3. Click OK.

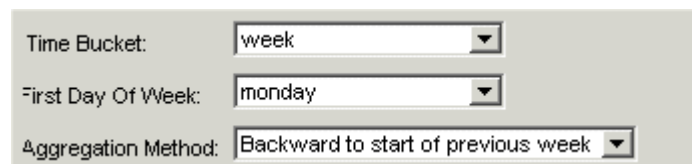
4. Click Next.

The wizard prompts you for basic information about this data model.

5. Specify a unique name and an optional description.

6. Click Next.

The wizard prompts you to specify the time unit for this data model.



The screenshot shows a configuration window with three rows of dropdown menus. The first row is labeled 'Time Bucket:' and has 'week' selected. The second row is labeled 'First Day Of Week:' and has 'monday' selected. The third row is labeled 'Aggregation Method:' and has 'Backward to start of previous week' selected.

Time Bucket:	week
First Day Of Week:	monday
Aggregation Method:	Backward to start of previous week

7. For Time Bucket, select one of the following base time units: Day, Week, or Month.
8. If you selected Weekly, fill in the other fields as follows:

First Day of Week	<p>Select the day of the week to use as the first day of any week. For example, if you select Monday, then all weekly data will be aggregated and displayed on the Monday of each week.</p> <p>(Monthly time units are always aggregated on the first day of the month).</p>
Aggregation Method	<p>Select how to aggregate data in time, either backward or forward. Oracle recommends that you select Backward to start of previous week, so that the data will be aggregated to the beginning of the previous week. For example, all the sales records of the week June 07 to June 13 will be aggregated to June 07.</p> <p>Monthly time units are always aggregated backwards to the first of every month. For example, sales for June will be aggregated to June 01.</p>

Note: The settings you choose here also apply to integration. See "Series and Level Integration". Note that the Data Model Wizard is the only place where you can specify these settings.

9. Click Next.

The wizard next prompts you for information about the source data. See "Describing the Staging Tables".

Describing the Staging Tables

If your sources are text files, the Data Model Wizard helps you map them into staging tables. If your sources are database tables, your tables are the staging tables. In either case, the wizard helps you describe the contents of the staging tables so that you can build a model on them.

To describe the staging tables:

1. In the Select Source Combination dialog box, you specify the number of source tables or files that you will use, as follows:

Single Table	Contains sales, items, and locations in a single file.
Two Tables	Contains sales and locations in the first file and items in the second file.
Three Tables	Contains sales, locations, and items in three separate files.

2. Click Next.

The Data Model Wizard now displays the Tables or Text Files as Sources dialog box.

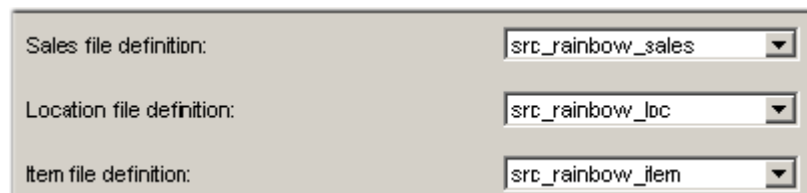
3. Specify whether the source data is in tables or text files:

Oracle Tables	Click if the data is in database tables.
Text Files	Click if the data is in text files.

4. Click Next.

5. The next step depends on whether the data is in the database or in text files:

- If the source data is in the database, the wizard now displays the Choose User Defined Source Tables dialog box. This dialog box varies, depending on the number of source tables you specified. The tables shown in drop-down lists depend on what you have previously loaded into the Demantra database.



Sales file definition:	src_rainbow_sales
Location file definition:	src_rainbow_loc
Item file definition:	src_rainbow_item

For example, if your source is in three tables, fill in the fields as follows and then click Next:

Sales file definition	Select the file where the sales history data is stored.
-----------------------	---------------------------------------------------------

Location file definition	Select the file where the location data is stored.
Item file name	Select the table where the item data is stored.

- If the source data is in files, the wizard now prompts you for information about the format of those files, and how to map the data into database staging tables. See "Specifying the Structure of the Staging Tables".
6. The next step depends on whether you specified multiple sources:
- If you specified multiple tables or files, you must specify how to join them. See "Joining Multiple Source Files or Tables".
 - If you specified only a single table or file, you are ready to define the data model. See "Defining the Minimal Required Data Model".

Specifying the Structure of the Staging Tables

If you specified two or three tables or files, the wizard displays the Text File dialog box. Here, you describe each text file you are going to import, as well as the staging table that corresponds to that file. If you are using multiple source files, the wizard prompts you for information for each one, in a specific order.

To specify the structure of the staging tables:

1. Read the title bar of the dialog box to make sure you know which data you are mapping. For example, if the wizard is expecting item data, the title includes "item."
2. In the upper part of the dialog box, describe the source file that you are using for this data, as follows:

File Directory	The location of the files.
Log Path	Path and filename of the log file that will store any load errors.
Delimiter Type	Select the delimiter type from the drop-down list.

Date Format	Select the date format from the drop-down list. If this source file does not contain dates, this is optional.
File Name Format	Select more than one file through the use of a wildcard (*). For example, dcl_his*. * selects every file with the prefix dcl_his.
Load Option	Insert is the only option currently available.
Column Delimiter	Select the character used in this source file to delimit fields.
No of lines to skip from begin	If there is a header, this gives the number of lines to miss at the top of the table.

3. In the lower part of the screen, define the table structure of the table you are defining in the staging area. To add a new entry to this information, right-click and then click Add. Fill in the details of the table structure as follows:

Name	Field name
Type	Data type for this field; specify one of the data types supported by the database.
Width	Maximum width of field
Dec	Decimal point
Default	The default value of the field if it is null in the source data.
Null	Click yes to allow field value to be null.
From Position	Use if Fixed was selected as the Delimiter Type. This is the position in the source text file where the field starts.
To Position	Use if Fixed was selected as the Delimiter Type. This is the position in the source text file where the field ends.

Const	Constant column width. If selected, the From Position and To Position fields are disabled for editing.
-------	--------------------------------------------------------------------------------------------------------

4. When you are done describing this source file and its corresponding staging table, click Next. If you are using multiple source files, Business Modeler displays a similar dialog box where you specify the next source file.
5. The next step depends on whether you specified multiple sources:
 - If you specified multiple tables or files, you must specify how to join them. See "Joining Multiple Source Files or Tables".
 - If you specified only a single table or file, you are ready to define the data model. See "Defining the Minimal Required Data Model".

Note: This dialog box provides the following additional options:

- The Create Load Batch button creates a batch file that will load the table. This button is only present in certain installations of the Business Modeler.
- The Table Syntax button displays the SQL syntax used by the database and the Business Modeler to create the table in the staging area. Only experienced consultants should use this feature.
- If you are using an Oracle database, Demantra uses the SQL*Loader tool (from Oracle) to load data. The Click CTL Syntax button displays the control file used by SQL*Loader. Only experienced consultants should use this feature.

Joining Multiple Source Files or Tables

If you specified two or three tables or files, the wizard displays the Join Sources dialog box. By defining the joins, you implicitly specify the primary keys in the data.

To join the source data:

1. For the first table or file to be joined, drag a field name from that table or file (on the left) to a blank space in the empty join structure on the right.
2. For the other field or file to be joined, drag a field name from the other table or file

(on the left) to the remaining blank space in the join structure.

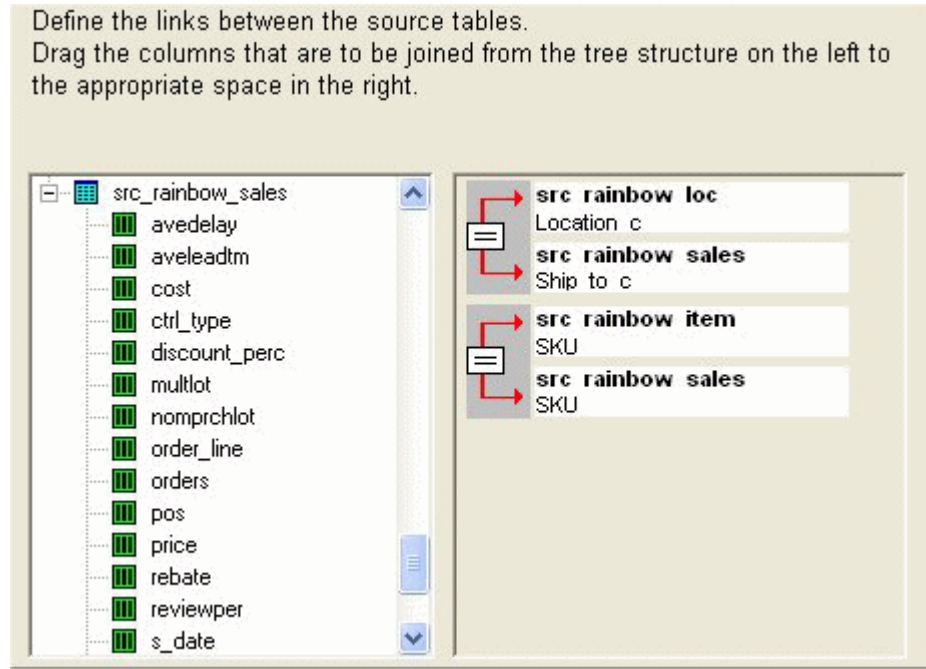
3. If you specified three tables or files as the source, you must create an additional join. To do so, right-click the right side of the dialog box and click New.

A new, empty join appears in the right side.

In the same way that you defined the previous join, create a join between the sales table and the remaining, non joined table.

Note: While you are defining these joins, it may be helpful to toggle between the table descriptions and the table names. To do so, click the toggle button Show Table Description or Show Table Name.

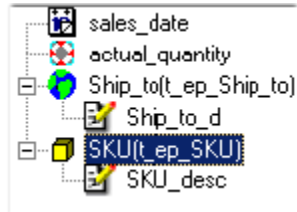
The result might look like this:



4. Click Next. The wizard now helps you define the data model. See "Defining the Minimal Required Data Model".

Defining the Minimal Required Data Model

This section describes how to define enough of the data model so that you can save it for future work. Here you will declare the sales date and actual quantity, and you will define an item level and a location level; you can define these four things in any order. The following shows an example.



In this stage, you specify how to use the fields in the staging tables, generally using each field in a level definition or in a series definition.

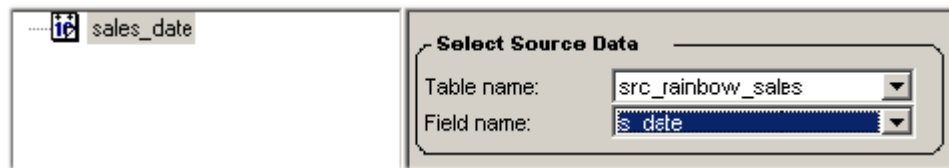
Note: Demantra imports only the fields that you specify how to use.

Declaring the Sales Date

To declare the sales date:

1. In the left box, right-click and then click Create Sales Date.
2. Select the table and field that contains the sales date for any given sale.

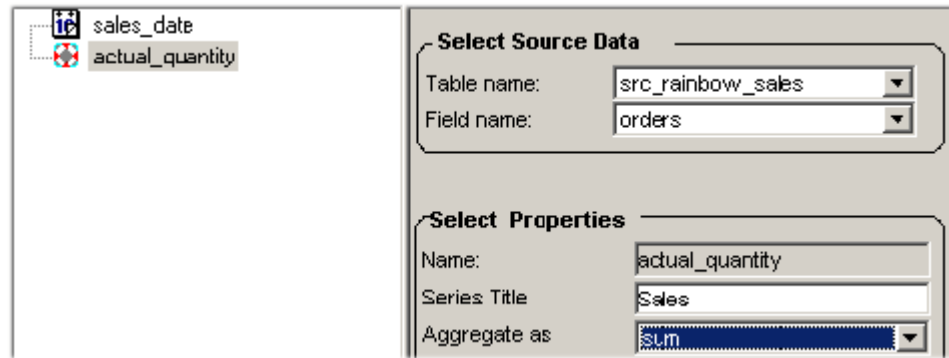
For example, in `src_rainbow_sales`, the `s_date` field might contain the sales date for each sales record.



Declaring the Sales Quantity

To declare the sales quantity:

1. In the left box, right-click and then click Create Sales Quantity.
2. Select the table and field that contains the item quantity for any given sale. For example, in `src_rainbow_sales`, the `orders` field might contain the sales quantity for each sales record.
3. In Series Title, specify a user-friendly name for this series.
4. For Aggregate as, select the function to use when aggregating multiple records.

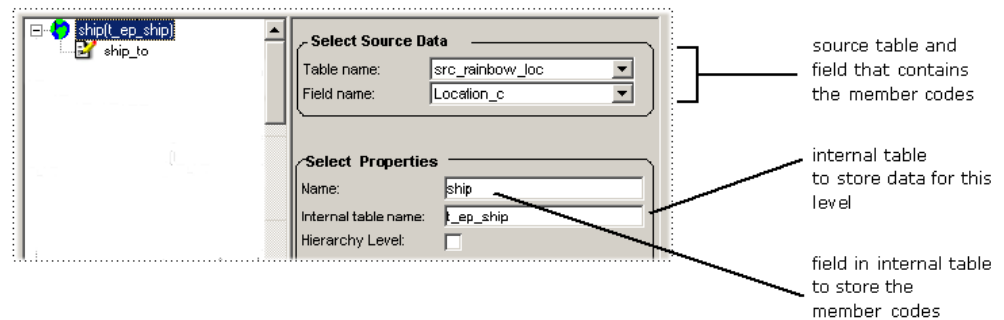


Defining an Item Level and a Location Level

The data model must include at least one item level and one location level. The first levels you define should be at the lowest desired level of aggregation. The lowest item level usually corresponds to specific SKUs, and the lowest location level usually corresponds to specific stores or ship-to locations.

To define an item or location level:

1. In the left box, right-click and then click Create Location Dimension or Create Item Dimension.
2. Then specify the level as follows:



1. In the Table Name field, click a source table from the drop-down list.
2. In the Field Name field, click the field within that source table that stores the unique codes that distinguish the members of this level.
3. In the Name field (in the Select Properties area), type a unique name of this level, for internal use.

Demantra automatically uses this as the basis for the Internal table name, which is the table in which Demantra will store information for each member of this

level. Within this table, the member codes will be stored in a field whose name is the Name that you specified.

3. In the left box, right click the level and then click Create Description.
4. Specify the level description as follows:

The screenshot shows the 'Data Model Wizard' interface. On the left, a tree view shows a level named 'ship_to'. The main panel is divided into two sections: 'Select Source Data' and 'Select Properties'. In 'Select Source Data', 'Table name' is set to 'src_rainbow_lbc' and 'Field name' is set to 'Location_d'. In 'Select Properties', 'Name' is 'ship_to', 'Level Title' is 'Ship to', 'Enable Level' is set to 'Enabled', and the checkbox 'Use autogenerated ID as a code for filter' is unchecked. Annotations with arrows point to these fields: 'src_rainbow_lbc' is labeled 'source table and field that contains the member descriptions', 'Location_d' is labeled 'field in internal table to store the member descriptions', 'ship_to' is labeled 'user-friendly level name', and 'Ship to' is labeled 'field in internal table to store the member descriptions'.

1. For Table Name, select the same table.
2. In Field Name, select the field that stores the descriptions of the members of this level.

If the source data does not include a specific field that you can use as the member descriptions, then just use the field that stores the member identifiers.
3. In Level Title, type the name of this level, as it should be shown in the user interfaces.
4. In Enabled Level, specify whether you want the level to be disabled. If the level is disabled, it is effectively removed from the data model. However, it can be reactivated if required later.
5. To use the internal, autogenerated ID as the code display field, click the check box Use auto generated ID as code for filter. Otherwise, Demantra uses the code field as the code display field.au

Saving the Data Model

After defining the minimum required data model, you can save all the work that you have done within the Data Model Wizard. You can return and continue work later.

To save the data model:

1. Click Save.
2. To exit the Data Model Wizard now, click OK. You can return later and continue your work.

Defining Additional Elements

This section describes how to define additional elements of the data model.

Declaring the Unit Price

The price per item is generally different at different locations and over time.

To declare the unit price:

1. In the left box, right-click and then click Create Item Price.
2. For Table Name and Field name, select the table and field that contains the unit price for an item, at a given location and time. For example, in `src_rainbow_sales`, the `orders` field might contain the sales quantity for each sales record.
3. For Unit Title, optionally modify the title, whose default is Price.
4. For Aggregate as, select the function to use when aggregating the price of multiple items.
5. If you selected `wavg` (weighted average), then specify a weight to use. For Dependent Field, select a field within the same table that Demantra should use as the weights.

The screenshot shows a software interface for defining a unit price. On the left, a tree view displays a folder icon, followed by 'sales_date', 'actual_quantity', and 'item_price'. On the right, there are two panels. The top panel, titled 'Select Source Data', contains 'Table name:' with a dropdown menu showing 'src_rainbow_sales' and 'Field name:' with a dropdown menu showing 'price'. The bottom panel, titled 'Select Properties', contains 'Name:' with a text box containing 'item_price', 'Unit Title' with a text box containing 'Price', 'Aggregate as' with a dropdown menu showing 'wavg', and 'Dependant Field' with a dropdown menu showing 'orders'.

Defining a Unit of Measure

When you define a unit, you specify a set of conversion factors for that unit. The conversion factors can be different for different items.

Note: There must be a one-to-one relationship between the unit values

and the members in the level where the unit is configured.

To define a unit:

1. In the left box, right-click and then click Create Unit.
2. For Table Name and Field name, select the table and field that contains the conversion factor for this unit, for each item. For example, in `src_rainbow_item`, the `pallet` field might contain the conversion factor for each item.

The conversion factor for a unit X should give the number of Xs per base unit. For example, for a given SKU if a pallet consists of 100 items, then `src_rainbow_item.pallet` should be 0.01 for that SKU.

3. For Aggregate as, select the function to use when aggregating the unit count of multiple sales. You usually use Sum.

The screenshot shows a software interface for defining a unit. On the left, a tree view contains the following items: `sales_date` (with a calendar icon), `actual_quantity` (with a red and blue icon), `item_price` (with a green coin icon), and `Pallet` (with a yellow pallet icon). On the right, a dialog box titled 'Create Unit' is open. It has two main sections. The first section, 'Select Source Data', contains two dropdown menus: 'Table name:' set to 'src_rainbow_item' and 'Field name:' set to 'pallet'. The second section, 'Select Properties', contains three text boxes and one dropdown menu: 'Name:' set to 'Pallet', 'Unit Title' set to 'Pallet', and 'Aggregate as' set to 'sum' (highlighted in blue).

See also

"Configuring Units"

Adding Higher Levels

Note: As you define levels, it is important to consider what sort of forecast tree you will need, as described in "Configuring the Forecast Tree"

To create a higher level:

1. Right-click the level to which you want to add a higher level. Note that any level can have multiple parent levels.
2. Click Create Relation.

3. Specify the following items:

Table Name	Select a source table from the drop-down list
Field Name	Select the field within that source table that stores the unique identifiers for members of this level.
Name	Specify a unique name of this level, for internal use.

As before, Demantra automatically creates a name for the internal table associated with this level.

4. Create a description for this dimension.

See also

"Creating a Level"

Adding Level Attributes

You can store additional level-specific information as attributes. Any number of attributes can be added to a level. Each attribute name must be unique within a given level. To view attributes of a member, the user can right-click the member within the Members Browser of a worksheet or in the Collaborator Workbench.

To create level attributes:

1. In the left box, right-click a level and select Create Attributes.
2. Complete the fields as follows:

Table name	Table in which attribute data is stored. This should be the source table for this level.
Field name	Field in table that stores this attribute.
Name	Unique internal name for this attribute.
Attribute Title	User-friendly name for this attribute.

See also

Defining Other Values

A value can be a causal factor or a data series.

To define a value:

1. In the left box, right-click and then click Create Value.
2. Select the value item, and then give the value a field name, a name, and a series title.

Impacts of Changing Levels, Series, and Units

This section describes the impact of adding, removing, or modifying levels, series, and units.

For more information about building or updating a model, see *Building the Data Model*, page 25-22.

Adding a Level

Consultants can add a 'Parentless' level only, in other words, they can add a parent level to an existing level or new branch only. Adding mid-tier levels is not an explicitly supported scenario. Levels can be renamed accordingly to reflect the insertion of a new level between existing levels. See "Creating a New Intermediate Level".

When a new level is created, the upgrade process:

- Creates default methods for this level including New, Edit, Delete, View
- Add this level to loading procedures
- Adds a line to the upgrade log file: 'Level XXX added to model'.

For more information about upgrading a model by adding parent levels, see *Creating a New Top Level*, page 14-71.

Removing a Level

When a level is removed, any parent levels are also removed. The upgrade script:

- Adds a line to the upgrade log file: 'Level XXX removed from model'
- Regarding series on a removed level:

- If the upgrade mechanism leaves the series in the model, in other words, the series is not deleted, the series is listed in the log file as needing reconfiguration. The upgrade log states:

```
Series YYY refers to level XXX. Please reconfigure.
```

This message appears after and indented from the 'Level XXX removed from model' message.

All worksheets, workflows, integration profiles, and methods referring to this series on the removed level remain intact. Any worksheet that refers to this series will open with all other data intact. The series on the removed level shows no data.
 - If the current upgrade mechanism deletes this series, the upgrade log states:

```
Series YYY referred to level XXX and was deleted.
```

This message appears after and indented from the 'Level XXX removed from model' message. References to this series on the removed level are deleted for all worksheets, workflows, integration profiles, and methods that refer to the series.
 - All worksheets, workflows, integration profiles, user-created methods, 'Open With' and Embedded worksheets that contain this level remain intact. Users may need to manually reconfigure these objects. For example, a worksheet may need to refer to a different level. Worksheets open for editing purposes.
- The level reference is replaced by the child level. For example, if we have an Item > Brand hierarchy and remove the Brand level, worksheet references to 'Brand' are replaced with references to the 'Item' level. The upgrade log lists all objects affected by removing this level.
- **Log entries:**

```
Worksheet ZZZ referred to level XXX. It now refers to child level  
YYY.  
Please reconfigure as necessary.  
Integration interface ZZZ referred to level XXX. It now refers to  
child level YYY.  
Please reconfigure as necessary.
```
 - All methods on this level are deleted, including default methods such as 'New' and 'Edit', and user-created methods. Any underlying workflows referenced in these methods are not deleted.
 - **Level table:**
 - If the upgrade mechanism deletes the table for the level, the Upgrade log states:

```
Table XXX for level YYY has been deleted.
```

- If the current upgrade mechanism does not delete the table for the level, the Upgrade log states:

```
Table XXX for level YYY has not been deleted. Please delete manually if not required.
```

- If the level was referenced in the forecast tree, a user warning appears in the Upgrade log file at the end of the upgrade:

```
Level XXX was referenced in the forecast tree. Please reconfigure forecast tree.
```

User security references to that level are removed and logged:

```
User/Group XXX reference to level YYY has been removed.
```

Modifying a Level

When a level is modified, the upgrade script:

- When adding an attribute – loading scripts are updated to include the new attribute. An attribute column is added to the level table. The consultant or user is responsible for manually modifying the source table to include the attribute column and values.
- When Deleting an attribute – loading scripts are updated to remove this attribute. An attribute column is deleted from the level table.
- Table references – specify what source column the level pulls data from. This has no impact on the existing model. The loading scripts are updated to reflect the new source column.

Adding a Series

All series that need to be part of the loading mechanism must be defined in the data model first in order to be reflected in the loading scripts. They can then be configured more fully by the consultant or user in 'Configure Series.'

Consultants can add a series to the data model. The series should be added to the loading procedure and related internal tables (sales_data, or mdp_matrix).

Removing a Series

The upgrade process does not remove an existing series from tables, for example, the computed_fields, table. Upgrade only removes the series from the data model, and then removes the series from the loading procedure. Therefore, removing a series will not invalidate objects because the series is still present. However, no data will be loaded to the removed series, so it becomes meaningless over time. Removing a series from the data model accomplishes the following:

- Add a message to the upgrade log:

```
Series XXX was removed from model. Please delete this series from
the series list using the 'Configure Series' option.
```

Oracle provides a reference list of objects affected by this deletion, such as worksheets, dependent series, and integration profiles that reference the series. Providing this output allows the user to go back and manually fix issues such as server expression references.

- Remove the series from the loading procedures.

The series is removed from internal tables, such as `sales_data`, `computed_fields`, and `mdp_matrix`, when the user deletes the series using the 'Configure Series' option. Deleting the series in this way removes references to the series in worksheets, workflows, integration profiles, and rolling profiles but *does not* remove or invalidate these objects. Some of these objects, such as worksheets, may require user re-configuration, but the worksheets will still open.

- If this series is a causal factor, display a warning message in the upgrade log:

```
Series XXX was a causal factor. Please reconfigure causal factors.
```

- Remove user security references to this series.

- Add the log message:

```
Series XXX is used in components A, B, C. Please reconfigure these
components.
```

Modifying a Series

If a series is modified, the upgrade process does the following:

- Modifications of the source field are reflected in the loading procedure.
- If the series aggregation has been modified, the loading procedure is updated. The new aggregation is not updated in computed fields, in case a custom client expression was configured.
- Add a log entry:

```
Series XXX has been modified. Please review its configuration in
'Configure Series' after completing the upgrade.
```

Adding a Unit

Adding a unit has the following impacts:

- Create reference in the level table
- Log message to state the need to add the Unit to the component and link the Unit to

the desired level.

Removing a Unit

Removing a unit has the following impacts:

- Remove the Unit from the model
- Delete the Unit from the model and the loading procedures
- Log message to state the need to delete the Unit in the Business Modeler

Modifying Time Aggregation

If the time aggregation is modified, this process typically occurs early in an implementation, prior to loading substantial data.

Caution: Any modification to the granularity of the model , whether more granular or less granular, will clear out all data. Reset all worksheets to the new model granularity.

See Setting and Modifying the Base Time Unit, page 7-5.

If a time granularity is modified, the upgrade process reflects the following changes:

- Modify loading procedure to respect new time aggregation
- Modify inputs table to respect new time buckets
- Add log message:
`Time aggregation has changed. Please review all worksheets and modify time definitions appropriately.`
- Set Integration profiles and worksheets to new model granularity, and leave start and end date time range as is.

Note: Consultant needs to review and re-configure worksheets, workflows, and integration profiles to fully respect the new granularity. The default settings are intended to ensure the worksheets open after the upgrade.

Modifying the First Day of Week

If the First Day of Week is modified, this process typically occurs early in an implementation, prior to loading substantial data.

Caution: Any modification to the First Day of the Week for a model will clear out all data.

- Loading procedures change to reflect new start of week
- Inputs table changes to reflect new start of week
- Leave start and end dates of the worksheet and integration profiles.

Note: The default settings are intended to ensure the worksheets open after the upgrade.

Moving Weekly Bucket Aggregation Forward or Backward

- Loading process changes to reflect new start of week
- Change parameter in sys_params table
- Data in the database is not cleared.

Navigating the Data Model

To navigate the data model:

1. Right-click the white background and then select one of the following options:

Expand All	Open the branches of a data model structure.
Collapse All	Collapse the branches of data model structure to its root level.
Refresh Tree	Update the tree display, with changes since previous refresh.

See also
"Configuring Levels"

Building the Data Model

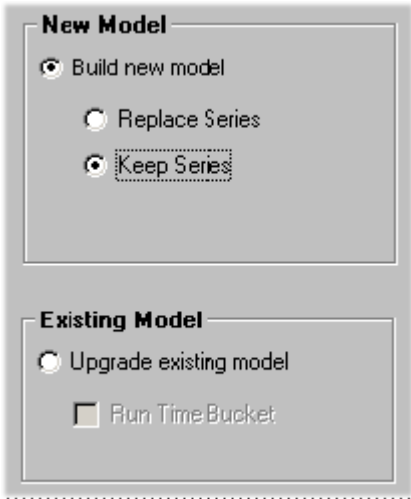
In the Finish Wizard dialog box, you build (or upgrade) the model itself. Here the Data Model Wizard creates all the internal structures that it needs for the data model you have specified.

To build the data model:

- 1. To Remove Illegal Characters, click Yes to check the source data and remove unwanted characters.
- 2. Click Finish or click Build Model.

If you click Finish, the Data Model Wizard closes.

If you click Build Model, the Build/Upgrade dialog box appears.



- 3. Now you can select whether to completely replace the existing data model or just modify it:
 - If you want to completely replace the existing data model, select one of the following options:

Replace Series

Click this to completely replace the existing series definitions.

Keep Series

Click this if you do not want to make any changes to the existing series. This option is suitable if you are in the process of working on the data model but do not want to spend the time updating the series definitions right now.

Caution: In either case, any previously existing levels are completely removed, and the new levels are initialized.

- Alternatively, if you are just modifying an existing data model, select Upgrade Existing Model. In this case, if you have made changes to the base time unit, select Run Time Bucket.

Upgrade Model does not modify the source tables in any way. Any additions, modifications and deletions to these tables would be done manually by consultants. Any level extensions that must be reflected in the loading procedures are managed by the Data Model. Changes made in Series and Level configuration screens are not supported or synchronized.

Note: There should be no conflicts with the data model definitions (file structures, file locations, and so on). Make sure that the customer's files include all the series and levels that you have configured.

If there are series or levels that appear in the Business Modeler but that are not included in the components structure, a warning will recommend that you manually remove these objects. A log file is created with a list of the objects. Otherwise, the system will automatically identify the series and levels of the selected components and truncate (make empty) the objects that do not participate in the selected components. Only then can the Upgrade Model work.

4. Click OK.

The process of building the data model begins. This may take a few minutes.

Business Modeler also creates the file load_data.bat. See Loading the Data into the Data Model, page 25-24.

Loading the Data into the Data Model

After completing the data model, you must load the data into it.

To load the data:

1. Run the file load_data.bat, which is in the Demantra/Desktop directory.

When the file is run, the script imports data from the source files into the staging area, and from there into the Demantra data model.

The load_data.bat file contains several procedures to help you to load data into the data model. The prepare_data procedure is empty by default, and can be edited to carry out procedures which precede the data loading

Caution: This should only be carried out by an experienced consultant. It is not recommended to edit the other procedures.

To check for errors:

1. After loading the files, check the following files for error messages:
 - For a single source table (Item+Location+Sales): SRC_sales_err
 - For two source tables (Item, Location + Sales): SRC_loc_err
 - For three source tables (Item, Location, Sales): SRC_item_err

Manipulating Existing Data Models

To open an existing data model or template:

1. Click Data Model > Open Data Model.

The Open Existing Data Model/Template screen appears.

2. Select Data Model or Template.

3. Select the button corresponding to the data model or template you want to modify and click OK.

The Start Wizard window appears. See "Getting Started with the Data Model Wizard".

To save a data model or template under a different name:

1. Click Save As (on the Data Model/Template window).

The Save As dialog box appears.

2. Type in the name of the data model or template.
3. Type in an optional description.
4. Click OK.

To delete a data model or template:

1. Select the Data Model/Template button.

2. Click Delete.

A warning window appears.

3. Click Yes to confirm the action.

To import or export a data model or template:

The import and export functions enable you to store a data model or template and share it with other users. Data models and templates are saved as database files with the suffix .dmw.

1. Select the Data Model/Template radio button.

2. Click Export.

A database file is saved in the current directory.

To import a data model or template:

1. Click Import.

A browser window appears.

2. Navigate to the required .dmw file, and then click OK.

Configuring Levels

This chapter describes how to configure levels with the Configure > Levels option.

This chapter covers the following topics:

- Before Configuring Levels
- Creating a Level
- Filtering a Level
- Configuring an Existing Level
- Adding Attributes to a Level
- Dropdown Security
- Filtering an Attribute Drop-down List
- Specifying Default Parents for a New Member
- Adding a Population Attribute to a General Level
- Creating a time aggregation
- Viewing the Members of a Level
- Removing Levels

Before Configuring Levels

Before you use Configure > Levels option, be sure to do the following:

- Read the "Levels" Chapter and make sure you understand the basic data requirements of Demantra.
- If you are using Promotion Effectiveness, DSM, or Promotion Optimization, use the Demantra database procedures to set up an initial set of levels for those products; see "Configuring Promotion Effectiveness", "Configuring DSM", and "Configuring Promotion Optimization for PTP".

- Load some sample data for items, locations, and sales, by using the batch script created by the Data Model Wizard. Make sure that this data contains all the codes needed to define the desired item and location levels.
- Carefully plan the levels you will configure.
- Use the Data Model Wizard as much as possible before using Configure > Levels.

Creating a Level

To create a new level:

1. Click Configuration > Configure Levels. Or click the Configure Levels button.

The system displays a screen showing the levels. The disabled levels are indicated by an X symbol.

2. Right-click the level button and then select New.

The first screen is General Properties.

3. Complete the fields in this screen as follows:

Title	Level name.
Type	Type of level. Note that Product is an item level.
Child Level	Select the level that should be the child of the one you are creating.
Create as attribute	Check this if you want to create a lookup level attribute for the child level of the current level. For example: the current level is level B that has a child level A. In this case, if you check Create as attribute, the wizard will create a lookup level attribute for level A. This field is disabled at the lowest level.

4. If you are creating a general level, the following fields are also required:

Icon Path	Path and filename for the GIF file that contains a graphic to represent this level in the desktop user interfaces. The path must be relative to the Demand Planner\Desktop directory, for example: bitmaps/location.bmp
Icon URL	Web address for the GIF file that contains a graphic to represent this level in the Web-based user interfaces.
Indicator URL	Web address for the GIF file that contains an indicator for this level. This option applies only to general levels that have no children. Worksheet tables use this graphic to indicate the combinations and times with which a member of this level is associated.

Note: For other kinds of levels, Demantra has default icons.

5. Click Next.

The Data Properties screen appears.

The screenshot shows the 'Data Properties' dialog box. It has the following fields and values:

- Table Name: t_ep_SKU
- Key Field: t_ep_SKU_EP_ID
- Code Display Field: SKU
- Description Field: SKU_D
- Code Field: SKU
- Extra From: (empty text box)
- Extra Join: (empty text box)
- Unlinked Level: ☐

6. In Table Name, specify the name of the table in which Demantra should store information related to this level. As soon as you enter this name, Business Modeler automatically populates the following four fields.

Key Field	Primary key of the table you have just created.
Code Display Field	Field containing the code display label for level members, as displayed in the filter window in the worksheet designer. This field accepts string values. Typically, you use one of the following: Field that stores the autogenerated ID for this level (same value as used in the Key Field) Field that stores the code for this level (same value used in the Code Field) Field that stores the description for this level (same value used in the Description Field).
Description Field	Field containing the description or pretty name for level members, as displayed in worksheets.
Code Field	Field containing the code for the level members.

7. If the level is to be unlinked, click Unlinked Level.

Unlinked levels are used only for a special kind of series aggregation within worksheets.

8. Click Next.

The General Attributes screen appears. If needed, add attributes as described in "Adding Attributes to a Level".

9. Click Next.

The Defaults screen appears. If needed, specify the default parents of any manually created member of this level. See "Specifying Default Parents for a New Member".

Filtering a Level

Most levels span all the sales data; any sales record is associated with exactly one member of each level. You can, however, create filtered levels. A filtered level contains a filtered subset of the sales data.

To create a filtered level, you add an SQL WHERE clause to filter the data. You can also join the underlying data to another table of your choice. Each level internally has an SQL query. Normally this query can refer only to fields in the following tables:

Level type	Table where code field is found
Item	Items
Location	Location
Combination	mdp_matrix
Time	Inputs

Specifying the "Extra From" Field for a Level

In rare cases, you may need to refer to data in other tables. In such a case, use the Extra From field. In this field, specify an optional list of additional tables (separated by commas) that contain data relevant to this level.

Specifying the "Extra Join" Field for a Level

If you need to filter the level, use the Extra Join field. Internally, the Extra Join field is added to the WHERE clause in the SQL query that retrieves data for this level.

The syntax of this field is as follows:

table.column operator other_table.other_column

Where, *operator* is a comparison operator that is one of the following:

=
<>
>
>=
<
<=

And *table.column* and *other_table.other_column* are key columns in the database.

Configuring an Existing Level

To configure an existing level:

1. Click Configuration > Configure Levels. Or click the Configure Levels button.

The system displays a screen showing the levels. The disabled levels are indicated by an X symbol.

2. Double-click the level or right-click the level and then select Open > General Properties.

The General Properties screen appears.

The screenshot shows the 'General Properties' dialog box for configuring a level. The fields are as follows:

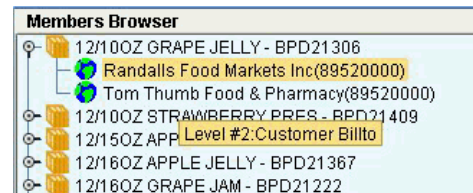
- Title:** A text box containing 'SKU'.
- Type:** A dropdown menu set to 'Product Level'.
- Icon Path:** A text box with a browse button (three dots) to its right. Below it, a note says: '(An example for a relative path: .\bitmaps\item.gif)'
- Icon Uri:** A text box.
- Indicator Uri:** A text box. Below it, a note says: '(An example for url: http://servername:8080/plan_ICO.gif)'
- Status:** A dropdown menu set to 'Enabled'.
- Child Level:** A dropdown menu set to '<None>'. To its right is a checkbox labeled 'Create as Attribute' which is currently unchecked.
- Order:** A spinner box.
- Hint Message:** A large text area.
- Display Width on Worksheet Table Axis:** A spinner box set to '20'.

3. Make edits as needed to the following fields:

Title	Level name.
Icon Path	Path and filename for the GIF file that contains a graphic to represent this level in the desktop user interfaces. The path must be relative to the Demand Planner\Desktop directory, for example: bitmaps/location.bmp

Icon URL	Web address for the GIF file that contains a graphic to represent this level in the Web-based user interfaces.
Indicator URL	<p>Web address for the GIF file that contains an indicator for this level.</p> <p>This option applies only to general levels that have no children. Worksheet tables use this graphic to indicate the combinations and times with which a member of this level is associated.</p>
Status	<p>Level status: Enabled or Disabled.</p> <p>Determines if the level is available to end users.</p>
Create as attribute	<p>Check this if you want to create a lookup level attribute for the child level of the current level. For example: the current level is level B that has a child level A. In this case, if you check Create as attribute, the wizard will create a lookup level attribute for level A.</p> <p>The new level is added immediately.</p> <p>This option is disabled at the lowest level.</p>
Order	This number determines where this level will be listed in filter and selection windows. (The lower the number, the closer the level appears to the top.)

Hint Message	Add or modify a message for the level. Demantra will display this message when the pointer hovers in the Members Browser in a worksheet, as follows:
--------------	------------------------------------------------------------------------------------------------------------------------------------------------------



Display Width on Worksheet Table Axis	Specify the default display width for this level when this level is included in a worksheet table.
---------------------------------------	----------------------------------------------------------------------------------------------------

Is Analytical	Applies only to general levels. Check this if you are creating a general level for use with Promotion Effectiveness. Enable this option only at the lowest level in the promotions hierarchy; Demantra can contain only one analytical level.
---------------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

4. Click Next.
The Data Properties screen appears.
5. Make edits as needed to the following fields:

Code Display Field	Field containing the code display label to use in filters. Typically, you use one of the following: <ul style="list-style-type: none"> • Field that stores the autogenerated ID for this level (the value given in the Key Field) • Field that stores the code for this level (the value given in the Code Field) • Field that stores the description for this level (the value given in the Description Field).
--------------------	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Extra From	See "Specifying Extra From for a Level".
------------	------------------------------------------

Extra Join	See "Specifying Extra Where (Extra Join) for a Level".
Unlinked Level	Select if the level is to be unlinked. Unlinked levels are used only for a special kind of series aggregation, not documented here.

6. Click Next.

The General Attributes screen appears. If needed, add attributes as described in "To add a new attribute to a level".

7. Click Finish.

Or, if you are configuring a general level, click Next. The Population Attributes screen appears; see "Adding a Population Attribute to a General Level".

Adding Attributes to a Level

Attributes provide additional information about a level. When you add an attribute to a level, Demantra automatically adds a new column to the internal table that it uses for that level.

To view attributes of a member, the user can right-click the member within the Members Browser of a worksheet.

To add a new attribute to a level:

1. Click Configuration > Configure Levels. Or click the Configure Levels button.

Business Modeler displays the Configure Levels screen.

2. Right-click the level and select Open > General Attributes.

Business Modeler displays the attributes associated with this level.

3. Right-click the empty space in the Attribute Name list and then select Add.
A new row is added to the list.
4. In Attribute Name, enter the name for the attribute.
5. Specify the following general information for the attribute

Column Name	<p>When you enter an attribute name, Business Modeler automatically generates a name to use for the column that it will add to the level table. You can enter a different column name, up to 30 characters long. The column name cannot include spaces.</p> <p>If you create a method on this level, this column name also serves as the name of a variable that can be passed to the method.</p>
Column Type	Select the data type of this attribute from the drop-down list.

Default Value	Specify the default value for this attribute, to be used when users manually create a new member of this level. If you click Create as level, do not use this setting, because it is ignored. Instead, see "Specifying Default Parents for a New Member".
Column Format	Select the display format for this attribute from the drop-down list.
Create as level	Check this box if you want Business Modeler to automatically create a parent level that uses this attribute to distinguish different members. (The new level is added immediately.)
Paste attribute values	Check this box if you want Demantra to copy and paste the value of this attribute when a user copies and pastes and level member.

6. If this attribute should have a drop-down list, then do the following:

- Select one of the following options for Lookup Type:
 - Select *Table* if the attribute values are in a table.
 - Select *Level* if the attribute values are members of a level.
- If you selected Table, then complete values for this attribute as follows:

Table Name	Select a table containing the reference values from the drop-down list.
Display Column	Select the column that has the user-friendly attribute descriptions.
Data Column	Select the column that has the corresponding numeric code values for the attribute.

- If you selected *Level*, then for *Level Name*, select the level that contains the attribute values.
- For either Table or Level, optionally specify additional criteria to control the drop-down list

Extra From	Comma-separated list of additional tables to include in the query that retrieves the drop-down list. See "Using Extra From for an Attribute"
Extra Where	True/false SQL expression that filters this list further. See "Using Extra Where for an Attribute".

- If you selected "Level," the fields "Security" and "Minimum Privilege displayed" will become enabled. To configure these, refer to the section "Dropdown Security" at the end of this procedure.

7. Click Next.

The Defaults screen appears. If needed, specify the default parents of any manually created member of this level. See "Specifying Default Parents for a New Member".

To delete an attribute from a level:

1. In the Attribute Name list, right-click the attribute.
2. Click Delete.

See also

"Before Configuring Levels"

Dropdown Security

The fields "Security" and "Minimum Privilege Displayed" are enabled when Lookup Type is set to Level, or when Lookup Type is set to Table and the specified Table Name is a level table. Examples of level tables include Location, Items, Promotion, and Settlement. These control which level members a worksheet will be able to access.

If the lookup type is set as 'Table' but the table name is a level table, as listed in GTABLE column of GROUP_TABLES, security will be applied as though the lookup was on a Level.

Security

This dropdown has the following four options:

None (default)	Dropdown security is turned off.
----------------	----------------------------------

Direct	Security will be respected on the level being looked up and its direct parent level. If security has been defined explicitly on the level (for example, Site) a user will see those Sites to which they have access. If security has been defined on the immediate parent (for example, Account), the user will see only those Sites they have access to, as inherited through Account restrictions.
Uni-Dimensional	Security will be respected within the complete dimensional hierarchy of the level being looked up—both the direct parent hierarchy and indirect sibling hierarchies within the single dimension (item, location or GL). For example, if security has been defined on the 'Customer' level and a Lookup is created on the 'Site' level, a user would be restricted to seeing only those Sites for which they have access, as inferred from 'Customer' security.
Cross-dimensional	Security will be inherited across hierarchies via matrix relationships. For example, if security has been defined on the 'Region' level and a dropdown is created to lookup on 'Item' level in the Item hierarchy, the user will be restricted to only those products selling into the Regions they have access to, as determined through mdp_matrix.

Minimum Privilege Displayed

When security is enabled (all but 'None' option), only those level members for which the user has Full Control or Read & Write access will be visible in the dropdown by default. If a user has no visibility or read-only visibility to a member, they will not be able to select that member as part of their planning process, particularly for hierarchical objects such as Accounts or Product Category.

However, in some instances a member may be secured as Read Only but accessible. For example, Promotion Type. The user will be unable to change the value, but should be able to select it when planning a promotion.

This access is controlled by the Minimum Privilege Displayed parameter, which has the following three options:

Read Only	User can view all members of this level but cannot select or modify them.
Read & Write (default)	User can view, select, and edit members of this level but cannot delete members.
Full Control	User can view, select, edit, and delete members of this level.

Filtering an Attribute Drop-down List

Sometimes it is useful to filter the drop-down list of an attribute, and to filter this list in a context-specific way. For example, the value of one attribute sometimes should restrict the list of choices for another attribute. Demantra provides options to enable you to filter the drop-down list.

Note: The `MaxAvailableFilterMembers` parameter specifies the maximum number of entries that a filtered drop-down list can display.

Using Extra From for an Attribute

For a drop-down attribute, the values are taken either from a table or from a level (which of course is also in a table). You can provide a comma-separated list of other tables that should be included in the query that returns the drop-down list.

Using Extra Where for an Attribute

For a drop-down attribute, you can specify a SQL expression that filters the drop-down list. The syntax of this expression is generally as follows:

- `table.column operator other_table.other_column`

Here *operator* is a comparison operator, one of the following:

- `=`
- `<>`
- `>`
- `>=`
- `<`
- `<=`

And `table.column` and `other_table.other_column` are key columns in the database.

A user sees the drop-down list for an attribute within the member properties window

(right-click > Edit) of the Web client. Your Extra Where clause may need to refer to the value of an attribute (or population attribute) that is present in that window. To do so, you can include either of the following syntax elements in your Extra Where clause:

#att.null-warning.attribute-name#

#pop.null-warning.attribute-name.level-name#

Here:

att or pop	Indicates the type of attribute that you are referring to: <ul style="list-style-type: none">• pop (indicates a population attribute)• att (indicates a regular attribute)
null-warning	Indicates what to do if the attribute has a null value. Use one of the following keywords: <ul style="list-style-type: none">• oknull (a null value is permitted for the attribute; the Extra Where clause will not throw an error)• nonnull (if the attribute has a null value, do not execute the SQL of the Extra Where clause) <p>Set this appropriately so that users do not see an error.</p>
attribute-name	Name of the attribute to consider. Specifically: <ul style="list-style-type: none">• For a population attribute, this should be the ATTRIBUTE_LABEL value in the GROUP_ATTRIBUTES_POPULATION table.• For a regular attribute, this should be the ATTRIBUTE_LABEL value in the GROUP_ATTRIBUTES table.
level-name	Name of the level (from the population attribute) whose member IDs will be accessed in this expression.

For example, the syntax **#pop.oknull.population.Selling Entity#** refers to the Selling Entity member of a population attribute.

Specifying Default Parents for a New Member

When a user manually creates a new member of a given level, the user must specify the parents of that member. You can optionally specify the default parent members to be used in this case.

For each level, Demantra provides a predefined default member, which is initially named Default level name. You can choose whether to display this member and you can rename it. This predefined default member is not normally associated with any data, however. If you have data loaded in the system, you can instead choose an existing member to use as the default member. So, for example, you could use any of the following as the default member of the Brand level:

- The predefined default member: Default Brand
- The predefined default member, renamed by you: Unspecified Brand
- An existing member: Acme

Remember that a given level can have multiple parent levels. This means that you can specify a default within each of those parent levels. For example, in the demo, the Promotion level has three parents: Promotion Status, Promotion Type, and Scenario. When a user creates a new promotion, you may want the user to have a default value for each of these.

To specify default parents:

1. Click Configuration > Configure Levels. Or click the Configure Levels button.

Business Modeler displays the Configure Levels screen.

2. Right-click the level and select Open > Defaults.

Business Modeler displays information about the parents of this level. For example, if you view the defaults for the Promotion level (in the demo), you will see the following:

Default Parent Level: Promotion Status

Default Member

☐ Hide Predefined Default Member

Rename Predefined Default To: Update

3. For each parent level of this level, optionally do the following:

- Select the parent level from Default Parent Level.

The Default Member area then lists all the members of this parent level.

Default Parent Level: Promotion Status

Default Member

- ☐ Default Promotion Status (Predefined Default)
- ☐ Committed
- ☐ Exposed
- ☐ Matched
- ☐ Planned
- ☐ Unmatched
- ☒ Unplanned

Total available members: 7

☒ Hide Predefined Default Member

Rename Predefined Default To: Update

- To indicate which member should be the default parent within this level, select

the check box next to that member.

- If you are not using the predefined default member (shown in blue) as the default, you might want to hide this member. To do so, select Hide Predefined Default.
- To rename the predefined default member, type a new name in Rename Predefined Default To and then click Update. You cannot rename this member if you have chosen to hide it.
- When you are done specifying the default for this parent level, select another parent level from Default Parent Level, and then repeat the preceding steps.

Adding a Population Attribute to a General Level

A general level can have population attributes in addition to general attributes. A *population attribute* specifies a set of item-location combinations and consecutive time buckets with which the general level is associated.

Note: General levels are not supported in Demand Planner and Demand Replenisher.

To add a population attribute to a general level:

1. Click Configuration > Configure Levels. Or click the Configure Levels button.
Business Modeler displays the Configure Levels screen.
2. Right-click a general level and select Open > Population Attributes.
Business Modeler displays the population attributes associated with this level.
3. Right-click the Attribute Name list and then select Add.
A new row is added to the list.
4. In Attribute Name, enter the name for the attribute.
As soon as you move the cursor out of this field, the Business Modeler automatically generates names for the tables associated with this level.
5. If you want the attribute to be visible, select the Visible check box.
If an attribute is visible, its properties are available for editing by the user in Demantra. It is recommended that an attribute be configured as non-visible when you do not wish the user to have the ability to edit the attributes. If the attribute is non-visible, it can be edited only in the database.

6. On the right side of the screen, complete the fields as follows:

Attribute Name	Visible
Population	<input checked="" type="checkbox"/>

General Properties
Attribute Name:
Population Type:

Population Tables Structure
Levels Table Name:
Members Table Name:
Dates Table Name:
Data Table Name:
Matrix Table Name:
Indicator: ☒

Population Type

Select Searchable or Descriptive. A general level can have only one searchable population attribute and any number of descriptive population attributes.

If a population attribute is searchable, then each member of the general level is directly linked with the associated item-location combinations and time buckets. Internally, Demantra automatically joins the data for use by the Analytical Engine.

If a population attribute is descriptive, it is available to the users but is not available to the Analytical Engine.

Indicator

Specifies whether the cells in a worksheet table should display an indicator if a general level is associated with them. It is useful to enable this indicator for the benefit of users of the worksheet. This option is enabled only for the searchable population attribute.

-
7. When you are done adding population attributes, click Finish.

Or, if you are configuring a general level at the lowest level, click Next. See "Configuring the Activity Browser".

To delete a population attribute from a general level:

1. In the Attribute Name list, right-click the attribute and then select Delete.

See also

"Before Configuring Levels"

Creating a time aggregation

A time aggregation aggregates data by time, and time aggregations are often used for custom calendars. Your solution can use time aggregations, custom time units, or a combination of both. Use the following guidelines to determine which you need:

	Names	Uses in worksheet
time aggregation	Each member can have a user-friendly name that you create.	You can use a time aggregation like any other level, such as placing it on a worksheet axis.
time unit	Each time bucket in the worksheet is automatically labeled with the start date of that bucket.	You can use time units only on the x-axis (the time axis) of the worksheet.

To create a time aggregation:

1. Within the database, either add either a column to the Inputs table or add an entire table to store the data.
2. Follow the procedure in "Creating a Level".

Note: time aggregations are supported only in the Web products. For the equivalent functionality in the desktop products, create a group expression; see "Configuring Desktop Group Expressions".

See also

- "Configuring Time Units"

Sorting the time aggregation:

time aggregations are populated such that an alphabetical sort on code generates a chronological result in the description. In other words, instead of an alphabetical result:

April, August, December, February, March, and so on; the result is January, February, March, April, and so on.

Please populate the code appropriately to generate the desired sort. The sort is controlled by an associated code, for example:

Code	Description
1	JAN-01
2	FEB-01
3	MAR-01
and so on	

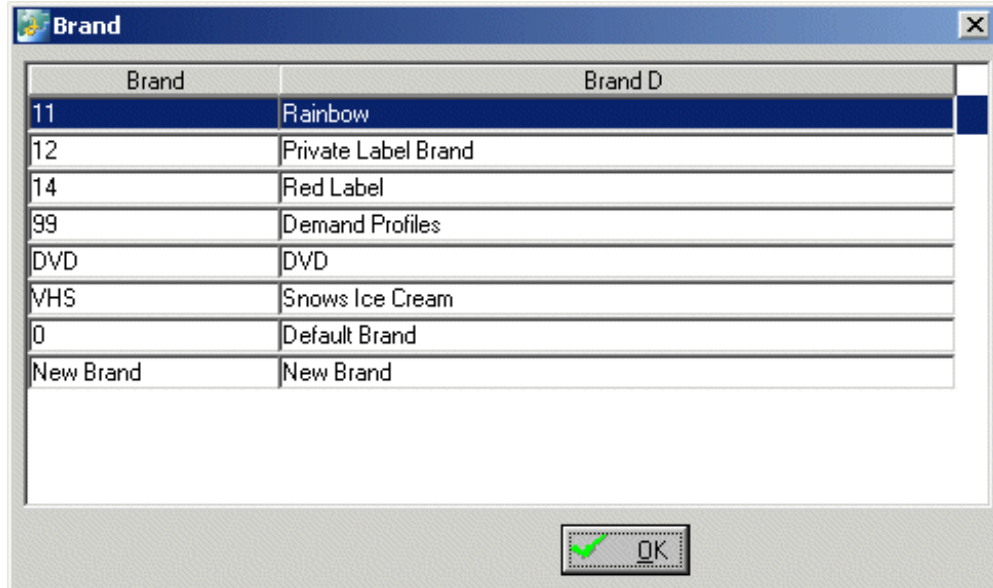
Viewing the Members of a Level

You can view the members of any level.

To view the members of a level:

1. Click Configuration > Configure Levels. Or click the Configure Levels button.
2. Right-click the level and select Level's Members.

Business Modeler displays a screen like the following:



Each row shows the code display field and description for one member of this level. The column headers show the name of the table fields that store these two labels.

The same data is displayed in the standard filter user interface in Demand Management and other Demantra products.

Removing Levels

You can disable a level, removing it from visibility in the user interfaces. You can also delete levels.

To disable a level:

1. Click Configuration > Configure Levels. Or click the Configure Levels button.

The system displays a screen showing the levels. The disabled levels are indicated by an X symbol.

2. Double-click the level or right-click the level and then select Open > General Properties.

The General Properties screen appears.

3. Change the Status field to Disabled.
4. Click Next repeatedly until the Finish button is no longer grayed out.
5. Click Finish.

You can enable the level later in much the same way.

To delete a level:

1. Click Configuration > Configure Levels. Or click the Configure Levels button.

The system displays a screen showing the levels.

2. Right-click the general level and then select Delete. This task applies to the Business Modeler. See "Logging onto the Business Modeler".

Configuring Series and Series Groups

This chapter describes how to configure series and series groups.

This chapter covers the following topics:

- Before Configuring Series
- Creating a Series
- Creating a New Series Based on an Existing Series
- Specifying General Properties of a Series
- Specifying How to Display a Series
- Configuring a Dropdown-Style Series
- Filtering a Series Drop-down List
- Specifying Data Properties of a Series
- Using the Expression Editors
- Syntax of Server Expressions
- Syntax of Client Expressions
- Specifying Server and Client Expressions
- Creating an Edit-Lock Expression
- Creating a Color Expression
- Controlling Access to Series
- Configuring Desktop Group Expressions
- Deleting a Series
- Enabling Series Caching By Item
- Specifying the Order of Series in Dynamic Open Link
- Creating or Modifying a Series Group
- Deleting a Series Group

- Viewing Dependencies Among Series

Before Configuring Series

Before you use Configure > Series option, be sure to do the following:

- Read the "Series" Chapter and make sure you understand how series are calculated and stored.
- Load some sample data for items, locations, and sales, by using the batch script created by the Data Model Wizard.
- If you are using DSM, use the Demantra database procedures to set up an initial set of series for that product; see "Configuring DSM".

Creating a Series

The following procedure describes the minimal set of steps needed to create a new series.

To create a series:

1. Click Configuration > Configure Series or click the Configure Series button.
2. Click the New button.

The series editor displays the General Properties screen, with a new series that has a default name and internal name.

3. Edit the Series name field as needed. This should be a user-friendly name, because it is displayed in the components.

You can include series names in client expressions. If a series name contains a mathematical expression, for example, Primary Forecast -10, Oracle Demantra evaluates the mathematical expression. Therefore, the client expression doesn't function as you intended. See Operators in Client Expressions, page 38-2 for the way to include mathematical expressions in series names.

4. Edit the Internal Name field as needed. Use a name that is easy to remember. The internal name cannot have spaces or special characters.

Note: Business Modeler uses this name as the name of the column in which it stores the series data. When you create server expressions, you refer to those column names.

5. Click Next repeatedly until the Data Properties screen appears.

6. In the Data Table field, select the table in which data for this series should be stored, if you choose to store the data. The choice depends on how you want to use the series, as follows:

sales_data	Use for data that varies by item, location, and time. In this case, you are creating a sales series.
mdp_matrix	Use for data that varies by item and location, but does not vary by time. In this case, you are creating a combination or matrix series.
promotion	Use for data that varies by item, location, promotion ID, and time. In this case, you are creating a promotion series, which is supported only in Promotion Effectiveness. (Note that the series is added to the promotion_data table, rather than the Promotion table as stated in the Business Modeler.)
Level name	Use for data associated with a specific level; all levels that you have defined are listed here; see "Configuring Levels". In this case, you are creating a level series.

Caution: If you change the selection in the Data Table field, Business Modeler automatically removes the existing data from the table where it had been originally stored. Business Modeler then creates a new, empty column in the newly selected table.

7. What happens next depends on the table you chose.
- If you selected sales_data, mdp_matrix, or promotion, Business Modeler asks you to confirm whether you want to create this series within that table.

If you want to store this series directly in the database, click Yes. Business Modeler automatically populates Update Field with the value you used for the internal name. Otherwise, click No.
 - If you selected the name of a level, then in Update Field, select the field that you want to use as this series.

Note: If you are familiar with database terminology, note that this option determines the primary key of the series.

8. At this point, you have entered enough information to save your work.

About the series editor

The series editor consists of a set of screens with different purposes. To move from screen to screen, click Next and Previous.

Screen	Purpose	For details, see...
General Properties	Specify the series name and other basic information.	"Specifying General Properties of a Series".
Display Properties	Specify how to display this series in tables and graphs; also specify numeric precision of series (number of decimal places).	"Specifying How to Display a Series".
Drop-down Properties	Optionally configure the series elements as drop-down lists.	"Configuring a Dropdown-Style Series".
Data Properties	Specify how this series will be stored in the database.	"Specifying Data Properties of a Series".
Expressions Properties	Specify either a server expression, a client expression, or both, that calculate values for this series.	"Specifying Server and Client Expressions".
	Optionally specify special-purpose expressions.	"Creating an Edit-Lock Expression" "Creating a Color Expression" .
Security	Specify which users can access this series	"Controlling Access to Series " .

Creating a New Series Based on an Existing Series

You can easily create a new series that has most of the properties of an existing series. This is useful when you need to define multiple series to use for multiple forecast versions or for use in rolling data sessions.

To create a series based on an existing series:

1. Click Configuration > Configure Series or click the Configure Series button.
2. Right-click the series and click Create As.
Business Modeler prompts you for the name and internal name of the new series.
3. For Series Name, specify a user-friendly name. This name is displayed in the worksheets.
4. For Internal Name, specify a unique name that has no spaces or special characters.
5. Click OK.
 - A new series is created, with everything copied from the original series, except for Update Field, on the Data Properties screen.
6. In the series list on the left side of the screen, right-click the series and then select Open > Data Properties.
7. For Update Field, select the field in which you want to store this series, if any.

See also

- "Creating a Series" "Configuring Rolling Data"

Specifying General Properties of a Series

To edit general properties of a series:

1. Click Configuration > Configure Series or click the Configure Series button.
2. Right-click a series and then select Open > General Properties.

Series Id:	160
Series Name:	Market Plan \$
Internal Name:	rev_target
Show as Default in New Query:	<input type="checkbox"/>
Period Association:	Forecast
Editable:	Yes
Hint Message:	Revenue Target in Dollars
<p>Note - you can use the #FDATE@<forecast version number># token in the Hint Message. When displaying the Hint Message, This Token will be replaced with the creation date of the appropriate forecast version.</p>	
Aggregated by Unlinked Level:	

3. Specify the following information about the series:

Series Name	User-friendly name for the series. This is displayed in the worksheets and is used as a reference when configuring spreadsheet expressions.
Internal Name	Internal name that Demantra uses to refer to the series. Important: By default, Business Modeler uses this name as the name of the column in which it stores the series data. When you create server expressions, you refer to those column names.
Show as Default in New Query	Check if you want this series to appear by default as an option for a new worksheet.

Period Association

Select one of the following choices, to specify the time periods during which the series can be edited, if at all:

History

Forecast

History and Forecast

For an editable series:

If the series is configured as history, then it is editable only in the current time bucket and previous time buckets.

If the series is configured as forecast, then it is editable only in the current time bucket and future time buckets.

If an edit-lock expression has been applied to this series, that can further restrict editing. See "Creating an Edit-Lock Expression" .

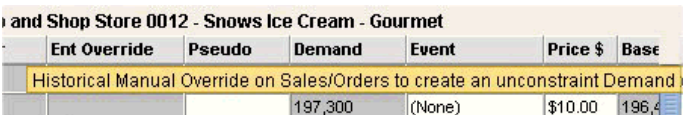
Editable

Specify whether the series will be editable.

If the series has a client expression, it must be read-only.

Hint Message

Short description of the series and its purpose. Demantra will display this message when the pointer hovers over a series name in a worksheet table.



and Shop Store 0012 - Snows Ice Cream - Gourmet					
Ent Override	Pseudo	Demand	Event	Price \$	Base
Historical Manual Override on Sales/Orders to create an unconstraint Demand		197,300	(None)	\$10.00	196,4

You can include the token #FDATE@<Version># to refer to the date on which a forecast version was generated. This is particularly useful if the server expression refers to multiple forecast versions.

Aggregated by Unlinked Level

Optionally specifies an unlinked level that aggregates data for this series when displayed in a worksheet that contains that unlinked level.

An unlinked level is a level that is flagged for use in this way.

Specifying How to Display a Series

To specify the display properties:

- 1. Click Configuration > Configure Series or click the Configure Series button.
- 2. Right-click a series and then select Open > Display Properties.

- 3. Specify the following information about the series:

Display Type	Specify where the series will be displayed in worksheets: Table and Graph Table only Graph only
Color	Colors to use in the graph, both online and printed.
Style	Styles to use for lines in the graph, both online and printed.
Symbol	Symbols to use for data points in the graph, both online and printed.

Display Format	<p>Format in which the series will be displayed in worksheet tables. This can be configured for commas, percentage sign, decimal point and so on. For example, ##,###.## and ##.##%</p> <p>Select a format and modify it if necessary. For example, you can more decimal places to the series by adding pound signs (#) after the decimal.</p> <p>For information on the date formats, see "Display Formats for Dates)".</p>
----------------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

4. For Summary Function, specify how to summarize data for this series within the Summary row in any worksheet table that includes this series.

Note: The Summary Function is used in all rows of the worksheet table if a level is hidden in the worksheet view.

Choose a function, specify a client expression, or select the No Summary option:

Function

- *Total* gives the numeric total of the non-null series entries that worksheet currently displays. (If all entries are null, the total is given as 0.)
- *Average* gives the numeric average of the non-null series entries that worksheet currently displays. (If all entries are null, the average is given as 0.)
- *Count* gives the number of series entries that worksheet currently displays, including any null entries.
- *Min* gives the smallest of the non-null entries.
- *Max* gives the largest of the non-null entries.
- *Common* gives the most common non-null entry. If multiple values appear the most times, an arbitrary one of them is displayed.
- *Uncommon* gives the least common non-null entry. If multiple values are the least common, an arbitrary one of them is displayed.
- *Latest* gives the last non-null entry in the column.

Client Expression

If you enter a client expression, that expression calculates the summary for this series. To enter a client expression, click the Client Expression option and then click the ellipses (...) button. The system displays a dialog box where you can create a client expression; see "Using the Expression Editors".

Note: If you use a client expression, the series is supported only in Web-based products, not in the desktop.

You can also create a weighted average. To do so, enter an expression in the sum_func column in the computed_fields table in the database. For example, to create a weighted sum for two series called batch_for and final_for, use the following expression in the sum_func column:

```
sum(cbatch_for for all) * sum(cfina_for for all)
```

Note: The series names are given the prefix c. Also note that "for" and "all" are protected names.

No Summary

No Summary gives a null value.

5. Specify the width of the columns in which to display this series in worksheet tables:

DP Series Width	Width of the column in the desktop. Each increment of 25 can display approximately one character, depending on the formatting. If the field width is 250, it can display about 9 characters.
DP Web Series Width	Width of the column in the Web-based products. Each increment of 2 can display approximately one character, depending on the formatting. If the field width is 25, it can display about 11 characters. See "Series Widths on the Web".

Series Widths on the Web

For Web worksheets, the following table provides common useful settings for DP Web Series Width:

DP Web Series Width	Sample Data of Maximum Displayable Width
5	\$ 99 -\$ 99 99% -99%
6	\$0.99 \$123 (\$123) -\$123
7	\$99.99 -\$99.99
8	\$ 999.00
9	\$99,999.00 10/10/2005
10	\$ 999,999.00
11	\$ 9,999,999.00

You should also consider the width needed to display the series title.

Display Formats for Dates

For a series that contains date values, you can use any of the following display formats. The date used in the examples is 28 January 1971.

Format	Example	Name of format
MM/dd/yyyy	01/28/1971	American slash
MM/dd/yy	01/28/71	American slash 2-digit year

Format	Example	Name of format
MM-dd-yyyy	01-28-1971	American dash
MM-dd-yy	01-28-1971	American dash 2-digit year
dd.MM.yyyy	28.01.1971	European dot
dd.MM.yy	28.01.71	European dot 2-digit year
dd/MM/yyyy	28/01/1971	European slash
dd/MM/yy	28/01/71	European slash 2-digit year
E, MMM. dd yyyy	Thu, Jan. 28 1971	American text long
E MM/dd/yyyy	Thu 01/28/1971	American number slash long
E MM-dd-yyyy	Thu 01-28-1971	American number dash long
E dd/MM/yyyy	Thu 28/01/1971	European number slash long
E dd.MM.yyyy	Thu 28.01.1971	European number dot long

Configuring a Dropdown-Style Series

You can configure the elements in a series as drop-down lists. When a user includes the series in a worksheet, he or she can set the value of a series element by selecting from the list, if the series is editable.

To specify a series as a dropdown-style series:

1. If you have not yet created the table or level that you want to use for series values, create the table or level. (A level is stored as a table, of course.)

To create a table, use a database tool or an SQL script.

For information on creating a level, see "Configuring Levels".

2. Click Configuration > Configure Series or click the Configure Series button.
3. Right-click a series and then select Open > Dropdown Properties.

Dropdown Type:

Lookup Properties

Level Name:

Display Field:

Data Field:

Security:

Minimum Privilege Displayed:

Extra From:

Extra Where:

4. Specify the dropdown style, one of the following:

List	Use this option if the list of choices is not available in the database as a level or as a regular table.
Table	Use this option if the database includes a <i>table</i> that contains the choices you want to present in the user interface.
Level	Use this option if the database includes a <i>level</i> that contains the choices you want to present in the user interface.

All three styles look the same to end users.

5. If you specified list style, click the Edit Dropdown List button. Then specify the list elements as follows:
1. Click Add.
 2. For Code, type a numeric value. This is an actual possible value of the series.
 3. For Description, type the string value to display when the corresponding

numeric code is selected.

4. Repeat as needed. When you are done, click OK.
6. If you specified table or level style, specify the following information:

Table Name or Level Name	Select the name of a table or a level, depending on the style you specified.
Display Field	Field that contains the values to display in the series drop-down list.
Data Field	Field that contains the values associated with the selected display field. The data field must contain numeric data. When the user selects a given display field, Demantra sets the series entry equal to the corresponding data field. Note that the Data Type for this series must also be numeric; see "Specifying Data Properties of a Series".
Extra From	Comma-separated list of additional tables to include in the query that retrieves the drop-down list. See "Using Extra From for a Series Dropdown".
Extra Where	True/false SQL expression that filters this list further. See "Using Extra Where for a Series Dropdown".

7. The dropdowns "Security" and "Minimum Privilege Displayed" are only enabled when the Dropdown Type is set to "Level." These fields enable you to specify which level members will be accessible by the series. For an explanation of these two fields, see the section "Dropdown Security" in the chapter "Configuring Levels."
8. (Optional) To see the syntax of the series, click the Show Syntax button, which appears after you have specified the required information.

Then, to copy the syntax to the Windows clipboard, click Copy. This button appears after you click the Show Syntax button.

Filtering a Series Drop-down List

Sometimes it is useful to filter the dropdown list of a series, and to filter this list in a context-specific way. For example, the value of one series sometimes should restrict the list of choices for another series. Demantra provides options to enable you to filter the dropdown list.

Note: The MaxAvailableFilterMembers parameter specifies the maximum number of entries that a filtered drop-down list can display.

Using Extra From for a Series Dropdown

For a dropdown-type series, the values are taken either from a table or from a level (which of course is also in a table). You can provide a comma-separated list of other tables that should be included in the query that returns the drop-down list.

Using Extra Where for a Series Dropdown

For a dropdown-type series, you can specify a SQL expression that filters the dropdown list. The syntax of this expression is generally as follows:

table.column operator other_table.other_column

Here *operator* is a comparison operator, one of the following:

=
<>
>
>=
<
<=

And *table.column* and *other_table.other_column* are key columns in the database.

A user sees the drop-down list for a series within a worksheet table in the Web client. Your Extra Where clause may need to refer to the value of a series or a level member that is present in that window. To do so, you can include either of the following syntax elements in your Extra Where clause:

#series.null-warning.series-name#

#level.null-warninglevel-name#

Where:

series or level	Indicates the type of object that you are referring to:
	<ul style="list-style-type: none">• series (indicates a series)• level (indicates a level)

<i>null-warning</i>	<p>Indicates what to do if the attribute has a null value. Use one of the following keywords:</p> <ul style="list-style-type: none"> • oknull (a null value is permitted for the attribute; the Extra Where clause will not throw an error) • nonnull (if the attribute has a null value, do not execute the SQL of the Extra Where clause) <p>Set this appropriately so that users do not see an error.</p>
<i>series-name or level-name</i>	<p>Name of the series or level to consider. Specifically:</p> <p>For a series, this should be the COMPUTED_NAME value in the COMPUTED_FIELDS table.</p> <p>For a level, this should be the TABLE_LABEL value in the GROUP_TABLES table.</p>

For example, the syntax #pop.oknull.population.Selling Entity# refers to the Selling Entity member of a population attribute.

Specifying Data Properties of a Series

To specify the data properties of a series:

1. Click Configuration > Configure Series or click the Configure Series button.
2. Right-click a series and then select Open > Data Properties.

3. In the Data Table field, select the table with which this series should be associated. (If you are familiar with database terminology, note that this option determines the primary key of the series.) The choices are as follows:

sales_data	Use for data that varies by item, location, and time. In this case, you are creating a sales series.
mdp_matrix	Use for data that varies by item and location, but does not vary by time. In this case, you are creating a combination or matrix series.
promotion	Use for data that varies by item, location, promotion ID, and time. In this case, you are creating a promotion series, which is supported only in the Web client.
Level name	Use for data associated with a specific level; all levels that you have defined are listed here; see "Configuring Levels". In this case, you are creating a level series, which is supported only in the Web client.

Note: If you change the selection in the Data Table field, Business Modeler automatically removes the existing data from the table where it had been originally stored. Business Modeler then creates a new, empty column in the newly selected table.

4. For Update Field:

- If you selected sales_data, mdp_matrix, or promotion, Business Modeler asks

you to confirm whether you want to create this series within that table.

If you want to store this series in the database, click Yes. Business Modeler automatically populates Update Field with the value you used for the internal name; see "Specifying General Properties of a Series". Otherwise, click No.

- If you selected the name of a level, then in Update Field, select the field that you want to use as this series.

5. Enter the rest of the information as follows:

Data Type	<p>Specify the type of data that this series will display, one of the following:</p> <p>Numeric</p> <p>Date</p> <p>String</p> <p>If this is a level-style or table-style drop-down series, the data type must be numeric.</p>
Branch_data Synchro Field	<p>Select the field from branch_data in which Demantra should cache data for this series (branch_data is a synonym for the sales_data or the promotion_data table, as needed).</p> <p>In almost all cases, you select the field with the same name as the Update Field.</p> <p>Make sure not to create two series that have the same synchronization field. Such series will result in an engine error.</p>
Available for Exceptions	<p>If this option is checked, you can use this series in an exceptions filter in a worksheet.</p>
Same Value Update	<p>The default value "0" (zero) means that if the value for a cell has been modified and then returned to the original value, do not send an update. If set to "1", (one) then send an update even if the cell's value has been returned to it's original value.</p>
Ignore Scale	<p>Specifies whether the series is divided by the scaling factor in a worksheet.</p> <p>Demantra automatically divides all numbers in the worksheet by that factor, except for any series that are marked as unscaled. Most series are scaled. A series that calculates a fraction, however, should be unscaled.</p>

Proportional	<p>Specifies whether to split a series value, in cases where data is edited or imported at an aggregate level.</p> <p>A series should be proportional only if the server expression is of the following form:</p> <p><code>sum (table_name.update_column_name)</code> Where <code>table_name.update_column_name</code> is the update field for this series.</p> <p>If a series is proportional, data for a given combination is divided among the child combinations according to the proportions given by the Proportion Calculation Series.</p>
Save Zero as Null	<p>If this option is checked, zero values are treated as null. That is, when a series value is set equal to zero and then saved, the value is automatically set equal to null.</p>
Proportion Calculation Series	<p>Select a reference series that you will use to calculate the proportions when splitting aggregated data for this series.</p> <p>The default series depends on whether the series is a historical series or a forecast series.</p> <p>In general, use a series that is stored in the same table as the series you are defining. For example, if you are defining a sales series, the Proportion Calculation Series should also be a sales series. See "Specifying a Proportions Reference Series".</p> <p>When defining a proportional series, only proportional series will be available in the "Proportion Calculation Series" drop down.</p>
Aggregated Base Level	<p>Applies only to sales series. This option lets you specify how this series is aggregated in a worksheet that includes a promotion level. See "Using the Aggregated Base Level Option" . For most series, use <code>sales_data</code>.</p>

Aggregation Function	<p>Specifies how to aggregate data for this series during the following operations: import, export, copy and paste of a promotion, editing of the duration of a promotion.</p> <p>Choose one of the following functions:</p> <p>Sum</p> <p>Max</p> <p>Min</p> <p>Avg</p> <p>Wavg</p>
WAVG By	<p>Specify the series to use as the weights if you use Wavg for the preceding option.</p>
Copy/Paste preservation type	<p>This option applies only to promotional series, which store time-varying data associated with promotions. It specifies how to handle the series data if a user copies and pastes a promotion, or moves an item from one promotion group to another. When set to Preservation, this option works in conjunction with the CopyPasteIntersect system parameter which determines how volume is distributed when an item is moved from one promotion group to another. For examples, see "Series". See also "System Parameters" for more information about the CopyPasteIntersect system parameter.</p>
Move preservation Type	<p>This option applies only to promotional series, which store time-varying data associated with promotions. It specifies how to handle the series data if a user moves a promotion, changing its dates. For examples, see "Series".</p>

Using the Same Value Update Option

Sometimes a user changes a worksheet cell value, and then changes the cell back to the original value without saving or rerunning the worksheet. This option lets you specify whether this series should send all modified cell values when updated, or send only the cells with changed values.

Using the Aggregated Base Level Option

This option lets you specify how this series is aggregated in a worksheet that includes a promotion level:

- If you choose sales_data, this series is aggregated by the items, locations, and dates selected in the worksheet. Most series are aggregated this way in a typical implementation.

- If you choose promotion, this series is aggregated by the items, locations, dates, and promotions selected in the worksheet. That is, when the series is aggregated, any data that is not associated with a promotion is ignored.

Within a worksheet that does not include a promotion, the series is aggregated based on the series setting; that is, it is aggregated by the items, locations, and dates if it aggregates by sales_data only, and additionally by promotions if aggregated by promotion.

Specifying a Proportions Reference Series

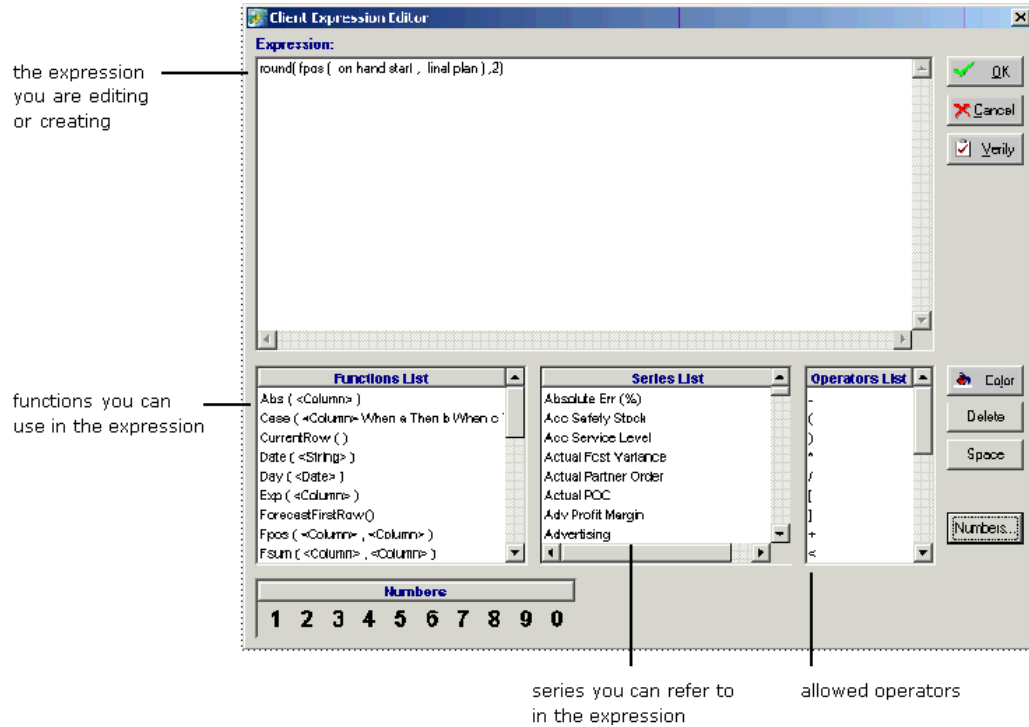
For best performance, Oracle recommends the following:

- Proportions from the same table are better than proportions from a different table.
- If the proportions are not in the same table that stores the series that uses those proportions, consider caching the proportions into the same table that stores the series. For example: create a cache of GLOB_PROP in sales_data and promotion_data.
- Use PROPORTION_COLUMN when the proportions are from the same table and do not require a server expression.
- Use a series that is not empty (most of the time) for the proportion reference.
- The series available in the proportional series drop-down depend upon the table on which the series is defined.
 - Series on Sales_Data can be split by series residing on Sales_Data, MDP_Matrix, Item or Location levels.
 - Series with a data table such as Promotion_Data or Settlement may only be split by series on the same table.
 - Series on a general level without population may be split by any series.
 - Series on MDP_Matrix may be split by any series on MDP_MATRIX, Item or Location levels but not on Sales_Data.

Using the Expression Editors

For server and client expressions, you use the Business Modeler expression editors, which are similar for these two types of expressions.

For example, the Client Expression Editor looks like this:



This editor has been designed so that you can create expressions without using the keyboard, so that you can avoid introducing errors. The number buttons at the bottom of the screen, the Space button, and the Delete button support this.

Note: You use the Color button only if you are creating a color expression; see "Creating a Color Expression".

The Server Expression Editor is similar, with the following main differences:

- The set of allowed functions is different.
- Rather than a list of series, the editor provides a list of the allowed database columns.
- The Server Expression Editor includes two extra fields, Extra From and Extra Where. For details, see "Specifying a Server Expression".

To edit or create an expression:

1. To insert an element at the position of the cursor, click that element. For example, to insert a function, scroll to that function and then click it.
2. To replace an element (such as a placeholder like <Column>), highlight that element and then click the element you want to replace it with.

3. When you are done, click either OK or Verify.

If the expression is not valid, you will receive the message "Expression is not valid." In that case, close the message box and correct the expression.

See also

"Syntax of Server Expressions" "Syntax of Client Expressions" "Specifying Server and Client Expressions" "Creating an Edit-Lock Expression" "Creating a Color Expression"

Syntax of Server Expressions

This section summarizes the allowed syntax for server expressions. For a more detailed discussion, see "Series".

A server expression must be an aggregating SQL expression that returns to a value with length greater than zero for each element. (If you never plan to use the series within a cached worksheet, it can return null or a zero-length value; but you may not be able to prevent the series from being misused.)

A server expression must have one of the following forms:

aggregate_function (branch_data.database_column * #UNIT#)

aggregate_function (branch_data.database_column)

aggregate_function (mdp_matrix.database_column * #UNIT#)

aggregate_function (mdp_matrix.database_column)

aggregate_function (other_expression)

Here:

- *aggregate_function* is one of the SQL aggregating functions, most commonly sum.
- *database_column* is a column of the branch_data or mdp_matrix table, most often the update field that corresponds to this series. That is, if SeriesA is associated with branch_data.SeriesA, then the server expression for SeriesA could be sum(branch_data.SeriesA)

Note: branch_data is a synonym for the sales_data table or the promotion_data table.

- #UNIT# represents the unit conversion factor. Within a worksheet, this token is automatically replaced by the conversion factor that corresponds to the unit that the worksheet is currently using.

In turn, *other_expression* can be made up of some or all of the following components:

- Other SQL functions.
- Constants and expressions that have numeric, string, date, and true/false values.

Note: Enclose any literal negative value within parentheses, as in this example: (-0.33)

- Operators such as +-*./.
- Demantra tokens such as #UNIT#. Note that #UNIT# currently supports ONLY multiplication (*).
- Columns of the branch_data and mdp_matrix tables.

You can use parentheses to control the precedence of calculations, according to standard algebraic rules.

Note: SQL expressions have a limit of 3000 characters. To avoid reaching this limit, use small field names.

For information on the supported operators, tokens, and SQL functions, see "Server Expression Functions and Operators".

Syntax of Client Expressions

This section summarizes the allowed syntax for client expressions. For a more detailed discussion, see "Series".

A client expression uses Demantra functions. The client expression can be made up of some or all of the following components:

- Constants and expressions that have numeric, date, true/false or null values.

Note: Enclose any literal negative constant within parentheses, as in this example: (-0.33)

- Demantra functions.
- Operators such as +-*./.
- References to other series. To refer to a series, you use the name of the series.
- References to series at other time periods. Here, you use the following syntax:
series_name[relative-time-bucket]

An expression like this is sometimes called a vertical formula. For example: Sales [-1] refers to the Sales series for the previous period. Sales [1] refers to the Sales series for the next period. [0] is not allowed.

Here relative-time-bucket must be any of the following:

- An integer
- A series name
- A simple expression using integers, series names, and the basic mathematical operators.

For information on the supported operators and functions, see "Client Expression Functions and Operators."

Specifying Server and Client Expressions

Specifying a Server Expression:

To edit a server expression

1. Click Configuration > Configure Series or click the Configure Series button.
2. Right-click a series and then select Open > Expression Properties.

Server	
Expression:	<code>sum(nvl(branch_data.orders_override,branch_data.orders)*#UNIT#)</code>
Extra From:	
Extra Where:	
Client Expression:	<code>if (mod check =2, if (supply <= minimum order [- lbytimebuck], minimum order [- lbytimebuck], if (supply >= maximum order [- lbytimebuck], maximum order [- lbytimebuck], if (truncate (supply / lot size ,0) = (supply / lot size), supply , (truncate (supply / lot size ,0) + 1)* lot size)) , int order plan)</code>
Edit Lock Expression:	
Color Expression:	

3. Click the button to the right of the Server Expression field.
The Server Expression Editor is displayed.
4. In the Expression field, create an expression as described in "Using the Expression Editors". For information on the syntax to use, see "Syntax of Server Expressions".
 - Enclose any literal negative value within parentheses, as in this example: (-0.33)
 - If this series is going to be used within cached worksheets, it cannot return null or zero-length values. Use the expression `to_number(null,0)` to express null values that can be cached.
 - `branch_data` is a synonym for the `sales_data` table.
5. When you are done, click either OK or Verify.
If the expression is not valid, you will receive the message "Expression is not valid." In that case, close the message box and correct the expression.

See also

"Server Expression Functions and Operators"

Specifying Extra From for a Server Expression

Normally the server expression can refer only to fields in the following tables:

For sales and matrix series	branch_data and mdp_matrix tables. Note that branch_data is a synonym for the sales_data table or the promotion_data table.
For promotion series	branch_data table.
For level series	Table associated with the level.

In rare cases, you may need to refer to data in other tables. In such a case, use the Extra From field. In this field, specify an optional list of additional tables (separated by commas) that contain data relevant to this series.

If you include a table here, the server expression can refer to columns in that table.

Note: Internally, these tables are added to the From clause in the SQL query that retrieves data for this series.

Specifying Extra Where for a Server Expression

If you need to filter the data further, use the Extra Where field. The syntax of this field is as follows:

table.column operator other_table.other_column

Here *operator* is a comparison operator, one of the following:

=

<>

>

>=

<

<=

And *table.column* and *other_table.other_column* are key columns in the database.

Note: Internally, the Extra Where field is added to the WHERE clause in the SQL query that retrieves data for this series.

Specifying a Client Expression:

To edit a client expression

1. Click Configuration > Configure Series or click the Configure Series button.
2. Right-click a series and then select Open > Expression Properties.
3. Click the button to the right of the Client Expression field.
4. In the Expression field, create an expression as described in "Using the Expression Editors". For information on the syntax to use, see "Syntax of Client Expressions".
 - Enclose any literal negative value within parentheses, as in this example: (-0.33)
 - To include a null value within a client expression, do the following:
 - Create a series named, for example, Null Value and give this series a server expression that evaluates to null.
 - Within the client expression, refer to the Null Value series.
5. When you are done, click either OK or Verify.

If the expression is not valid, you will receive the message "Expression is not valid." In that case, close the message box and correct the expression.

Verifying All Expressions:

To verify all server and client expressions

1. Click the Verify Expressions button in the toolbar. Or click File > Verify All Expressions.

See also

"Client Expression Functions and Operators"

Creating an Edit-Lock Expression

An editable series can have an optional edit-lock expression, which can make series cells uneditable based on a condition. For each editable cell in a series, an edit-lock expression evaluates to true or false.

- If the expression evaluates to true, the cell is automatically made uneditable.

- If it evaluates to false, the cell is left in its original state, which can be either editable or not.

To create an edit-lock expression:

1. Click Configuration > Configure Series or click the Configure Series button.
2. Right-click a series and then select Open > Expression Properties.
3. Click the button to the right of the Edit Lock Expression field.

The Client Expression Editor appears.

4. Create an expression that evaluates to true or false; see "Syntax of Client Expressions".

See also

"Client Expression Functions and Operators"

Creating a Color Expression

Any series can have a color expression, which controls only the appearance of the series. For each editable cell in a series, a color expression evaluates to either a numeric color value or null. This expression must have one of the following forms:

If condition, numeric-color-value

If condition, expression-with-numeric-value

Then for each cell in the series:

- If the expression evaluates to a number, the cell is displayed in the corresponding color.
- If the expression evaluates to null, the color of the cell is left unchanged.
 - In its basic form, a color expression returns one numeric color value based on one condition. To return different color values for multiple conditions, use the second form, and use an If-Then-Else expression for the *expression-with-numeric-value*.
 - A color expression cannot include time-shift expressions such as [- 1], Fpos, and Fsum.

To create a color expression:

1. Click Configuration > Configure Series or click the Configure Series button.

2. Right-click a series and then select Open > Expression Properties.

3. Click the button to the right of the Color Expression field.

The Client Expression Editor appears.

4. Construct an expression (using the If function) that evaluates to a numeric color value. For example, the following color expression makes the cell background red (red=255) if the absolute value of the order variance is greater than the order tolerance:

if (-order variance > order tolerance , 255)

To insert a numeric color value, do the following:

1. Place the cursor where the numeric color value should appear in the expression.
2. Click Color to display the Color dialog box.
3. Select an existing color. Or click Define Custom Colors and define a color. See "To define a custom color".
4. Click OK.

Demantra finds the numeric value that corresponds to the color you selected, and places that number into the expression.

Standard Colors

For reference, the following table lists the standard colors in numeric order. You can use this table to look up a color without having to use the Business Modeler user interface. This may be useful when you are working with unfamiliar client expressions.

Color	Description	Color	Description	Color	Description
0	black	4210816	brown	8454143	yellow
64	dark brown	4227072	green	10485760	dark blue
128	brown	4227200	olive green	12615680	steel blue
255	red	4227327	orange	12615808	blue
16384	dark green	4259584	bright green	12615935	pink
16512	brown	8388608	dark blue	12632256	gray

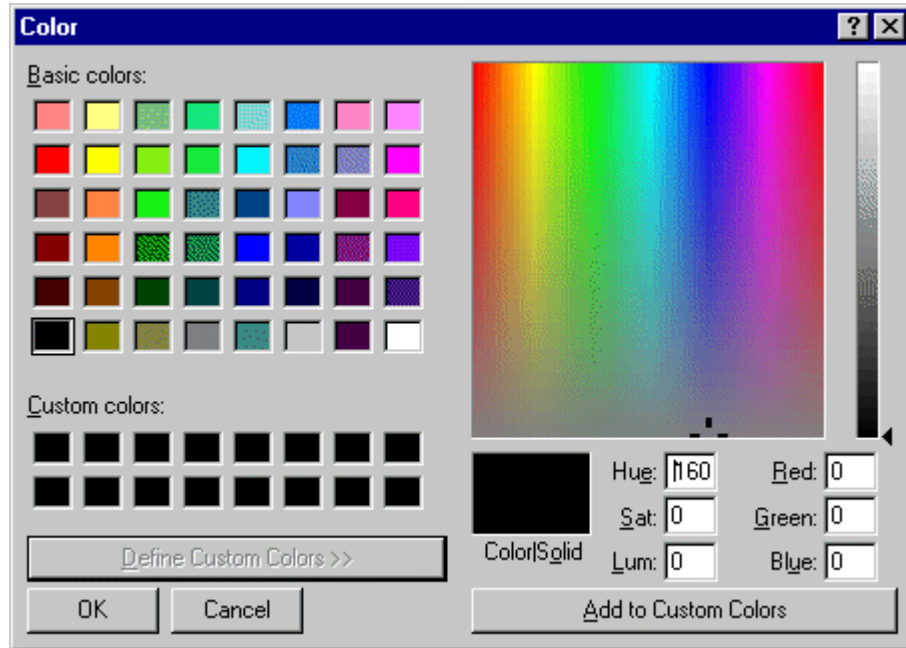
Color	Description	Color	Description	Color	Description
32768	green	8388672	purple	16711680	royal blue
32896	olive green	8388736	purple	16711808	purple
33023	orange	8388863	pink	16711935	pink
65280	bright green	8404992	dark blue	16744448	blue
65408	bright green	8421376	blue green	16744576	blue
65535	yellow	8421440	blue green	16744703	pink
4194304	very dark blue	8421504	gray	16776960	aqua
4194368	very dark purple	8421631	pink	16777088	aqua
4194432	brown	8453888	aqua	16777215	white
4210688	very dark green	8454016	light green		

Defining Custom Colors:

To define a custom color

1. Click Define Custom Color in the Color dialog box.

The Color dialog box expands to display a color palette.



2. Click the color palette to select a color.
3. Drag the luminance pointer (on the right of the dialog box) up to the required luminance (according to the color/solid display).
4. Click Add to Custom Colors to add the color to the Custom Colors list.
5. Click OK.

Controlling Access to Series

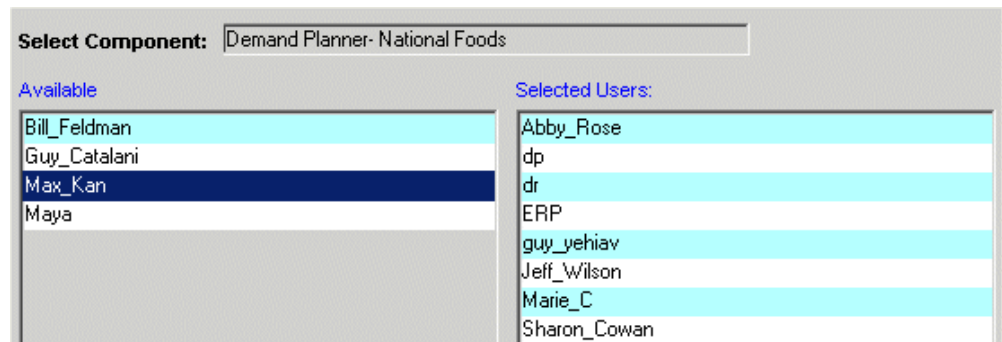
When you create a series in the Business Modeler, Demantra automatically adds that series to your component. You can give access to this series to other users of your component.

To control access to a series:

1. Click Configuration > Configure Series or click the Configure Series button.
2. To see which components include a specific series, click the plus sign (+) to the left of the series name. The display expands to list all the components that include this series:



3. To make changes, right-click the series and then select Open > Expression Properties.
4. Click Next to access the Security page.



5. If you logged into Business Modeler with one of the internal Demantra passwords, you can select any component. Otherwise, you can make changes only within the component with which your ID is associated.
6. For each user of this component who needs access to this series, double-click the user name to move the user name from the Available list to the Selected Users list.

Configuring Desktop Group Expressions

Group expressions specify how to group data for display purposes, into different blocks of time such as quarters, months, or half years, when the user chooses to group data in that way.

Note: Group expressions are supported only in Demand Planner and Demand Replenisher. For the equivalent functionality in the Web products, create a time aggregation; see "Creating a Time Aggregation".

The following figure shows an example of a worksheet when it is ungrouped and also when it is grouped by month.

ungrouped					grouped by month				
	Date	Demand	Base Fcst	Base Fcst 1		Date	Demand	Base Fcst	Base Fcst 1
→	12/31/2001	29,580	30,769	30,785		12/31/2001	29,580	30,769	30,785
	1/7/2002	35,400	33,825	38,999			29,580	30,769	30,785
	1/14/2002	49,300	39,253	45,116		1/7/2002	35,400	33,825	38,999
	1/21/2002	93,700	35,696	40,734		1/14/2002	49,300	39,253	45,116
	1/28/2002	69,980	40,057	42,489		1/21/2002	93,700	35,696	40,734
	2/4/2002	101,140	44,765	52,014		1/28/2002	69,980	40,057	42,489
	2/11/2002	70,752	49,605	58,038			248,380	148,832	167,339
	2/18/2002	85,000	47,785	53,153		2/4/2002	101,140	44,765	52,014
	2/25/2002	62,520	39,957	47,972		2/11/2002	70,752	49,605	58,038
	3/4/2002	78,930	34,562	52,965		2/18/2002	85,000	47,785	53,153
	3/11/2002	141,720	72,742	74,567		2/25/2002	62,520	39,957	47,972
	3/18/2002	397,680	212,143	68,424			319,412	182,112	211,178
	3/25/2002	139,130	48,630	57,793		3/4/2002	78,930	34,562	52,965
	4/1/2002	501,680	238,580	59,210		3/11/2002	141,720	72,742	74,567
	4/8/2002	139,020	69,144	57,194		3/18/2002	397,680	212,143	68,424
	4/15/2002	397,900	207,601	61,982		3/25/2002	139,130	48,630	57,793
	Summary	2,393,432	1,245,114	841,436			757,460	368,077	253,749
	Min	29,580	30,769	30,785	→	4/1/2002	501,680	238,580	59,210
	Max	501,680	238,580	74,567		4/8/2002	139,020	69,144	57,194
						4/15/2002	397,900	207,601	61,982
							1,038,600	515,325	178,386
						Summary	2,393,432	1,245,114	841,436
						Min	29,580	30,769	30,785
						Max	501,680	238,580	74,567

A group expression specifies the group (in time) to which each row belongs. It does not specify how the subtotals are calculated. The subtotals are calculated as specified by the summary function; see "Specifying How to Display a Series".

Demantra provides a set of possible group expressions, which are all pre-configured. You can reconfigure these as needed.

To configure a group expression:

1. Click Configuration > Configure Group Expressions.
The Edit Group Expression screen appears.
2. Select an expression name from the list at the left.
3. In the Expression Description field, edit the name of this expression.
This name appears in the list of choices when the user clicks Data > Define Group... (in Demand Planner or Demand Replenisher).
4. In the Expression field, edit the expression itself.
5. Click File > Save to save this change.
6. Click the close button at the top right of the window to close the dialog box.

Deleting a Series

To delete a series:

1. Right-click the series and click Delete.
Business Modeler prompts you to confirm that you want to delete this series.
2. Click Yes.
Deleting a series may take a couple of minutes, depending on the size of the database.
3. Click File > Save to save this change.

Enabling Series Caching By Item

You can cache series data that is aggregated by item, in the `branch_data_items` table; this improves performance. This technique can be used only in worksheets that meet certain conditions.

To make sure cached data can be accessed from a worksheet:

1. Make sure that the `DYNAMIC_SYNC` procedure is scheduled to run. See "Database Procedures".
This procedure updates `branch_data_items` based on changes elsewhere in the database.
2. Make sure that the worksheet is defined as follows:
 - It does not include filters of the location or matrix types.
 - It does not include levels of the location or matrix types.
 - It does not include contain any matrix series.
3. For the user who is running the worksheet, make sure that no security filters have been defined. see "Creating or Modifying a User".
4. Make sure that the `UseItemsAggri` parameter is set to Yes. For information on this parameter, see "Non-Engine Parameters".

Specifying the Order of Series in Dynamic Open Link

You can control the order in which series are displayed when linked into a third-party

tool through Dynamic Open Link (DOL).

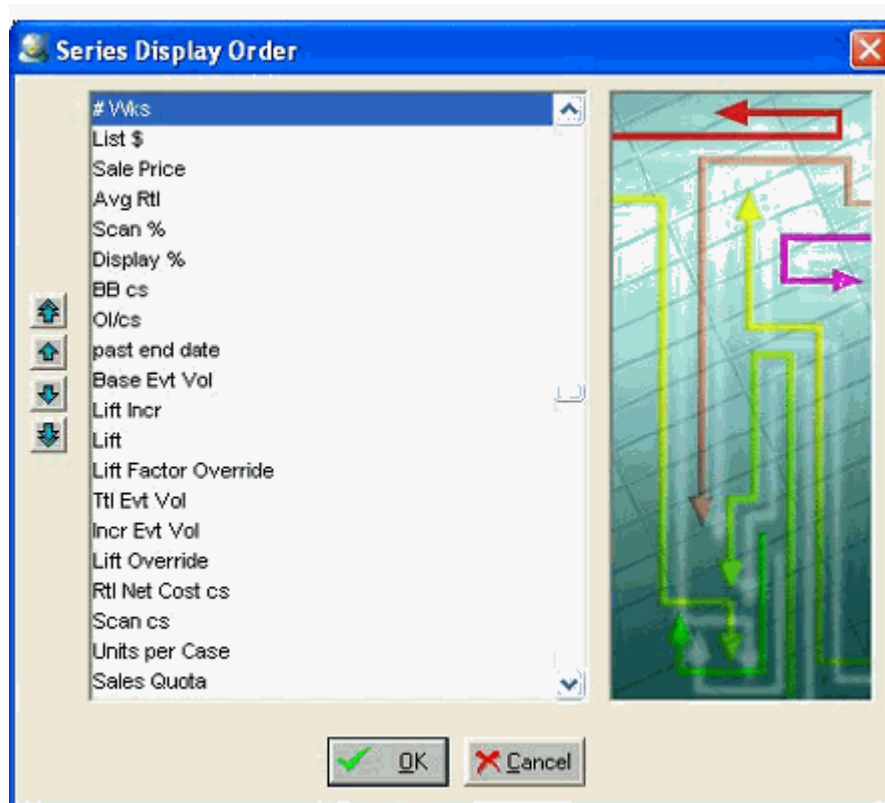
Note: This option controls the order of series in the worksheet wizard used by the following:

- Default display order in the Desktop
- Order of series exported via Demantra Dynamic Open Link
- Default display order of Web client Worksheets

To specify the order of series in the desktop:

1. Click File > Define Series Display Order.

The Business Modeler displays the following screen:



2. To move a series, click it and then click the arrows to move the series to the beginning of the list, up one, down one, or to the end of the list.
3. Click OK.

Creating or Modifying a Series Group

To create or modify a series group:

1. Click Configuration > Configure Series Groups.
2. Next:
 - To create a new series group, double-click the New Group icon. Or click the icon and then click OK.
 - To edit a series group, double-click the icon corresponding to the series group. Or click the icon and then click OK.

The Business Modeler displays the General Properties screen.

3. In Group Name, type a unique name for the new series group.
4. In Group Description, type an optional description for the new series group.
5. Click Next.

The Business Modeler displays the Series screen.

6. To select the series to include in this series group, move series from the left list to the right list.
7. When you are done specifying series, click Finish.

Deleting a Series Group

To delete a series group:

1. Click Configuration > Configure Series Groups.
2. Click the icon corresponding to the series group.
3. Click Delete.
4. Click Yes to confirm the deletion.

Viewing Dependencies Among Series

The Business Modeler provides a simple tool you can use to check the dependencies among the series in your system. The results for a given series depend on whether that

series is editable.

To view series dependencies:

1. Click Tools > Series Dependencies.

The Business Modeler displays the Series Dependencies screen.

2. In Select Series, select a series.

If the series is not editable, the Business Modeler updates the screen, as follows:

The screenshot shows the 'Series Dependencies' window. At the top, the title bar reads 'Series Dependencies'. Below it, on the left, is a 'Select Series:' dropdown menu with 'CY MDF Avail' selected. To the right of this dropdown is a button labeled 'Export All Dependencies'. Below the dropdown menu, there are four distinct panels arranged in a 2x2 grid. The top-left panel is titled 'CY MDF Avail depends On:' and contains a list of series: 'Bill Back', 'CY MDF Actual Spent', 'CY MDF Auth', 'Off Invoice', and 'Promo Cost'. The top-right panel is titled 'Color Expression:' and contains the text 'CY MDF Avail'. The bottom-left panel is titled 'Lock Expression:' and is currently empty. The bottom-right panel is titled 'Summary Line Expression:' and is also empty.

These fields have the following meanings:

Selected series depends on

All the series on which the selected uneditable series depends, directly or indirectly, through its client expression.

If this series does not have a client expression, this field is blank.

Color Expression

Series whose color expressions refer to the series you selected.

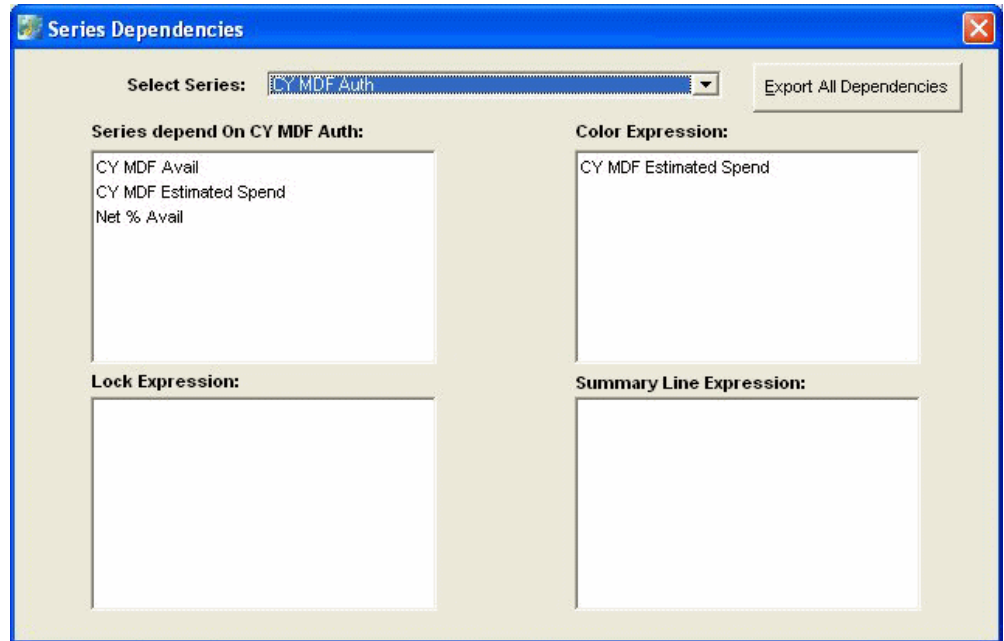
Lock Expression

Series whose edit-lock expressions refer to the series you selected.

Summary Line Expression

Series whose summary row expressions refer to the series you selected.

However, if the series is **editable**, the screen is slightly different:



Notice that the first field has a different label and meaning:

Series depend on *selected series*

All the series that depend on the selected editable series, directly or indirectly, through their client expressions.

The other fields have the same meanings as in the other case.

To export dependencies of all series:

1. Click Export All Dependencies.

The Business Modeler generates the file Demantra_root/Demand Planner/Desktop/Dependencies.xls.

Configuring Units, Indexes, and Update-Lock Expressions

This chapter describes how to perform miscellaneous configuration tasks.

This chapter covers the following topics:

- Before Configuring Units and Indexes
- Configuring Indexes and Exchange Rates
- Editing Values for Indexes and Exchange Rates
- Configuring Units
- Associating Units with Levels
- Configuring Time Units
- Configuring Update-Lock Expressions

Before Configuring Units and Indexes

Before you configure units and indexes, be sure to do the following:

- Read the "Units, Indexes, and Exchange Rates" Chapter and make sure you understand how unit conversion data and the #UNIT# token are used.
- Load unit version data.

Configuring Indexes and Exchange Rates

Monetary units of measure can use financial indexes and exchange rates. Each index and exchange rate is stored in a different table, except for the placeholder index (constant, equals one for all dates).

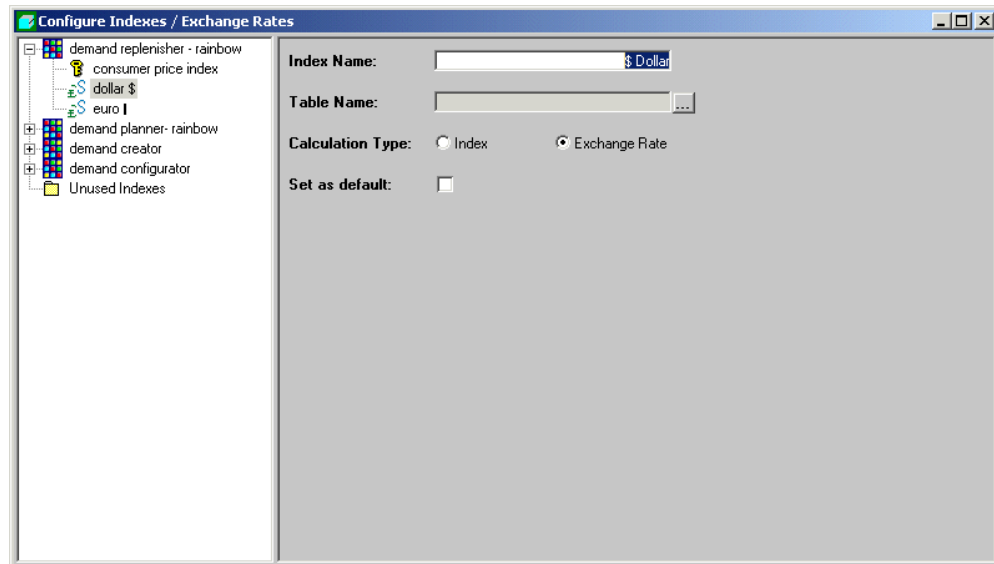
The placeholder index is used to switch a worksheet back to the same monetary units that are used in the imported data. By default this is called **dollar \$**, because monetary

values are usually imported in dollars.

To create an index or exchange rate:

1. Click Configuration > Configure Indexes.

The Configure Indexes and Exchange Rates dialog box appears.

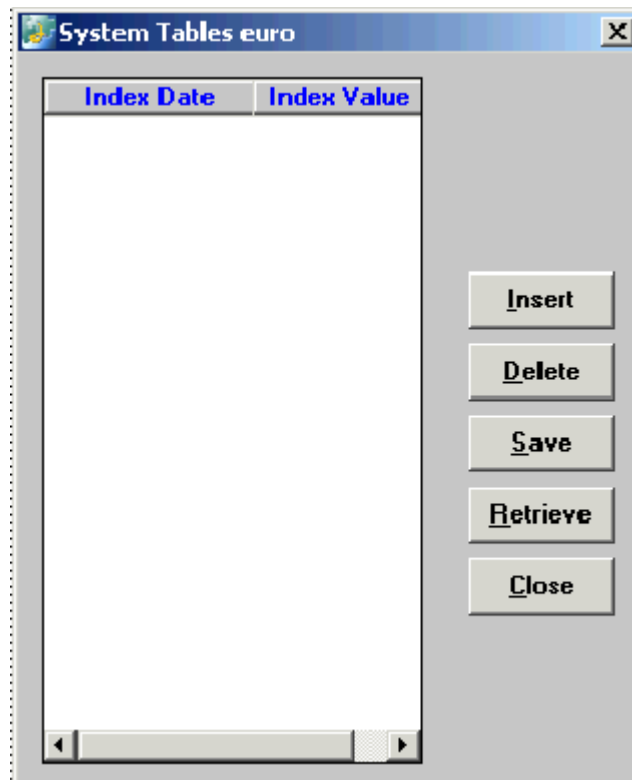


2. Click File > New. Or click the New button.
3. In Index Name, type the name of the index or exchange rate, as it should appear in worksheets.
4. In Table Name, type the name of the table in which Demantra should store information for this index or exchange rate. Demantra will automatically create this table.

Note: For simplicity, use the same name as you used for the index or exchange rate.

5. For Calculation Type, click one of the radio buttons to indicate whether this is an index or an exchange rate.
6. If this should be a default option, click Set as default.
7. Click File > Save.
8. To enter data for this index or exchange rate:
 1. Click the ellipsis (...) button next to the Table Name field.

Business Modeler displays the following window:



2. To add an entry, click Insert.
 3. For Index Date, specify a date, using the date format required by the database.
 4. For Index Value, specify the value that takes effect on the specified date. This value is multiplied by the base unit price.
 5. Repeat as needed. When you are done, click Save.
 6. Click Close.
9. The new index or exchange rate is not associated with any component. See "Creating or Modifying a Component".

To edit an index or exchange rate:

1. In the left side of the dialog box, click the name of any component that includes the index or exchange rate. This expands the display so that you can see the indexes and exchange rates in that component.
2. Click the index or exchange rate.

3. Modify as needed.
4. Click File > Save.

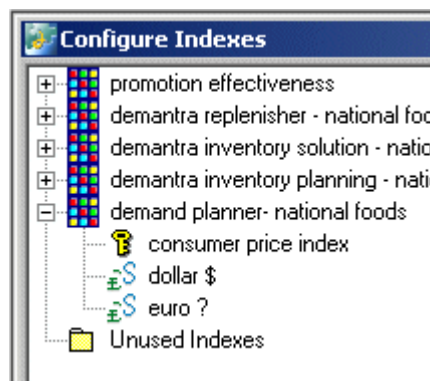
To delete an index or exchange rate:

1. Select an index or exchange rate.
2. Click Delete.
3. Business Modeler prompts for confirmation.
4. Click OK.

To see which indexes a component uses:

1. In the left side of the dialog box, click the plus sign (+) to the left of the component name.

The hierarchy expands to display all the indexes and exchange rates that are used in this component.



To assign indexes and exchange rates to a component, see "Creating or Modifying a Component".

Editing Values for Indexes and Exchange Rates

To edit values for the indexes:

1. Click System > Maintain > Edit Installed Indexes.
The Installed Indexes List dialog box appears.
2. Click a table and then click OK.

A dialog box appears for the selected table.

3. To insert new values, click Insert and then type values in the new row that appears.
4. To update the list after inserting new values, click Update. Or, to reset the list to the original values, click Reset.

Configuring Units

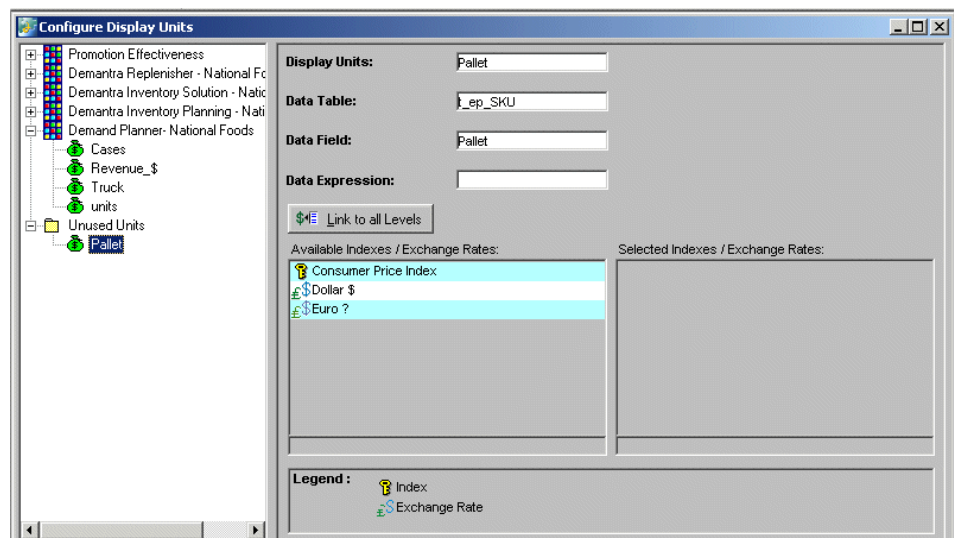
If the database contains the appropriate unit conversion data, you can define two general kinds of units of measure to use in Demantra:

- Size units, which measure the size of a sale: cases, truckloads, and so on.
- Monetary units, which measure the value of a sale. You can also specify indexes and exchange rates associated with the unit.

The procedure is slightly different for these two kinds of units.

To create a size unit:

1. Click Configuration > Configure Display Units.
 - The Configure Display Units dialog box appears.



2. Click File > New. Or click the New button.
3. In Display Units, type the name of this unit of measure, as it should appear in worksheets.

4. In Data Table, type the name of the table (such as t_ep_sku) in which Demantra contains conversion factors for this unit.

Note: The conversion factors must be imported, because these factors are generally different for each SKU and may vary over time.
5. In Data Field, type the name of the field in this table that contains the conversion factors for this unit.
6. In Data Expression, type an expression that retrieves the conversion factors for this unit.
7. Click File > Save.
8. The new unit is not associated with any component. To make this unit available to users, see "Creating or Modifying a Component".

To create a monetary unit:

1. Click Configuration > Configure Display Units.
The Configure Display Units dialog box appears.
2. Click File > New. Or click the New button.
3. In Display Units, type the name of this unit of measure, as it should appear in worksheets.
4. In Data Table, type the name of the table in which Demantra contains conversion factors for this unit.
5. In Data Field, type the name of the field in this table that contains the conversion factors for this unit.
6. In Data Expression, type an expression that retrieves the conversion factors for this unit.
7. For each index and exchange rates by which this unit could potentially be multiplied, drag the index/exchange rate from the left list to the right list. Within a worksheet, the user will be able to select one of these at a time.
8. Click File > Save.
9. The new unit is not associated with any component. To make this unit available to users, see "Creating or Modifying a Component".

To edit a unit:

1. Click Configuration > Configure System Units.

The Configure Display Units dialog box appears.

2. In the left side of the dialog box, click the name of any component that includes the unit. This expands the display so that you can see the units in that component.
3. Select a unit.
4. Modify as needed.
5. Click File > Save.

To delete a unit:

1. Click Configuration > Configure System Units.

The Configure Display Units dialog box appears.

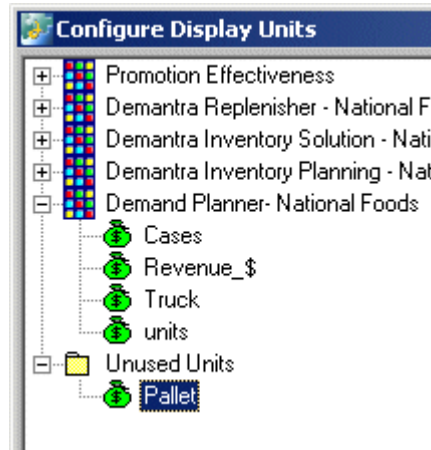
2. In the left side of the dialog box, click the name of any component that includes the unit. This expands the display so that you can see the units in that component.
3. Select a unit.
4. Click File > Delete.
 - Business Modeler prompts for confirmation.
5. Click OK.

To see which units a component uses:

1. Click Configuration > Configure System Units.

The Configure Display Units dialog box appears.

2. In the left side of the dialog box, click the plus sign (+) to the left of the component name. The hierarchy expands to display all the units that are used in this component.



To assign units to a component, see "Creating or Modifying a Component".

Associating Units with Levels

Before users can access a unit of measure, you must associate that unit of measure with each aggregation level with which it could conceivably be used. Within a worksheet, the user will be able to select any unit associated with any aggregation level that the worksheet uses.

There are two equivalent approaches you can use:

- You can use an option that associates a unit with all existing levels, and then you can make further changes for any levels that should not use this unit.
- You can associate the unit with each level one at a time.

The approach you choose depends on how many levels you have and how many of them should use a given unit.

To associate a unit with all levels:

1. Click Configuration > Configure System Units.

The Configure Display Units dialog box appears.

2. In the left side of the dialog box, click the name of any component that includes the unit. This expands the display so that you can see the units in that component.
3. Select a unit.
4. Click Link Unit to All Levels.
5. Optionally remove this unit from specific levels, if needed, as described below.

To associate a unit with a level:

1. Click Configuration > Configure Units for Levels.

The system then displays the following list, which includes one line for each existing level-unit association.

Level Name	Level Unit
Lowest Location Level	units
Lowest Location Level	Revenue
Lowest Location Level	Revenue \$
Lowest Location Level	Cases
Lowest Item Level	units
Lowest Item Level	Revenue
Lowest Item Level	Revenue \$
Lowest Item Level	Cases
Forecast User Status	units
Forecast User Status	Revenue
Forecast User Status	Revenue \$
Forecast User Status	Cases
Aggregation User Stat	units
Aggregation User Stat	Revenue
Aggregation User Stat	Revenue \$
Aggregation User Stat	Cases
Forecast System Statu	units

2. Click Add.
3. In the Group column, select the level.
4. Do one of the following:
 - Click the corresponding cell in the Group Unit column and select a unit value.
 - Click the appropriate cell in the Group Unit column and modify its value.
5. Click Save to save the configuration.
6. Click Close to close the dialog box.

To remove a unit from a level:

1. Click Configuration > Configure Units for Levels.

The system then lists all the existing level-unit associations.

2. Click the line that corresponds to the association you want to remove.
3. Click Delete.
4. Click Save to save the configuration.
5. Click Close to close the dialog box.

Configuring Time Units

Any Demantra solution has a base time unit (often weeks or months). Demantra provides some larger predefined time units, and you can add others. In general, there are two types of time units:

- Simple time units (such as quarters) are simple multiples of the base time unit. For these, you just provide a scaling factor. For example, for a weekly system, a quarter consists of 13 time units. These time units are assumed to divide evenly into one year, and Demantra automatically figures out which base time bucket each date belongs to.
- Data-dependent time units (such as 4-4-5 time units) require explicit data. That is, they must be assigned explicitly to each date in the system, within the Inputs table. For an example, see "Units, Indexes, and Exchange Rates".

To configure a simple time unit:

1. Click Tools > Maintain > Edit Time Resolution.
2. Click Insert.
3. In the Description column, type a name for the time unit.
4. In the Time Scale column, type the number of base time units in this new time unit. Ignore the Inputs Column field.
5. To save changes, click Save. Or to exit without saving changes, click Cancel.

To configure a data-dependent time unit:

1. Using a database tool, add a column to the Inputs table that indicates how to group the base time buckets.
2. Within the Business Modeler, click Tools > Maintain > Edit Time Resolution.
3. Click Insert.

4. In the Description column, type a name for the time unit.
5. In the Inputs Column field, select the column from Inputs that contains the data for this time unit. Ignore the Time Scale column.
6. To save changes, click Save. Or to exit without saving changes, click Cancel.

To delete a time unit:

1. Click Tools > Maintain > Edit Time Resolution.
2. Click a row in the Time Resolution screen.
3. Click Delete.
4. To save changes, click Save. Or to exit without saving changes, click Cancel.

See also

"Creating a Time Aggregation"

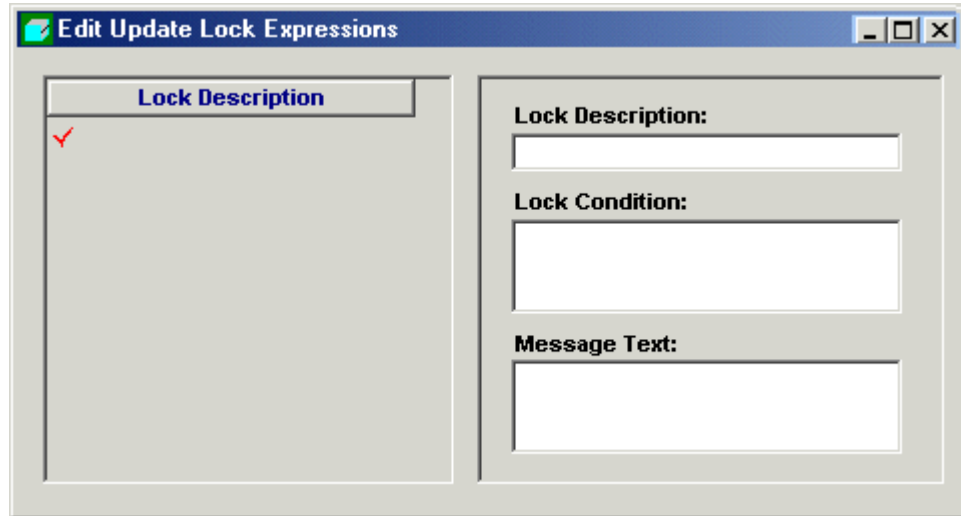
Configuring Update-Lock Expressions

An update-lock expression checks to see if a condition is met for each combination and if so, and prevents users from updating the database (saving the changes to the combination). This expression evaluates to either true or false. If the expression evaluates to true, then when the user tries to save the combination, a message is displayed and the worksheet data is not saved. The user must correct the data before the worksheet can be saved.

To configure an update-lock expression:

1. Click Configuration > Configure Update Locks.

The Edit Update-Lock Expressions dialog box appears.



2. Click the New button.
3. In the Lock Description field, add or edit the title of the expression.
4. Click the Lock Condition field.
The Client Expression Editor appears.
5. Create or edit an expression that evaluates to either true or false. See "Specifying Server and Client Expressions".

Note: Be sure that any constant values are expressed as the correct type of data (numeric, string, or date) for the expression you use. For example, be sure to use double quotes around constant string values if your expression uses a string-type series.

6. In the Message Text field, create or edit the message to be displayed when the update-lock expression returns true. This message should be as informative as possible so that the user knows which of his or her edits is responsible for the update-lock condition.
7. Click the Save button.

To delete the contents of a field:

1. Click the Delete button.

To refresh data from the database:

1. Click the Retrieve button.

Series and Level Integration

This chapter describes how to use the Integration Interface Wizard, which you use to import or export series data and level members.

This chapter covers the following topics:

- Before Using the Integration Interface Wizard
- Creating or Modifying an Integration Interface
- Creating a Data Import Profile
- Creating a Data Export Profile
- Specifying Series Data to Import or Export
- Creating an Export Profile for Any Level
- Creating an Import Profile for a General Level
- Deleting an Integration Interface
- Details of the Staging Tables
- Executing an Integration Interface
- Checking the Integration Queue

Before Using the Integration Interface Wizard

Before you use the Integration Interface Wizard, be sure to do the following:

- Read the "Integration" Chapter and make sure you understand the different options for importing and exporting data.
- Define the series and levels that you plan to import or export.
- If you are importing data, make sure you know what database procedures to run in order to keep the Demantra tables synchronized.

- Consider the following option provided by the Business Modeler: When you create an integration interface, the Business Modeler can automatically create a database user/schema that is pre-filtered to display only the data selected in that integration interface. This database user is provided for convenience and can be used within the Demantra solution or elsewhere. Demantra does not use this database user directly.

If you choose to create this user; the Business Modeler automatically uses the user name and password that you used to log into the Business Modeler. This means that you must use the following overall process for each integration interface you want to create:

1. Decide what database user name and password you want to associate with that integration interface.

Note: Make sure that the database does not yet 'contain a user with this name. The Business Modeler will not change an existing user.

2. Log onto the Business Modeler as a component owner. Create the user.
3. Log into the Business Modeler as that user.
4. Within the Business Modeler, create the integration interface. Be sure to enable the option Create DB User for Interface Objects.

Note: Whenever the Demantra user's password is changed, the database user's password must also be changed (if the user is an interface database user).

Creating or Modifying an Integration Interface

Before you perform data import or export, you define a reusable integration interface. This will contain any number of data profiles and level profiles. The profiles define the details of the integration.

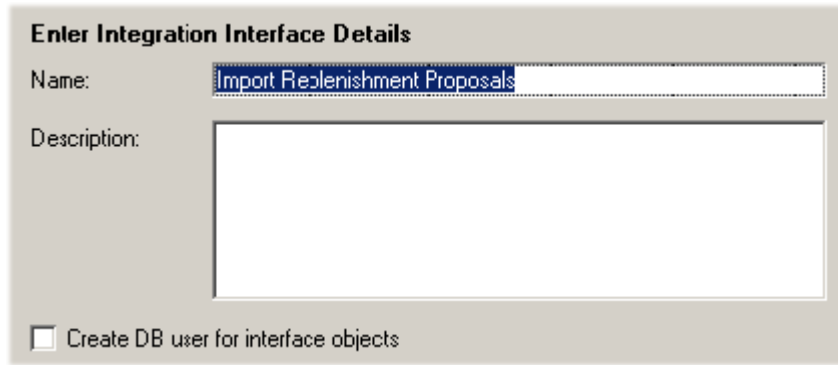
To create or modify an integration interface:

1. Click Tools > Integration Interface.
The Create/Modify Integration Interface screen appears.
2. Do one of the following:
 - Click New Integration Interface and then click OK. Or double-click New

Integration Interface.

- Click the button corresponding to the integration interface that you want to edit.

The Integration Interface Details screen appears.



Enter Integration Interface Details

Name:

Description:

☐ Create DB user for interface objects

1. Specify or edit the name and description for this integration interface.
2. Select or clear the Create DB User for Interface Objects check box.

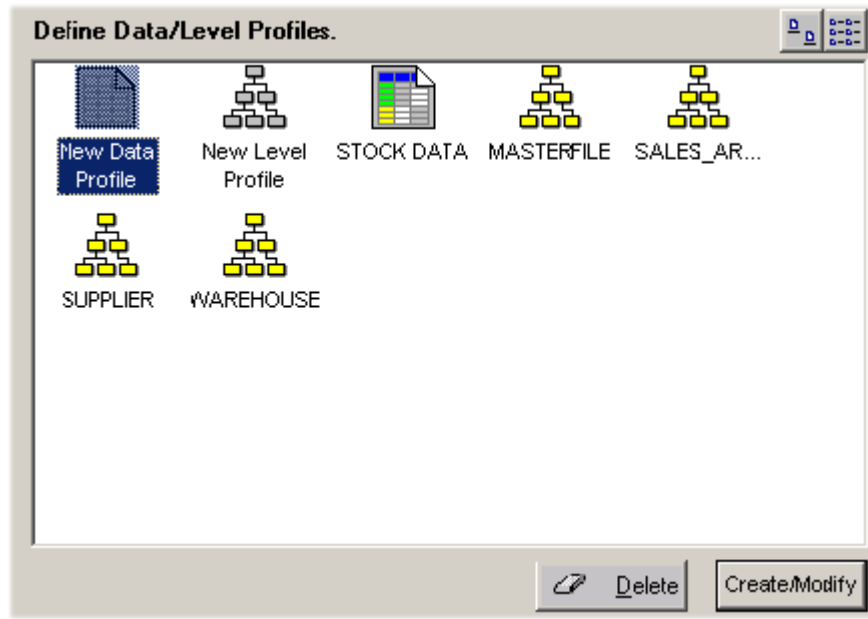
If selected, you will create a database user who can view only the data as configured in this integration interface.

Note: The new database user will receive the username and password with which you logged into the Business Modeler.

If the database user already exists, it is not changed.

3. Click Next. Or click Cancel and proceed to the next stage without saving changes.

The Integration Interface Selection screen appears. Here you define data profiles and level profiles within this integration interface.



4. To display the profiles in list format instead, click the list format icon on the upper right.
5. Do at least one of the following:
 - Define a data profile, as described in "Creating a Data Import Profile" or "Creating a Data Export Profile".
 - Define a level profile, as described in "Creating an Import Profile for a General Level".
6. Click Finish.

See also

Making Changes Available to Users

Creating a Data Import Profile

A data import profile describes how to import series data aggregated to a specific aggregation level or levels, with optional filtering. You can include sales series or promotion series, but not matrix series.

Important: Create import integration profiles using the Business Modeler installed *on the server* where Oracle Demantra is installed, and not using Business Modeler installed on the client personal computer.

In the case where the E-Business Suite Administrator navigates to the Collaborator Workbench > Planning Applications > Business Modeler, this deploys the silent install of the Business Modeler locally on C:\Program Files\Oracle Demantra Spectrum.

See Desktop and Business Modeler Automatic Install, page 23-4.

When the user creates an Integration Profile, the system creates a batch file on the machine from which Business Modeler is initiated. An executable step of the corresponding workflow references the batch file location on the server. In order for the workflow to process the batch file, the path to the batch file in the executable step of the workflow needs to reference the correct location (on the server). In the case where the user launches Business Modeler from the client PC, the workflow errors, because it looks for the batch file on the server, while it exists on the client PC.

To create a data import profile:

1. Open the integration interface, as described in "Creating or Modifying an Integration Interface".
2. Click Next to display the Define Data/Levels Profile screen.
3. Click New Data Profile and then click Create/Modify. Or double-click New Data Profile.

The Integration Interface Wizard displays the following screen.

Enter Data Profile Details

Data Profile Name: REPLENISH_REQMT_PWS

Description:

Presentation Type: Code

Integration Type: Import Export Data:

Create Worksheet: ☒

Import From File: ☒

Create Workflow: ☒ Workflow Group: Integration

4. Specify the following details for the data profile.

Name	Unique name for the data profile.
Description	Optional description.
Presentation Type	<p>Specifies how level members are identified in the profile:</p> <p>If you select Description, each member is identified by its description field.</p> <p>If you select Code, each member is identified by its code display field.</p>
Integration Type	<p>Import.</p> <p>For information on export profiles, see "Creating a Data Export Profile".</p>
Create Worksheet	Select this check box if you want Demantra to create a worksheet that you can use to view the data.
Import from file	Select this check box if you are importing data from a file. Or leave this check box deselected if you are importing data from a table that is in the Demantra database.
Create Workflow	Select this check box to automatically create a workflow schema that uses this data profile. This option is available only if your system includes the Workflow module.
Workflow Group	Specify the schema group that this workflow should belong to. This option is available only if your system includes the Workflow module.

5. If your system does not include the Workflow module, then the preceding screen includes the following field:

Create Process Batch	Select this check box to automatically create a batch file that uses this data profile. The name of the batch file is given in the final step of the wizard.
----------------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------

6. Click Next.
7. In the next set of screens, the Integration Interface Wizard prompts you for information about the data to import. This process is similar to creating worksheets. If you are not familiar with that process, see "Specifying Series Data to Import or

Export".

8. After you specify the data to import, the Integration Interface Wizard prompts you for additional import details.

Import to Forecast

Insert New Combinations: ☐

Populate Forecast Horizon: ☐

Split Proportions

☐ Matrix Proportions

☒ Actual Proportions

Table Name:

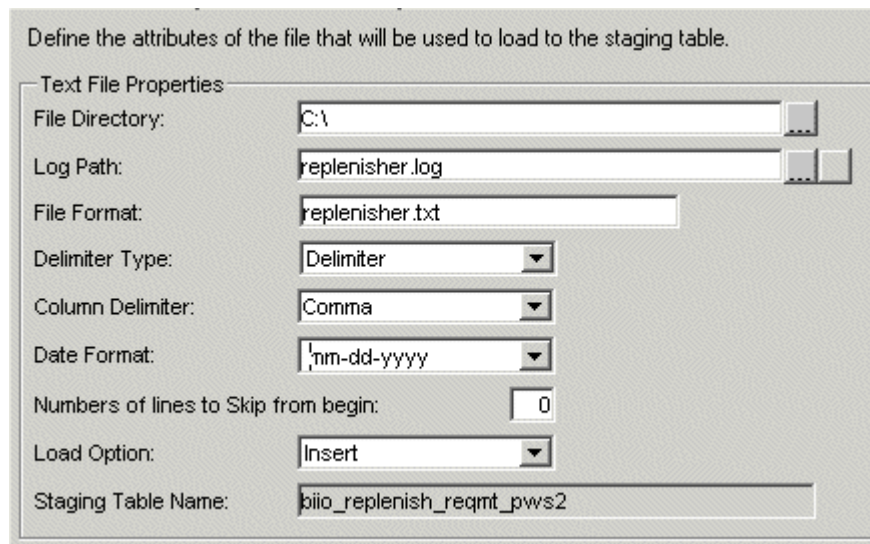
9. Specify the following information:

Insert New Combinations	<p>If selected, all new combinations will be imported into Demantra.</p> <p>The Insert New Combinations process creates new combinations in the MDP_MATRIX table only when both the item and location are provided in the data import. For example, if the Location part of the combination is missing, the process will be unable to create a new matrix combination for that record.</p>
Populate Forecast Horizon	<p>Inserts new forecast records into the sales_data table for the entire forecast horizon data for the new combinations. This option is available only if you selected Insert New Combinations.</p>
Split Proportions (ignored if you import at the lowest level)	<p>This option is relevant only if you are importing data at an aggregated level. Select one of the following:</p> <p>Matrix Proportions - Uses the precalculated proportions that are stored in the mdp_matrix table. This means that if your aggregate data is imported at a weekly granularity, the split at the lowest level will be based on the corresponding monthly proportions.</p> <p>This option shortens the calculation process before any split action. Also, if you use this option, you will not need to run the MANUALS_INS_INTEGRATION procedure.</p> <p>Actual Proportions - Calculates proportions based on historical data. This option splits the aggregate data down to the lowest level based on sales or forecast values at the lowest level in the time bucket of interest. If no sales or forecast data exists, matrix proportions will be used.</p>

Table Name	<p>The meaning of this field depends on whether you are importing data from a file:</p> <p>If you are importing data from a file, this field specifies the name of the internal staging table into which Demantra will import this data.</p> <p>If you are instead importing data from a table in the database, this field specifies the name of that table.</p> <p>In either case, Demantra creates the table for you.</p> <p>If you change this name, it must not include spaces.</p>
------------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Note: Make a note of the name of the staging table. For an introduction, see "Details of the Staging Tables".

10. Click Next.
11. If you are importing data from a staging table rather than a file, skip ahead to Step 17.
12. If you are importing data from a file, the Integration Interface Wizard prompts you for details about that file.



Define the attributes of the file that will be used to load to the staging table.

Text File Properties

File Directory: C:\

Log Path: replenisher.log

File Format: replenisher.txt

Delimiter Type: Delimiter

Column Delimiter: Comma

Date Format: mm-dd-yyyy

Numbers of lines to Skip from begin: 0

Load Option: Insert

Staging Table Name: pio_replenish_reqmt_pws2

13. Specify the following information:

File Directory	Directory that contains the source file.
Log Path	Directory and filename of the log file into which Demantra will write any messages when you execute this integration profile.
File Format	The name of the file from which you want to import data.
Delimiter Type	Choose either Fixed Width or Delimiter.
Column Delimiter	Choose the delimiter that separates columns of data. This option is enabled only if Delimiter Type is Delimiter.
Number of Lines to Skip	Specify the number of lines at the top of this file that Demantra should ignore. For example, if the file contains a single header line, specify 1.
Load Option	Choose one of the following: Insert—use this to add new rows to the staging table Replace—use this to replace existing rows in the staging table

14. Click Next.

15. If you are importing from a file, the Integration Interface Wizard displays a screen that shows additional details of the imported data.

Name	Type	From Pos	To Pos	Active
sdate	date			<input checked="" type="checkbox"/>
level1	varchar2			<input checked="" type="checkbox"/>
aggri_98	varchar2			<input checked="" type="checkbox"/>

This screen shows one row for each field that the imported data contains.

1. Specify which columns to import. By default, all columns are imported; to prevent a column from being imported, select the Active check box.
2. Specify the order of the fields. To move a field, click the field and then click the up or down arrows until the field has reached the appropriate location.
3. If you chose the fixed width option earlier, you must now specify the widths of each of these fields. To do so, enter the starting and ending column position.

16. Click Next.
17. The Integration Interface Wizard displays a screen that reviews your selections.
18. Review the displayed information and do one of the following:
 - To make further edits, click Back.
 - To finish the data profile, click Finish.

The Integration Interface Selection screen appears. See "Creating or Modifying an Integration Interface".

Creating a Data Export Profile

A data import profile describes how to export series data aggregated to a specific aggregation level or levels, with optional filtering. You can include sales series or promotion series, but not matrix series.

If you want to export a series that uses a client expression, you must first run the Business Logic Engine to evaluate the expression, split the resulting data to the lowest level, and save it to the database. If you are exporting data within a workflow, you use the BLE step type for this purpose.

To create a data export profile:

1. Open the integration interface, as described in "Creating or Modifying an Integration Interface".
2. Click Next to display the Define Data/Levels Profile screen.
3. Click New Data Profile and then click Create/Modify. Or double-click New Data Profile.

The Integration Interface Wizard displays a screen where you name the new data profile.

Enter Data Profile Details

Data Profile Name: REPLENISH REQMT - CW

Description:

Presentation Type: Code

Integration Type: Export Export Data: Full

Create Worksheet: ☒

Import From File: ☐

Create Workflow: ☒ Workflow Group: Integration

4. Specify the following details for the data profile.

Name	Unique name for the data profile.
Description	Optional description.
Presentation Type	<p>Specifies how level members are identified in the profile:</p> <p>If you select Description, each member is identified by its description field.</p> <p>If you select Code, each member is identified by its code display field.</p>
Integration Type	<p>Export.</p> <p>For information on import profiles, see "Creating a Data Import Profile".</p>
Export Data	<p>Select one of the following options:</p> <p>Full - use this to export all data</p> <p>Incremental - use this to export only changed data</p>
Create Worksheet	<p>Select this check box if you want Demantra to create a worksheet that you can use to view the data.</p>

Create Workflow	Select this check box to automatically create a workflow schema that uses this data profile. This option is available only if your system includes the Workflow module.
Workflow Group	Specify the schema group that this workflow should belong to. This option is available only if your system includes the Workflow module.

- If your system does not include the Workflow module, then the preceding screen includes the following field:

Create Process Batch	Select this check box to automatically create an external batch file that uses this data profile. The name of the batch file is given in the final step of the wizard.
----------------------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------

- Click Next.
- In the next set of screens, the Integration Interface Wizard prompts you for information about the data to export. See "Specifying Series Data to Import or Export".
- After you specify the data to export, the Integration Interface Wizard prompts you for additional export details.

Data Profile Export Properties
File and View properties for the export of the data

Export to File options

File Name: F:\CW-export.bt

Fixed Width: ☒ Delimiter: [Dropdown]

Select View Type for export

☒ View
☐ Materialized View

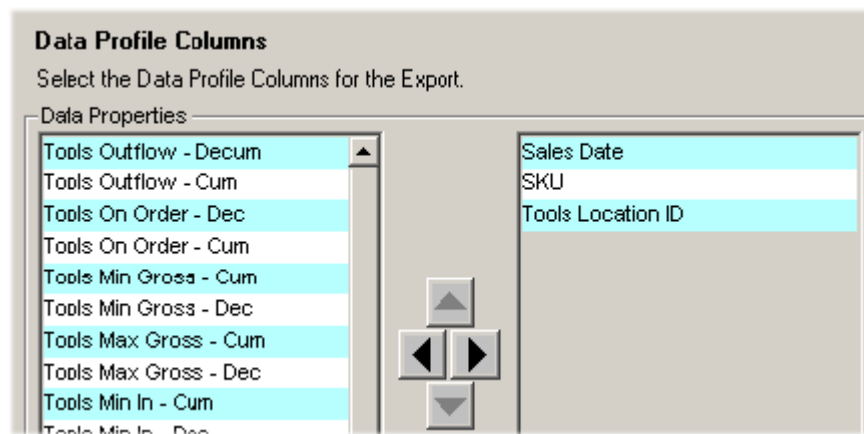
View Name: BEO_REPLENISH_REQMT_CW

- Specify the following information:

File Name	Full directory path and filename of the file to which you are exporting data. Note that Demantra appends a unique time stamp to the end of the file so that you can export multiple times and see each result.
Fixed Width	Select if you want to export data in fixed width format. Leave this option unselected if you want to specify a field delimiter instead.
Column Delimiter	Choose the delimiter that Demantra should use between fields when it exports the data. This option is enabled only if Fixed Width is not selected.
Select Type of View for Export	<p>Choose one of the following:</p> <p>View - Retrieves data from the database.</p> <p>Materialized view - Saves data as a cube allowing quick and flexible analysis. (This option is enabled only for Oracle.)</p>
View Name	<p>Name of internal table into which Demantra will export this data. If you change this name, it must not include spaces.</p> <p>Also, the view is created only after you run the export process. The columns in the view are not necessarily in the same order as in the exported file.</p>

10. Click Next.

The Integration Interface Wizard prompts you to specify the fields to export.



- For each field that you want to export, click that field in the left list and then click the right-pointing arrow.

2. To change the order of the fields, click a field and then click the up or down arrow.
 3. Click Next.
11. If you chose the fixed width option earlier, you must now specify the widths of each of these fields. The Integration Interface Wizard displays the following screen.

Data Profile Column's Width
Populate the Data Profile Column's Width to Export.

Field Name	Field Type	Field Width
Sales Date	Date	10
SKU	Char	10
Tools Location ID	Char	3

1. For each field, specify a numeric field width.
 2. Click Next.
12. The Integration Interface Wizard displays a screen that reviews your selections.
13. Review the displayed information and do one of the following:
- To make further edits, click Back.
 - To finish the data profile, click Finish.

The Integration Interface Selection screen appears. See "Creating or Modifying an Integration Interface".

Specifying Series Data to Import or Export

When you define a data profile for import or export (or both), a series of screens prompt you for information about the data.

Tip: It is generally good to import data at the lowest level possible.

If you need to load both sales series and promotion series, you probably should define separate data profiles for the two sets. You would likely want to import the promotion series at the promotion level, while you would import the sales series at some other level.

To specify series data for import or export:

1. The Integration Interface Wizard prompts you for the series to import or export in this data profile.

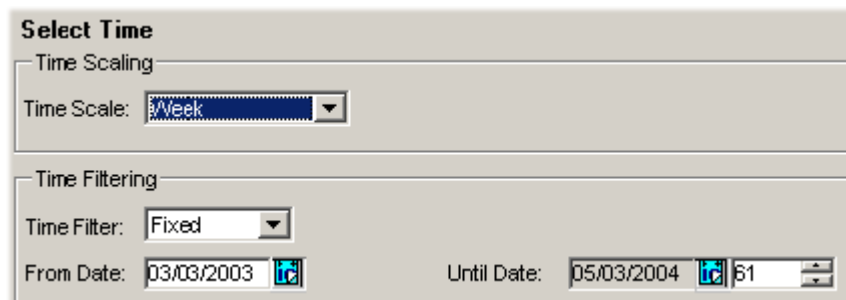


2. Click each series you want to include so that it appears with a check mark.

Note: Be sure to select only series that have an update field.

3. Click Next.

The Integration Interface Wizard prompts you for information about the time resolution and span of time to include in this data profile.



4. In the **Time Scale** box, specify the time resolution of the data to import or export.
5. In the **Time Filter** box, choose one of the following:
 - Choose **Relative** if you always want the import or export to use a time range

relative to the date when you run the integration profile.

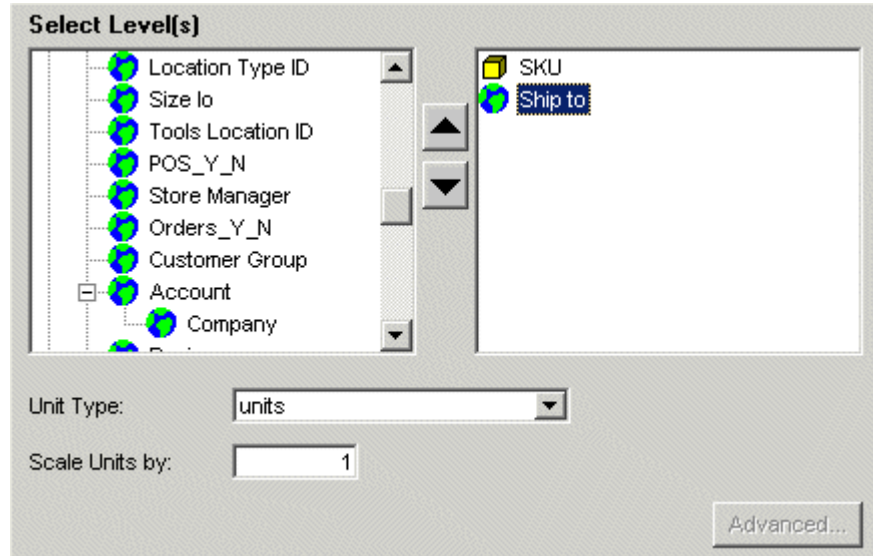
- Choose **Fixed** if you always want the import or export to use a specific time range, regardless of when you run the integration profile.

6. In the **From Date** and **To Date** boxes, enter values depending on the time filter you have chosen, as follows:

Time Filter	Box	Action
Relative	From Date/ To Date	Specify periods in both From and To with the current (computer) date as the reference point. For example: If the Time Scale is <i>Month</i> , and you want data starting from six months before today until six months after, enter -6 (negative) in From Date, and 6 in To Date.
Fixed	From Date	Enter a specific date as a starting point. This is enabled only from the calendar.
	To Date	Specify the number of periods you want to include, starting from the From date. The unit period is what you selected in Time Scale. For example: If the Time Scale is <i>Year</i> , From Date is 01/01/96, and you want data from then until 12/31/98, enter 3 in To Date.

7. Click Next.

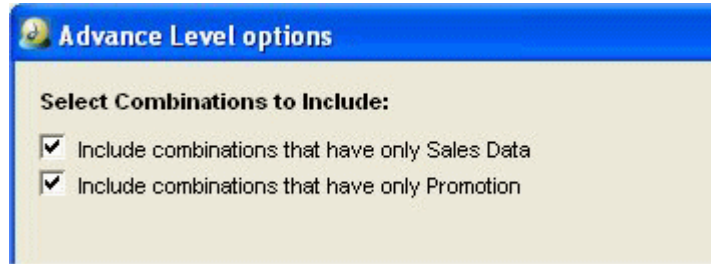
The Integration Interface Wizard prompts you for the levels at which the imported or exported data should be aggregated. For example, in the data profile shown here, data is imported for SKU-Ship to combinations.



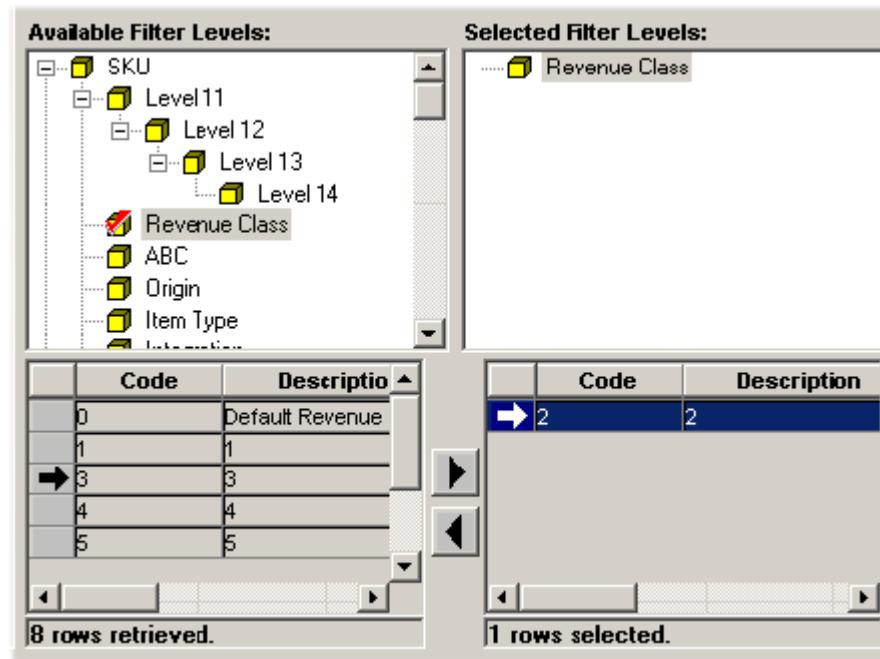
8. Double-click each level you want to use, so that it appears in the right column.
9. Optionally move a selected level to an earlier or later position in the right column.

Note: The order in which the levels are listed in the right column controls the order of the fields in the staging table that Demantra creates for this data profile.

1. In the Scale Units by box, specify the factor by which all data in the import or export is to be divided.
2. In the Unit Type box, select the unit of measure to use in the imported or exported data.
3. If the Index box is displayed, choose an index from the drop-down list.
4. The Index menu lists all the time-dependent indexes and exchange rates that are associated with this unit.
5. Optionally click the Advanced button in the lower right. Oracle displays a dialog box with additional options.



6. Select or deselect these check boxes as needed, and then click OK.
7. Click Next.
10. Now you can optionally filter the imported or exported data.



1. Find the aggregation level at which you want to filter data and move it from the Available Filter Levels list into the Selected Filter Levels list,
2. In the Available Members list, find a member that you want to include in the imported or exported data and move it into the Selected Members list.
3. Continue to move members from the Available Members list into the Selected Members list, until the latter list includes all the members you want.
4. Click Next.

11. If you are defining an *export* integration profile, optionally define one or more exceptions.

Select Exception

Exceptions		
Demand	<=	10000.0

Add **Delete** **Verify**

Use :

☒ **AND** ☐ **OR**

1. Click Add.
 2. In the new row, click the arrow to the right of the series box and select a series from the dropdown list.

Note: Not all series are necessarily available for exceptions, depending upon the series definition. See "Available for Exceptions".
 3. Click the arrow to the right of the operator box and select an operator from the dropdown list.
 4. In the number box, type the required number.
 5. You can apply additional exceptions. Select the AND or the OR radio button to specify that the relationship between the exceptions.
 6. Click Next.
12. Continue to define the data profile as in the following topics.
 - Creating a Data Import Profile: Step 8
 - Creating a Data Export Profile: Step 8

Creating an Export Profile for Any Level

You can define a profile that describes how to export the members and attributes of any level.

Note: An export profile creates a database view, and the data in that view is then exported to the specified export file. The view is created only after you run the export process.

To create a level export profile for any level:

- 1. In the Integration Interface Selection screen, double-click the New Level button.
- 2. Specify a name and description for the level profile.

Enter Level Profile Details

Level Profile Name:

export level

Description:

Presentation Type:

Code

Integration Type:

Export

Export Data:

Full

Create Workflow:

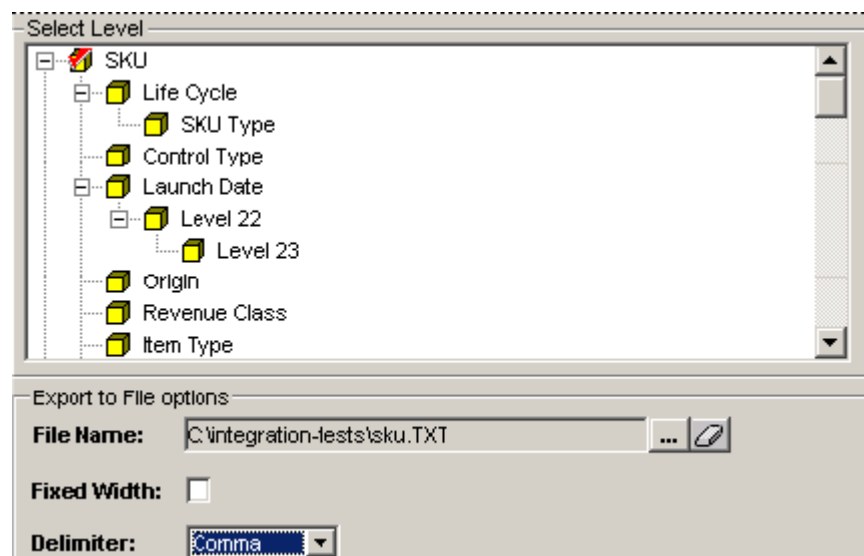
☐ Workflow Group:

Name	Unique name for the level profile.
Description	Optional description.
Presentation Type	<div>Specifies how level members are identified in the profile:</div> <div>If you select Description, each member is identified by its description field.</div> <div>If you select Code, each member is identified by its code display field.</div>
Integration Type	<div>Export.</div> <div>For information on level import.export profiles, see "Creating an Import Profile for a General Level".</div>

Export Data	<p>Select one of the following options:</p> <p>Full—use this to export all data</p> <p>Incremental—use this to export only changed data</p>
Create Workflow	<p>Select this check box to automatically create a workflow schema that uses this data profile. This option is available only if your system includes the Workflow module.</p>
Workflow Group	<p>Specify the schema group that this workflow should belong to. This option is available only if your system includes the Workflow module.</p>

3. Click Next.

- The Integration Interface Wizard displays a screen like the following.

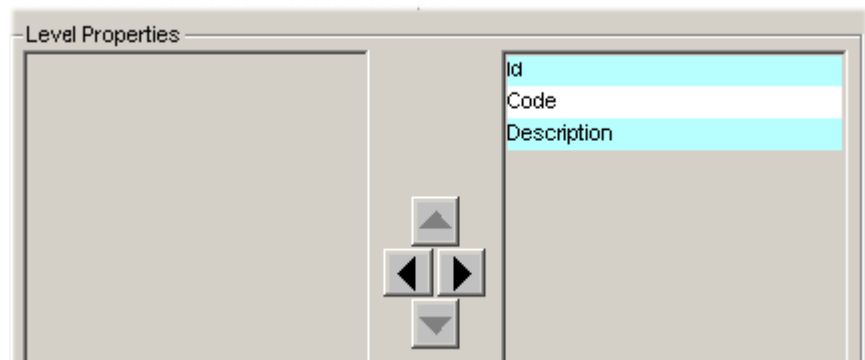


- Click the level to import or export.
- Specify information about the file to export to, as follows. This file does not have to exist beforehand:

File Name	Specify the path and filename of the export file.
Fixed Width	If selected, each data item will be trimmed or filled to a specific field width, as specified later.
Delimiter	A symbol used to separate the data elements in the file. Use this option only if you do not check Fixed Width.

6. Click Next.

The Integration Interface Wizard displays a screen that lists the columns you can export for this level. In general, these columns correspond to identifiers and attributes of members of this level.



1. For each field that you want to export or import, click that field in the left list and then click the right-pointing arrow.
 2. To change the order of the fields, click a field and then click the up or down arrow.
 3. Click Next.
7. If you chose the fixed width option earlier, you must now specify the widths of each field of the export file.

The Integration Interface Wizard displays the following screen.

Field Name	Field Type	Field Width
Id	Num	10
Code	Char	10
Description	Char	200

8. In the Field Width box, enter the required width of each field, in characters.
9. Click Preview Structure.

The Integration Interface Wizard displays a screen that reviews your selections.

10. Review the displayed information, and then do one of the following:
 - To make further edits, click Back.
 - To finish the data profile, click Finish.

The Integration Interface Selection screen appears.

See "Creating or Modifying an Integration Interface".

Creating an Import Profile for a General Level

For a general level, you can define a profile that you can use to import or export the members and attributes of the level.

- You can export members and attributes of a general level, but you cannot export the population attributes of the members. (The population attributes specify the item-location combinations to which each promotion applies.)
- An export profile creates a database view, and the data in that view is then exported to the specified export file. The view is created only after you run the export process.
- This import/export profile addresses only the members and attributes of the level. In the case of promotions, you generally also need to import or export the associated promotion series. To do so, create a data profile that includes the promotion series; see "Creating a Data Import Profile".
- When you import a member that already exists, Demantra updates the member with the new attribute values.

To create a level profile for an integration interface:

1. In the Integration Interface Selection screen, double-click the New Level button.
2. Specify a name and description for the level profile.

Level Profile Name: Import Promotions

Description:

Presentation Type: Code

Integration Type: Import Export Data:

Create Workflow: ☐ Workflow Group:

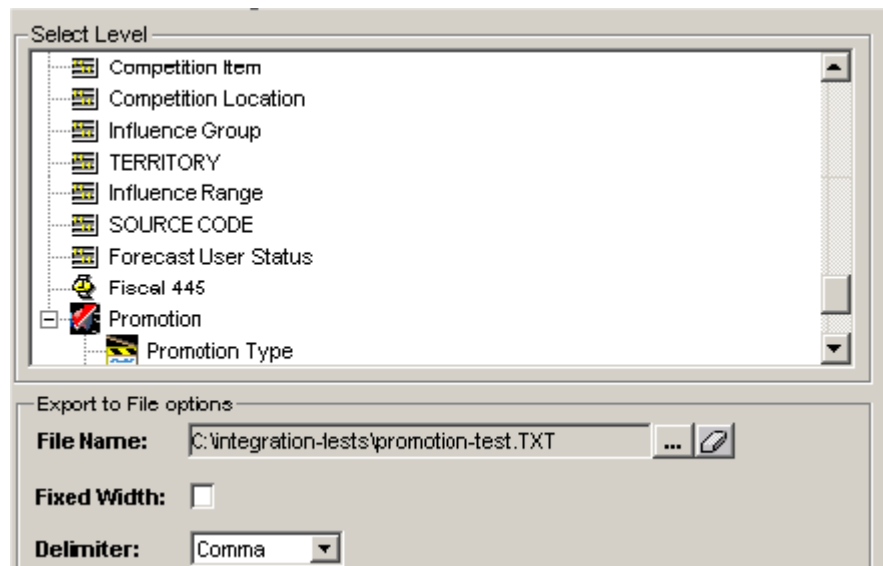
Insert Matrix Combination: ☒ Create fictive combinations: ☐

Name	Unique name for the level profile
Description	Optional description
Presentation Type	<p>Specifies how level members are identified in the profile:</p> <p>If you select Description, each member is identified by its description field.</p> <p>If you select Code, each member is identified by its code display field.</p>
Integration Type	<p>Import.</p> <p>For information on level export profiles, see "Creating an Import Profile for a General Level".</p>
Create Workflow	Select this check box to automatically create a workflow schema that uses this data profile. This option is available only if your system includes the Workflow module.
Workflow Group	<p>Specify the schema group that this workflow should belong to.</p> <p>This option is available only if your system includes the Workflow module.</p>

Insert Matrix Combinations	This option specifies whether to create the combinations of the associated population attribute, if those combinations do not already exist.
----------------------------	----------------------------------------------------------------------------------------------------------------------------------------------

3. Click Next.

- The Integration Interface Wizard displays a screen like the following.

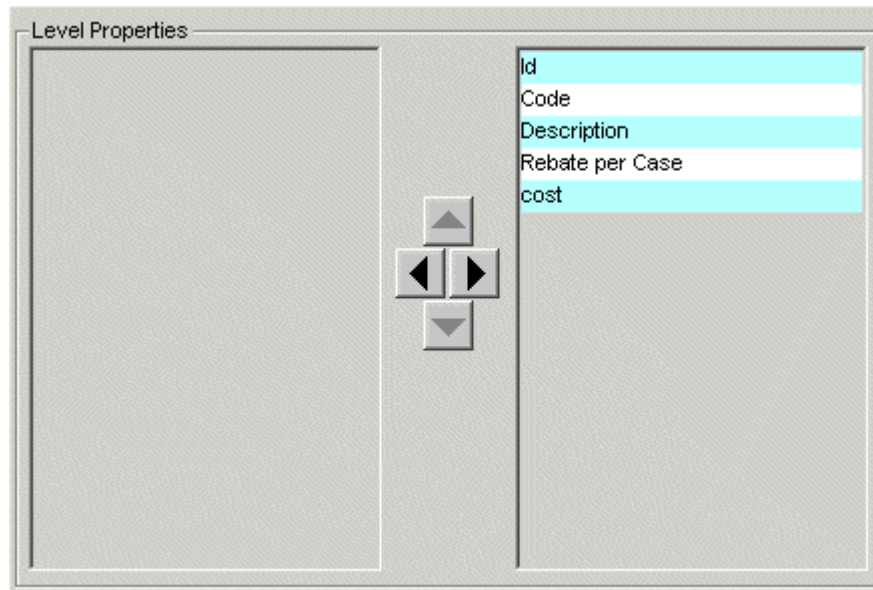


4. Click a general level.

5. If you want to export this general level, specify the following:

File Name	Specify the path and filename of the export file. This is the file to which the members and attributes of the general level will be exported. The associated population attributes are not exported.
Fixed Width	If selected, each data item will be trimmed or filled to a specific field width, as specified later.
Delimiter	A symbol used to separate the data elements in the file. Use this option only if you do not check Fixed Width.

6. Click Next.
7. If you specified an export file, the Integration Interface Wizard displays a screen where you specify the columns in the file. In general, these columns correspond to identifiers and general attributes of members of this general level. Population attributes are not shown.



For each field that you want to export or import, click that field in the left list and then click the right-pointing arrow.

8. To change the order of the fields, click a field and then click the up or down arrow. Click Next.

If you chose the fixed width option earlier, the Integration Interface Wizard displays the following screen.

Field Name	Field Type	Field Width
Id	Num	10
Code	Char	10
Description	Char	200

9. In the Field Width box, enter the required width of each field, in characters. Click Next.

The Integration Interface Wizard displays a screen where you specify the staging table or tables for this level.

Members Table:

Attribute Table:

You will import data into or export data from these tables. These tables have the following meaning:

Members Table	Table that contains the members of this general level
Attribute Table	Typically applies only at the lowest general level. This table contains the population attributes for each member. This table is used only for import and is ignored for export.

Note: Make a note of the names of the staging tables. For an introduction, see "Details of the Staging Tables".

10. Click Preview Structure.

The Integration Interface Wizard displays a screen that reviews your selections.

11. If you are importing data from a staging table rather than a file, click Create to create that staging table.
12. Review the displayed information and do one of the following:
 - To make further edits, click Back.
 - To finish the data profile, click Finish.

The Integration Interface Selection screen appears.

See "Creating or Modifying an Integration Interface".

Deleting an Integration Interface

To delete an integration interface:

1. Click Tools > Integration Interface.

The Create/Modify Integration Interface screen appears.

2. Click the button corresponding to the integration interface that you want to delete.

3. Click Delete.

Details of the Staging Tables

It is critical to understand the structure and purpose of the staging tables generated by this tool, especially as you will have to share this information with other groups or organizations who will provide the data to load. Demantra generates three kinds of staging tables, each with its own structure. This section describes the general rules that Demantra uses to create these tables.

Note: It is outside the scope of this documentation to describe how to load data into the staging tables.

Introduction

Oracle Demantra automatically creates staging tables as follows:

- For a data import profile, Oracle Demantra creates a staging table to contain the series data aggregated to the specified level or levels.
- For a general level import profile, Oracle Demantra creates a staging table to contain the level members that will be added. If the general level also includes a population attribute (as in the case of promotions), then Oracle Demantra creates an additional staging table to contain the population associated with each level member.

The Integration Interface Wizard initializes the names of these staging tables, but you can rename the tables within the wizard if needed. The default names start with `biio_`, but you should make a note of the names of your tables, as displayed within the Integration Interface Wizard.

You will also have to examine the structure of your staging tables, to ensure that the correct data is loaded into them. The SQL command `desc tablename` is useful for this task; see your database documentation. The following sections provide guidelines for understanding the fields in these tables.

For each staging table, the Integration Interface Wizard also creates an error table. The error table has the same structure as the corresponding staging table, with the addition of a new column that is meant to contain any error message associated with that record. The error table has the same name as the corresponding staging table, with `_err` appended to the end.

Staging Table for a Data Profile

The staging table associated with a data profile is intended to contain one row for a given date and combination. For that date and combination, this row contains data for

each series that is being loaded. The staging table has the following fields:

1. The first field (Sdate) is meant to contain the date for the data you are loading. This can be any bucket date that lies within the span of time specified by the data profile; see Step 2.

A bucket date is the calendar date of the first day within a given bucket. For example, if you are using a weekly system starting on Monday, all dates must fall on Mondays.

2. The next fields (Level1, Level2, and so on) are meant to contain either the codes or the descriptions for each member to which this data applies. These are character fields and are required.

- Pay attention to the Presentation Type used in the data profile. If the Presentation Type is Code, then these fields should contain the codes of the parent members. Likewise, if the Presentation Type is Description, then these fields should contain the descriptions.
- As you can see, these fields are not labeled to indicate the level to which they refer. Demantra considers the order in which you listed the levels within the integration wizard; see Step 3. The Level1 field contains the first level, Level2 contains the second level, and so on.

3. The rest of the fields are meant to contain the values for each series for the selected member(s) and date. The following rules apply:

- These fields are not required.
- The name of each field is the same as the **update field name** that you specified for the series. (See "Specifying Data Properties of a Series".) This can be confusing if your series names and update field names are not the similar.
- Except in the case of dropdown series, each of these fields has the same data type as the series itself.
- For a drop-down series, the import field is expecting to contain the value that is shown within the drop-down list in the worksheet (rather than the internal value that is instead stored). Oracle Demantra automatically converts the imported value to the appropriate internal value.

For example, consider a drop-down series that uses the Promotion Type Level as a lookup:

Promotion Type	
Promotion Type Id	Promotion Type Desc
42	DISPLAY
43	F. SHOPPER
45	FEATURE
44	F&D
46	TPR
0	default Promotion Type
47	NATIONAL TV

This series stores the values shown in the left column but displays the values shown in the right. When loading data for this series, you would provide values as shown in the right column.

Oracle Demantra looks up the value that you provide, finds the corresponding internal value, and imports the internal value into the database. If Oracle Demantra cannot find the value that you provide, this record is not imported and Oracle Demantra writes an error to the corresponding error table.

Main Staging Table for a Level Profile

The main staging table associated with a level profile is intended to contain one row for each member. That staging table has the following fields. For all these fields, field names depend on the level you are loading.

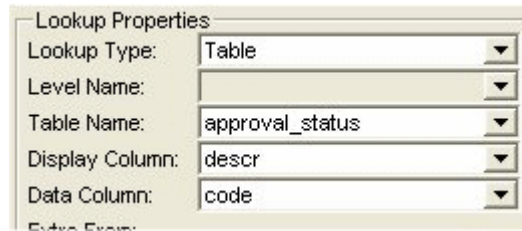
1. The first field is meant to contain the codes of the members that you are loading. This is a character field and is required.
2. The second field is meant to contain the descriptions of the members that you are loading. This is a longer character field and is required.
3. The next fields are meant to contain the **codes** for each immediate parent. These are character fields and are required.
4. The rest of the fields are meant to contain the attribute values for each member that you are loading. The following rules apply:
 - None of these fields are required.

Note: If you omit values for these fields, Demantra leaves the corresponding attributes as null. Oracle Demantra does not use any level or attribute defaults during import.

- The name of each field is the same as the **column name** that you specified for

the attribute. This can be confusing if your attribute names and column names are not the same.

- Except in the case of lookup attributes, each of these fields has the same data type as the attribute itself.
- For a lookup attribute, the import field is always a character field.
- For a lookup attribute of type table, the import field is meant to contain the same data as used in the display column of the table, as specified in the level editor here:



Lookup Properties

Lookup Type: Table

Level Name:

Table Name: approval_status

Display Column: descr

Data Column: code

For example, suppose that the approval_status table is as follows:

Code	DESCR
1	Submit
2	Submitted
3	Approved
4	Re-Review

In this case, the staging table is expecting to receive the description fields, such as "Submit." Oracle Demantra uses the lookup table, finds the corresponding codes, and inserts those into the level table. If Oracle Demantra cannot find the given field in the lookup table, the promotion is not imported and Oracle Demantra writes an error to the corresponding error table.

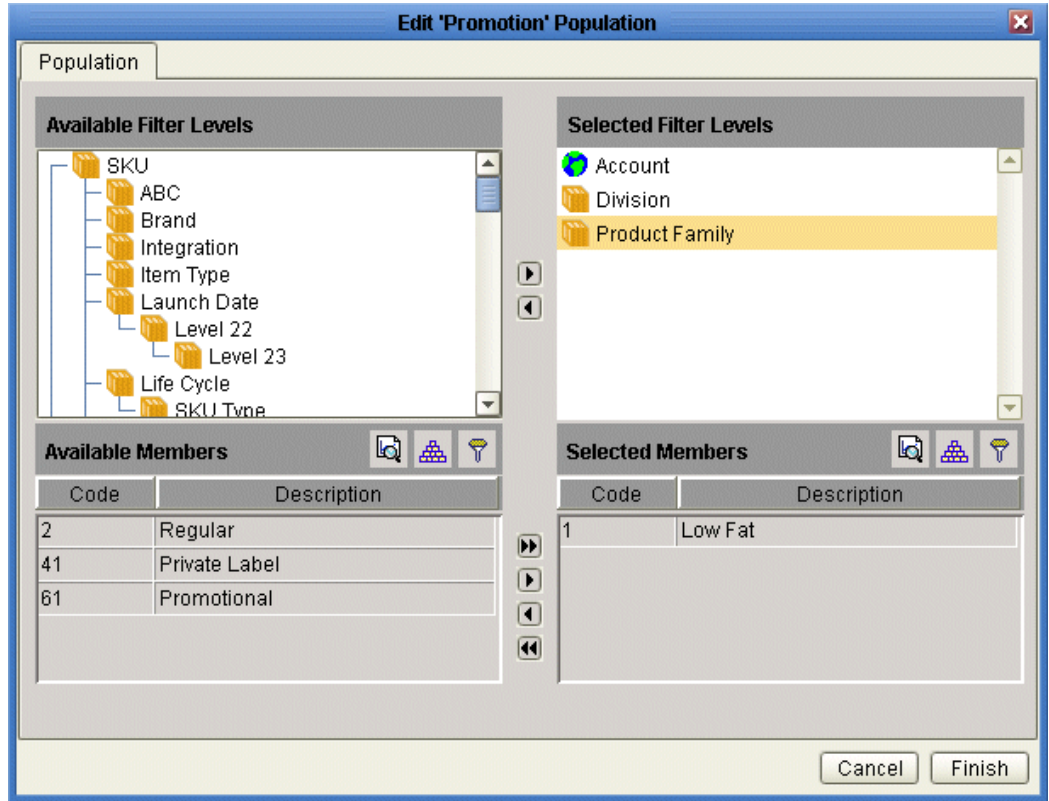
- For a lookup attribute of type level, the import field is meant to contain the description field of the member, as displayed in the right side of the Level Members screen:

Promotion Type	
Promotion Type Id	Promotion Type Desc
42	DISPLAY
43	F. SHOPPER
45	FEATURE
44	F&D
46	TPR
0	default Promotion Type
47	NATIONAL TV

For example, you would load the data "Feature" into the field for this level-type attribute. Oracle Demantra looks up this description in the level table, finds the corresponding code, and inserts that into the level table for the member that you are importing. If Oracle Demantra cannot find the given description, the promotion is not imported and Oracle Demantra writes an error to the corresponding error table.

Population Staging Table for a Level Profile

If you define an import profile for a promotion or other general level, Oracle Demantra also creates a staging table (for example, `biio_population`) to hold the population of the promotions that you are loading. This staging table describes the population of each promotion. Specifically, it contains the same information as this window:



For each promotion, the table can contain multiple rows. Each row specifies a level and a member of that level, just as the preceding screen does (the previous screen shows that this promotion is associated with the Low Fat member of the Product Family). This table has the following structure:

Field	Data Type	Purpose
LEVEL_MEMBER	varchar2 (40)	Code of the promotion (or other general level) member that you are loading.
FROM_date	date	Start date for this promotion member.
UNTIL_date	date	End date for this promotion member.
FILTER_LEVEL	varchar2 (50)	Name of a level, for example "Product Family" or "SKU".

Field	Data Type	Purpose
LEVEL_ORDER	number (15)	Use 1 for a location-type level or 2 for an item-type level.
FILTER_MEMBER	varchar2 (50)	Description of a member of this level, for example "Low Fat".

Executing an Integration Interface

Once you have created an integration interface, you can use it in either of two ways:

- You can incorporate the integration interface in a workflow controlled by the Workflow Manager.

To automate import or export, you add the appropriate integration interface to the workflow or workflows that you have defined for the users. You use the workflow step Transfer Step to initiate the import/export process. Internally, in this case, the APS layer performs the integration.

- You can use the separate Standalone Integration Tool, which is `Demantra_root/Demand Planner/Integration/aps.bat`. (This tool consists of a subset of the APS, packaged as an executable file.) To use this tool, open a shell, change to the directory in which the tool resides, and enter the following command:

```
aps.bat option "integration interface name" "profile name"
```

Here, option must be one of the following options, to specify what to import or export:

- EXPORT_DATA
- IMPORT_DATA
- EXPORT_LEVEL
- IMPORT_LEVEL

Also, integration interface name must be the name of an integration interface that has already been defined, and profile name is the name of a data profile or level profile within that interface

The double quotes are needed only if there are spaces within the interface name or the profile name.

Note: If you are importing data, and if the data profile specifies actual proportions (rather than matrix proportions), be sure to run the MANUALS_INS_INTEGRATION procedure after you run the integration interface.

Checking the Integration Queue

You can check the status of the integration tasks.

To view the integration queue:

- 1. Start Demand Planner or Demand Replenisher.
- 2. Click Window > Integration Queue Monitor.

The Integration Queue Monitor window appears. The window shows the Oracle Demantra tasks initiated for the integrated application. You can see the following information:

Timestamp	Date where the task was created.
ACTCODE	Codes specifying tasks to be performed in the integrating application.
PARAMSTRING	Parameter that is connected to the task.
Status	Status of the task, indicating the phase that the task is in.
Last Refresh	Time when this point of the process was reached.

Importing Supplementary Data

This chapter describes how to import data into the Demantra database by using Tools > Import File. You use this tool to import supplementary data such as lookup tables.

This chapter covers the following topics:

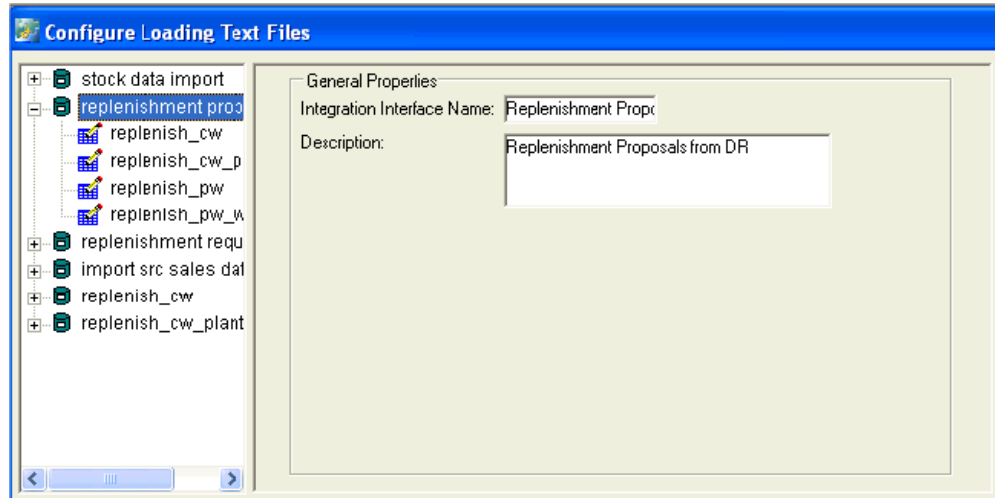
- Creating or Modifying a File Load Interface
- Deleting a File Load Interface
- Creating an Import Profile
- Creating the Import Batch File

Creating or Modifying a File Load Interface

An import interface consists of one or more profiles. Each profile corresponds to one table; note that multiple files can be loaded into a single table.

To create or modify a file load interface:

1. Click Tools > Import File.
2. The Configure Loading Text Files screen appears.



3. Next:

- To create a new interface, click File > New. Or click the New button in the toolbar.

A new file load interface is displayed in the left side of the dialog box.

- To edit an existing interface, click the interface.

4. Specify a name and description.

5. Make sure the interface includes at least one import profile. See "Creating an Import Profile".

Deleting a File Load Interface

To delete a file load interface:

1. In the tree pane, right-click the interface and then select Delete Import Interface.

Creating an Import Profile

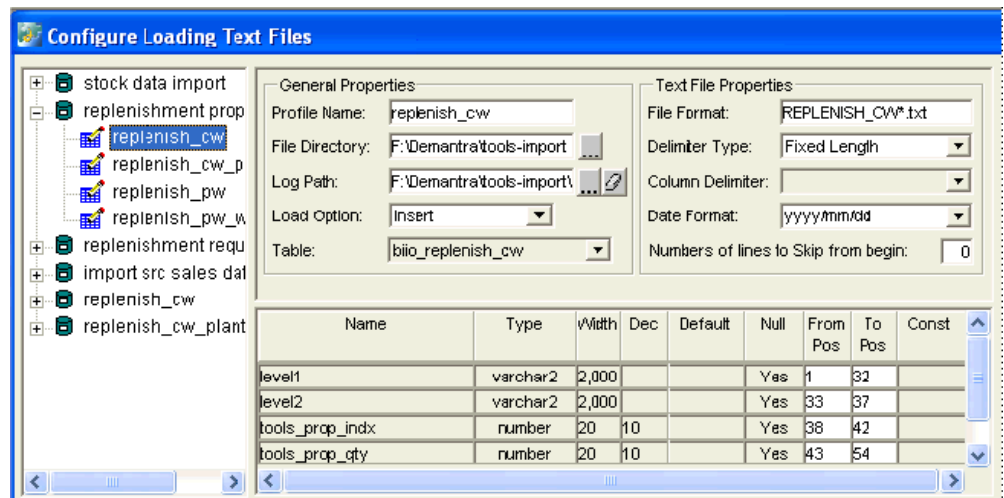
Each file load interface must include at least one import profile, which describes how to import data into a single table.

To create an import profile:

1. In the tree pane, right-click the interface and then select Create Import Profile.

A new import profile is added to the tree. The right side of the dialog box prompts

you for details on this profile.



2. Complete the fields as follows:

Profile Name	Specify a unique name for this import profile.
File Directory	The location of the files.
Log Path	Path and filename of the log file that will store any load errors.
Load Option	Select Insert to add the imported data after the last row in the table. Or select Replace to replace all the data in the table.
Table	The name of the existing table into which you want to load the text file data.
File Format	The file name, which can include wild cards.
Delimiter Type	Select either fixed length or delimiter.
Column Delimiter	Select the character used in this source file to delimit fields.

Date Format	Select the date format from the drop-down box. If this source file does not contain dates, this is optional.
File Name Format	Select more than one file through the use of a wildcard (*). For example, dcl_his*. * selects every file with the prefix dcl_his.
Number of lines to skip from begin	If there is a header, this gives the number of lines to miss at the top of the table.

If fixed length was selected as the delimiter type, the table column pane is activated.

Name	Type	Width	Dec	Default	Null	From Pos	To Pos	Const
application_id	number	10	0		No			
series_id	number	10	0		No			

- To specify table columns (where fixed length columns have been selected), complete the fields as follows:

From Pos	Position in the source text file where the field starts.
To Pos	Position in the source text file where the field ends.
Const	Constant column width. If selected, the From Pos and To Pos fields are disabled for editing.

- Click File > Save. Or click the Save button.

Creating the Import Batch File

To use a file load interface, you create and run a batch file that imports the data as described in the interface.

To create an import batch file:

1. Select a file load interface in the tree pane.
2. Click Create Load Batch.

The system displays a message is displayed when it creates the batch file.

The batch file is named `load_text_file_model_name.bat`

Note: You can also view the CTL syntax, which is the control file that SQL*Loader uses to map data into the database. Only experienced consultants should use this feature. To view the CTL syntax, click Show CTL Syntax.

Creating Workflows

This chapter describes how to create Demantra workflows, which are automated or semi-automated processes that you can use for a wide variety of purposes.

This chapter covers the following topics:

- Creating or Editing Workflow Schemas
- Parameters Used as Arguments for a Workflow
- Deleting Workflow Schemas

Creating or Editing Workflow Schemas

You can edit any workflow schema that you created, but you cannot edit schemas created by other users. When you edit a schema, the changes will be used in any new instances of the workflow schema. Any instances of the schema that are currently running are unaffected.

Note: If you are configuring a method that changes attribute values, the workflow must include an Edit Member Step as its first step. Otherwise, the changed values will not be saved to the database.

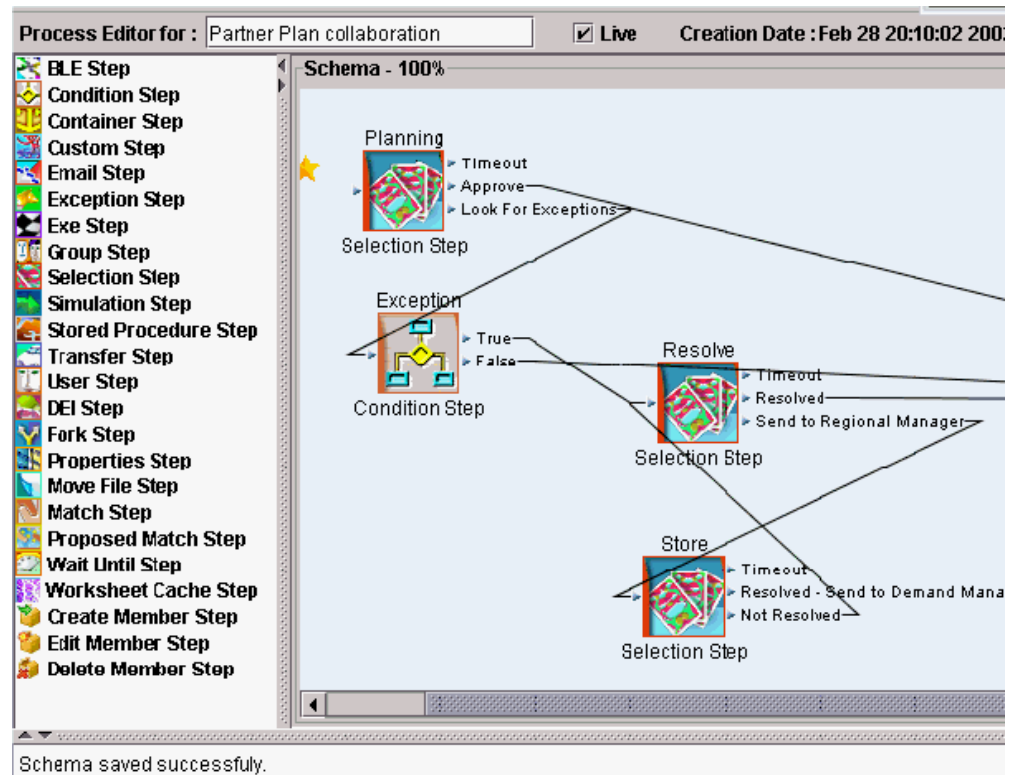
Note: You do not have to create the whole schema in one editing session. You can save your changes and then return to edit the schema.

Note: You can mark a schema as archived, which prevents it from being used. It is good practice to archive any schema that is not yet finalized, to prevent it from being used before it is ready.

To create or edit a workflow schema:

1. Log onto the Workflow Editor as described in "Logging into the Workflow Manager."
2. Do one of the following:
 - To create a new schema, click New Schema at the bottom of the page.
 - To edit a schema, click Edit in the row corresponding to that schema. Or click the schema name.

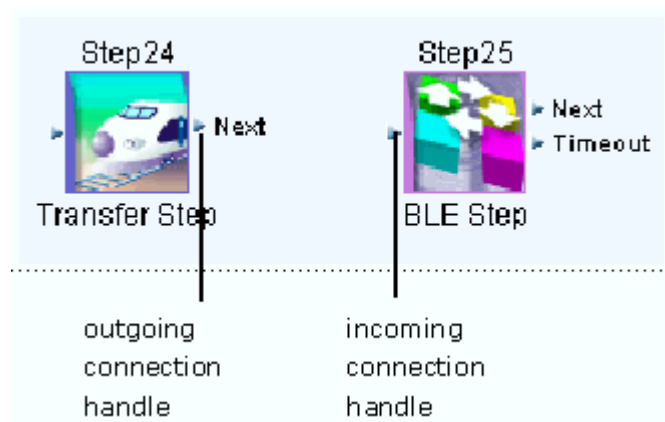
The Workflow Editor appears.



This screen has three panes:

- The left pane lists all the available steps that you can include in the workflow.
 - The right pane shows the definition of the workflow itself.
 - The message pane at the bottom of the window contains information, such as a warning of a failed validation or confirmation of a successful save.
3. Do one of the following:

- To make the schema to be available for use, select the Live check box.
 - To make the schema unavailable, clear the Live check box.
4. Add the required steps to the workflow. To add a step to the workflow:
 1. In the left pane, double-click the icon corresponding to the step.
The Workflow Editor displays a popup window that prompts you for information about the step.
 2. Specify the step properties. The properties depend on the kind of step you are adding. For details, see "Workflow Steps".
 3. Click OK.
The step is added to the right pane.
 4. Within the right pane, optionally reposition the step icon to improve readability.
 5. Connect the steps in the required order. To link two steps, click the outgoing connection handle of the first step and drag to the incoming connection handle of the next steps.



Note the following rules:

- An outgoing connection handle can be linked to only one following step. The Container Step is the exception.
- A step that is contained within a Container Step cannot have any of its outgoing connection handles connected to other steps.
- The Selection Step has multiple outgoing steps. To define the outgoing connection handles and to add the links, you use the properties dialog box. See

"Defining and Linking to Selections".

6. To specify which step starts the workflow, right-click the step and then select Set as Start step.

This step is labelled with a yellow star. If a different step had previously been marked as the starting step, that mark is cleared automatically.

Each schema must specify a step from which the execution starts.

7. To save your workflow to the database, click Save. Or click Back to cancel your changes.

Note: When you save a schema, it is automatically checked for validity. You cannot save an invalid schema. Click Verify to verify before saving.

8. Click Back to return to the main screen.

To edit an existing step:

1. Right-click the step and then click Properties.

2. To delete a step

Click the step and press Delete.

3. To delete a link

Click the link and press Delete.

See also

"Managing Workflows"

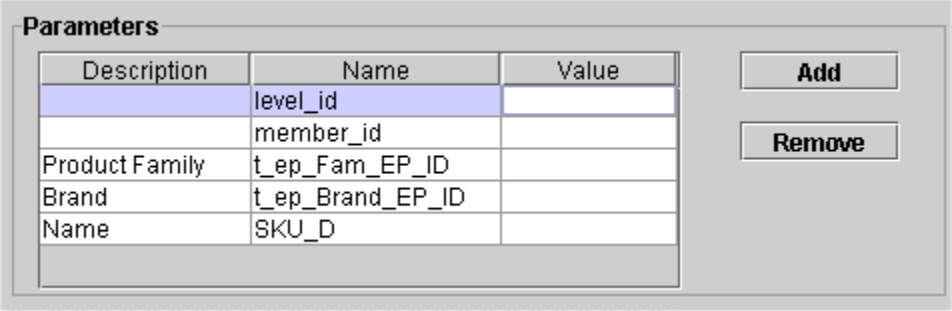
Parameters Used as Arguments for a Workflow

When you configure a workflow as a method, Demantra can pass arguments in memory to any method. Considered as a group, these arguments are the context dictionary for the method. For each argument, Demantra passes a variable name and its associated value. The available arguments generally fall within three categories:

- System information, such as the ID of the worksheet from which the method was launched
- Member information, such as the unique identifier of the member
- User inputs, such as specific, keyed-in details provided in the method input dialog

box. For example, a maximum budget can be supplied to the Optimization method.

In order to make these arguments available to a workflow step, you must explicitly configure the variables that each workflow step should receive. To do so, you type each variable name in the Name column of the Parameters list for that step, as follows:



Description	Name	Value
	level_id	
	member_id	
Product Family	t_ep_Fam_EP_ID	
Brand	t_ep_Brand_EP_ID	
Name	SKU_D	

Buttons: Add, Remove

In this example, the first two arguments are standard member variables, from the table in "Available Arguments". These arguments can be used in any method.

The remaining three arguments are input variables; these variables refer to attributes of the member. Specifically these are the names of the columns in which these attributes are stored (Product Family, Brand, and Name).

Note: In the Parameters list:

- The parameter names are case-sensitive.
- The descriptions are not used by the method.
- If a value is null in this table, then the value is taken from the member from which the method was launched.
- If the value is not null, then it is used instead of the value taken from that member.

Dictionary Step

Functionality to update member attributes as part of a Workflow process can be beneficial:

- As part of a Workflow to load new data from an external data source, such as a Supply Plan from Strategic Network Optimization. For example, set the status of Supply Plan members to 'Load In Process' to indicate to users who are viewing data that this data is being updated.
- As part of a Workflow to approve a Supply Plan. For example, set the status of the Supply Plan members to 'Approved'.

The dictionary allows for get and set capabilities on member attributes. A custom step called the 'DictionaryStep' (com.demantra.workflow.step.DictionaryStep) allows users to specify an attribute name, and a value to assign to that attribute.

To accomplish this, two parameters must be specified:

- **Name** – Attribute name. This is the name of the parameter on the dictionary that will be modified. When a level method is executed, it places all the level attributes as parameters on the dictionary. The attribute names are the internal column name in uppercase letters. The reference is case insensitive. To support passing parameters between steps, the dictionary also adds the parameter to the dictionary, if the attribute does not already exist on the member.
- **Value** – Attribute value. This is the value that the attribute receives. The dictionary step modifies the value only if the attribute type matches the value. In other words, if the name is of type "character," it can be updated with any numeric and alphanumeric character. If the name is of type "number," it will only allow numeric updates. Values can also reference a different attribute, in this case you enclose the reference attribute internal name with parenthesis {}.

Protected attributes.

The dictionary step assumes the following attributes as protected because of dangerous side affects of modifying them:

- member_id
- current_worksheet
- METHOD_STATUS
- appserver_root
- application_root
- create_combinations_param
- default_filters
- combination_path
- view_name
- user_id
- worksheet_filter
- level_id
- ws_id

The reference to the attribute value can be one of the following:

- A fixed varchar or numerical value, such as Approved or 10,000. Strings do not require quotes around them.
- A lookup on another attribute value. These are indicated using parenthesis {}, such as {other_attribute}. If a token is used, the step will look up the value of {other_attribute} and apply that value to the attribute named, such as {attribute_name}. In this case, the attributes must be of the same type, such as varchar, number, or date.
- Lookups on other tables. For complex situations such as attributes that are lookups on another table, the value itself is stored in Oracle Demantra as an integer that maps to a value, such as 'Unplanned', 'Planned', 'Closed'. The dictionary step works in such a way that the value can be specified as the lookup value, such as 'Unplanned' and not the integer. This makes workflows easier to review and understand. In other words, it is easier to parse a workflow in which a step sets the status to 'Planned' than one that sets the status to '1' – the internal id of the 'Planned' status. If the attribute is defined as a lookup on a set of values, Status = {Unplanned, Planned, Closed}, the set value must be one from this set. Otherwise an exception will be thrown.
- A date attribute.

Broader lookups are not supported. This means the value cannot be a SQL query, such as:

```
value = "select name from TABLE_A where x=y"
```

The DictionaryStep ensures the value is an acceptable value for the attribute named. For example, if the attribute is of type 'number', the value must be a number. If the attribute is a lookup on a set of values, the value must be one from this set.

The value is set in the dictionary. In order to ensure the value is saved to the dictionary, the final step in the workflow must be an 'Edit Member' step to ensure the dictionary context of the level method is saved back to the database.

Logging is handled by removing the comment "appserver.workflow.general" in the logconf.lcf file.

In case there are conflicts of parameter names a properties file ("DictionaryStep.properties") in conjunction with DictionaryStep in Common.jar that controls the name, value parameters.

To leverage the Dictionary Step functionality:

1. Login to Workflow Manager.
2. Create a New Schema by selecting New Schema.

3. On the left hand side, select Custom Step from the list of available steps.
4. Give the step a unique identifying name. Specify the class name as `com.demantra.workflow.step.DictionaryStep`. Also specify the name of the attribute the step will set and the value; rows are added to the parameters table by selecting Add on the right hand side.

Example

In the example below, the DictionaryStep will lookup the attribute/parameter "scenario_status" on the dictionary, and then set it to "approved".

Custom Step's Properties

Custom Step

Step Id DictionaryStep

Class Name: com.demantra.workflow.step.DictionaryStep

Parameters

Description	Name	Value
	name	#scenario_status#
	value	approved

Add **Remove**

Recovery: Ask ▼

General **Time**

Mandatory Fields

OK **Cancel**

Important: In order for methods to be able to save values to the database there should always be an "EditMemberStep" preceding the DictionaryStep.

```
##
## DictionaryStep properties
##

# the step parameter name-token used for DictionaryStep
step.param.name=name
# the step parameter value-token used for DictionaryStep
step.param.value=value
# the step parameter reference tokens
step.param.reference.left={
step.param.reference.right=}
# log4j logger category to use
logger.category=appserver.workflow.general
```

Level Methods. The ability to set an attribute value through a level method continues. This functionality supports workflow-based setting of member attributes, such as setting values implicitly through a workflow, versus explicitly through a user-initiated method.

Relative Path Requirements

Some workflows contain executable (.exe) steps that point to specific batch files to be executed. Since those paths are absolute, there is a need to correct paths to the executable files for each installation. In order to remedy the use of an absolute path in workflow steps, Oracle Demantra provides support for a *relative* path to files or applications referenced by workflow steps in the Demantra Workflow Manager.

This parameter is automatically assigned to every launched workflow dictionary. It can be used to define a *physical root* path to a folder containing key files or executables leveraged by Workflow schemas in the Workflow Manager.

Relative path requirements fall under two cases:

- Case 1: Relative to web application folder structure, in other words, one folder above the WEB-INF folder
- Case 2: Relative to the Demantra application folder, in other words, the folders in which the engine, Demand Planner desktop, Business Modeler and the integration standalone aps.bat are installed.

To accommodate these two relative path requirements, two new path tokens are provided. Every running workflow schema will have these parameters in their dictionary. They will be automatically assigned by the Workflow process creator. Using these parameters eliminates the need to change workflow steps that run external files. In most cases, the reference to these files can now be relative to the install folder.

Case 1: Using the appserver_root Parameter for Web Application

The appserver_root parameter, located within the sys_params table, provides support for a *relative* path to files or applications referenced by workflow steps in the Demantra Workflow Manager. If the appserver_root parameter is null, or empty, then its default value is one folder above the WEB-INF folder of the Demantra application install.

Note: Typically the folder located directly above WEB-INF is labeled as the 'demantra' folder, but this name can be changed during installation.

Workflow token #appserver_root#

1. By default, this token is calculated dynamically from the context of the Demantra WEB APPLICATION, in other words, one folder above the WEB-INF folder. When the Oracle Demantra application server starts, a servlet is executed that calculates the physical path to the root of the virtual folder where Demantra is installed, and

then assigns it to a global variable that is visible in the application scope.

2. Parameter value: The parameter `appserver_root` exists by default in `SYS_PARAMS` with a null value. If its value is non null, it will take precedence over the previously calculated value.
3. Parameter description: The directory where the Demantra Web Application is deployed. If it is NULL, the default is one folder above WEB-INF
4. Security: Regular users can read the parameter values in the Business Modeler, but only consultants can modify them.

For example, the relative path: `#appserver_root#\optimization`,

where `#appserver_root#` is defined as `C:[demantra_install_directory]\demantra`

results in the physical path: `C:[demantra_install_directory]\demantra\optimization`.

In this way, workflows can be pre-configured in an application independently of where the application is eventually installed. In other words, C: drive versus D: drive, Oracle folder versus demantra folder, and so on.

The `appserver_root` parameter can then be used by the executable step in a workflow schema to run an executable file independently of where the application is eventually installed, thus achieving functionality that is outside the prepackaged scope of the Demantra platform.

For example: `appserver_root='F:\Program Files\[demantra_install_directory]\Custom'`. This takes precedence and every path that references the `#appserver_root#` token is calculated relative to this parameter.

Note: This functionality works "out-of-the-box" on all Windows or LINUX or UNIX files systems, provided Demantra is installed cohesively, in other words, under one root folder. Any arrangement deviating from that will require custom configuration.

Case 2: Using the `application_root` Parameter for Demantra Application

1. By default, this token takes its value from the existing installer-populated parameter "AppServerLocation".
2. Parameter value: An additional parameter `application_root` in `SYS_PARAMS`. Its default value is null. If its value is non null, it will take precedence over AppServerLocation..
3. Parameter description: The directory where the Demantra Application is deployed. If it is NULL, the value is taken from the AppServerLocation parameter setting.

4. Security: Regular users can read the parameter values in the Business Modeler, but only consultants can modify them.

For example: application_root='F:\Program

Files\[demantra_install_directory]\Custom'. This takes precedence and every path that references the #application_root# token is calculated relative to this parameter.

Example

Example 1:

Assuming you have following path to Demantra WEB-INF:

F:\[demantra_install_directory]\Collaborator\demantra\WEB-INF,

#appserver_root# will be dynamically assigned a value of:

F:\[demantra_install_directory]\Collaborator\demantra

Example

Example 2:

In order to run the Engine from an Exe Step, the following command line would be leveraged:

```
#application_root#\..\..\Demand Planner\Analytical  
Engines\bin\EngineManager.exe
```

- The \..\..\ path segment 'backs out' of the 'demantra' folder, and then navigates to the appropriate Engine folder.
- Note that the workflow parameter is referenced as a token, #application_root#.

Important: The selection of the value for the application_root parameter must be such that all files referred to by the Workflow are in folders below the common application_root.

Relative Path to Promotion Optimizer

To run Promotion Optimization in a new install, configure the parameters in the OPLstep of the Call Promotion Optimizer workflow schema to leverage the #application_root# parameter.

For example, the following parameters are passed to the OPLStep:

- Parameter: MODEL_PATH
- Value: #application_root#\optimization\OPL\promoopt.opl

By default, the #application_root# parameter is left blank, or null. Therefore, if Demantra is installed as C:\[demantra_install_directory]\ with a virtual directory of

'demantra', this example parameter value results in the physical path:
C:\[demantra_install_directory]\Collaborator\demantra\optimization\OPL\promoot.opl

Deleting Workflow Schemas

You can delete any workflow schema that you created, as long as no instances of that schema are currently running. You cannot delete schemas created by other users.

To delete a workflow schema:

1. Log onto the Workflow Editor as described in "Logging into the Workflow Manager".
2. On the Workflow Management page, click Delete next to the schema.
3. Click OK to confirm the deletion.

Configuring Methods

This chapter describes how to configure methods that the users can run within worksheets or within a Members Browser content pane.bp

This chapter covers the following topics:

- Configuring a New Level Method
- Passing Arguments to a Method
- Modifying a Level Method
- Deleting a Level Method

Configuring a New Level Method

Note: For information on setting the Security Threshold for a method, see the section *Custom Methods* in the chapter *Levels*.

To configure a level method:

1. Define a workflow schema to use as the method. See "Creating or Editing Workflow Schemas."

Note: If you are configuring a method that changes attribute values, the workflow must include an Edit Member Step as its first step. Otherwise, the changed values will not be saved to the database.

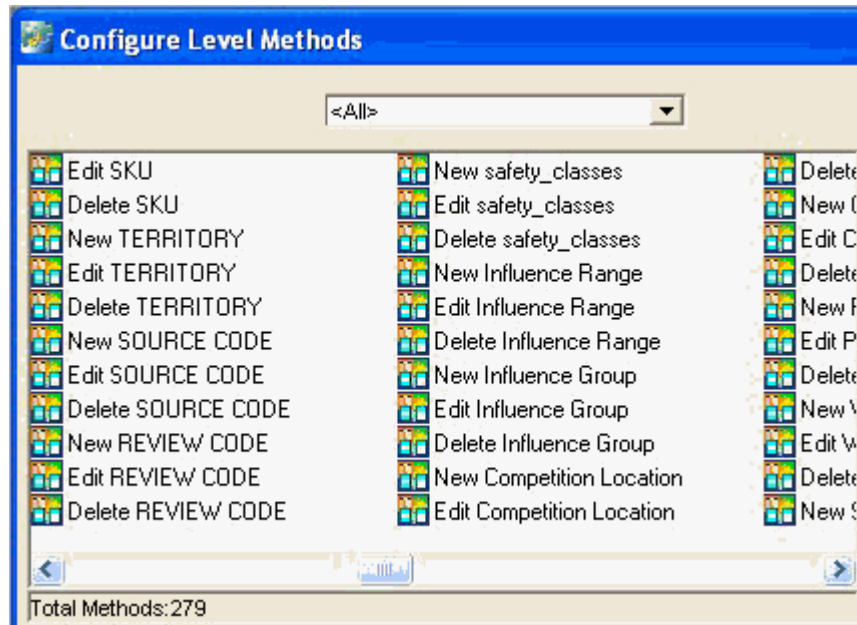
2. If the workflow includes a Custom Step, create the Java class that the step should invoke.
3. If this method should be available only within a specific worksheet, define that worksheet. See the Oracle Demantra Demand Management User's Guide or other user manual.

If the method applies to all worksheets, this step is not necessary.

4. Log into the Business Modeler.
5. Click Configuration > Configure Methods.

The system displays a screen showing the existing methods, including all the predefined methods.

6. Optionally click the Detail Style icon to re-display this screen with the full method names:



7. Optionally click a level name in the drop down list at the top of the screen. The screen is re-displayed with only the methods associated with that level.
8. Click the New Method icon and then click OK. Or double-click New Method.
The first screen is General Properties.

Welcome to the Level Method Wizard

This wizard lets you configure workflow schemas as level methods that the user can execute from within worksheets.

Name:

Level:

Enabled at:

Worksheet:

Workflow:

Display in menu: ☒

Synchronous WF: ☒ Refresh Cache: ☐

Dialog invoke button label:

Action when initiated:

Action when complete:

Method type:

Destructor type:

Execute client class:

9. Complete the fields in this screen as follows:

Constructor	Prompts the user for values of the attributes of the new member and then adds the member in the database.
Destructor	Removes the member from the database.
Edit	Prompts the user for new values of the attributes for this member and then saves the changes.
View	Displays the values of the attributes for this member.
Custom	Optionally prompts the user for new values of the attributes for this member and then runs a workflow.

Name	Name of this method, to display in the right-click menu of the worksheet. The method will be visible only in the Web-based worksheets.
------	----------------------------------------------------------------------------------------------------------------------------------------

Level	Level at which this method will be available.
Enabled at	<p>Specifies where this method will be available.</p> <p>All to make this method available in all worksheets.</p> <p>Worksheet to make this method available in a single worksheet.</p> <p>No to make this method unavailable.</p>
Worksheet	Worksheet where this method will be available.
Workflow	The workflow that will be executed when users run this method. This field is required for methods of type Custom.
Display in menu	Deselect this check box if you want to hide this method or leave it selected to have it displayed in the right-click menu, as specified.
Synchronous WF	<p>Specifies whether this workflow should be run synchronously or asynchronously.</p> <p>If the workflow runs synchronously, that means that before doing anything else, the user must wait until the workflow has completed.</p>
Refresh Cache	Specifies whether this method should refresh the local members cache.
Dialog invoke button label	<p>When the user selects this method from the right-click menu, the worksheet displays a dialog box that asks the user to confirm whether to proceed or not.</p> <p>This option specifies the text on the "OK" option.</p>

Action when initiated	<p>Specifies what Demantra will do when the method is initiated. The choices are as follows:</p> <p>Do nothing (typically setting)</p> <p>Save Data (immediately saves the worksheet data, automatically)</p> <p>Ask to Save (displays a dialog box to ask if the user wants to save the worksheet data immediately)</p> <p>Explanation: Level methods operate on the server. If you create a level method that does something based on client expressions in the worksheet, it is necessary to save the worksheet data before launching that method. The Save Data setting is useful in such a case.</p> <p>In other cases, you may want to give users the option of saving data before running the method. For these cases, use the Ask to Save setting.</p>
Action when complete	<p>Specifies what Demantra will do when the method has completed its execution. The choices are as follows:</p> <p>Reload (reruns the worksheet)</p> <p>Reload and Message (reruns the worksheet and displays the output dialog box)</p> <p>Message and Ask (displays the output dialog box and asks if the user wants to rerun the worksheet)</p> <p>None (displays the output dialog box)</p> <p>The output dialog box is not displayed unless you have specified a message or at least one attribute to display on it.</p>
Method type	<p>For a list of Method Types see the Methods and Workflows, page 9-3 chapter.</p> <p>Constructor, Destructor, and Edit type methods can also run workflows. The workflow is run after the level member is created, removed, or edited.</p>

Destructor type	<p>Applies to destructor methods. This option specifies the type of deletion:</p> <p>Delete member only. This deletes the member from the level table but does not delete data related to that member. Note that when parent members are deleted, their child members will be deleted as well.</p> <p>Delete member and data. This deletes the member from the level table and all related history/forecast data. Specifically, it deletes associated data in the MDP_MATRIX and SALES_DATA tables.</p>
Execute client class	Indicates the Java class that Demantra runs when a user executes this method.
Display population	Controls the style of user interface to use for the population attribute, if the member has a population attribute. See "Presentation Styles for Population Attributes".

10. Click Next.

The next screen is Input.

Select Input Arguments.

Message:

Available Attributes:

Selected Attributes: Editable: Mandatory:

Name

Total Available Attributes: 0 Total Selected Attributes: 1

Here you specify the appearance of the input screen of the method.

1. Optionally specify a message to display at the top of the input screen.

2. For each attribute that the user should be able to edit, double-click that attribute to move it from the Available Attributes column to the Selected Attributes column.
3. If the attribute should be editable, make sure the Editable check box is selected.

Note: If you are configuring a method that changes attribute values, the workflow must include an Edit Member Step as its first step. Otherwise, the changed values will not be saved to the database.

4. If the attribute should also be mandatory, make sure the Mandatory check box is selected.
5. To change the order in which these attributes are displayed, use the up and down buttons on the right of the screen.
6. Click Next.

Note: If you do not specify a message or at least one attribute, the method input dialog box is not displayed when the user runs the method.

11. The next screen is Output.

Select Output Arguments.

Message:

Available Attributes:	Selected Attributes:
	Name

Total Available Attributes: 0 Total Selected Attributes: 1

Here you specify the appearance of the output screen of the method.

1. Optionally specify a message to display at the top of the output screen.
2. For each attribute to display, double-click that attribute to move it from the Available Attributes column to the Selected Attributes column.
3. To change the order in which these attributes are displayed, use the up and down buttons on the right of the screen.

Note: If you do not specify a message or at least one attribute, the method output dialog box is not displayed when the user runs the method.

12. Click Finish.

See also

"Making Changes Available to Users"

Presentation Styles for Population Attributes:

You can choose the presentation style for the population attribute of any level that has this kind of attribute. A population attribute is a set of item-location combinations and a range of dates. Promotions, for example, have population attributes. Other general levels could also have population attributes.

Detail Style:

With this style, part of the method input dialog box summarizes the item-location combinations and shows the range of dates, as follows:



The screenshot shows a dialog box titled "New Promotion - Input". Inside, there is a section labeled "Population". Below this label, there are two lines of text: "Brand:Rainbow" and "Account:Rainbow Company". To the right of these lines are up and down arrow buttons. Below the "Population" section is an "Edit" button. At the bottom of the dialog box, there is a "Dates:" label followed by "Start:" and a date field containing "03/10/2003" with a dropdown arrow, and "End:" and a date field containing "03/06/2006" with a dropdown arrow.

Here, when the user clicks Edit, Demantra displays a screen where the user can select the levels and the members of those levels.

Simple Style:

With this style, the method input dialog box does not summarize the population

attribute; instead it displays just the Define (or Edit) button:



When the user clicks this button, Demantra displays a screen with two tabs. On one tab, the user can select the levels and the members of those levels. On the other tab, the user can specify the range of dates.

Passing Arguments to a Method

Demantra can pass arguments in memory to the method. Considered as a group, these arguments are the context dictionary. For each argument, Demantra passes a variable name and its associated value.

Available Arguments

The available arguments are as follows.

Variable *	Value	Data Type
ws_id	Identifier of the worksheet from which the method was launched.	Java.util.String
_filter	The filter population of the worksheet from which the method was launched. Represented as a list of pairs of level_id and member_id.	java.util.String level_id,member_id; pairs separated by comas and semicolons.
view_name	The name of the active view from which the method was called.	java.util.String
level_id	Identifier of the level from which the method was launched.	java.util.String
member_id	Identifier of the member from which the method was launched.	java.util.String

Variable *	Value	Data Type
Combination_path	The context of the selected member for themethod. Will be represented as a list of pairs of level_id and member_id.	java.util.String level_id,member_id
population.filters (example)	Applies only to promotion levels. The population attribute of the selected member. The name of this variable is based on the name of the population attribute as follows: <i>population_attribute_name</i> .filters	Array of: com.demantra.applicationServer.metadataObjects.level.levelFilters.LevelFilterGetters
population.from_date (example)	Applies only to promotion levels. The from_date attribute of the selected member. The name of this variable is based on the name of the population attribute as follows: <i>population_attribute_name</i> .from_date	java.util.Date
population.to_date (example)	Applies only to promotion levels. The to_date attribute of the selected member. The name of this variable is based on the name of the population attribute as follows: <i>population_attribute_name</i> .to_date	java.util.Date

Variable *	Value	Data Type
Attribute_column_name	Values of the attributes of the selected member that are specified as inputs to the method (all attributes on the Select Input Arguments screen). The name of each variable is the same as the name of the column in which the attribute is stored.	java.util.Object

Passing Arguments

In order to pass arguments to the method, you must explicitly configure the variables that each workflow step should receive. To do so, you type the parameter names on the Parameters list for that step; see "Properties Used as Arguments for a Method".

Note: The parameter names are case-sensitive.

For the input variables, you also specify which variables to pass when you configure the method. Specifically you select the desired attributes on the Select Input Arguments screen.

Modifying a Level Method

To modify a level method:

1. If necessary, redefine the workflow schema that you are using within the method. See "Creating or Editing Workflow Schemas".
2. Log into the Business Modeler.
3. Click Configuration > Configure Methods.
The system displays a screen showing the existing methods, including all the predefined methods.
4. Optionally click a level name in the drop-down list at the top of the screen. The screen is re-displayed with only the methods associated with that level.
5. Click the method icon and then click OK. Or double-click the method name.
6. Make changes as needed and click Finish.

Deleting a Level Method

To delete a level method:

1. Click Configuration > Configure Methods.
The system displays a screen showing the existing methods, including all the predefined methods.
2. Optionally click a level name in the drop-down list at the top of the screen. The screen is re-displayed with only the methods associated with that level.
3. Click the method icon and then click Delete.
4. Click OK.
5. If the workflow schema is not used elsewhere, delete it. See "Deleting Workflow Schemas".

Using the Desktop BLE User Interface

This chapter describes how to configure and use the Business Logic Engine (BLE) to evaluate client expressions in worksheets.

This chapter covers the following topics:

- Overview of the Business Logic Engine

Overview of the Business Logic Engine

Purpose of the Business Logic Engine

Typically many of your series use client expressions. When users run a worksheet, the client expressions are evaluated. The worksheet re-evaluates the client expressions whenever the user changes data in the worksheet. If the worksheet is at an aggregated level, Demantra Spectrum then splits the values to the lowest level (for example, SKU-store), and writes them to the database.

However, when you change data in the database (for example, by importing new data), the client expressions are not automatically re-evaluated. This can be an issue if you need to use the resulting data without first opening a worksheet. For example, suppose you want to import data and then export other data calculated from the imported data. In such a case, you need to force Demantra Spectrum to evaluate the relevant client expressions. To do so, you use the Business Logic Engine, which evaluates the client expressions in a given worksheet.

In general, you can run the Business Logic Engine in two different ways:

- You can call the Business Logic Engine from within a workflow step. See "BLE Step".
- You can use the desktop BLE user interface, described here

Types of BLE Requests

There are three general types of requests:

- Start the desktop BLE user interface and submit a request to process a worksheet. These requests are called manual requests. The desktop BLE user interface displays the status of all these requests.
- Run the Business Logic Engine from the command line, passing to it the ID of a previously defined request. These requests are called automatic requests. The desktop BLE user interface displays the status of instances of the automatic requests.
- Within a workflow, include a BLE Step, which starts the Business Logic Engine if necessary and evaluates the specified worksheet. These requests are called workflow-launched requests. The desktop BLE user interface does not display the status of these requests. To monitor them, you use the Workflow Manager; see "Viewing Workflow Status".

Logging onto the Desktop BLE User Interface

Before starting the desktop BLE user interface, make sure that the database is running.

To open the desktop BLE user interface

1. Do one of the following:
 - Within Collaborator Workbench, click Tools and Applications > Business Logic Engine
 - The MANUALS_POPULATE_INS procedure should be running, in order to maintain the database whenever the Business Logic Engine evaluates data at an aggregated level. This procedure uses the values calculated by the Business Logic Engine, splits them to the lowest level, and writes them to the database. See "Introduction to Database Procedures and the Scheduler"
 - If you are using the branch_data_items table, the DYNAMIC_SYNC procedure should be running
 - The Business Logic Engine currently ignores all non-numeric series. No error occurs
 - On the desktop Start menu, click Programs. Then click Demantra > Demantra Spectrum release > Business Logic Engine. A login window appears.
2. Enter your user name and password
3. Click Login. The Business Logic Engine window appears.

Viewing Requests

The Business Logic Engine window displays the status of manual and automatic requests (it does not display requests submitted from within a workflow).

To view requests

1. Start the desktop BLE user interface, as described in "Logging onto the Desktop BLE User Interface"
2. Click one of the following radio buttons, depending on the category of request you want to view:

Note: This menu option may or may not be available, depending on how Collaborator Workbench is configured.

Use one of the following user names:

- User name of the owner of a specific component.
- User with the System Manager permission level (see "Permission Levels").

In either case, your changes in Business Logic Engine are visible to all components.

Automatic Requests

Displays automatic requests. See "Types of BLE Requests". The rest of the window displays information about that category of requests. If a task has not yet completed or if it failed during one of the stages, it is highlighted in red. Its message box lists the reason for its failure.

To refresh the Batch Monitor display

- Click Monitor > Refresh Monitor. Or click the Refresh Monitor button.

To specify the refresh delay

1. Modify the value in the Refresh Delay box by clicking one of the arrows to its right. This box displays the refresh delay in seconds.
2. Click Monitor > Set Refresh Delay. Or click the Set Refresh Delay button.

Starting the Business Logic Engine

If you are submitting requests within the desktop BLE user interface, you must first start the Business Logic Engine. (If you submit requests from a workflow, Demantra Spectrum automatically starts the Business Logic Engine as needed.)

To start the Business Logic Engine

1. Start the desktop BLE user interface, as described in "Logging onto the Desktop BLE User Interface".

2. Click Monitor > Start Engine. Or click the Start Engine button.

New Requests Displays manual requests that the Business Logic Engine has not yet processed.

Completed Requests Displays requests that the Business Logic Engine has processed.

Request Date The date and time when the request was submitted.

Run Date The required execution date and time of the request.

End Date Date and time when the request was completed, if applicable

Query Name The selected worksheet.

Status Numeric code for the current status of the request, as described below.

Messages Explanatory message for the status.

User Name The user who submitted the request.

Comments Any comments that were entered as part of the request.

Note: The bce_log_parm parameter controls how long the completed requests are saved.

An animated button appears at the upper right corner of the Batch Monitor dialog box to indicate that the Business Logic Engine is running.

Stopping the Business Logic Engine

To stop the Business Logic Engine

- Within the desktop BLE user interface, click Monitor > Stop Engine. Or click the Stop Engine button.

The animated button disappears from the upper right corner of the Batch Monitor dialog box.

Creating and Running a Manual Request

To create or modify a manual request

1. Start the desktop BLE user interface, as described in "Logging onto the Desktop BLE User Interface"
2. Make sure the New Requests radio button is selected.
3. Do one of the following
 - For a new request, click Monitor > Add Request. Or click the Add Request button.

- To edit an existing request, click the row corresponding to the request. A dialog box appears.
4. In the Run Date box, enter the required execution date and time of the request. The default is the current system date/time. You can modify each part of the default date and time (day, month, hour, and so on) by placing the cursor in it and clicking the arrow buttons to the right of the box until the required value is displayed.
 5. . In the Query Name box, select a worksheet from the list of available worksheets.
 6. (Optional) In the Comments box, type a comment
 7. Click OK

To run a manual request

- Make sure that the Business Logic Engine is running so that it can process the request; see "Starting the Business Logic Engine".

Creating and Running an Automatic Request

To create or modify an automatic request

1. Start the desktop BLE user interface, as described in "Logging onto the Desktop BLE User Interface"
2. Make sure the Automatic Requests radio button is selected.
3. Do one of the following:
 - For a new request, click Monitor > Add Request. Or click the Add Request button
 - To edit an existing request, click the row corresponding to the request. A dialog box appears
4. In the Run Date box, enter the required execution date and time of the request. The default is the current system date/time. You can modify each part of the default date and time (day, month, hour, and so on) by placing the cursor in it and clicking the arrow buttons to the right of the box until the required value is displayed.
5. In the Query Name box, select a worksheet from the list of available worksheets.
6. . (Optional) In the Comments box, type a comment
7. Click OK.

To run an automatic request

1. In the desktop BLE user interface, make a note of the numeric ID of the automatic

request you want to run.

2. In a shell, enter the following command:

rulescalc.exe username?password?request_ID Here username and password are the username and password you use to log onto the desktop BLE user interface, and request_ID is the numeric ID of the automatic request.

Note: This field is not editable for an automatic request. A date is shown, but this date is not used.

Deleting a Request

To delete a request

1. Start the desktop BLE user interface, as described in "Logging onto the Desktop BLE User Interface"
2. In the Batch Monitor window, select the request that you want to delete
3. Click Monitor > Delete Request. Or click the Delete Request button

Reloading the Configuration

When you make changes that affect a worksheet (either Business Modeler or in the worksheet designer), you must reload those changes before using the Business Logic Engine to evaluate that worksheet.

To reload the configuration

- Click Window > Reload Configuration.
- Or exit and reopen the desktop BLE user interface.

Non-Engine Parameters

This chapter describes parameters unrelated to the Analytical Engine and lists their default values, if any. As indicated, most parameters are visible to all users; a few are visible only if you log in as the owner of the component.

This chapter covers the following topics:

- About These Parameters
- System Parameters

About These Parameters

To access these parameters within Business Modeler, click Parameters > System Parameters.

The parameters listed on the Database, System, and Worksheet tabs are in the sys_params table in the database.

The parameters listed on the App Server tabs are in the aps_params table in the database.

System Parameters

Parameter	Location	Default	Details
A			

Parameter	Location	Default	Details
AccountLockoutDuration	System	60	Represents the number of seconds a user is locked out of system after 'AccountLockoutThreshold' has been reached. Value = -1: the account will be locked out until administrator unlocks it; Value ≥ 0 : the account will be locked for number of seconds specified.
AccountLockoutThreshold	System	-1	Controls the number of failed login attempts by the user before locking user account. Value = -1 : the account will not be locked after failed login attempts. Value ≥ 0 : the account will be locked after failed login attempts according to this value.
accumulatedOrUpdate	System	update	For integration, this parameter specifies how to handle existing data. Use one of the following values: accumulate: Add the new data to the existing data. update: Overwrite the existing data.
active_forecasts_versions	System	5	Specifies how many forecast versions the Demantra database should store. Use a positive integer.
AlignNumericSeries	System Parameters > Worksheet	right	This parameter controls alignment of numeric series only (i.e. not date or string series). Feasible values are (left, center, right).

Parameter	Location	Default	Details
align_sales_data_levels_in_loading	Database	no	<p>Visible only to owner. Specifies whether to maintain matrix information (combination information that is time-independent) within the sales_data table. If the matrix information is within the sales_data table, then the Analytical Engine can use that table directly instead of internally creating the sales_data_engine table.</p> <p>Use one of the following values:</p> <p>no: Do not adjust the sales_data table for direct use by the engine.</p> <p>yes: Adjust the sales_data table for direct use by the engine, so that the engine can run more quickly. This adjustment is made when data is added through loading, integration, or other mechanisms. If you set this parameter to yes, it is also necessary to rewrite some database procedures. For configuration steps, see "Reconfiguring the sales_data_engine Table".</p>
applicationDateFormat	Application Server> App. Server	mm-dd-yyyy	The system date format.
applicationDateTimeFormat	Application Server> App. Server	mm-dd-yyyy hh:mm:ss	The system date/time format, used where both a date and time are displayed.
application_root			Provides support for a relative path to files or applications referenced by workflow steps in Workflow Manager. See Parameters Used as Arguments for a Method, page 31-4.

Parameter	Location	Default	Details
ApprovalProcessScope	System	"0" indicates approval scope as Global to maintain backwards compatibility	Approval process refers to Forecast approval process, which in turn locks approved combinations. This parameter determines the scope of the approval process. When set to '0' the scope is said to be global and causes the Approval series to be added to every worksheet in the application, thus enforcing the approval process everywhere. When set to '1' the scope is said to be specific to SALES_DATA. In this case, approval process series is added to a particular worksheet only if other SALES_DATA series are selected in the worksheet, either explicitly or through a client expression. SALES_DATA series are also referred to as "BASE 0" series.
AppServerLocation	System		Visible only to owner. Path of the directory that contains the Web Platform Server; this is the directory that contains the Web_inf subdirectory.
AppServerSupport	System		Do not change this parameter; this is for internal use only.
AppServerURL	System	http://localhost:8080/demantra	<p>Visible only to owner. URL of the Web Platform Server.</p> <p>In some cases, external notifications, such as simulation and security changes, are missed because the Web server is overloaded. If you are using IIS and JRun, you can work around this by pointing all the external notifications to the internal JRun web server rather than to the IIS and JRun connector. To do so, set the AppServerURL parameter to use the default JRun port (8100). For example:</p> <p>http://frodo:8100/demantra</p> <p>See the Oracle Demantra Installation Guide.</p>

Parameter	Location	Default	Details
audit_history_length	System	12	Number of months of audit data to keep. Used by the CLEAN_LOG_TABLES procedure.
AuditMailAddress	Application Server> DP Web		Visible only to owner. Mail address of the BCC (blind carbon copy) recipient of Demantra email messages.
AutoRunMode	Worksheet	false	<p>Specifies whether Demantra automatically runs worksheets. This applies to all worksheets, in Demand Planner Web, Promotion Effectiveness, and Settlement Management as well as Demand Planner and Demand Replenisher. Use one of the following values:</p> <p>true: Demantra automatically runs any worksheet as soon as it is opened. It will not rerun automatically if changes are made to the data, for example, after clicking the update button. Also, it will not rerun automatically if changes are made in the wizard (apart from the Layout Designer).</p> <p>false: Demantra does not automatically run worksheets</p>
B			
bce_log_parm	Database	100	Ignore this parameter.
BLEThreadPoolSize	Application Server> App. Server	4	Visible only to owner. Number of threads that the Business Logic Engine can use.
BLETimeOut	Application Server> App. Server	5000	Visible only to owner. Length of time, in milliseconds, before an idle thread (of the Business Logic Engine) times out.

Parameter	Location	Default	Details
buildNumber	Application Server > Collaborator		Read-only. Current build number of Demantra.
C		.	.
client.dropdownCacheLimit	APS_PAPRAMS table	50	The number of dropdown lists for series that the client can cache in memory. This is an internal client implementation parameter. Client Dropdown Cache is implemented in an LRU manner so that when the limit is reached, the oldest entry will be deleted from the client cache memory. When the system needs to access this deleted dropdown list, it will be reloaded again from the server. Changing this parameter may have an effect on client memory consumption as well as on client performance. This parameter originally prevented an OutOfMemory exception on a client with huge dropdown lists.
client.ssl.authentication	System Parameters > Application Server > DP Web	false	This parameter controls two-way (mutual) SSL authentication.
client.uilimitations.maxcombs.ws	Application Server > Collaborator	2000	The maximum amount of combinations that can be retrieved for a single worksheet
client.uilimitations.maxcells.ws	System Parameters > Collaborator	100,000	This parameter determines the maximum number of cells that will be retrieved in a single worksheet.

Parameter	Location	Default	Details
client.uilimitations.maxcells	System Parameters > Collaborator	300,000	This parameter determines the maximum number of worksheet cells that will be retrieved for a single client machine. The value is based on the total number of cells in all open worksheets. .
client.uilimitations.maxdiskspace	System Parameters > Collaborator	200,000	This parameter sets the maximum disk space that a single client machine will allow per worksheet (in KB). It should not be necessary to change this parameter unless there is a disk space issue.
client.uilimitations.warning	Application Server > Collaborator	80	The percentage of the allowable maximum combinations, cells, or disk space at which to display to a warning message. For example, if maxcombs is 1000 and this value is 80, then a warning message will be displayed after loading 800 combinations (80% of max). If user has entered 0, it is ignored and the default value is used instead. If this value is 100, then no warnings will be displayed, since the error messages will occur at the same
charsetEncoding	Application Server > DP Web	UTF-8	The encoding character set for the XML.
check_validity_installation	Database	yes	Visible only to owner. Specifies whether to check the validity of the installation.
client.enableOpenNoteWithDoubleClick			<p>Double-click within the worksheet table to specify whether users can access the notes dialog box by . Use one of the following values:</p> <p>0 = false (users cannot use double-click to access this dialog box)</p> <p>1 = true</p> <p>In any case, it is always possible to access this dialog box by using the right-click menu, as in Microsoft Excel.</p>

Parameter	Location	Default	Details
client.JREMaxMemory	Application Server> DP Web		Visible only to owner. Maximum amount of memory (in MB) that JRE can use. The Web worksheets (Demand Planner Web, Promotion Effectiveness, and Settlement Management) use JRE.
client.lookAndFeel	Application Server> DP Web	com.demantra.common.ui.Inf.DemantraLookAndFeel	Renamed in 7.0.2. Specifies the user interface style. Do not change this.
client.MaximumCombsLoadChunkSize	Application Server> DP Web	20	Visible only to owner. Maximum number of combinations to load each time the user chooses to rerun a worksheet.
client.task.attachments.security.filetypes	Application Server> Collaborator	empty	Defines which file types are allowed as Task attachments. default value is empty - meaning all file types are blocked. Each added extension is separated by comma. Need to add a file extension to this parameter in order to allow it as a task attachment. Example: *.html,*.htm,*.xml . *. - means all files are allowed (mainly for backward compatibility use).
client.worksheet.privateAccessType	Application Server> DP Web		Visible only to owner. Specifies the default setting of the public/private option in the worksheet designer (used by the Web products).
ClientDDCacheCapacity		10	Specifies the maximum number of distinct drop-down lists per worksheet that any client should cache. These caches are cleared when the worksheet is closed.
collaborator.supportURL	APS_PARAMS table		URL of the Support link, relative to http://server name/virtual directory/portal/. This link is in the upper right corner of Collaborator Workbench.

Parameter	Location	Default	Details
collaborator.searchURL	APS_PA RAMS table		URL of the Search link, relative to http://server name/virtual directory/portal/. This link is in the upper right corner of Collaborator Workbench.
company.name	Applicati on Server> Workflo w	Demantr a	Name of your company; the Workflow Engine uses this string when it sends email messages when a workflow step fails.
ColorCodingLevel	Workshe et		PE only; visible only to owner. Specifies the ID of the level that will be used to color code promotions.
CopiedMembersPerProce ss	Database		Specifies the maximum number of members that can be copied in a single copy/paste process.

Parameter	Location	Default	Details
CopyPasteIntersect	Database	0	<p>Specifies whether the total volume is preserved when an item is removed from a promotion group. This is a global parameter that applies to the entire application installation. This parameter works in conjunction with the Copy/Paste preservation Type setting on the series configuration when it is set to Volume Preservation.</p> <p>Valid values:</p> <ul style="list-style-type: none"> 0 - Total volume of the promotion is preserved and distributed among the remaining items in that promotion as defined by the proportionality configuration of the series. 1 - Total value of the promotion is reduced by the volume of the item removed from the promotion if the promotion was defined at an item or aggregation level. The volume of the individual items in the promotion is retained. <p>Important: Ensure that the proportionality of the volume series is set to the desired proportions so that volume is distributed correctly. It is recommended that you set the volume series to be proportional to itself in the series configuration.</p>
CTO_Enable_Worksheet_Calculations	System Parameters > System	Yes	This parameters controls whether planning percentages are automatically calculated when a worksheet is updated.
CTO_Calc_PP_Forecast	System Parameters > System	No	This parameters controls whether to forecast planning percentages.

Parameter	Location	Default	Details
CTO_HISTORY_PERIOD S	System Parameters > System	52	This parameter specifies the number of history periods to use when calculating planning percentages based on the history of the options.
CTO_PLANNING_PERCENTAGE_CHOICE	System Parameters > System	Existing	This parameter specifies the planning percentage to use when series Planning Percentage Choice calculates the dependent information in Forecast Dependent Demand. Setting this to "Existing" means to use the downloaded planning percentages from Oracle e-Business Suite that are in series Planning Percentages - Existing. The other option is to have Oracle Demantra calculate planning percentages based on history and forecast of options.
D			
DatabaseEncoding	Application Server > DP Web		Visible only to owner. Encoding style for Oracle Web products. For a list of possible encoding sets, see http://java.sun.com/j2se/1.4.1/docs/guide/intl/encoding.doc.html . Use the "Canonical Name for java.io and java.lang API."
DBConnectionTimeout	Application Server > App. Server	5000	The database connection time-out period, in milliseconds.
DBDateFormat	Application Server > App. Server	{call pre_logon() }	The date format used in the database.
DBIdleTimeOut	Application Server > App. Server	1800000	The connection idle time-out period, in milliseconds.

Parameter	Location	Default	Details
DBName	Application Server> App. Server		<p>The meaning of this parameter depends on the database used:</p> <p>For Oracle databases, this is the Oracle SID.</p> <p>See the Oracle Demantra Installation Guide.</p>
DBPassword	Application Server> App. Server		Password of the database user in which the Demantra data schema resides. See the Oracle Demantra Installation Guide.
DBPort	Application Server> App. Server		The port number of the database server. See the Oracle Demantra Installation Guide.
DBType	Application Server> App. Server	oracle	Indicates the type of database that Demantra is using.
DBUser	Application Server> App. Server	.	Database user in which the Demantra data schema resides. See the Oracle Demantra Installation Guide.

Parameter	Location	Default	Details
Debug	Worksh et	false	<p>Applies to the desktop product Visible only to owner. Specifies the debug mode, if any, in which the Demantra application server should run. In debug mode, the application server writes debug messages to the console. Use one of the following values:</p> <p>false (0): Do not write any debug messages at all.</p> <p>true (1): Write debug messages from all possible causes. Use this setting sparingly, because it generates a large number of messages.sql</p> <p>(2): Write SQL syntax from all database interactions. You can then copy and execute the SQL statements outside of the application to verify that desired results are achieved by the statements.</p> <p>mem (3): Write messages related to memory usage. Specifically it provides information about memory consumption at various stages of the application. You can use this information to find memory leaks.</p> <p>init(4): Write messages related to object initialization. This information helps you see which objects come into being as the application is working on various tasks.</p> <p>synch (5): Write notifications related to synchronization. Use this information to see if the various parts of the application are synchronized with one another. It is often useful to eliminate the issue of synchronicity as the source of whatever problem is being investigated.</p> <p>update (6): Write messages related to data updates, so that you can see what updates are passing through the system. If you are updating but do not see expected results, use this debug setting to verify first that the update has passed through in the correct way through the internal libraries.</p>

Parameter	Location	Default	Details
			ts; visible only to owner. Specifies whether Demand Planner or Demand Replenisher runs in debug mode. In debug mode, these products pop up debug windows when errors occur.
DebugMode	Application Server> App. Server	off	.
DefaultContentSecurityAccess	System		<p>Specifies the default security access per member, as seen in the Security menu options. Use one of the following values:</p> <p>1 (no access)</p> <p>2 (read access)</p> <p>3(read/write access)</p> <p>4 (full control; includes delete access)</p>
DefaultLevelSecurityAccess	System		Specifies the default security access per member, as seen in the Component menu options. Use any of the values listed for DefaultContentSecurityAccess.
DeskTopNotification	System	http notification	<p>Applies to the desktop products; visible only to owner. Use one of the following values:</p> <p>http notification</p> <p>db notification</p>
dir.onlineHelp	Application Server> Collaborator		URL of the Help link, relative to http://server name/virtual directory/portal/. This link is in the upper right corner of Collaborator Workbench.

Parameter	Location	Default	Details
DSMAllShipDateDifference	System		DSM only. Specifies the window of time that Demantra uses to search for a promotion that matches a given settlement. Express this as the number of time buckets between the promotion end date and the deduction date. The promotion end date is given by the series specified by the DSMPEShipDateSeries parameter.
DSMOICheckProduct	System		Not used.
DSMOIPercentDifference	System		DSM only. The maximum percent difference (of monetary amount) permitted when matching an off-invoice settlement to possible promotions.
DSMOIShipDateDifference	System		DSM only. Specifies the window of time that Demantra uses to search for a promotion that matches a given off-invoice settlement. Express this as the number of time buckets between the promotion end date and the settlement date. The promotion end date is given by the series specified by the DSMPEShipDateSeries parameter.
DSMPEOIAmountSeries	System		DSM only. Specifies the ID of the series that stores the monetary off-invoice amounts for the promotions. This series must have an aggregating server expression.
DSMPEShipDateSeries	System		DSM only. Specifies the ID of the series that stores the ship dates of the promotions. This series must have an aggregating server expression.
DSM PromotionBudgetSeries	System		DSM only; parameter is visible only to owner. Specifies the ID of the promotion budget series to which adjustments will be made when settlements are matched to a promotion.

Parameter	Location	Default	Details
DSMWriteOffThreshold	System		DSM only. Specifies the monetary amount below which Demantra automatically writes off a settlement. See "Configuring DSM" for information on configuring the writeoff process.
E			
EnableIncrementalLoading	System	true	Enables the Demantra incremental loading feature, for faster worksheet reruns. There is no user impact apart from performance.
EngineBaseUrl	System	http://server:port/EngineManager/	Parameter defining path to web-based analytical engine. Server:port should be modified to match application server details
EnginePlatform	System	1	Parameter indicates whether engine will be executed on Linux (1=default) or Windows (0) environment
EnableWorksheetCaching	System	true	Specifies whether Demantra can cache the Web worksheets.
execution.shell	Application Server>Workflow		Applies to the Executable Step. This parameter specifies any prefix that is needed in order to run executable steps. For example, you may need to specify the following for Unix: "./" (i.e. dot slash).
ExternalLogoutURL	System	none	The URL that points to the landing page after logging out from a remote system, such as the Oracle E-Business Suite.
F			

Parameter	Location	Default	Details
FillValueExportToExcel	System Parameters > Worksheet	no	This parameter controls whether all rows or columns are labeled with combination information versus one label only at first row of data per combination. Enabling this parameter allows the user to more easily create reports, such as pivot tables, within Excel. Values (yes = fill all rows or columns with combination details; no = display combination information only once)
FIRSTDAYINWEEK	Database	monday	First day of week to use when binning sales data into base time buckets.
format.dates.longFormat	Application Server > Collaborator	mm/dd/yy hh:mm:ss a	Long date format.
format.dates.shortFormat	Application Server > Collaborator	mm/dd/yy	Short date format.
G			
GatherStatisticsThreshold	Worksheet	2000	For a temporary table, minimum number of rows needed in order to automatically run Analyze Table. Used by the ANALYZE_TABLE_TMP procedure.
general.copyright	Application Server > Collaborator	Copyright Demand a; 2005	Visible only to owner. Copyright notice displayed in lower left of the Collaborator Workbench window.
general.homepage.title	Application Server > Collaborator	My Demand Collaborator	Title of the Collaborator Workbench home page, as used within the Collaborator Workbench title bar.

Parameter	Location	Default	Details
general.title.text	Application Server> Collaborator	Demand Collaborator	Title of the browser window when it displays Collaborator Workbench.
general.userList.tasks	Application Server> Collaborator	true	Not used.
general.userList.whoisonline	Application Server> Collaborator	true	Specifies whether the Who's Online module is displayed.
general.userList.worksheets	Application Server> Collaborator	true	Not used.
Graph.MaxLabelWidth	Application Server> Collaborator	20	Maximum width of labels in graphs in Collaborator Workbench, Demand Management, and Promotion Effectiveness. If a label is longer than this width, the last characters of the label are represented by three periods (...).
HintDisplayTime	Parameters > System Parameters > DP Web	4 seconds	Global parameter controlling duration of hint visibility when user places mouse over any object that supports tooltips, measured in seconds. Set to -1 to display the hint as long as the mouse is over the object. Set to 0 if you do not want to display hints.
ImportBlockSize	Application Server> App. Server	5000	The number of rows for each commit, used during import.

Parameter	Location	Default	Details
indexspace	Database	TS_FOR ECAST_ X	Database table space that stores the forecast table indexes, as specified during installation.
indicator.synchronizeSalesData	Application Server> Collaborator	yes	Controls the synchronization of the general level indicators in the sales_data table.
initial_param	Database	20M	Default initial size of system table spaces. See also next_param.
InsertDateTimeFormat	Application Server> App. Server	MM-dd- yyyy 00:00:00	The date-time format that Demantra uses when writing to the database. When you enter dates in a worksheet, Demantra converts them to this format before writing them to the database.
Insertmissingvalues	Worksheet	yes	Not used.
J			
L			
Legend.MaxLegendItem Width	Application Server> Collaborator	30	Maximum width (in characters) of the legend in a graph-type content pane in Collaborator Workbench. If any lines of the legend are too longer, the last characters of those lines are represented by three periods (...).
L icense.ExpirationMessage	Application Server> App. Server	Your Security License File has expired. Please contact support.	Visible only to owner. Message shown when the system license has expired.

Parameter	Location	Default	Details
LinkedDataAutoCopy	System	1 - continue copy	Specifies whether to continue linking source and target data after initial link. If active, the linking process continues until the launch date is reached. Possible values are: (0) do not copy or (1) continue copy.
LoadDataStop	System	yes	Specifies whether Demantra should stop loading data when it finds an error in the data.
LockTimeout	Database	10	Specifies the period (in seconds) between killing a database session and releasing the lock for that session.
log.history	Applicati on Server> Workflo w	1	The number of days for which workflow history is kept.
M			
mail	Applicati on Server> DP Web	on	Specifies whether Demantra is enabled to automatically send email, as specified within a workflow. Use one of the following values: yes: no: You can set this parameter during installation or later. See the Oracle Demantra Installation Guide.
mail.outgoing.server	APS_PA RAMS table		

Parameter	Location	Default	Details
mail.server	Application Server> Collaborator		<p>Name or IP address of the SMTP mail server that Demantra should use when it sends email.</p> <p>For example: mayflower.demantra.net</p> <p>You can set this parameter during installation or later. For details, see the Oracle Demantra Installation Guide.</p>
mail.strings.from.system	Application Server> Workflow	See details.	<p>Visible only to owner. Specifies the title of the sender of Demantra email messages, for example "Demantra Solution Manager". Default:</p> <p>Demantra Solution</p>
mail.strings.internalerror.message	Application Server> Workflow	See details.	<p>Text of email message sent in case of internal error. Default:</p> <p>Internal error: please check database and network connections</p>
mail.strings.internalerror.subject	Application Server> Workflow	See details.	<p>Subject of email message sent in case of internal error. Default:</p> <p>Workflow internal error</p>
mail.strings.processfailure.subject	Application Server> Workflow	See details.	<p>Text of email message sent by a Fail-To-Execute Step. Default:</p> <p>Error in Process Execution</p>
mail.strings.processterminated	Application Server> Workflow	See details.	<p>Message sent when a process is terminated. Default:</p> <p>The following process is terminated</p>

Parameter	Location	Default	Details
mail.strings.recovery	Application Server> Workflow	See details.	String included in recovery email message. Default: Please handle recovery for the following process:
mail.strings.taskfailuresubject	Application Server> Workflow	See details.	Subject of email message sent by a Fail-To-Execute Step. Default: Workflow process has failed
mail.strings.tasktimeoutsubject	Application Server> Workflow	See details.	Message sent when a task is timed out. Default: Task(s) timed out in workflow
mail.strings.timeout.group	Application Server> Workflow	See details.	Message sent when a task is timed out in a group step. Default: Treatment period for this task(s) was finished and one or more of the group members haven't responded. \n\nThe process moved to alternative treatment.
mail.strings.timeout.user	Application Server> Workflow	See details.	Message sent when a task is timed out in a user step. Default: Treatment period for this task(s) was finished and the process moved to alternative treatment
mail-password	APS_PARAMS table		Visible only to owner. Password of the administrator account; this is also usually the network username of the administrator. You can set this parameter during installation or later. For details, see the Oracle Demantra Installation Guide.

Parameter	Location	Default	Details
mail_recipient	System	no send	<p>Specifies where to send a message when error is found during data load. Use one of the following values:</p> <p>no send</p> <p>send to administrator and changer</p> <p>send to administrator</p>
mail-username	APS_PARAMS table		<p>Visible only to owner. Username of the administrator account from which Demantra sends email; this is usually the network username of the administrator.</p> <p>For example: admin</p> <p>You can set this parameter during installation or later. For details, see the Oracle Demantra Installation Guide.</p>
mailAddress	Application Server> DP Web.		<p>The email address of the administrator email account. For example:</p> <p>demantra-admin@acme.com</p> <p>You can set this parameter during installation or later. For details, see the Oracle Demantra Installation Guide.</p>
mailProtocol	Application Server> DP Web	mail.smtp.host	<p>Server for sending email. Demantra supports only SMTP servers.</p>
ManualRefreshAsDefault	System	true	<p>Visible only to owner. Specifies the default setting of the Refresh Type caching option in the worksheet designer.</p> <p>This parameter has an effect only if worksheet caching is enabled (through EnableWorksheetCaching). Also see WorksheetCachingAsDefault.</p>

Parameter	Location	Default	Details
MatrixCombs			Read-only. Indicates the number of lowest-level combinations in the mdp_matrix table. For use in helping you set SimMaxSize and SimWarnSize.
MaxAvailableFilterMembers			<p>Specifies the maximum number of members that can be retrieved in the worksheet filter screen. If the user selects more members than allowed, a message asks the user to add further filtering.</p> <p>This limit is also applied to dropdown lists for series and attributes.</p>
MaxBackgroundLoad	Application Server> DP Web	20	Maximum number of worksheets that can be loaded in the background.
MaxDBConnections	Application Server> App. Server	50	<p>The maximum number of database connections for the Demantra database user.</p> <p>Recommended: the number of concurrent users multiplied by 2.</p>
max_fore_sales_date	System		Visible only to owner; read-only. The latest possible forecast sales date.
max_initial_members	Worksheet	100	Applies to Member Management in Demand Planner. Specifies how many members to display immediately when clicking on a parent member.
max_records_for_commit	Database	10000	The number of records that Demantra will insert into the database before performing a COMMIT operation. If you increase this number, the insertion will run more quickly, but you risk losing all uncommitted records in case of a crash.
max_sales_date	Worksheet		The latest sales date loaded as history in the system.

Parameter	Location	Default	Details
MaxSaleVal	System	999999999	Maximum allowed sale value within Demantra, for any possible unit of measure.
MaxUpdateThreads	Application Server> App. Server	2	The maximum number of threads to use for the update mechanism. Must be an integer and should equal the number of database CPUs plus one.
MinDBConnections	Application Server> App. Server	4	The minimum number of database connections for the Demantra database user.
min_fore_sales_date	System		Visible only to owner; read-only. The earliest possible forecast sales date.
min_sales_date	System		Earliest possible sales date.
mix_max_members	Worksheet	50	The maximum number of members that a user will be allowed to work on.
N			
navBarContentProvider.addNewContentLink.Text	Application Server> Collaborator	New	Text of the New link, which is shown at the top of the Contents menu in Collaborator Workbench.
navBarContentProvider.igniteEnterpriseContent.text	Application Server> Collaborator	EnterpriseContent	Not used.
next_param	Database	20M	Incremental amount of storage that is added to a table space when more space is needed. See also initial_param.

Parameter	Location	Default	Details
O			
NpiUseWsFilter	Worksheet	No	<p>Specifies whether or not worksheet filters are used when executing the Find Similar, Edit Lifecycle Definitions and Remove Combinations workflows. Choose one of the following values:</p> <p>No (0): Do not apply worksheet filters.</p> <p>Yes (1): Apply worksheet filters.</p>
O			
OpenWithContext	Worksheet	Selected member	<p>Specifies the default setting of the Open With Context setting of the worksheet designer. Choose one of the following values:</p> <p>Selected member (filters only to the selected member)</p> <p>Selected context (filters to the selected member and any additional context where the user clicks within the Members Browser)</p>
P			
PasswordHistoryCount	System	5	<p>Represents the number of historical passwords which cannot be used when resetting the password. Value = -1: do not restrict reuse of historical passwords; value > 0 : restrict password reuse to specified number of passwords.</p>
PasswordResetTime	System	-1	<p>Frequency with which passwords must be reset, measured in days. Warnings will be sent within 10 days of the reset deadline. Users who do not reset passwords prior to expiration will require administrator support to regain access to the application. Default reset interval is set to -1 to emulate no reset. Otherwise, should set to number of days between resets.</p>

Parameter	Location	Default	Details
PasswordRulesEnforcedGlobal	System	No	Controls whether the system enforces password policies. For details, see Password Policies.
parts_factor_source	System	bom_factor. or.factor	Visible only to owner. Table and column of BOM factor.
pct_increase_for_analyze	Database	10	Visible only to owner. Percentage of data increase for a given table, beyond which Demantra automatically increases the table size.
phase_in_out	System	1	Visible only to owner.
portalVersion	Application Server> Collaborator		Read-only. Current version number of Demantra.
PromoDefaultSpan	Worksheet		PE only; parameter is visible only to owner. Specifies the default length of time for promotions created within a worksheet.
PromoDefaultStart	Worksheet	start date of the worksheet	PE only; parameter is visible only to owner. Specifies the default start date for promotions created within a worksheet. Use one of the following values: today (0) last loaded sales date (1) start date of the worksheet (2)
ProportDefaultEngineProfile	Database	0	Engine profile to be used when executing the proport procedure as part of data load or update process. 0 (default) points to the base engine profile. Value should be modified if a different engine profile is to serve as basis for proportion calculations.
Q			

Parameter	Location	Default	Details
Query.MaxCombinations	Application Server> Collaborator	10	Maximum number of combinations that can be displayed in a graph-type content pane in Collaborator Workbench, when you display a single series plotted for multiple combinations. The user receives an error if a content pane contains more than this number of combinations.
Query.MaxSeries	Application Server> Collaborator	10	Maximum number of series that can be displayed in a graph-type content pane in Collaborator Workbench. The user receives an error if a content pane contains more than this number of series.
Query.TopBottom.MaxCombinations	Application Server> Collaborator	30	Maximum number of combinations that can be displayed in a Collaborator Workbench content pane that contains a stacked bar chart or pie chart. The user receives an error if a content pane contains more than this number of combinations.
QueryMechanismTimeout	Application Server> App. Server	5000	Visible only to owner. The time-out period for the query notification listener, in milliseconds. The APS uses this parameter.
QueryRunMaximumThreadUser	ds.ini		
R			
Rebuild_Sales_Table	Database	yes	<p>Oracle only; visible only to owner. Controls the REBUILD_TABLES procedure. Use one of the following values:</p> <p>yes: The REBUILD_TABLES procedure rebuilds the sales_data table.</p> <p>no: The procedure ignores this table.</p>

Parameter	Location	Default	Details
ReceiveCollaborationMessages	Worksheet	true	<p>Applies to the desktop products. Specifies whether desktop users should receive messages when data is changed. Use one of the following values:</p> <p>yes: Users receive messages when the currently displayed data has been changed by another user (who is logged onto either the desktop or the Web products).</p> <p>no: Users do not receive these messages.</p> <p>This parameter has no effect on the Web products.</p>
RefreshTimer	Worksheet	10	<p>Applies to the desktop products. Idle event interval in seconds.</p>
RestartOnFailureInterval	System	20	<p>Ignore this parameter.</p>
Run_full_matrix_proport	Database	no	<p>Visible only to owner. Specifies whether to run the proport mechanism on all the item-location combinations. Use one of the following values:</p> <p>all combinations: Run proport on all combinations in mdp_matrix.</p> <p>only flagged combinations: Run proport only on the combinations that have prop_changes=1.</p> <p>all new combinations (2), run proport on all combinations that have new_member=1.</p>
RunProportInMdp_add	Database	yes	<p>Visible only to owner. Specifies whether to call the proport mechanism from the MDP_ADD procedure. Use one of the following values:</p> <p>yes: The MDP_ADD procedure runs the proport mechanism.</p> <p>no</p>
S			

Parameter	Location	Default	Details
sales_data_engine_index_space	Database	TS_FOR ECAST_ X	Database table space that stores indexes for the sales_data_engine table, as specified during installation.
sales_data_engine_space	Database	TS_FOR ECAST_	Database table space that stores the sales_data_engine table, as specified during installation.
SecurityFromSystem	Worksheet	false	When starting the system it will try to log as the O.S user with password "system"
server.generalurl	Application Server> Workflow		URL for the workflow server, not including the portal/workflow directory.
Server.SessionExpiration	Application Server> Collaborator	1200	Specifies how long (in seconds) before an idle Collaborator Workbench session expires. Does not affect worksheet sessions.
ServerName	Application Server> App. Server		Database server name (host machine or IP address on which database resides). For example: @wysiwyg See the Oracle Demantra Installation Guide.
SettlementLocationExtension			DSM only. Specifies the internal identifier of the location-type level with which settlements should be associated. This generally represents the entity that is being billed or refunded. To set this parameter, use the installer. See the Oracle Demantra Installation Guide.

Parameter	Location	Default	Details
SettlementProductExtension			<p>DSM only. Specifies the internal identifier of the item-type level with which settlements should be associated. This generally represents a promoted product or a product group.</p> <p>To set this parameter, use the installer. See the Oracle Demantra Installation Guide.</p>
show_legend	Worksheet	no	<p>Applies to the desktop products. Specifies whether to display the legend in new worksheets.</p>
SimMaxSize	System	0.1	<p>Specifies the threshold size of a simulation that is too large to run. If a user tries to perform a simulation of this size, Demantra displays a message and does not attempt the simulation. Specify this as a percentage of the total number of combinations in the mdp_matrix table. Also see SimWarnSize.</p> <p>The MatrixCombs parameter displays the number of combinations currently contained in this table.</p>
SimWarnSize	System	0.03	<p>Specifies the threshold size of a simulation that is large enough to trigger a warning message to the user. Specify this as a percentage of the total number of combinations in the mdp_matrix table. Also see SimMaxSize.</p> <p>Note that the MatrixCombs parameter displays the number of combinations currently contained in this table.</p>
simulationindexspace	Database	TS_FOR_ECAST_X	<p>Database table space that stores indexes for the simulation tables, as specified during installation.</p>
simulationspace	Database	TS_FOR_ECAST	<p>Database table space that stores the simulation tables, as specified during installation.</p>

Parameter	Location	Default	Details
SortMemberbyCodeinFilter	Application Server> App. Server	1	Controls the order in which the Worksheet Manager displays the level members on the filter screen.
sortSeriesByDisplayOrder	Application Server> App. Server	1	<p>Controls the order in which Worksheet Designer displays series</p> <p>If sortSeriesByDisplayOrder is 1, display the series by display_order.</p> <p>If sortSeriesByDisplayOrder is 0, display the series alphabetically.</p>
SPF_Default_Profile	System	1	Default profile list for SPF Failure Rate calculation. If listing more than one profile ID, separate with commas.
SPF_Enable_Worksheet_Calculations	System	Yes	<p>Automatically calculates planning percentages during worksheet updates. Updates to relevant series trigger BOM tree propagation if set to yes.</p> <p>Note: Enabling this parameter may negatively affect worksheet performance. When set to No (disabled), propagation occurs when the user modifies worksheet values. However, the new planning percentages are only displayed after the end user reruns the worksheet.</p>
SPF_History_Periods	System	26	Periods of history used for to calculate planning percentages.
START_OR_END_WEEK			Visible only to owner. Specifies whether data is aggregated forward or backward in time, when loaded into Demantra through any Demantra loading mechanism.

Parameter	Location	Default	Details
StartUpdateQueue	Application Server> App. Server	yes	Visible only to owner. Specifies whether to start the manual update listener. The APS uses this parameter.
SynchRangeStart	System	1/1/1900 will be automatically substituted by first period in future horizon.	Controls beginning of date range to be copied from GL data to Sales_Data.
SynchRangeEnd	System	1/1/1900 will be automatically substituted by last period in future horizon.	Controls end of date range to be copied from GL data to Sales_Data.
SynchDefaultProfile	System	1	The default profiles for GL synchronization. (Example default profiles: "1", "2", "1,2,3", "1,3")
SynchEnableBatch	System	Yes	Enable GL synchronization in batch mode, if set to NO GL synchronization will not copy data when called.
SynchEnableAdHoc	System	Yes	Enable ad hoc GL synchronization, if set to NO ad-hoc synchronization will not copy data when called.
sysadmin_email_address	System		Not used.
sysadmin_password	System		Not used.

Parameter	Location	Default	Details
sysadmin_username	System		Not used.
SystemStatus	Database	enabled	Read-only. Indicates the status of the system, according to the sessions monitor: enabled read-only mode disabled
T	.	.	.
tablespace	Database	TS_FOR ECAST	Database table space that stores the system tables, as specified during installation.
task.default.option	Applicati on Server> Collabora tor	Select Option	The default option for a selection task.
task.message.length	Applicati on Server> Collabora tor	254	Maximum length of task description field in characters.
task.message.taskadded	Applicati on Server> Collabora tor	A task has been added to your tasklist	Do not change.
tasklist.showTimeoutTasks	Applicati on Server> Collabora tor	yes	Specifies whether Collaborator Workbench should show tasks that are past their due date. Possible values are 1 (show tasks that are overdue) and 0 (don't show such tasks). Default value is 1.
threadpool.attributesUpdate.per_comb			Maximum number of threads that a single thread can use.

Parameter	Location	Default	Details
threadpool.attributesUpdate.size			Maximum number of allowed threads for this thread pool. This should be less than MaxDBConnections.
threadpool.attributesUpdate.timeout			Idle time-out period. This specifies how long (in milliseconds) a thread is left unused before it is ended automatically.
threadpool.copy_paste.perr_process	Application Server> App. Server		Visible only to owner. Maximum number of allowed threads for the copy/paste mechanism in any given process. Must be an integer and should be less than MaxDBConnections.
threadpool.copy_paste.size	Application Server> App. Server		Visible only to owner. Maximum number of allowed threads for the copy/paste mechanism. Must be an integer and should be less than MaxDBConnections. Also be sure to leave room for system processes.
threadpool.copy_paste.timeout	Application Server> App. Server		Visible only to owner. Idle time out period. This specifies how long (in milliseconds) a copy/paste thread is left unused before it is ended automatically.
threadpool.default.size	Application Server> App. Server	10	Visible only to owner. Not used.
threadpool.default.timeout	Application Server> App. Server	10000	Visible only to owner. Not used
threadpool.level_method.block	Application Server> App. Server	wait	Visible only to owner. Specifies how the level methods should access this thread pool. Use one of the following values: wait: Wait for a free thread. abort: Do not wait for a free thread.

Parameter	Location	Default	Details
threadpool.level_method.size	Application Server> App. Server		Visible only to owner. Maximum number of allowed threads for the level method mechanism. Must be an integer and should be less than MaxDBConnections. Also be sure to leave room for system processes.
threadpool.level_method.timeout	Application Server> App. Server		Visible only to owner. Idle time out period. This specifies how long (in milliseconds) a level method thread is left unused before it is ended automatically. Recommended setting: 300000 (5 minutes)
threadpool.query_run.size	Application Server> App. Server		Visible only to owner. Maximum number of allowed threads that Demantra can use to run a worksheet. If this number is missing or negative, the worksheet run mechanism does not use threads. Must be an integer. Should be less than MaxDBConnections. Also be sure to leave room for system processes.
threadpool.query_run.timeout	Application Server> App. Server	180000	Visible only to owner. Idle time out period. This specifies how long (in milliseconds) a worksheet thread is left unused before it is ended automatically. This parameter is ignored if threadpool.query_run.size is negative or missing.
threadpool.update.size	Application Server> App. Server	10	Visible only to owner. Maximum number of allowed threads for the update mechanism. Must be an integer and should be less than MaxDBConnections. Also be sure to leave room for system processes.
threadpool.update.timeout	Application Server> App. Server	10000	Visible only to owner. Idle time out period. This specifies how long (in milliseconds) an update thread is left unused before it is ended automatically.

Parameter	Location	Default	Details
Timeresolution	System	week	<p>Read-only. The base time unit. That is, the smallest possible unit of time visible in Demantra: day, week, month</p> <p>Read-only. The base time unit. That is, the smallest possible unit of time visible in Demantra: day, week, month</p>
TrendDampPeriod	System	0	The number of time periods used in application of residual dampening. The last group of periods is not dampened with previous groups dampened in an exponential manner. When set to 0 no dampening is applied.
U			
update_dead_comb	Worksheet	"1"	The parameter determines whether the update mechanism saves information to dead combinations. If set to "1", dead combinations are allocated forecast. If set to "0" dead combinations are not allocated forecast. In this latter case, if all combinations that are part of an update are dead, the updated values will be lost.
update_units_by_items	Database	by items	<p>Specifies how to update units for the INSERT_UNITS procedure. Use one of the following values:</p> <p>by items: The procedure operates item by item. This is faster but less accurate.</p> <p>by combinations: The procedure operates on individual combinations. This is slower but accurate.</p>
UpdateAudit	Database	no	Visible only to owner. Specifies whether the MANUALS_INS_INTEGRATION procedure updates the audit tables. This procedure is used when data is imported at an aggregated level.

Parameter	Location	Default	Details
UpdateQueueTimeout	Application Server> App. Server	5000	Visible only to owner. The time-out period for the manual update listener, in milliseconds. The APS uses this parameter.
UpdateThreadTimeout	Application Server> App. Server	5000	The time-out period for the update threads, in milliseconds. The APS uses this parameter.
UseDateRangeMatrix			<p>This parameter controls whether the worksheet mechanism uses internal data structures can improve the performance of worksheets that include promotions. You can control whether worksheets use these structures; the system uses them automatically for other purposes. If you enable this option, the largest benefit occurs in cases where promotions are long (and have many rows of data).</p> <p>This affects only worksheets that include general levels that have population attributes.</p>

Parameter	Location	Default	Details
useGLEExceptions	Application Server> Collaborator	true	<p>Visible only to owner. Specifies whether Demantra respects worksheet exceptions that refer to promotion series. Use one of the following values:</p> <p>true: Demantra does consider worksheet exceptions that refer to promotion series. This affects the behavior of the worksheet and of the BLE Step. The worksheet behavior, however, is not intuitive; the Members Browser displays all the combinations. When the user clicks on a combination that does not meet the criteria, the worksheet then displays a message saying that the combination is empty.</p> <p>false: Demantra ignores worksheet exceptions that refer to promotion series. This means that such exceptions are useless, but the worksheets behave more intuitively.</p>
UseItemsAggri	Worksheet	no	<p>Specifies whether to use data from the branch_data_items rather than the usual table. The branch_data_items table aggregates data by item, for improved performance in worksheets or within the Business Logic Engine. You can use this table only when you do not need to view specific locations or filter by location.</p> <p>Use one of the following values:</p> <p>yes: Whenever possible, Demantra uses data from the branch_data_items. In this case, the first time the engine runs, there will be a long update of all rows in sales_data. Also make sure that the DYNAMIC_SYNC procedure runs periodically to keep the branch_data_items table current.</p> <p>no: Demantra uses the usual tables.</p>

Parameter	Location	Default	Details
UserListContentProvider. commontitle	Application Server> Collaborator	Who's Online	The title of the Who's Online pane in Collaborator Workbench.
UserTitleContentProvider .TimeToSleep	Application Server> Collaborator	10000	To update the Who's Online pane in Collaborator Workbench, Demantra polls at regular intervals to see which users are online. This parameter specifies the length of time (in milliseconds) between successive polls.
V			
VisibleUserName		User Name	Determines how a user is identified in Demantra. This parameter is only available to administrators. Possible values are: User Name First Name Last Name
.			
W			
workflow.group	Application Server> Workflow		Comma-separated list of groups whose users are authorized to log into the Workflow Editor. Use the group names as specified in the Business Modeler. In order to log into the Workflow Editor, these users also must have System Manager permission level. See "Providing Access to the Workflow Editor".
worksheet.data.comb.block_size	Application Server> App. Server	1	Indicates whether we want to retrieve all WS Combinations in one sql (value=0) or with a block of Combinations in the sql (value>0).

Parameter	Location	Default	Details
WorksheetBeanContentProvider. memberCombinationsLimit	Application Server> Collaborator	4	Maximum number of combinations in a worksheet.
WorksheetCachingAsDefault	System		Specifies the default setting for the Cache Worksheet Data check box in the worksheet designer. This parameter has an effect only if worksheet caching is enabled (through EnableWorksheetCaching). Also see ManualRefreshAsDefault.
WorksheetDefaultDateChoice Method	Worksheet	relative to today	<p>Visible only to owner. Controls the default start date for Web worksheets, as seen in the worksheet designer. Use one of the following values:</p> <p>relative to today (0)</p> <p>relative to last loaded sales date (1)</p>
WorksheetDefaultSpan	Worksheet	104 (for a weekly system)	Visible only to owner. Specifies the default length of time for a Web worksheet, in base time units. Must be an even number, 2 or greater.

The message that a user receives when he or she receives a task. Title of the "enterprise" content link. Specifies whether the My Tasks module is displayed. If you disable this module, Collaborator Workbench also removes related options such as the Send as Task button. Specifies whether Collaborator Workbench displays worksheet content panes. If implemented, specifies whether to check whether the product matches between a settlement and the possible promotions. This parameter controls matching for all kinds of settlements, not just off-invoice settlements.

Database Procedures

This chapter lists the most commonly needed database procedures.

This chapter covers the following topics:

- Recommended Procedure Scheduling
- ANALYZE_SCHEMA
- API_CONFIG_SETTLEMENT
- APPPROC_REBUILD_CONTEXT_CACHES
- CHAINING
- CLEAN_LOG_TABLES
- COMPUTE_STD_ERR
- CREATE_OPT_STRUCT
- CREATE_PE_STRUCT
- DELETE_FORE_COL
- DELETE_INVALID_GL_POPULATION
- DROP_TEMP_TABLES
- EP_LOAD_MAIN
- EP_LOAD_MDP_LEVEL
- EP_LOAD_ITEMS
- EP_LOAD_LOCATIONS
- EP_LOAD_SALES
- EXECUTE_PROFILES
- EXPOSE_PROMOTIONS
- INSERT_UNITS
- MANUALS_INS

- MANUALS_INS_INTEGRATION
- MANUALS_INS_RECOVERY
- MDP_ADD
- POP_ALL_MATCH_PROPOSAL
- POP_OI_MATCH_PROPOSAL
- PRE_LOGON
- REBUILD_INDEXES
- REBUILD_SCHEMA
- REBUILD_TABLES
- REPLACE_APOSTROPHE_LEVELS
- SCHEDULED_CLEANUP_TASKS
- UPGRADE_TO_SHAPE_MODELLING

Recommended Procedure Scheduling

The following table lists the most important predefined procedures provided by Demantra. You should schedule all these procedures to run periodically from workflows.

Procedure	Recommended to run...
ANALYZE_SCHEMA	run after the first import, and then once per week (for example, by running the Scheduled Cleanup Tasks workflow) run after running REBUILD_INDEXES and REBUILD_TABLES
CHAINING	every 10 seconds
COMPUTE_STD_ERR	once a week, if you are performing standard error calculations*
DROP_TEMP_TABLES	once per week (for example, by running the Scheduled Cleanup Tasks workflow)
MANUALS_INS	every 10 seconds, if you are using the desktop products (Demand Planner and Demand Replenisher)

Procedure	Recommended to run...
REBUILD_INDEXES**	once a week, on a weekend day*
REBUILD_TABLES**	run after the first import run once a week, on a weekend day*
SCHEDULED_CLEANUP_TASKS	once a week, on Saturdays (this is the default)
<p>*It is recommended that you run each of these procedures at off hours and at staggered times to avoid deadlock issues. **These procedures require tablespace equal in size to the current tablespace.</p>	

ANALYZE_SCHEMA

Oracle only. Analyzes all the tables in the schema. By default, this procedure runs once a week as part of the Scheduled Cleanup Tasks workflow.

API_CONFIG_SETTLEMENT

DSM associates settlements with an item and a location level and is shipped pre-configured to an existing item and location level. If you want to change this information, run the procedure API_CONFIG_SETTLEMENT.

This procedure:

- Updates the item and location levels with which settlements should be associated
- Upgrades the Settlement Location alias level to point to the new location level
- Ensures that all standard procedures reflect the new settlement levels

For details, see "Setting up Database Structures" in the *Configuring DSM* chapter.

APPPROC_REBUILD_CONTEXT_CACHES

The APPPROC_REBUILD_CONTEXT_CACHES stored procedure automates the creation of caches for worksheets that use the **Open With** functionality and where the **Open With Context** for the worksheet is set to Selected Member.

This procedure simulates the user open with action which allows worksheet caches to be created in a batch process thereby eliminating the need to build the cache the first time the user opens the worksheets.

Note: This procedure improves the efficiency of building common caches by creating a cache for one user and copying it to other users instead of creating the cache for each user. This is accomplished by specifying a User Group. When common caches are detected, the cache is created for one of the Users in the User Group and copied to other Users in the User Group that require the same cache.

This procedure checks User security only on the Open With Level when creating and copying caches. It is important to structure the User Groups to be used by this procedure so that all Users have common security for all levels other than the Open With Level of the Worksheet that the cache is being built for.

This procedure must be incorporated into a workflow that also runs the Worksheet Cache Step. See *Worksheet Caching*, page 8-11 > *Creating a Workflow to Build Worksheet Caches Automatically*.

The APPPROC_REBUILD_CONTEXT_CACHES stored procedure has the following requirements:

- Worksheet
- Level
- User Group

The worksheet to be cached:

- Must be a public worksheet
- Must have caching enabled
- Must be selected as a Worksheet for Levels in the Component definition in the Business Modeler

The level

- Must be selected for Open With on the worksheet
- Must contain an attribute named CAN_CREATE_CW which must be set to 1 for all members to build caches for. For details, see *Identifying the Members of the Open With Level to Cache* in this section.

In addition, a User Group must exist that contains all of the Users for which caches are to be built.

Following are the parameters for the APPPROC_REBUILD_CONTEXT_CACHES stored procedure. You set these parameters in the Stored Procedure Step Properties dialog in Workflow Manager:

- WORKSHEET_ID: The internal ID of the worksheet to be cached

- `LEVEL_ID`: The internal ID of the Open With level for the worksheet
- `GROUP_ID`: The internal ID of the User Group containing the Users for which caches will be created
- `RUN_MODE`:
 - 1: Pre-process called before the Worksheet Cache Step to create master caches for the first User in the User Group and Level members
 - 2: Post-process called after the Worksheet Cache Step to create copies of master caches for remaining Users in the User Group

Identifying the Members of the Open With Level to Cache

For each Member of the Open With Level for the Worksheet being cached, the `APPPROC_REBUILD_CONTEXT_CACHES` stored procedure requires a flag to indicate whether or not the member should be cached.

To set this up, perform the following:

1. In Business Modeler (**Configure > Configure Levels**), on the Open With level for the worksheet that you want to cache, create an attribute named `CAN_CREATE_CW`.
2. Set the value of the `CAN_CREATE_CW` attribute to 1 for all Level Members that you want to build caches for. You can set this value through any method you select such as a worksheet, data load process, or database procedure.

This setting applies to all worksheets that are cached using the `APPPROC_REBUILD_CONTEXT_CACHES` stored procedure for the particular Open With level.

CHAINING

Checks the chaining queue for any pending chaining operations; performs those operations. By default, this procedure runs once a week as part of the Scheduled Cleanup Tasks workflow.

CLEAN_LOG_TABLES

Removes old data from the `db_exception_log` and `audit_trail` tables. To determine which data to keep, this procedure checks the value of the `audit_history_length` parameter, which you can access in the Business Modeler. By default, this procedure runs once a week as part of the Scheduled Cleanup Tasks workflow.

COMPUTE_STD_ERR

Performs any pending standard error calculations.

CREATE_OPT_STRUCT

Creates the default database structures needed for the Promotion Optimization module. The installer runs this procedure, if you choose to set up a default Promotion Optimization environment.

For details, see "Other Configuration for PTP".

CREATE_PE_STRUCT

Creates the default database structures needed for Promotion Effectiveness.

Demantra provides the Promotion Effectiveness (PE) structures by default. Therefore, the CREATE_PE_STRUCT procedure should be run only after creating a new or custom data model from scratch (this is not common or recommended); only in this scenario would these structures not already be in place. To determine whether PE structures exist, search the DB for table names containing %PROMOTION%.

For details, see "Configuring Promotion Effectiveness".

DELETE_FORE_COL

Deletes all the history data from the current forecast column, for all combinations for which prediction_status equals 1. The procedure deletes history, starting at the date given by max_sales_date - ForecastGenerationHorizon.

ForecastGenerationHorizon is an engine parameter; see "Engine Parameters".

DELETE_INVALID_GL_POPULATION

Checks the promotion_data table for invalid records and deletes them.

DROP_TEMP_TABLES

Deletes temporary database tables that are created, for example, when modifying worksheets, launching integration profiles, or running the Analytical Engine.

By default, this procedure runs once a week as part of the Scheduled Cleanup Tasks workflow.

EP_LOAD_MAIN

Performs the data loading described in the Data Model Wizard. Specifically, this procedure loads data from the staging tables specified within the Data Model Wizard and writes records in sales_data and other internal tables as needed.

This procedure is created by the Data Model Wizard and thus is different in different implementations.

EP_LOAD_MDP_LEVEL

EP_LOAD_MDP_LEVEL is the main method of populating matrix levels. During data load, information from sales master that is not time dependant can be loaded to these levels.

The following parameters are used with the EP_LOAD_MDP_LEVEL procedure:

Ep_Load_Mdp_LevelUseParallelDML

Description	If you run DML statements, any objects with a parallel degree configured will be run in parallel. Individual parallel hints setting will supersede this setting.
Values	TRUE - LOAD_MDP_LEVEL should use PARALLEL DML on objects that are set as parallel. FALSE - LOAD_MDP_LEVEL should not use PARALLEL DML on objects that are set as parallel.
Default Value	FALSE

Ep_Load_Mdp_LevelUseParallel

Description	Set this parameter to FALSE when you do not want to use parallel hints in the EP_LOAD_MDP_LEVEL script. Should be set to FALSE when database resources may be limited.
Values	TRUE - MDP_ADD will use PARALLEL hints. FALSE - MDP_ADD will not use PARALLEL hints.

Default Value	TRUE
---------------	------

Ep_Load_Mdp_LevelUseOneCommit

Description	Set this parameter to FALSE if the database is relatively small and has smaller rollback segments. If your database is powerful and thus sufficient resources are available, then set this parameter to TRUE.
Values	TRUE - EP_LOAD_MDP_LEVEL executes only one commit.FALSE - EP_LOAD_MDP_LEVEL executes multiple commits.
Default Value	FALSE

Ep_Load_Mdp_LevelSetProp_Changes

Description	This parameter determines whether the MDP_ADD process will set PROP_CHANGES to 1. PROP_CHANGES are used to calculate status and proportions of MDP_MATRIX combinations.This parameter should be set to FALSE if another process calls the PROPORT procedure or status and proportions are defined during a different process.
Values	TRUE - Set PROP_CHANGES = 1 during EP_LOAD_MDP_LEVEL.FALSE - Do not set PROP_CHANGES = 1 during EP_LOAD_MDP_LEVEL.
Default Value	FALSE

Ep_Load_Mdp_LevelDisableEnableIndexes

Description	This parameter determines whether the MDP_ADD process will set PROP_CHANGES to 1. PROP_CHANGES are used to calculate status and proportions of MDP_MATRIX combinations. This parameter should be set to FALSE if another process calls the PROPORT procedure or status and proportions are defined during a different process.
Values	TRUE - Set PROP_CHANGES = 1 during EP_LOAD_MDP_LEVEL. FALSE - Do not set PROP_CHANGES = 1 during EP_LOAD_MDP_LEVEL.
Default Value	FALSE

Ep_Load_Mdp_LevelDisableEnableIndexes

Description	Controls whether to drop indexes on a table when loading MDP_LEVEL information.. MDP indexes are rebuilt after execution.
Values	TRUE - Disable and then enable (rebuild) indexes. FALSE - Indexes remain in place during the updates.
Default Value	FALSE

Ep_Load_Mdp_LevelIndexRebuildDegree

Description	If rebuilding indexes using Ep_Load_Mdp_LevelDisableEnableIndexes, this parameter specifies which degree to use.
Values	---
Default Value	4

Ep_Load_Mdp_LevelUseMerge

Description	This parameter controls the MDP_MATRIX update method. Choose either Merge or Cursor. For larger data sets, with appropriate database resources, MERGE should be faster.
Values	TRUE - Use the MERGE update method. FALSE - Use the CURSOR update method.
Default Value	TRUE
Details	DELETE mdp_matrix WHERE NOT EXISTS (SELECT ... SALES_DATA s) AND NOT EXISTS (SELECT hint PROMOTION_DATA s))

DBHintEp_Load_Mdp_LevelMerge

Description	SQL hint to use when Ep_Load_Mdp_LevelUseMerge is set to MERGE. This parameter defines the degree used during the Merge.
Values	TRUE - Use the MERGE update for the MDP_MATRIX cache columns - FROM_DATE and UNTIL_DATE. FALSE - Use a cursor loop.
Default Value	+ full (a) parallel (a 20)
Details	Example: MERGE hint UPDATE INTO MDP_MATRIX

EP_LOAD_ITEMS

The EP_LOAD_ITEMS procedure is the main method by which items are loaded into Demantra internal tables. The procedure validates item hierarchy compliance and uploads all item dimension information into internal tables.

The following parameters can be used with the EP_LOAD_ITEMS procedure:

Ep_Load_ItemsUseParallelDML

Description	If you run DML statements, any objects with a parallel degree configured will be run in parallel. Individual parallel hints setting will supersede this setting.
Values	
Default Value	FALSE

Ep_Load_ItemsUseParallel

Description	Set this parameter to FALSE when you do not want to use parallel hints in the EP_LOAD_ITEMS script. This parameter should be set to FALSE if your database has limited resources.
Values	TRUE - EP_LOAD_ITEMS will use parallel hints. FALSE - EP_LOAD_ITEMS does not use parallel hints.
Default Value	TRUE

Ep_Load_ItemsUseOneCommit

Description	Set this parameter to FALSE if your database is relatively small and has smaller rollback segments. If your database is powerful and thus sufficient resources are available, then set this parameter to TRUE.
Values	TRUE - EP_LOAD_ITEMS executes only one commit. FALSE - EP_LOAD_ITEMS may execute more than one commit.
Default Value	FALSE

DBHintEp_Load_ItemsMergeMDP

Description	Hint for sub-select in merge statement with MDP_MATRIX. Used in the same statement with hint DBHintEp_Load_ItemsMergeMDP. The total degree of both hints should not exceed number of database CPUs.
Values	---
Default Value	+ parallel (a4)
Details	MERGE INTO mdp_matrix (SELECT hint)

Ep_Load_ItemsDisableEnableMDPIndexes

Description	This parameter disables indexes before performing the update on MDP_MATRIX and then rebuilds them when the update is complete. This parameter should be used only if the customer data set is very large and rebuilding indexes is faster than updates with indexes enabled. Note: This parameter is not currently used.
Values	TRUE - Indexes on columns updated in the MDP_MATRIX MERGE update are disabled before the update and are rebuilt after the update. FALSE - Indexes remain during the update.
Default Value	FALSE
Details	This parameter is not currently used.

Ep_Load_ItemsIndexRebuildDegree

Description	If EP_LOAD_ITEMS rebuilds indexes, this parameter controls what parallel degree to use. The value of this parameter should not exceed the number of available CPUs.
Values	---
Default Value	4
Details	This parameter is not currently used.

EP_LOAD_LOCATIONS

The EP_LOAD_LOCATIONS procedure is the main method by which locations are loaded into Demantra internal tables. The procedure validates location hierarchy compliance and uploads all Location Dimension information into internal tables.

The following parameters can be used with the EP_LOAD_LOCATIONS procedure:

Ep_Load_LocationsUseParallelDML

Description	If you run DML statements, any objects with a parallel degree configured will be run in parallel. Individual parallel hints setting will supersede this setting.
Values	TRUE - EP_LOAD_LOCATIONS should use PARALLEL DML on objects that are set as parallel. FALSE - EP_LOAD_LOCATIONS should not use PARALLEL DML on objects that are set as parallel.
Default Value	FALSE

Ep_Load_LocationsUseParallel

Description	Set this parameter to FALSE when you do not want to use parallel hints in the EP_LOAD_LOCATIONS script. Set this parameter to FALSE if your database has limited resources.
Values	TRUE - EP_LOAD_LOCATIONS will use parallel hints. FALSE - EP_LOAD_LOCATIONS will not use parallel hints.
Default Value	TRUE

Ep_Load_LocationsUseOneCommit

Description	Set this parameter to FALSE if your database is relatively small and has smaller rollback segments. If your database is powerful and thus sufficient resources are available, then set this parameter to TRUE.
Values	TRUE - EP_LOAD_LOCATIONS executes only one commit. FALSE - EP_LOAD_LOCATIONS may execute more than one commit.
Default Value	FALSE

DBHintEp_Load_LocationsMergeMDP

Description	Hint for merge statement with MDP_MATRIX. Used in the same statement with hint DBHintEp_Load_Locations_MergeMDP_GLOBAL. The total degree of both hints should not exceed number of database CPUs.
Values	---

Default Value	+ parallel (m4)
Details	MERGE (hint) INTO mdp_matrix (SELECT)

DBHintEp_Load_LocationsMergeMDP_GLOB

Description	Hint for sub-select in merge statement with MDP_MATRIX. Used in the same statement with hint DBHintEp_Load_LocationsMergeMDP. The total degree of both hints should not exceed number of database CPUs.
Values	---
Default Value	+ parallel (a4)
Details	MERGE INTO mdp_matrix (SELECT hint)

Ep_Load_LocationsDisableEnableMDPIndexes

Description	This parameter disables indexes before performing the update on MDP_MATRIX and then rebuilds them when the update is complete. This parameter should be used only if your data set is very large and rebuilding indexes is faster than updates with indexes enabled. Note: This parameter is not currently used.
Values	TRUE - EP_LOAD_LOCATIONS disables indexes on columns updated in the MDP_MATRIX MERGE update. FALSE - EP_LOAD_LOCATIONS does not disable indexes on columns updated in the MDP_MATRIX MERGE update.
Default Value	FALSE

Ep_Load_LocationsIndexRebuildDegree

Description	If EP_LOAD_LOCATIONS rebuilds indexes, this parameter controls what parallel degree to use. The value of this parameter should not exceed the number of available CPUs.Note: This parameter is not currently used.
Values	---
Default Value	4

EP_LOAD_SALES

The EP_LOAD_SALES procedure loads lowest level time-dependant information into the Demantra SALES_DATA table. This information includes historical data as well as other time-varying information such as price, booking, and shipments.

The following parameters can be used with the EP_LOAD_SALES procedure.

DBHintEp_Load_SalesUseParallelDML

Description	If you run DML statements, any objects with a parallel degree configured will be run in parallel.Individual parallel hints setting will supersede this setting.
Values	TRUE - LOAD_MDP_LEVEL will use PARALLEL DML on objects that are set as parallel.FALSE - LOAD_MDP_LEVEL will not use PARALLEL DML on objects that are set as parallel.
Default Value	FALSE

DBHintEp_Load_SalesUseParallel

Description	Set this parameter to FALSE when you do not want to use parallel hints in the EP_LOAD_SALES script. Should be set to FALSE when database resources are limited.
Values	TRUE - EP_LOAD_SALES will use PARALLEL hints. FALSE - EP_LOAD_SALES will not use PARALLEL hints.
Default Value	TRUE

DBHintEp_Load_SalesInsertErr

Description	When checking data for errors. EP_LOAD_SALES will send invalid rows to error tables. This parameter enables hints on the SELECT statement to determine if there is an error. If loading very large source tables, providing a hint is recommended.
Values	---
Default Value	+ parallel (@T_SRC_SALES@ 4) full(@T_SRC_SALES@)
Details	Example: INSERT INTO ... SELECT hint ... MINUS SELECT hint ... Note: Leave the token @T_SRC_SALES@ to be used for the real T_SRC sales table.

DBHintEp_Load_SalesInsertLd

Description	Use this parameter when data from a staging table is aggregated to match internal time resolution. The hint defines the degree used during insert for aggregation.
Values	---

Default Value	+ parallel (@T_SRC_SALES@ 4)
Details	Example Usage: INSERT EP_(t_src_sales)_LD... . This is the insert statement from the T_SRC_SALES table to the EP_T_SRC_SALES_LD table. Note: Leave the token @T_SRC_SALES@ to be used for the real T_SRC sales table.

DBHintEp_Load_SalesMergeMDP_LA

Description	When inserting information into MDP_MATRIX_LOAD_ASSIST, this hint defines the degree used in the subselect statement.
Values	---
Default Value	+ parallel (@T_SRC_SALES_LD@ 4) full (@T_SRC_SALES_LD@)
Details	Example Usage: MERGE INTO mdp_load_assist USING (SELECT hint ... FROM EP_(t_src_sales)LDNote: Leave the token @T_SRC_SALES_LD@ to be used for the real T_SRC Load sales table,

DBHintEp_Load_SalesMergeSALES_DATA

Description	When merging data into SALES_DATA table, this hint defines the degree used in the sub-select from intermediate data table.
Values	---
Default Value	+ parallel (@T_SRC_SALES_LD@ 4) full (@T_SRC_SALES_LD@)

Details	Example:MERGE INTO SALES_DATA USING (SELECT hint ... FROM EP_ (t_src_sales)LDNote: Leave the token @T_SRC_SALES_LD@ to be used for the real T_SRC Load sales table.
---------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Ep_Load_Sales_IgnoreNullActualQty

Description	This parameter is used when inserting a row into the MDP_MATRIX table based on historical demand records.Set this parameter to FALSE if you are loading all rows in sales history. Set this parameter to True if you want to ignore rows where historical demand is null.Setting of FALSE is typically faster.
Values	TRUE -Ignore/do not process for NULL values.FALSE - Process NULL values.
Default Value	FALSE
Details	---

Ep_Load_Sales_MDP_LOAD_ASSIST_GatherStatsIfNoStats

Description	Gathering statistics can be very slow in large environments.Initially, you would want to gather statistics but subsequently you may not want to gather statistics to improve performance. Parameter controls whether statistics are gathered on the MDP_LOAD_ASSIST table. When set to FALSE, statistics are always gathered. When set to TRUE statistics are gathered only if none exist.
Values	TRUE - Gather statistics ONLY if MDP_LOAD_ASSIST has no table statistics.FALSE - Always gather statistics for MDP_LOAD_ASSIST.

Default Value	FALSE
Details	---

Ep_Load_Sales_MDP_LOAD_ASSIST_GatherStats_1

Description	Gather statistics for MDP_LOAD_ASSIST. Gathering executed first merge into MDP_LOAD_ASSIST.
Values	TRUE - Gather statistics after the first merge into MDP_LOAD_ASSIST if no statistics are available.FALSE - Always gather statistics for MDP_LOAD_ASSIST after the first merge.
Default Value	TRUE

Ep_Load_Sales_MDP_LOAD_ASSIST_GatherStats_2

Description	Gather statistics for MDP_LOAD_ASSIST. Will be executed after second merge into MDP_LOAD_ASSIST.
Values	TRUE - Gather statistics after the second merge into MDP_LOAD_ASSIST if no statistics are available.FALSE - Always gather statistics for MDP_LOAD_ASSIST after the second merge.
Default Value	TRUE

Ep_Load_Sales_MDP_LOAD_ASSIST_GatherStats_3

Description	Gather statistics for MDP_LOAD_ASSIST. Will be executed later in code, after delete from MDP_LOAD_ASSIST. This parameter will not be used if delete duplicates is not used (see Ep_Load_Sales_MDP_LOAD_ASSIST_DelDuplicates).
Values	TRUE - Gather statistics after deleting duplicates from MDP_LOAD_ASSIST if no statistics are available. FALSE - Always gather statistics for MDP_LOAD_ASSIST after deleting duplicates.
Default Value	TRUE

Ep_Load_Sales_MDP_LOAD_ASSIST_DelDuplicates

Description	Controls whether MDP_LOAD_ASSIST deletes any duplicates found in MDP_LOAD_ASSIST. Should be enabled if rows frequently are added to MDP_LOAD_ASSIST from a variety of sources.
Values	TRUE - Delete duplicates from MDP_LOAD_ASSIST. FALSE - Do not delete duplicates from MDP_LOAD_ASSIST.
Default Value	TRUE

Ep_Load_Sales_Run_CHANGE_IS_FICTIVE

Description	Controls whether the procedure CHANGE_IS_FICTIVE is executed. This parameter must be set to TRUE if you are using Members Management or Chaining.
-------------	---------------------------------------------------------------------------------------------------------------------------------------------------

Values	TRUE - Run CHANGE_IS_FICTIVE.FALSE - Do not run CHANGE_IS_FICTIVE.
Default Value	TRUE

Ep_Load_Sales_SALES_DATA_Merge_InOneMerge

Description	Controls whether rows are inserted into SALES_DATA from staging table in one MERGE or using a loop.Should be set to FALSE for smaller environments, with smaller database and rollback settings.For more powerful environments, setting this parameter to TRUE will improve performance.
Values	TRUE - Use a single MERGE into SALES_DATA.FALSE - Merge is run in a location level loop.
Default Value	FALSE

Ep_Load_Sales_SALES_DATA_Merge_LoopControl

Description	If Ep_Load_Sales_SALES_DATA_Merge_InOne Merge is set to FALSE, then this level will define the commit points. The name of a suitable T_SRC_SALES_TMPL location column should be entered as an input.If left null, the generated default is the one of the first Location level columns available. Level chosen should have an average number of combinations matching desired commit point.
Values	---
Default Value	---

EXECUTE_PROFILES

Executes all the active rolling data profiles in the system. These are defined through the Business Modeler and are stored in the `rolling_profiles` table. A profile is active if it is selected in the Configure Rolling Session dialog box.

For each active profile, Business Modeler copies the source data into the target series. Data for any given time bucket is copied into the same time bucket in the target series.

The profiles are executed in the order in which they are listed in the Configure Rolling Session dialog box.

EXPOSE_PROMOTIONS

Iterates through the promotions listed in the promotion table, checks the status of each (the status field), and does the following:

- If the current status is 3 (planned) and if the current date is after the `from_date` of the promotion, change the status to 4 (committed).
- If the current status is 4 (committed) and if the current date is after the `until_date` of the promotion, change the status to 5 (executed).

For details, see "Configuring Promotion Effectiveness".

INSERT_UNITS

INSERT_UNITS is the mechanism by which future rows are inserted into the SALES_DATA table. Before each engine task is executed, all combinations belonging to the task which will receive a forecast are evaluated and future rows are inserted where appropriate. Ensuring the appropriate rows are available enables the engine output to be written quickly and efficiently.

The Analytical Engine calls this procedure at the start of an engine run. For performance reasons, Insert_Units automatically runs as a distributed process. No setup is required.

This procedure is controlled by the `RunInsertUnits` parameter and can do several things, depending on the value of that parameter:

Makes sure the engine has rows to write into when generating the forecast. In particular, for *all non-dead* combinations, this procedure does the following:

- Checks to see if the database contains records for this combination for all dates in the span of time from `max_sales_date` to `max_sales_date + lead`.
 - For any dates when the combination does not have records, this procedure inserts records with zero sales, into which the Analytical Engine can then write

the forecast.

- Records with dates in the past are ignored.
- Runs the EXECUTE_PROFILES procedure, which executes the active rolling data profiles.

RunInsertUnits and lead are engine parameters; see "Engine Parameters".

The following parameters can be used with both the INSERT_UNITS and the DYN_INSERT_UNITS_BRANCH_ID procedures:

iu_do_commits

Description	Should be set to TRUE for smaller databases or those with limited resources and smaller rollback segments. For larger databases with sufficient resources, set to FALSE.
Values	TRUE – The EP_LOAD procedure will do commits as needed.FALSE – Use only one commit per run of a DYN_INSERT_UNITS_BRANCH_ID procedure.
Default Value	TRUE

ep_load_do_commits

Description	Used with the EP_LOAD_SALES procedure. Defines whether the procedure will commit as needed or wait until the entire process is complete. When DB resources may be limited, set this parameter to False.
Values	TRUE - EP_LOAD will use the commit control and issue commits per n transitions.(Where n = SYS_PARAMS setting max_records_for_commit) FALSE - Use only one commit should be used per ep_load procedure.
Default Value	TRUE

MANUALS_INS

Updates all the appropriate database tables after a user makes a change in a worksheet.

Note that the `update_dead_comb` parameter controls whether this procedure considers dead combinations.

You use this procedure only if you are using the desktop product (Demand Planner and Demand Replenisher) rather than the Web products. The Web products (Demand Planner Web, Promotion Effectiveness, and Settlement Management) use a Java-based mechanism to update the same tables.

`update_dead_comb` is an engine parameter; "Engine Parameters".

MANUALS_INS_INTEGRATION

Updates all the appropriate database tables after data has been imported. Make sure you run this procedure after running an import integration interface.

MANUALS_INS_RECOVERY

Performs a recovery if `MANUALS_INS` or `MANUALS_POPULATE_INS` has failed. This procedure retrieves the temporary records that were written to the `update_batch_trai_err` and `update_batch_values_err` tables and completes the processing.

MDP_ADD

The `MDP_ADD` procedure is the main method by which Item/Location combinations are created in Demantra. Combinations are created and maintained based on Item/Location combinations existing in the `SALES_DATA` table. `MDP_ADD` also populates and maintains hierarchy information which is stored on `MDP_MATRIX` and executes the `PROPORT` procedure.

This procedure is run automatically when needed. It is responsible for maintaining the `mdp_matrix` table.

If the Analytical Engine fails with an error ("node not found in map"), you can correct the error by setting the `align_sales_data_levels_in_loading` parameter to true and then manually running the `MDP_ADD` procedure.

Depending on the setting of the `RunProportInMdp_add` parameter, this procedure may or may not call the `proport` mechanism when it runs.

The following parameters can be used with the `MDP_ADD` procedure:

Mdp_AddUpdates_Fictive

Description	Defines whether the Is_fictive column is to be maintained. Is_fictive is used only in Chaining and Members Management. If you are using Chaining and Members Management, this parameter should be set to TRUE. If you are not using these features, performance may improve by setting this parameter to FALSE.
Values	TRUE - MDP_ADD should update IS_FICTIVE. FALSE - MDP_ADD should not update IS_FICTIVE.
Default Value	TRUE

Mdp_AddUpdates_Linked

Description	Defines whether the Is_linked column is to be maintained. Is_fictive is used only in Chaining and Members Management. If you are using Chaining and Members Management, this parameter should be set to TRUE. If you are not using these features, performance may improve by setting this parameter to FALSE.
Values	TRUE - MDP_ADD should update IS_LINKED. FALSE - MDP_ADD should not update IS_LINKED.
Default Value	TRUE

Mdp_AddDeleteUnlinkedMdp_Matrix

Description	This parameter controls whether MDP_MATRIX combinations not found in historical data tables are removed from the matrix. Setting depends on functionality. In large implementations substantial saving can be achieved by setting this parameter to FALSE.
-------------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Values	TRUE - MDP_ADD should DELETE MDP_MATRIX rows not linked with SALES_DATA. In a PE schema, the delete also includes rows not linked with PROMOTION_DATA.FALSE - MDP_ADD should not DELETE MDP_MATRIX rows not linked with SALES_DATA.
Default Value	FALSE

DBHintMdp_AddUseParallelDML

Description	If you run DML statements, any objects with a parallel degree configured will be run in parallel.Individual parallel hints setting will supersede this setting.
Values	TRUE - Use PARALLEL DML on objects that are set as parallel.FALSE - Do not use PARALLEL DML on objects that are set as parallel.
Default Value	FALSE

DBHintMdp_AddUseParallel

Description	Set this parameter to FALSE if you want to use parallel hints in the MDP_ADD script. This parameter should be set to FALSE when DB resources may be limited.
Values	TRUE - EP_LOAD_ITEMS will use parallel hints.FALSE - EP_LOAD_ITEMS will not use parallel hints.
Default Value	TRUE

DBHintMdp_AddInsert

Description	Controls the degree used during the MDP_ADD insert statement when inserting rows into MDP_MATRIX. Used in the same statement as hint DBHintMdp_AddInsertSelect .The combined value of both parameters should not exceed the number of available CPUs.
Values	+ parallel(mdp_matrix 4) append
Default Value	---
Details	INSERT hint INTO mdp_matrix

DBHintMdp_AddInsertSelect

Description	Controls the degree used during the internal select of the MDP_ADD insert statement when inserting rows into MDP_MATRIX. Used in the same statement as hint DBHintMdp_AddInsert.The combined value of both parameters should not exceed the number of available CPUs.
Values	---
Default Value	+ parallel (@TBL_NAME@ 2) parallel (location 2) parallel (items 2)
Details	INSERT INTO mdp_matrix SELECT hint.Note: Leave the token @TBL_NAME@ unchanged as it is to be used for the real table, usually MDP_LOAD_ASSIST.

DBHintMdp_AddMdpInsInMdpUseIn

Description	If this parameter is set to TRUE, the MDP_ADD procedure checks if a row already exists before inserting to MDP_MATRIX. This parameter also controls whether the statement uses IN or NOT EXISTS. If set to TRUE the statement uses IN; otherwise it uses NOT EXISTS.
Values	---
Default Value	TRUE
Details	The default setting (TRUE) was chosen for backwards compatibility. However, setting this parameter to FALSE may improve performance.

DBHintMdp_AddMdpInsInTbl

Description	SQL hint for inserting into MDP_MATRIX when choosing the IN mechanism via DBHintMdp_AddMdpInsInMdpUseIn. Used with hint DBHintMdp_AddMdpInsInMdp. The combined value of both parameters should not exceed the number of available CPUs.
Values	TRUE - EP_LOAD_LOCATIONS should use PARALLEL DML on objects that are set as parallel. FALSE - EP_LOAD_LOCATIONS should use PARALLEL DML on objects that are set as parallel.
Default Value	+ parallel (a 2)
Details	INSERT INTO mdp_matrix SELECT ... WHERE ... IN (SELECT hint mdp_load_assist ...)

DBHintMdp_AddMdpInsInMdp

Description	Controls minus sub-select of insert into MDP_MATRIX statement when choosing the IN mechanism. Used with DBHintMdp_AddMdpInsInTbl. The combined value of both parameters should not exceed the number of available CPUs.
Values	---
Default Value	+ parallel (b 2)
Details	INSERT into mdp_matrix SELECT ... WHERE ... IN (SELECT mdp_load_assist MINUS SELECT hint mdp_matrix)

DBHintMdp_AddMdpCount

Description	SQL hint on select count from MDP. This hint should be used in environments that have a large amount of data. This parameter should not be set to a value that exceeds the number of available CPUs.
Values	---
Default Value	+ full (MDP_MATRIX) parallel (MDP_MATRIX 4)
Details	SELECT hint COUNT (*) FROM mdp_matrix

DBHintMdp_AddUpdates_Fictive0MDP

Description	If updating Is_fictive, set degree parallel when updating MDP_MATRIX IS_FICTIVE value to 0. If you are not using Is_fictive, then this parameter is not used.
Values	---

Default Value	---
Details	UPDATE hint mdp_matrix SET is_fictive =...

DBHintMdp_AddUpdatels_Fictive0SD

Description	This parameter is used in the subselect for updating Is_fictive value in MDP_MATRIX to 0.The degree set for this parameter, combined with DBHintMdp_AddUpdateIs_Fictive0MDP, should not exceed the number of available CPUs.
Values	---
Default Value	+ parallel(s 2)
Details	UPDATE mdp_matrix SET is_fictive = 0 EXISTS (SELECT hint FROM sales_data ...) AND is_fictive = 2

DBHintMdp_AddUpdatels_Fictive2SD

Description	Used in the subselect for updating Is_fictive value in MDP_MATRIX to 0.The degree set in this parameter combined with DBHintMdp_AddUpdateIs_Fictive0MDP should not exceed the number of available CPUs.
Values	---
Default Value	+ parallel(s 2)
Details	UPDATE mdp_matrix SET is_fictive = 2 NOT EXISTS (SELECT hint FROM sales_data ...) AND is_fictive = 0

DBHintMdp_AddDeleteMdp

Description	If deleting MDP_MATRIX rows, use this parameter to set the degree used in delete.Used together with hints DBHintMdp_AddDeleteMdpNotInSD and DBHintMdp_AddDeleteMdpNotInPD. Total values of hints should not exceed number of available CPUs.
Values	---
Default Value	+ parallel(s 2)

DBHintMdp_AddDeleteMdpNotInSD

Description	Hint used in the sub-select of delete population when viewing SALES_DATA.Used together with hints DBHintMdp_AddDeleteMdp and DBHintMdp_AddDeleteMdpNotInPD. Total values of hints should not exceed number of available CPUs.
Values	---
Default Value	+ parallel(s 2)
Details	DELETE mdp_matrix WHERE NOT EXISTS (SELECT hint FROM SALES_DATA s...))

DBHintMdp_AddDeleteMdpNotInPD

Description	Hint on the subselect of delete population when viewing promotion_data. Only used in environments that support promotions.Used together with hints DBHintMdp_AddDeleteMdp and DBHintMdp_AddDeleteMdpNotInSD. The total value of hints should not exceed the number of available CPUs
-------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Values	---
Default Value	+ parallel (p 2)
Details	DELETE mdp_matrix WHERE NOT EXISTS (SELECT ... SALES_DATA s) AND NOT EXISTS (SELECT hint PROMOTION_DATA s))

DBHintMdp_AddUseMdpCacheMerge

Description	When updating from_date and until_date in MDP_MATRIX, the process be done via a single merge or a cursor loop.The Merge statement should be faster if there is a large amount of data with sufficient database capacity.
Values	TRUE - Use the MERGE update for the MDP_MATRIX cache columns - FROM_DATE and UNTIL_DATE.FALSE - Use a cursor loop.
Default Value	TRUE

POP_ALL_MATCH_PROPOSAL

Only for DSM. This procedure iterates through all settlements (except for off-invoice settlements), finds promotions that meet all the match criteria, and writes a record into the proposed_match table for each match.

This procedure performs the following comparisons, which are controlled by parameters:

- It compares the promotion date (DSMPEShipDateSeries) to the settlement date. Only promotions with close enough dates are considered possible matches.

The DSMAllShipDateDifference parameter specifies the window of time that Demantra uses to search for a promotion that matches a given settlement. Express this as the number of time buckets between the promotion end date and the deduction date.
- It compares the promotion budget (DSMPromotionBudgetSeries) to the monetary settlement amount. A promotion is a possible match only if its remaining budget is

at least as large as the settlement amount.

POP_OI_MATCH_PROPOSAL

Only for DSM. This procedure iterates through all off-invoice settlements, finds promotions that meet all the match criteria, and writes a record into the proposed_match table for each match.

This procedure performs the following comparisons, which are controlled by parameters:

- It compares the promotion budget (DSMPromotionBudgetSeries) to the off-invoice amount. For this comparison, the DSMOIPercentDifference parameter specifies the maximum percent difference (of monetary amount) permitted when matching an off-invoice settlement to possible promotions.
- It compares the promotion date (DSMPEShipDateSerie) to the off-invoice date. Only promotions with close enough dates are considered possible matches. You use the DSMOIShipDateDifference parameter to specify the closeness of these dates.
- It can also check that the off-invoice settlement and the possible promotions use the same product. To control this check, you use the DSMOICheckProduct parameter.

PRE_LOGON

Sets the database date and time formats.

Many other predefined procedures automatically call this procedure.

REBUILD_INDEXES

Oracle only. Rebuilds table indexes, a necessary maintenance task for the database.

Note: This procedure requires additional space (equal to the current tablespace) and can take a long time.

REBUILD_SCHEMA

Rebuilds all tables, a necessary maintenance task for Oracle databases.

Note: This procedure requires additional space (equal to the current tablespace) and can take a long time.

REBUILD_TABLES

Oracle only. Rebuilds the sales_data and mdp_matrix tables, a necessary maintenance task for Oracle databases.

Note: This procedure requires additional space (equal to the current tablespace) and can take a long time.

Arguments

This procedure has the following optional positional arguments:

- The first argument indirectly specifies which tables to rebuild. If null, the procedure rebuilds tables according to the Rebuild_Sales_Table parameter. If this argument is 1, the procedure rebuilds the sales_data table. If this parameter is 0, the procedure skips the sales_data.
- If the second argument is 0, the procedure rebuilds the sales_data (if permitted by the previous argument), mdp_matrix, items and location tables. If this parameter is 1, the procedure rebuilds all tables listed in user_tables that need to be rebuilt.

REPLACE_APOSTROPHE_LEVELS

Iterates through the level tables and replaces any apostrophes in the column names with underscore characters.

SCHEDULED_CLEANUP_TASKS

Runs the Analyze Schema, Drop Temp Tables, Clean Log Tables, and Rebuild Tables workflows. By default this procedure runs once a week, on Saturdays.

UPGRADE_TO_SHAPE_MODELLING

Creates samples for activity shape modeling. Specifically this procedure does the following:

- Creates two sample activity causal factors: Product_launch and Price_change.
- It creates four editable series for the benefit of end users, described in the following table.

Series Name	Data Association	Series Purpose
Price_change	Sales	Lets the user indicate the start and duration of the price change shape associated with a specific combination. Within this series, for each date, the user chooses "Start" or "Active" from a drop-down menu to specify the promotion start and continuation dates. The default is "None," meaning no promotion. The user identifies past activities and marks where future activities will occur.
Price_change_QAD	Combination	Controls whether the Analytical Engine re-scales the generated shape to align with the amplitude of the most recent observed instance of this shape, for a given combination. Specify the number of buckets for which the shape alignment should occur, starting with the beginning of the shape. Typically you use either 0 or the length of the shape.
Product_launch	Sales	Like Price_change, but applies to the product launch shape instead of the price change shape.
Product_launch_QAD	Combination	Like Price_change_QAD, but applies to the product launch shape instead of the price change shape.

See "About Activity Shape Modeling".

Key Tables

This chapter provides reference information for some of the most important tables in Demantra, especially the data fields used by or written by the Analytical Engine. Unless otherwise noted, this information applies to all Demantra products.

This chapter covers the following topics:

- Sales_Data
- Mdp_matrix
- Promotion_Data

Sales_Data

The following table lists the most important fields in the sales_data table. The Analytical Engine reads from and writes to some of these fields, which you use mainly to create series.

Field Name	Use	Field Purpose
item_id	Read-only	Unique identifier for the item. Together, item_id, location_id, and sales_date form the primary key for rows in the sales_data table.
location_id	Read-only	Unique identifier for the location.
sales_date	Read-only	Date for this record.

Field Name	Use	Field Purpose
item_price	Read-only (imported)	Price for this item, at this location, on this date.
actual_quantity		
manual_fact		
manual_stat		
salesplus	Read-only (imported)	The demand used by the Analytical Engine.
orders		
FORE_0, FORE_1, FORE_2, ...	Read-only	The forecasts generated by the Analytical Engine. The Analytical Engine cycles through these columns. Each time, it writes the current forecast into one column (overwriting the oldest forecast). The Analytical Engine then adds a row to the forecast_history table that describes this forecast and that indicates which column it is stored in.
OBS_ERROR_STD	User input	<p>Specifies how the Analytical Engine should consider this observation when fitting each engine model. Specify a positive number, to be used as a weight for this observation. Use 1 to treat this observation as a standard observation.</p> <p>This field is ignored unless UseWeightedRegression is specified as yes (1).</p> <p>UseWeightedRegression is an engine parameter; see "Engine Parameters" .</p>

Field Name	Use	Field Purpose
outlier	Read-only	Indicates whether the Analytical Engine has marked this row as an outlier.
regime_change	Read-only	Indicates whether the Analytical Engine has marked this combination as an regime change.
approve	User input	
final approve	User input	
batch	Read-only	
PD1, PD2, PD3, PD4, PD5, PD6, PD7	No	Available only in a daily system. Daily proportions for this combination, for different days of the week.
PW1, PW2, PW3, PW4, PW5, PW6	No	Available only in a weekly or daily system. Weekly proportions for this combination, for different weeks of a month. When calculating these proportions, Demantra factors in the number that this week has.

Mdp_matrix

The following table lists the most important fields in the mdp_matrix table. You can use these fields to create series or levels that provide information about different combinations or that enable the user to manipulate different combinations.

Field Name	Use	Field Purpose
item_id	Read-only	Unique identifier for the item. Together, item_id and location_id form the primary key for rows in the mdp_matrix table.

Field Name	Use	Field Purpose
location_id	Read-only	Unique identifier for the location.
aggri_98	User input	Specifies whether to aggregate demand for this item-location combination, if this combination is young. See "prediction_status".
aggri_99	User input	Specifies whether to aggregate demand for this item-location combination, if this combination is dead. See "prediction_status".
delta	User input	Used in the proposit calculation as in the following example: $P1 = \text{glob_prop} * \text{delta} + (\text{monthly demand}) * (1 - \text{delta})$
delta_d	User input	Specifies the day-to-day smoothing of the daily proportions, which are calculated as in the following example: $D1 = (\text{actual average for day 1}) * \text{delta_d} + (\text{weekly proportion}) * (1 - \text{delta_d})$ Here D1 is the proportion for the combination for the first day of the week.
delta_w	User input	Specifies the week-to-week smoothing of the weekly proportions, which are calculated as in the following example: $PW1 = (\text{actual average for week 1}) * \text{delta_w} + (\text{monthly proportion}) * (1 - \text{delta_w})$ Here PW1 is the proportion for the combination for the first week of the month.
do_aggri	User input	Specifies whether to perform aggregation on this item-location combination. Choose one of the following values: 0—Will Not Be Used in Aggregation 1—Will Use Aggregation

Field Name	Use	Field Purpose
do_fore	User input	<p>Specifies whether to perform forecasting on this item-location combination. Choose one of the following values:</p> <p>0—Do Not Do Forecast.</p> <p>1—Do Forecast (the default).</p> <p>2—Do Zero Forecast. This combination is not used in aggregation.</p>
dying_time	Yes	<p>If no sales occurred during the length of time specified by dying_time, the combination will be marked as dead. If this field is null for a given combination, Demantra uses the dying_time parameter instead.</p>
glob_prop	Read-only	<p>Rolling average demand for this combination, averaged over the recent past, as specified by the length of time given by the hist_glob_prop setting.</p>
hist_glob_prop	User input	<p>Number of base time buckets worth of data to use to calculate the rolling average, glob_prop, for this combination. If this field is null for a given combination, Demantra uses the hist_glob_prop parameter instead.</p>

Field Name	Use	Field Purpose
is_fictive	Read-only	<p>Indicates whether this combination is real or fictive. This field is set automatically by Demantra. It has one of the following values:</p> <p>1 means that the combination was created through Member Management and no data has been loaded for it yet.</p> <p>0 means that there are sales for this combination.</p> <p>2 means that there are no sales for this combination.</p> <p>3 means that an error occurred while loading this combination or while redefining this combination. (When Demantra loads a new combination or changes the definition of a combination, it temporarily sets is_fictive equal to 3. When Demantra finishes the action, it then resets is_fictive equal to 0 or 2.)</p> <p>The engine does not consider the is_fictive setting.</p>
missing_all_sources	Read-only	<p>Used during chaining. Indicates whether the source data for this combination is complete. For each combination, this field has one of the following values:</p> <p>Yes means that there is no data for this combination.</p> <p>Partial means that there is data for this combination only for some of the dates.</p> <p>No means that there is data for this combination for all dates.</p>
missing_some_sources	Read-only	<p>Used during chaining. For each combination, this field indicates one of the following:</p> <p>Yes means that there is no data for one of the items in the combination.</p> <p>Partial means that there is data for this combination only for some of the dates.</p>

Field Name	Use	Field Purpose
models	Read-only	<p>Indicates the engine models that the Analytical Engine used when forecasting this combination, during the most recent engine run. Demantra uses a single letter to indicate each model:</p> <p>A: ARLOGISTIC</p> <p>B: BWINT</p> <p>C: CMREGR</p> <p>D: DMULT</p> <p>E: ELOG</p> <p>F: FCROST</p> <p>G: LOGISTIC</p> <p>H: HOLT</p> <p>K: ICMREGR</p> <p>J: IREGR</p> <p>L: LOG</p> <p>M: MRIDGE</p> <p>N: NAIVE</p> <p>R: REGR</p> <p>T: NAIVE HOLT</p> <p>V: ARIX</p> <p>X: ARX</p> <p>To specify multiple models, Demantra concatenates the letters together. For example, BDF means the BWINT, DMULT, and FCROST models.</p> <p>For information on engine models, see "Theoretical Engine Models".</p>
new_member	Read-only	<p>Specifies whether to run proposit on this combination; used by the Run_full_matrix_proposit parameter.</p>

Field Name	Use	Field Purpose
outlier	Read-only	Indicates whether the Analytical Engine has marked this combination as an outlier, for any time bucket.
P1, P2, P3, P4, P5, P6, P7, P8, P9, P10, P11, P12	Read-only	Monthly proportions for this combination. Each proportion represents the level-adjusted sales for this combination, for each month of the year, as averaged over multiple years
PD1, PD2, PD3, PD4, PD5, PD6, PD7	No	Available only in a daily system. Daily proportions for this combination, for different days of the week.
post_effect	User input	<p>PE only. For each combination, specifies how the Analytical Engine should search for the effects of any given promotion, after the end of that promotion. Specify this as the number of base time buckets after the end of a promotion.</p> <p>Null is treated as zero.</p> <p>Searching for post-promotional effects can slow the engine down, so Oracle recommends doing this only for a few combinations. For those combinations, Oracle recommends specifying a value of 2–4, to avoid possible overlaps between different promotions.</p>
pre_effect	User input	<p>PE only. For each combination, specifies how the Analytical Engine should search for the effects of any given promotion, before the start of that promotion. Specify this as the number of base time buckets before the start of a promotion.</p> <p>Null is treated as zero.</p> <p>Searching for pre-promotional effects can slow the engine down, so Oracle recommends doing this only for a few combinations. For those combinations, Oracle recommends specifying a value of 2–4, to avoid possible overlaps between different promotions.</p>

Field Name	Use	Field Purpose
prediction_status	Read-only	<p>Controls how the Analytical Engine uses this combination. Each combination has one of the following prediction status values:</p> <p>96 (No Forecast) — This status means that the Analytical Engine will completely ignore this combination.</p> <p>97 (Create Zero Forecast) — A user has set do_fore equal to 2 manually. This status means that the Analytical Engine will insert a zero forecast for this combination but otherwise ignore it.</p> <p>98 (Young) — Sales for this combination are too new to be used for prediction.</p> <p>99 (Dead) — Sales for this combination are not recent enough to be used for prediction.</p> <p>1 (Live or Active)—Neither young nor dead.</p> <p>The Analytical Engine ignores any young or dead combinations, except when it is necessary to aggregate. In case of aggregation, Demantra considers the do_aggri, aggri_98, or aggri_99 flag of the combination.</p> <p>Demantra sets the prediction_status indicator as follows.</p> <p>For fictive combinations (is_fictive = 1), Demantra automatically sets the prediction status to 98.</p> <p>For real combinations (is_fictive equal to 0 or 2), Demantra uses the following rules:</p> <p>If do_fore is 0, then prediction_status will be 99.</p> <p>If do_fore is 1, then prediction_status is set as follows:</p> <p>If the combination is dead because of the dying_time parameter, then prediction_status is set to 99.</p> <p>If the combination is young because of the mature_age parameter, then prediction_status is set to 98.</p>

Field Name	Use	Field Purpose
		Otherwise, the prediction_status is set to 1. If do_fore is 2, then prediction_status will be 97. dying_time and mature_age are engine parameters; see "Non-Engine Parameters".
prop_changes	Read-only	Specifies whether to run proposit on this combination; used by the Run_full_matrix_proport parameter.
PW1, PW2, PW3, PW4, PW5, PW6	No	Available only in a weekly or daily system. Weekly proportions for this combination, for different weeks of a month. When calculating these proportions, Demantra factors in the number that this week has.
level_id	Read-only	The strategy of the forecast tree where the forecast for this combination was generated.
item_node	Read-only	Item member in that level.
loc_node	Read-only	Location member in that level.

Promotion_Data

PE only. The following table lists the most important fields in the promotion_data table. The Analytical Engine reads from and writes to some of these fields, which you use mainly to create series that show the forecast results.

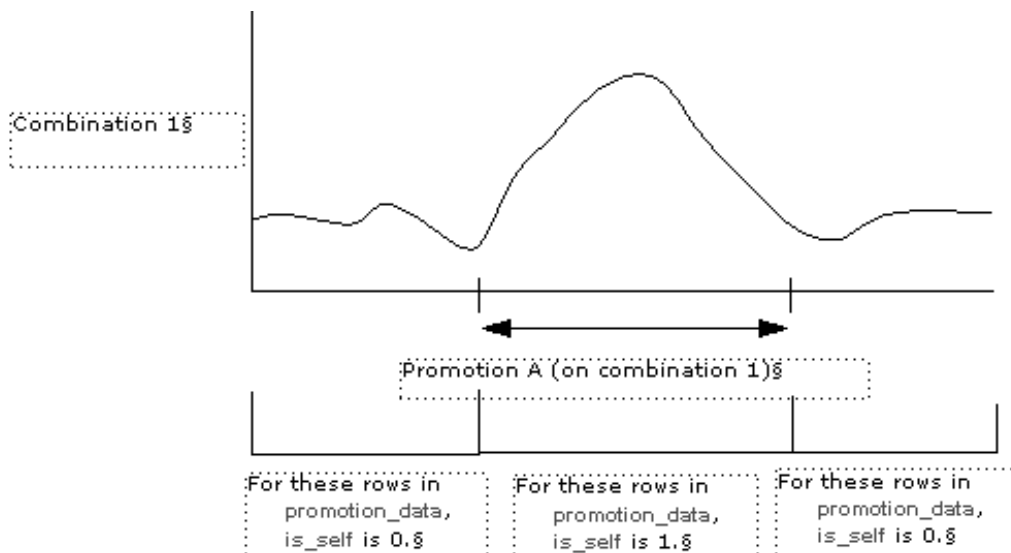
Field Name	Use	Field Purpose
item_id	Read-only	Unique identifier for the item. Together, item_id, location_id, sales_date, and promotion_id form the primary key for rows in the promotion_data table.

Field Name	Use	Field Purpose
location_id	Read-only	Unique identifier for the location.
sales_date	Read-only	Date for this record.
promotion_id	Read-only	Unique identifier for a promotion.
is_self	Read-only	Equals 1 if the lifts in this row (uplift, pre- and post-effect, and switching effects) are associated with the promotion and date of this row. See "Is_Self".
fore_0_uplift	Read-only	Basic lift due to this promotion, during the dates of the promotion.
fore_0_sw_channel	Read-only	Effects of channel switching as described in "Switching Effects".
fore_0_store	Read-only	Effects of store switching.
fore_0_product	Read-only	Effects of store switching.
fore_0_brand	Read-only	Effects of brand or category switching.
fore_0_pre_effect	Read-only	Pre-promotional effect due to this promotion.
fore_0_post_effect	Read-only	Post-promotional effect due to this promotion.

Field Name	Use	Field Purpose
norm	Read-only	<p>Normalized versions of the forecast data, if requested via the NormalizeResults parameter. When the Analytical Engine normalizes its results, it re-scales the historical engine results so that the observed baseline values are preserved.</p> <p>NormalizeResults is an engine parameter; see "Engine Parameters".</p>

Is_Self

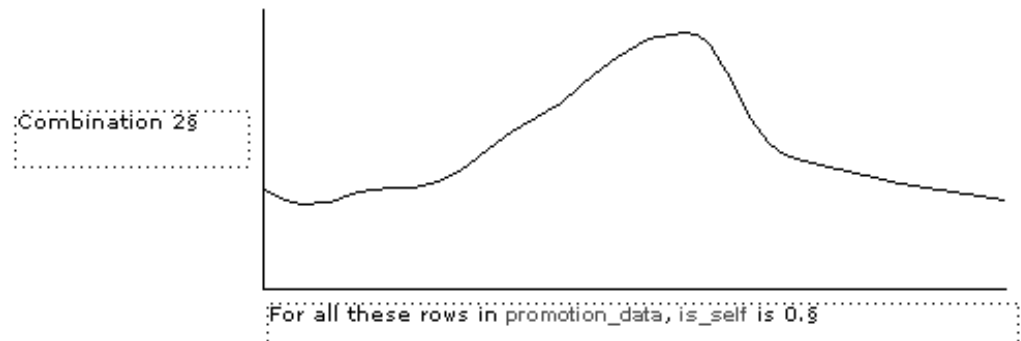
In any given row of the promotion_data, the Analytical Engine uses the is_self field to indicate whether the lifts in that row are associated with the promotion and date of that row. (Specifically, this refers to the uplift, pre and post-effect, and switching effects.) Consider the following example, with a combination that has a promotion (Promotion A) on it for some dates. For simplicity, this graph shows just the uplift on this combination due to the promotion. Each time point in this graph corresponds to a row in promotion_data. The notes at the bottom of the figure show the value of is_self for different times.



As you can see, during the dates of the promotion itself, is_self is 1. Outside those dates, is_self is 0 because these dates fall outside the promotion.

Now consider another combination and the same period of time. The sales for this other

combination were lifted, even though the promotion was not applied to this combination. The following graph shows the uplift on this combination, due to Promotion A (which ran only on the other combination). Again, each time point corresponds to a row in promotion_data:



For all these rows in promotion_data, is_self is 0 because these lifts are due to a promotion that did not run on this combination.

Server Expression Functions and Operators

This appendix provides reference information for the operators and functions that are allowed in server expressions.

This chapter covers the following topics:

- Supported SQL Functions
- Operators in Server Expressions
- Oracle Tokens

Supported SQL Functions

You can use the following SQL functions in a Oracle server expression:

Function	Aggregating?	Description	Supported on	
			Yes	Yes
<i>Avg (column)</i>	Yes*	Returns the average of the values of a group.	Yes	Yes
<i>Count (column)</i>	Yes***	Returns the number of members of a group.	Yes	Yes

Function	Aggregating?	Description	Supported on	
Decode (<i>expression</i> , <i>value1</i> [, <i>return1</i> , <i>value2</i> , <i>return2</i> , ...], <i>defaultreturn</i>)	No	Matches <i>expression</i> to the test cases <i>value1</i> , <i>value2</i> , and so on, and returns the return value (<i>return1</i> , <i>return2</i> , and so on) that corresponds to the matched value.	No	Yes
Lower (<i>column</i>)	No	Returns a string in lower case.	Yes	Yes
Ltrim (<i>column</i>)	No	Removes characters from the left side of the string.	Yes	Yes
Max (<i>column</i>)	Yes*	Returns the maximum of the values of a group.	Yes	Yes
Min (<i>column</i>)	Yes*	Returns the minimum of the values of a group.	Yes	Yes
Nvl (<i>expression1</i> , <i>expression2</i>)	No	If <i>expression1</i> is not null, returns <i>expression1</i> . Otherwise, returns <i>expression2</i> .	No	Yes
Round (<i>number</i> [, <i>m</i>])	No	Rounds the given number to the specified number <i>m</i> of decimal places (zero by default).	Yes	Yes

Function	Aggregating?	Description	Supported on	
Rtrim (<i>column</i>)	No	Removes characters from the right side of the string.	Yes	Yes
Safe_Division (<i>argument1</i> , <i>argument2</i> , <i>argument3</i>)	No	Custom function created by Oracle. This function returns <i>argument1</i> divided by <i>argument2</i> , unless <i>argument2</i> is null. If <i>argument2</i> is null, then the function returns <i>argument3</i> .	Yes	Yes
SubStr (<i>expression</i> , <i>start</i> , <i>length</i>)	No	Returns the substring of a given length that starts at the given position.	No	Yes
Sum (<i>column</i>)	Yes**	Returns the sum of the values of a group.	Yes	Yes
Sysdate()	No	Returns the current date and time.	No	Yes
To_char (<i>date</i> , [<i>format</i>]) or To_char (<i>number</i> , [<i>format</i>])	No	Returns the input, a date or number, converted to a string using the given format.	No	Yes
To_date (<i>date</i> , [<i>format</i>])	No	Returns a formatted date.	No	Yes

Function	Aggregating?	Description	Supported on	
To_number (<i>date</i> , [<i>format</i>])	No	Returns a number.	No	Yes
Upper (<i>column</i>)	No	Returns a string in upper case.	Yes	Yes

*If you use this function as the aggregating function for a series, the series should be non-proportional. **If you use this function as the aggregating function for a series, the series should be proportional. ***If you use this function as the aggregating function for a series, the series should be non-editable.

Note: A server expression must be an aggregating expression that returns numeric, date, string, or true/false values.

Note: If a series is going to be used within cached worksheets, its server expression cannot return null or zero-length values. Use the expression to_number(null,0) to express null values.

Note: In these reference sections, square brackets indicate optional parts of the syntax.

For details on these functions, consult the appropriate Oracle database documentation.

Operators in Server Expressions

You can use the following operators in a Oracle server expression:

- +
- -
- *
- /
- ()
- <
- <=

- \diamond
- =
- >
- \geq
- And
- Else
- In
- Not
- Or
- Then
- When

Calculations follow standard algebraic rules of precedence.

Oracle Tokens

You can use the following special-purpose tokens in a Oracle server expression:

Token	Allowed in	Automatically replaced by
#CONFIDENCE_LEVEL#	Server expressions	Confidence level associated with the forecast. Not supported in the Web user interfaces.
#FDATE@<Version>@<Profile_id>#	Series hint messages	Date of the specified engine profile's forecast version*. For example, #FDATE@0@25# is replaced by the date on which current forecast for engine profile 25 was generated.

Token	Allowed in	Automatically replaced by
#FORE@<Version>@<Profile_id>#	Server expressions	The specified forecast version*. For example: #FORE@1@25# is replaced by the second most current forecast version generated using engine profile 25.
#POST_EFFECT@<Version>@<Profile_id>#	PE server expressions	The post-promotional effect associated with the specified forecast version generated using a specified engine profile. *
#PRE_EFFECT@<Version>@<Profile_id>#	PE server expressions	The pre-promotional effect associated with the specified forecast version generated using a specified engine profile. *
#SIMULATION_TABLE#	Server expressions	.
#SW_BRAND@<Version>@<Profile_id>#	PE server expressions	The brand switching associated with the specified forecast version generated using a specified engine profile. *
#SW_CHANNEL@<Version>@<Profile_id>#	PE server expressions	The channel switching associated with the specified forecast version generated using a specified engine profile *
#SW_PRODUCT@<Version>@<Profile_id>#	PE server expressions	The product switching associated with the specified forecast version. *
#SW_STORE@<Version>@<Profile_id>#	PE server expressions	The store switching associated with the specified forecast version generated using a specified engine profile. *.

Token	Allowed in	Automatically replaced by
#UNIT#	Server expressions	The unit conversion factor that corresponds to the unit used in the worksheet. See "Configuring Units, Indexes, and Update-Lock Expressions".
#UPLIFT@<Version>@<Profile_id>#	PE server expressions	The uplift associated with the specified forecast version generated using a specified engine profile. *
* The most recent forecast is 0, the previous forecast is 1, and so on.		

Client Expression Functions and Operators

This appendix provides reference information for the operators and functions that are allowed in client expressions.

This chapter covers the following topics:

- About This Reference
- Operators in Client Expressions
- Abs Function
- Case
- CurrentRow
- Date
- Day
- Exp
- ForecastFirstRow
- Fpos
- Fsum
- GetRow
- GetWorksheetLevelNumber
- If
- Is_Modified
- IsNull
- Mod
- Month
- Pi
- Rand

- Round
- RowCount
- Sqrt
- SummaryAVG
- SummaryCount
- SummaryMax
- SummaryMin
- SummarySum
- SummaryWAVG
- Today
- Truncate
- Year
- Z_Val

About This Reference

This appendix provides reference information for the operators and functions that are allowed in client expressions.

- Edit-lock expressions must evaluate to true or false values.
- In these reference sections, square brackets indicate optional parts of the syntax.

Operators in Client Expressions

You can use the following operators in a client expression:

- +
- -
- *
- /
- ()
- []
- <

- <=
- <>
- =
- >
- >=
- And
- Else
- In
- Not
- Or
- Then
- When

Precedence of calculations follows standard algebraic rules.

Finally, to specify a series, use either of the following syntaxes:

- `series_name`
- `series_name[relative-time-bucket]`

For example: Sales [-1] refers to the previous period(*column*). Sales [1] refers to the next period (*column*). [0] is not allowed.

Abs Function

Returns the absolute value of a number.

Syntax

- `Abs (argument)`

The value of *argument* must be either numeric or null.

- If *argument* is numeric, the function returns the absolute value of *argument*.
- If *argument* is null, the function returns null.

Case

Tests the values of a series or expression and returns values based on the results of the test. If more than one WHEN clause matches the given input, the function returns the result corresponding to the first matching one.

Syntax

- `Case (test WHEN value1 THEN result1 [WHEN value2 THEN result2] [additional WHEN-THEN clauses] [ELSE else])`

The square brackets indicate optional parts of the syntax. The arguments are as follows:

- *test* is the series or expression whose values you want to test. You can use a column name or a column number preceded by a pound sign (#).
- *value1*, *value2*, and so on are possible values that *test* can have. Each value can be any of the following:
 - A single value
 - A list of values separated by commas (for example, 2, 4, 6, 8)
 - IS followed by a relational operator and comparison value (for example, IS>5)
 - Any combination of the preceding expressions, separated by commas (for example, 1,3,5,7,9, IS>42)

In this case, the function implicitly behaves as if the expressions were combined by a logical OR.

- *result1*, *result2*, and so on are the results to return for the possible values. For example, if *test* evaluates to *value1*, then the function returns *result1*. All returned values must have the same data type.
- *else* specifies the value to return if *test* does not equal any of the given cases (*value1*, *value2*, and so on). The default for *else* is null.

Examples

The following expressions are valid uses of Case:

- `Case (Input1 When is > 1 Then 10 When 2 Then 20 Else 30) Case (Input1 When is < 10 Then 5 When is < 20 Then 50) Case (Input1 When is > 100, is < 0 Then 5 Else 30) Case (Input1 When is > 1 Then 10)`

See also

- "If"

CurrentRow

Returns the number of the worksheet row that currently has the focus, that is, the worksheet row that the user has selected.

This function is easiest to use in color expressions.

Within the main client expression for a series, you can use this function to find the relative position of a row, in relation to another row. For example, you can use it indirectly to find the last row number.

Note: Remember that apart from the color expression and the edit-lock expression, the worksheet does not reevaluate client expressions until a value changes in the worksheet. That is, the action of moving the cursor does not force the main client expressions to be reevaluated.

Note: This means that if you use this function within the main client expression for a series, the expression should also refer to a series that the user will change.

If the function fails, it returns 0.

Syntax

- CurrentRow ()

Notes

When used in a general client expression (that is, neither a color expression nor an edit-lock expression), this function always returns 1.

When used in a color or edit-lock expression, this function returns the number that corresponds to the row that currently has the focus, that is, the row that the user has currently highlighted.

You can use this function in an edit-lock expression, but the GetRow function is a better choice. For example, consider the following possible edit-lock expressions:

- CurrentRow() < ForecastFirstRow() GetRow() < ForecastFirstRow()

Both expressions make it impossible to edit the series for the rows before the start of the forecast. However, if you use the former expression, all the cells will appear editable if the user selects a row before the start of the forecast, which would be very confusing.

See also

- "ForecastFirstRow" "GetRow"

Date

Given a string argument, returns a date. This function works only in the desktop.

Syntax

- `Date (string)`

See also

- `"Day" "Month" "Today" "Year"`

Day

Returns an integer indicating the day of the month of the given date value.

Syntax

- `Day (argument)`

The value of *argument* must be either a date or null.

- If *argument* is a date, the function returns an integer representing the day (1–31) of the month in that date.
- If *argument* is null, the function returns null.

Exp

Returns the number e raised to the specified power.

Syntax

- `Exp (argument)`

The value of *argument* must be either numeric or null.

- If *argument* is numeric, the function returns the number e raised to the power of *argument*.
- If *argument* is null, the function returns null.

Example

This expression returns 7.38905609893065:

- `Exp(2)`

These statements convert a natural logarithm (base e) back to a regular number. When executed, Exp sets value to 200:

- double value, $x = \log(200)$
value = Exp(x)

ForecastFirstRow

Returns the number of the row in the current worksheet where the forecast begins. It refers to the batch forecast.

Syntax

- ForecastFirstRow()

See also

- "CurrentRow" "GetRow"

Fpos

Periods of supply. Checks the period of the first arguments against the period (from the next period) of the second argument. This function works only on the forecast data.

Note: You cannot use this function in color expressions.

Syntax

- Fpos (series1,series2)

Each argument should be a series.

Examples

- If (GetRow() >= ForecastFirstRow()) Fpos(SupplyTinv, SupplyTinvFinalFcst) Null)

The section that uses "GetRow() >= ForecastFirstRow()" is essential because the expression works only in forecast.

Date	Inventory (SupplyTinv)	Forecast (SupplyTinvFinalFcst)	Third Series with FPOS
January	12	5	2.5
February		5	
March		4	

Date	Inventory (SupplyTinv)	Forecast (SupplyTinvFinalFcs t)	Third Series with FPOS
April		6	

$F_{pos} = 5 + 4 + 6 \times 0.5 = 2.5$. The inventory (12) will cover the forecast for 2.5 periods.

Fsum

Returns a series that adds multiple future consecutive items from a given series. The second argument, either a number or another series, specifies the number of time periods to use for each sum.

Note: You cannot use this function in color expressions.

Syntax

- `Fsum (series , count)`

The arguments are as follows:

- *Series* should be an actual series name.
- *Count* should be either a numeric series or an actual integer between 1 and 15, inclusively. If count is more than 15, the function returns null.

The parameters can be only the names of data series or actual values. It is not permitted to enter other functions or [] brackets inside the FSUM function.

Examples

- `Fsum (Series1, Series2)`

Date	Series1	Series2	=Fsum	Note about this entry
Jan	100	3	360	Series2 for Jan is 3, so the function finds the next 3 time periods within Series 1; that is, 110, 130, and 120. The sum of those numbers is 360.
Feb	110	2	250	Series2 for Feb is 2, so the function finds the next 2 time periods within Series 1; that is, 130 and 120. The sum of those numbers is 250.
March	130	2	290	Series2 for March is 2, so the function finds the next 2 time periods within Series 1; that is, 120 and 170. The sum of those numbers is 290.
April	120	2	...	
May	170	3	...	
April	...			

GetRow

Returns the number of the worksheet row. You generally use this function in edit-lock

expressions that lock worksheet rows depending on position.

If the function fails, it returns 0.

Syntax

- `GetRow ()`

See also

- "CurrentRow" "ForecastFirstRow"

GetWorksheetLevelNumber

Returns an integer that indicates the relative level of the summary row where this function is used.

Note: This function is available only for use within the summary row of a series.

Syntax

- `GetWorksheetLevelNumber ()`

You use this function to achieve different kinds of summaries for a series in different contexts in a given worksheet. Specifically, when a worksheet uses one or more levels on the x-axis, the worksheet table includes intermediate summary rows, for example:

09/08/2003	Stop and Shop Store 001:	Rainbow LF Chocolate Chip	\$5,548,561	
		Rainbow LF Oatmeal Raisin	\$2,165,763	
		Rainbow LF Peanut Butter	\$12,421	
		Rainbow Reg Chocolate Chip	\$1,199,128	
		Rainbow Reg Peanut Butter	\$1,909,086	
		Rainbow Spc Biscuits	\$6,487,385	
		Summary	\$17,342,364	
12/08/2003	Stop and Shop Store 001:	Rainbow LF Chocolate Chip	\$5,316,576	
		Rainbow LF Oatmeal Raisin	\$2,167,184	
		Rainbow LF Peanut Butter	\$51,299	
		Rainbow Reg Chocolate Chip	\$1,439,943	
		Rainbow Reg Peanut Butter	\$1,976,406	
		Rainbow Spc Biscuits	\$7,845,928	
Summary		Summary	\$18,797,336	
			\$77,725,936	

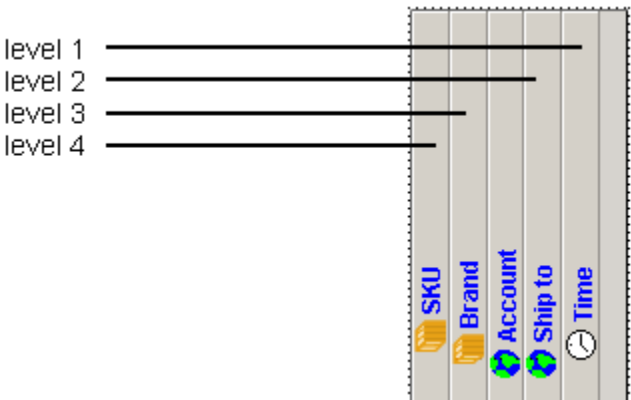
intermediate summaries

final summary

By default, the final summary row and all the intermediate summary rows are calculated in exactly the same way. The `GetWorksheetLevelNumber` function provides a way to distinguish each summary row, so that you can create a different summary functions as needed, at each of those levels.

To determine the relative aggregation level of the summary rows, the function

considers the layout of the worksheet. The following figure relates the summary levels to the x-axis layout:



For level 1 in the worksheet, GetWorksheetLevelNumber returns 1, and so on. If there are n levels on the x-axis of the worksheet, the function returns n+1 for the final summary.

Example

Suppose that you want to display only the intermediate summary but not the final summary, as follows:

Account	Time	Sample Max
Stop and Shop	02/17/2003	100
	08/18/2003	200
	02/16/2004	150
	Summary	200
WalMart	02/17/2003	125
	08/18/2003	220
	02/16/2004	170
	Summary	220
Summary		

To achieve this, you could use a summary row expression of the following form:

- If (GetWorksheetLevelNumber() =1, SummaryMax (Sample Max) , Null Value)

Note: If you configure a series with a context-sensitive summary row like this, be careful to use that series only within worksheets that have the appropriate layout. You may want to add a usage note to the series hint message to guide users.

If

Tests a true/false expression and returns one of two possible values based on the results of the test.

Syntax

- `IF (test , trueresult) IF (test , trueresult, falseresult)`
- The arguments are as follows:
- *test* is the series or expression that has true or false values.
- *trueresult* is the result to return if *test* equals true.
- *falseresult* is the result to return if *test* equals false. The default for *falseresult* is null. That is, if you do not specify this argument, the function returns null.

Note: Within a color expression, only the first syntax variant is allowed. That is, a color expression cannot include *falseresult*.

Examples

This expression returns 7 if Retail_history is greater than Retail_model; otherwise it returns Demand:

- `If (Retail_History > Retail_Model, 7, Demand)`

See also

- "Case"

Is_Modified

Returns true or false depending on whether a given series has been edited since the last time data was saved.

Syntax

- `Is_Modified (series)`

Here *series* should be an editable series. The function does not detect whether a calculated series has been changed by having its inputs edited.

Examples

The following expression returns the Inventory series if Pseudo has been modified and returns the Safety series if Pseudo has not been modified:

- If (Is_Modified(Pseudo), Inventory, Safety)

IsNull

Returns true or false depending on whether a given series equals null.

Syntax

- IsNull(*series*)

Mod

Returns the remainder (modulus) of a division operation.

Note: The results of this function take slightly longer to display than for other functions.

Syntax

- Mod (*argument1*, *argument2*)

The value of *argument1* and *argument2* must be either numeric or null.

- If both *argument1* and *argument2* are numeric, the function returns the remainder (modulus) of a division operation. Specifically, it returns the following result:

argument2 - round(*argument2*/*argument1*) where round(*argument2*/*argument1*) equals *argument2*/*argument1* rounded to the nearest integer.

The returned value is the data type of whichever argument has the more precise data type.

- If *argument2* or *argument2* is null, the function returns null.
- If *argument2* is 0, the function returns null.

Examples

- Mod(20, 6) returns 2
Mod(25.5, 4) returns 1.5
Mod(25, 4.5) returns 2.5

Month

Returns an integer indicating the month of the given date value.

This can be used to show or use dates in calculations. Also, a planning process can be

maintained, while the locking expression is based dynamically on dates. For example, override is allowed only in the first two weeks of the month.

Syntax

- Month (*argument*)

The value of *argument* must be either a date or null.

- If *argument* is a date, the function returns an integer (1 to 12) representing the month in that date.
- If *argument* is null, the function returns null.

Pi

Returns the number pi multiplied by a specified number.

Syntax

- Pi (*argument*)

The value of *argument* must be either numeric or null.

- If *argument* is numeric, the function returns the number pi multiplied by *argument*.
- If *argument* is null, the function returns null.

The function returns -1 if an error occurs.

Examples

You can use this function to convert angles to and from radians. For example, because pi equals 180 degrees, you can convert 60 degrees to radians as follows:

- $60 * 180 / \text{pi}(1)$

Rand

Returns a random integer between 1 and a specified upper limit.

Note: This function does not generate true random numbers. If you repeatedly call this function, you will receive a pseudo random sequence.

Syntax

- Rand (*argument*)
- The value of *argument* must be either numeric or null. If numeric, the argument

must have a value between 1 and 32767, inclusive.

- If *argument* is a number greater than or equal to 1, the function returns a random integer between 1 and *argument*, inclusive.
- If *argument* is null, the function returns null.

Examples

The following expression returns a random whole number between 1 and 10:

- Rand(10)

Round

Returns a number rounded to a specified number of decimal places.

Syntax

- Round (*argument1*, *argument2*)
- The value of *argument1* and *argument2* must be either numeric or null. If numeric, *argument2* should be a non negative integer between 0 to 18, inclusive.
- If *argument1* is numeric and *argument2* is a non negative integer, the function returns the value of *argument1*, rounded to the number of decimal places specified by *argument2*.
- If *argument1* or *argument2* is null, the function returns null.
- If the function fails, it returns null.

Examples

The following expression returns 9.62:

- Round(9.624, 2)

The following expression returns 9.63:

- Round(9.625, 2)

The following expression returns 9.600:

- Round(9.6, 3)

The following expression returns -9.63:

- Round(-9.625, 2)

see also

- "Truncate"

RowCount

Returns the numbers of rows in the worksheet where this function is used.

Syntax

- RowCount ()

See also

- "ForecastFirstRow" "GetRow"

Sqrt

Returns the square root of a non negative number.

Syntax

- Sqrt (*argument*)

The value of *argument* must be either a non negative number or null.

- If *argument* is a non negative number, the function returns the square root of *argument*.
- If *argument* is null, the function returns null.

Examples

This expression returns 1.414213562373095.

- Sqrt(2)

This expression results in an error at execution time.

- Sqrt(-2)

SummaryAVG

Returns the average value of the displayed rows of the specified series.

Note: This function is available only for the summary row of a series.

Syntax

- SummaryAvg(*argument*)

Here *argument* is the name of a series that has a numeric value.
Any null value is treated as zero.

SummaryCount

Returns the total count of the displayed rows of the specified series.

Note: This function is available only for the summary row of a series.

Syntax

- SummaryCount(*argument*)

Here *argument* is the name of a series that has a numeric value.

SummaryMax

Returns the maximum value of the displayed rows of the specified series.

Note: This function is available only for the summary row of a series.

Syntax

- SummaryMax(*argument*)

Here *argument* is the name of a series that has a numeric value.

Any null value is treated as zero.

SummaryMin

Returns the minimum value of the displayed rows of the specified series.

Note: This function is available only for the summary row of a series.

Syntax

- SummaryMin(*argument*)

Here *argument* is the name of a series that has a numeric value.

Any null value is treated as zero.

SummarySum

Returns the sum of the displayed rows of the specified series.

Note: This function is available only for the summary row of a series.

Syntax

- SummarySum(*argument*)

Here *argument* is the name of a series that has a numeric value.

Any null value is treated as zero.

SummaryWAVG

Returns the weighted average of the displayed rows.

Note: This function is available only for the summary row of a series.

Syntax

- SummaryWavg(*argument1* , *argument2*)

Here *argument1* and *argument2* are the name of series that have a numeric value. This function performs a weighted average of the values in *argument1*, using the values in *argument2* as the weights.

Any null value is treated as zero.

Today

Returns the current system date.

This can be used to show or use dates in calculations. Also a planning process can be maintained, while the locking expression is based dynamically on dates. For example, override is allowed only in the first two weeks of the month, and so on.

Syntax

- Today ()

Truncate

Returns a number truncated to a specified number of decimal places.

Syntax

- Truncate (*argument1*, *argument2*)

The value of *argument1* and *argument2* must be either numeric or null. If numeric, *argument2* should be a non negative integer between 0 to 18, inclusive.

- If *argument1* is numeric and *argument2* is a non negative integer, the function returns the value of *argument1*, truncated to the number of decimal places specified by *argument2*.
- If *argument2* or *argument2* is null, the function returns null.
- If the function fails, it returns null.

Examples

The following expression returns 9.2:

- Truncate(9.22, 1)

The following expression returns -9.2:

- Truncate(-9.29, 1)

See also

- "Round"

Year

Returns a four-digit integer indicating the year of the given date value.

This can be used to show or use dates in calculations. Also a planning process can be maintained, while the locking expression is based dynamically on dates. For example, override is allowed only in the first two weeks of the month, and so on.

Syntax

- Year (*argument*)

The value of *argument* must be either a date or null.

- If *argument* is a date that includes a four-digit year, the function returns a four-digit integer representing the year.
- If *argument* is a date that includes a two-digit year, the function returns a four-digit integer representing the year, as follows:
 - If the two-digit year is between 00 to 49, Demantra assumes 20 as the first two digits.
 - If the two-digit year is between 50 and 99, Demantra assumes 19.
- If *argument* is null, the function returns null.
- If an error occurs, the function returns 1900.

Demantra handles years from 1000 to 3000 inclusive. If your data includes date before 1950, such as birth dates, always specify a four-digit year so that Year and other functions, such as Sort, interpret the date as intended.

Z_Val

Z_val is used in safety stock calculation. Given a specified service level, this function returns a value to use in the safety stock calculation, by looking up values in a table.

Syntax

- Z_Val (*argument*)

The value of *argument* must be either numeric or null.

- If *argument* is less than or equal to 0, the function returns null.
- If *argument* is greater than 0 but less than the largest max_value, the function returns the z_val from the following table.
- If *argument* is greater than the largest max_value, the function returns null.
- If *argument* is null, the function returns null.

Z_Val Table

If the argument is...		The function returns this z_val...
greater than the min_level	and less than or equal to the max_level	
0	0.85	1.04
0.85	0.86	1.09
0.86	0.87	1.13
0.87	0.88	1.18
0.88	0.89	1.23
0.89	0.9	1.282
0.9	0.91	1.34

If the argument is...		The function returns this z_val...
greater than the min_level	and less than or equal to the max_level	
0.91	0.92	1.41
0.92	0.93	1.48
0.93	0.94	1.56
0.94	0.95	1.645
0.95	0.96	1.75
0.96	0.97	1.88
0.97	0.98	2.06
0.98	0.99	2.33
0.99	0.995	2.576
0.995	0.999	3.09
0.999	0.9995	3.291
0.9995	0.99995	3.891
0.99995	0.999995	4.417

Example

If we select a service level of 95.5% then the expression will look at the table at the following line, because this lies between 0.95 and 0.96 (the minimum and maximum values in the range). A z_val value of 1.75 is returned.

min_level	max_level	z_val
0.95	0.96	1.75

Workflow Steps

This chapter provides reference information for the available workflow steps.

This chapter covers the following topics:

- Specifying a Task
- BLE Step
- Condition Step
- Container Step
- Create Member Step
- Custom Step
- Delete Member Step
- Edit Member Step
- Email Step
- Exception Step
- Executable Step
- Group Step
- Paste Member Step
- Refresh Population Step
- Selection Step
- Simulation Step
- Stored Procedure Step
- Transfer Step
- Update Data Step
- User Step
- Wait Until Step

- Worksheet Cache Step
- Worksheet Step

Specifying a Task

Several of the step types include a task or list of tasks, for use in Collaborator Workbench. A task consists of a set of properties, described here.

Each user who logs onto Collaborator Workbench sees an individualized list of tasks in a module called My Tasks on the Collaborator Workbench page.

Each task in the My Tasks module has a subject line which can be a link to a URL, and a description that explains more about the purpose of the task. The subject line link can refer to a worksheet, a file, a URL that initiates an external application, or any other URL.

Task Properties (in a Workflow Step)

Property	Description
Message	Message to include in My Tasks. It is also used in the email message, if you use that option.
URL	<p>Optionally specify a URL, including the prefix <code>http://</code>. For example,</p> <p><code>http://www.acme.com/page.html</code></p> <p>If you do not include the prefix <code>http://</code> then the URL is read relative to the local host's Workflow Engine root directory. For example, <code>buyer/start.html</code> is read as:</p> <p><code>http://localhost/demantra/Portal/buyer/start.html</code></p> <p>You must specify either a URL or a worksheet to open.</p>
Worksheet to open	<p>Optionally select a worksheet, from the list of public worksheets defined in Demantra. This worksheet is listed in My Tasks. If you use the email option, the subject line of the message includes a link to this worksheet.</p> <p>You must specify either a URL or a worksheet to open.</p>

Property	Description
Source name	Optionally specify the originator of this task.
Description	Optionally specify a longer description of this task, up to 255 characters. This text will show in the Description field in My Tasks. It is also used in the email message, if you use that option.
File	Optionally specify a local file to send to the task recipient. Specify a full path and filename that is accessible to the Workflow Manager, on the machine that is running that software.
Send as email as well	Select this check box if the system should also send an email message containing this task. Make sure that each user has an email address. You use the Business Modeler to configure email addresses for the users. See "Creating or Modifying a User".

See also

- Oracle Demantra Demand Management User's Guide

BLE Step

This kind of step submits a worksheet to the Business Logic Engine queue. The worksheet is run when its turn is reached. The Business Logic Engine then evaluates all the client expressions, splits the resulting data to the lowest level, and saves it to the database.

Note: The BLE Step starts the Business Logic Engine if necessary.

End Conditions

This step is completed when the Business Logic Engine finishes processing the specified worksheet.

Properties

Properties	Description	Default
Step ID	Unique identifier for the step.	
Worksheet Name	Select the worksheet to evaluate. The list of worksheets includes all public worksheets and all worksheets that you own.	
Comments	Optional comments.	
Recovery	<p>Specify what the Workflow Engine should do if the system crashes while performing this step:</p> <p>Ask—the engine follows a Fail-To-Execute procedure for the step.</p> <p>Retry—the engine executes the step again.</p> <p>Continue—the engine continues with the next step.</p> <p>Abort—the engine terminates the workflow instance.</p>	Ask
Check Finish After	Specify how long to wait after starting the step and before first checking to see if the end conditions have been met.	0 seconds
Check Finish Every	The Workflow Engine checks periodically to see whether or not the end conditions have been met. This property specifies how long to wait between two successive checks.	60 seconds
Pause	Specify how long to wait after a step has been completed, before starting the next step.	0 seconds

Properties	Description	Default
Timeout>Timer	Specify a timeout for the step.	never
Timeout>Alert Time	Specify the alert phase that can occur just before the step times out. During the alert phase, the task due date is displayed in red in the user's My Task module.	never

Notes

If a step ends before the Check Finish After period, or during a Check Finish Every period, then the Workflow Engine still waits for that counter to finish before checking if the step has finished.

In timers, a month is measured as a calendar month.

If this step times out, the worksheet request is not deleted from the Business Logic Engine queue. You must manually delete the worksheet request from the Business Logic Engine queue.

Condition Step

This kind of step tests a worksheet, an SQL statement, or a Java class, and proceeds to either the True step or the False step, based on the result of that test.

The Workflow Engine continues with the True step in any of the following cases:

1. If the worksheet contains data
2. If the SQL statement returns data
3. If the Java class returns the True value

The Workflow Engine continues with the False step in any of the following cases:

- If the worksheet is empty
- If the SQL statement returns no data
- If the Java class returns the False value

End Conditions

This step is completed when the test has successfully been performed.

Properties

Properties	Description	Default
Step ID	Unique identifier for the step.	
Pause	Specify how long to wait after a step has been completed, before starting the next step.	0 seconds
Worksheet Name	Select the worksheet to test. The list of worksheets includes all public worksheets and all worksheets that you own.	
SQL	Specify an SQL statement.	
Class name	Specify a custom Java class that returns either True or False.	
Recovery	Specify what the Workflow Engine should do if the system crashes while performing this step: Ask—the engine follows a Fail-To-Execute procedure for the step. Retry—the engine executes the step again. Abort—the engine terminates the workflow instance.	Ask

Properties	Description	Default
Population	<p>Specifies the population attribute that further filters the worksheet used by this step. Specify the following:</p> <p>Name—specify the name of the population attribute, as given in GROUP_ATTRIBUTES_POPULATION.ATTRIBUTE_LABEL. In the demo, this attribute is named Population.</p> <p>The Value field is not used.</p> <p>This property is useful when you use a workflow as a method; otherwise it has no effect.</p>	

Container Step

This kind of step is used to execute multiple single steps simultaneously and independently. The Container step is completed when all steps in it are completed.

Note: In order to run a sequence of steps from within a Container step, you can use an Executable step that initiates a workflow instance that itself contains the required series of steps. However, you should remember that new workflow instance is run separately and does not affect the time-out, fail-to-execute, or end conditions of the original Container step.

Included Steps

A Container step can contain any number of steps. All the steps proceed independently of each other. The steps do not have to be for the same user.

You cannot include the following kinds of steps:

- Condition Step
- Container Step
- Exception Step

Also, the order in which the steps are included is not relevant to their processing.

End Conditions

This step is completed when all the steps that it contains are completed.

Properties

Properties	Description	Default
Step ID	Unique identifier for the step.	
Check Finish After	Specify how long to wait after starting the step and before first checking to see if the end conditions have been met.	0 seconds
Check Finish Every	The Workflow Engine checks periodically to see whether or not the end conditions have been met. This property specifies how long to wait between two successive checks.	60 seconds
Pause	Specify how long to wait after a step has been completed, before starting the next step.	0 seconds
Recovery	Specify what the Workflow Engine should do if the system crashes while performing this step: Ask — the engine follows a Fail-To-Execute procedure for the step. Retry — the engine executes the step again. Abort — the engine terminates the workflow instance.	Ask

Fail-To-Execute

If any of the steps in a container step fail to execute, the engine waits until all other steps have either finished before performing a Fail-To-Execute procedure on the entire container step.

An email notification is sent to the process initiator and shows which specific steps failed to execute. For example:

- **Subject:** Failure in Process execution. **Message Text:** Process ID: 19 Schema ID: 7 Step ID: ContainerStep Error description: Invalid user name: Brian, internal step in container step: 'Action Required Activities'. Schema name: 'Check for Action Required', step name: 'Notify user of Action Required'

See also "Fail-To-Execute Step".

Timeout

If an individual step included within a Container step times out, it does not continue to its own time-out step. The time-out procedure is executed but the time-out step is not activated.

Notes

If a step ends before the Check Finish After period, or during a Check Finish Every period, then the Workflow Engine still waits for that counter to finish before checking if the step has finished.

In timers, a month is measured as a calendar month.

Create Member Step

This kind of step should be used only as a method. It uses the passed arguments, and creates the specified level member.

End Conditions

This step is completed when the member has been created.

Properties

Properties	Description	Default
Step ID	Unique identifier for the step.	

Properties	Description	Default
Recovery	<p>Specify what the Workflow Engine should do if the system crashes while performing this step:</p> <p>Ask — the engine follows a Fail-To-Execute procedure for the step.</p> <p>Retry — the engine executes the step again.</p> <p>Continue — the engine continues with the next step.</p> <p>Abort — the engine terminates the workflow instance.</p>	Ask
Pause	Specify how long to wait after a step has been completed, before starting the next step.	0 seconds

See also

- "Delete Member Step" "Edit Member Step"

Custom Step

This kind of step executes a Java class from within a workflow instance. You can use this kind of step to add functionality to your workflow or interact with external applications without having to change the workflow structure itself. For example, you can use Custom Step to update member attributes as part of a workflow process. Incorporate this step in workflow processes to which you want to pass member attribute values. Add attribute names and values as properties of the step.

If a user has launched the workflow from within a worksheet, Demantra automatically passes arguments to the workflow, which Custom Step can use.

See *Parameters Used as Arguments for a Workflow*, page 31-4.

End Conditions

This step is completed when the executed Java class is completed.

Properties

Properties	Description	Default
Step ID	Unique identifier for the step.	
Class Name	The Java class to execute.	
Parameters	Any input parameters that are needed by the Java class. For each parameter, specify the parameter name and value, as well as an optional description.	
Recovery	<p>Specify what the Workflow Engine should do if the system crashes while performing this step:</p> <p>Ask—the engine follows a Fail-To-Execute procedure for the step.</p> <p>Retry—the engine executes the step again.</p> <p>Continue—the engine continues with the next step.</p> <p>Abort—the engine terminates the workflow instance.</p>	Ask
Check Finish After	Specify how long to wait after starting the step and before first checking to see if the end conditions have been met.	0 seconds
Check Finish Every	The Workflow Engine checks periodically to see whether or not the end conditions have been met. This property specifies how long to wait between two successive checks.	60 seconds
Pause	Specify how long to wait after a step has been completed, before starting the next step.	0 seconds

Java Class Functions

The Java class should contain these two functions:

- `public int executeStep() { write your code here }`
`setParameters(Hashtable params) { write your code here }`

The workflow step executes the `executeStep()` function.

`setParameters(Hashtable params)` defines parameters for the class. This function is called before execution.

Available Arguments

If a user has launched the workflow from within a worksheet, Demantra automatically passes arguments to the workflow.

Example

- ```
package com.demantra.workflow.step;
import com.demantra.workflow.parameters.*;
public class SampleCustomStep implements CustomStep
{
 public SampleCustomStep() {}
 public int executeStep(Parameter[] parms)
 {
 int i, length = parms != null ? parms.length : 0;
 for(i=0; i<length; i++)
 System.out.println("Parameter name : " + parms[i].getName() + " value: " +
 parms[i].getValue());
 // write your own logic here
 //
 return LinkedStep.ST_COMPLETED;
 }
}
```

## Notes

If a step ends before the Check Finish After period, or during a Check Finish Every period, then the Workflow Engine still waits for that counter to finish before checking if the step has finished.

In timers, a month is measured as a calendar month.



## Delete Member Step

This kind of step should be used only as a method. It deletes the specified level member.

### End Conditions

This step is completed when the level member has been removed from the database.

### Properties

| Properties | Description                                                                                                                                                                                                                                                                                                                              | Default   |
|------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------|
| Step ID    | Unique identifier for the step.                                                                                                                                                                                                                                                                                                          |           |
| Recovery   | Specify what the Workflow Engine should do if the system crashes while performing this step:<br><br>Ask—the engine follows a Fail-To-Execute procedure for the step.<br><br>Retry—the engine executes the step again.<br><br>Continue—the engine continues with the next step.<br><br>Abort—the engine terminates the workflow instance. | Ask       |
| Pause      | Specify how long to wait after a step has been completed, before starting the next step.                                                                                                                                                                                                                                                 | 0 seconds |

See also

- "Create Member Step" "Edit Member Step"

## Edit Member Step

This kind of step should be used only as a method. It uses the passed arguments, and modifies the specified level member.

**Note:** If you are configuring a method that changes attribute values, the

workflow must include an Edit Member Step as its first step.  
Otherwise, the changed values will not be saved to the database.

### End Conditions

This step is completed when the level member has been edited.

### Properties

| Properties         | Description                                                                                                                                                                                                                                                                                                                              | Default   |
|--------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------|
| Step ID            | Unique identifier for the step.                                                                                                                                                                                                                                                                                                          |           |
| Recovery           | Specify what the Workflow Engine should do if the system crashes while performing this step:<br><br>Ask—the engine follows a Fail-To-Execute procedure for the step.<br><br>Retry—the engine executes the step again.<br><br>Continue—the engine continues with the next step.<br><br>Abort—the engine terminates the workflow instance. | Ask       |
| Check Finish After | Specify how long to wait after starting the step and before first checking to see if the end conditions have been met.                                                                                                                                                                                                                   | 0 seconds |
| Pause              | Specify how long to wait after a step has been completed, before starting the next step.                                                                                                                                                                                                                                                 | 0 seconds |

See also

- "Create Member Step" "Delete Member Step"

## Email Step

This kind of step is used to send an email message to a user. This step allows a connection to the installed messaging application using SMTP protocol.

## End Conditions

This step is completed when the email message is successfully delivered to the installed messaging system.

**Note:** This step does not check if or when the message is read.

## Properties

| Properties | Description                                                                                                                                                                                                                                                                                                                              | Default   |
|------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------|
| Step ID    | Unique identifier for the step.                                                                                                                                                                                                                                                                                                          |           |
| Pause      | Specify how long to wait after a step has been completed, before starting the next step.                                                                                                                                                                                                                                                 | 0 seconds |
| To User    | Select the user who should receive the email. The list of possible users includes all users defined within this component.                                                                                                                                                                                                               |           |
| Subject    | The subject line of the email message                                                                                                                                                                                                                                                                                                    |           |
| Message    | The email message text                                                                                                                                                                                                                                                                                                                   |           |
| Recovery   | Specify what the Workflow Engine should do if the system crashes while performing this step:<br><br>Ask—the engine follows a Fail-To-Execute procedure for the step.<br><br>Retry—the engine executes the step again.<br><br>Continue—the engine continues with the next step.<br><br>Abort—the engine terminates the workflow instance. | Ask       |

## Note

**Make sure that the user has an email address.** You use the Business Modeler to

configure email addresses for the users. See "Creating or Modifying a User".

## Exception Step

This kind of step sends tasks to users depending on specific conditions in the Demantra database. This step runs a worksheet, normally a worksheet in which an exception condition has been defined. (If you attach an exception to a worksheet, Demantra checks the values of the worksheet data and displays only the combinations that meet the exception criteria.)

If the worksheet returns data, then the Workflow Engine sends a task to each specified user. When all users have marked as this task as done, the workflow continues to the next step.

### End Conditions

This step is completed when all users mark the task as Done.

### Properties

| Properties     | Description                                                                                                                                                                                                                                     | Default |
|----------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------|
| Step ID        | Unique identifier for the step.                                                                                                                                                                                                                 |         |
| Worksheet Name | Select the worksheet to test, normally a worksheet in which an exception condition has been defined. The list of worksheets includes all public worksheets and all worksheets that you own. Be sure to select a public worksheet for this step. |         |
| Task           | Specify a task. See "Specifying a Task".                                                                                                                                                                                                        |         |
| User           | Select the user or users who should receive tasks in the case of this exception. Press and hold down the Ctrl or Shift key while selecting multiple users. The list of possible users includes all users defined within this component.         |         |

| Properties         | Description                                                                                                                                                                                                                                                                                                                              | Default    |
|--------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------|
| Group              | Select the group or groups who should receive tasks in the case of this exception. Press and hold down the Ctrl or Shift key while selecting multiple users. The list of possible groups includes all users defined in Demantra.                                                                                                         |            |
| Recovery           | Specify what the Workflow Engine should do if the system crashes while performing this step:<br><br>Ask—the engine follows a Fail-To-Execute procedure for the step.<br><br>Retry—the engine executes the step again.<br><br>Continue—the engine continues with the next step.<br><br>Abort—the engine terminates the workflow instance. | Ask        |
| Check Finish After | Specify how long to wait after starting the step and before first checking to see if the end conditions have been met.                                                                                                                                                                                                                   | 0 seconds  |
| Check Finish Every | The Workflow Engine checks periodically to see whether or not the end conditions have been met. This property specifies how long to wait between two successive checks.                                                                                                                                                                  | 60 seconds |
| Pause              | Specify how long to wait after a step has been completed, before starting the next step.                                                                                                                                                                                                                                                 | 0 seconds  |
| Timeout>Timer      | Specify a time-out for the step.                                                                                                                                                                                                                                                                                                         | never      |

| Properties         | Description                                                                                                                                                                                                                                                                                                                                                                                                                    | Default |
|--------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------|
| Timeout>Alert Time | Specify the alert phase that can occur just before the step times out. During the alert phase, the task due date is displayed in red in the user's My Task module.                                                                                                                                                                                                                                                             | never   |
| Population         | <p>Specifies the population attribute that further filters the worksheet used by this step. Specify the following:</p> <p>Name—specify the name of the population attribute, as given in GROUP_ATTRIBUTES_POPULATION.ATTRIBUTE_LABEL. In the demo, this attribute is named Population.</p> <p>The Value field is not used.</p> <p>This property is useful when you use a workflow as a method; otherwise it has no effect.</p> |         |

### Fail-To-Execute

Aside from the expected cases of Fail-To-Execute such as an invalid worksheet ID, you should take care to avoid the following circumstances which will also cause a Fail-To-Execute:

- Workflow Initiator does not have privileges to execute the condition worksheet.
- An invalid Group id or User id in the ExceptionStep.

This applies if a Group contains an invalid User id or any individual User id listed to receive the Exception step task is invalid. Exception step will Fail-To-Execute even if all other Group ids or User ids in the Exception step are valid.

In the event of a Fail-To-Execute, none of the user groups or users listed in the Exception step receive the Exception step task.

### Notes

If a step ends before the Check Finish After period, or during a Check Finish Every period, then the Workflow Engine still waits for that counter to finish before checking if

the step has finished.

In timers, a month is measured as a calendar month.

Exception steps take advantage of these two performance enhancing features:

1. Exceptions run off of cached worksheet data when the cache is enabled for a worksheet.
2. Exceptions on combinations are evaluated until the first exception value is found. That is, no further evaluation of series values is done once one a violation has been found for a combination.

## Executable Step

This kind of step runs applications from within the workflow instance. The applications can be external executable files such as .exe and batch files, or Demantra executables (such as the Analytical Engine).

### End Conditions

This step is completed when the executed program ends and sends an interrupt to the Workflow Engine. The Workflow Engine then continues with the workflow instance.

### Properties

| Properties   | Description                                                                                                                                                                                                       | Default   |
|--------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------|
| Step ID      | Unique identifier for the step.                                                                                                                                                                                   |           |
| Pause        | Specify how long to wait after a step has been completed, before starting the next step.                                                                                                                          | 0 seconds |
| Command Line | The location of the network of the file to be executed, and its full name. (Must be in double quotes). Specify the full path to the file.<br><br><b>Note:</b> The file location is always from the server's view. |           |

| Properties | Description                                                                                                                                                                                                                                                                                                                                             | Default |
|------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------|
| Recovery   | <p>Specify what the Workflow Engine should do if the system crashes while performing this step:</p> <p>Ask — the engine follows a Fail-To-Execute procedure for the step.</p> <p>Retry — the engine executes the step again.</p> <p>Continue — the engine continues with the next step.</p> <p>Abort — the engine terminates the workflow instance.</p> | Ask     |

## Notes

The Workflow Engine executes the command that you specified for the Command Line option. For this reason, you can only run files that can be opened by using a single DOS prompt instruction. This is important if you are accessing files over a network.

Also, if you are invoking a batch file, be sure to do the following within that batch file:

- Enclose all paths within double quotes
- Use complete paths and file names

For information on running the Analytical Engine from the command line, see "Running the Engine from the Command Line".

When executing batch files, Fail-to-Execute is triggered only if there is an error when executing the batch file. If the batch file fail after execution, then Fail-to-Execute will not be triggered.

To run a .bat file from the Workflow Engine, do the following:

## Example:

1. Create the .bat file. Within this file, ensure that all paths are enclosed in double quotes. For example:
  - `cd "E:\Demantra\Spectrum610\scripts_vtk\Biio_load_proms"`
  - IF EXIST
  - "E:\Demantra\Spectrum610\scripts\_vtk\Biio\_load\_proms\cust\_prom\_flag.dat" (exit) ELSE (myplus vk\_check\_file)



- Within the Workflow Manager, create a new workflow.
- Insert an Executable Step.
- In the Command Line option of this step, put the full path to the location of the file you wish to run.

## Group Step

This kind of step sends tasks to a user group or groups. Tasks are shown in My Tasks in Collaborator Workbench, and a task is usually associated with a worksheet. The purpose of a task is typically to draw attention to exceptions, or to request that the user review and possibly edit the worksheet.

In contrast to the User Step, this step includes the Manager property, which specifies the user who is in charge of or manages the process.

You can use this step to send a worksheet to users with different security permissions. Each user is then presented with a worksheet, with contents filtered by the user's permissions.

You can also configure the step to automatically send email notification of the new task.

**Important:** If you do so, make sure that each user has an email address.

You use the Business Modeler to configure email addresses for the users. See "Creating or Modifying a User".

### End Conditions

This step is completed when all the users mark all the tasks as Done.

### Properties

| Properties | Description                                                                                                                                                                                                            | Default |
|------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------|
| Step ID    | Unique identifier for the step.                                                                                                                                                                                        |         |
| User       | Select the user or users who should receive this task or tasks. Press and hold down the Ctrl or Shift key while selecting multiple users. The list of possible users includes all users defined within this component. |         |

| Properties         | Description                                                                                                                                                                                                                                                                                                                                     | Default    |
|--------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------|
| Group              | Select the group or groups who should receive this task or tasks. Press and hold down the Ctrl or Shift key while selecting multiple users. The list of possible groups includes all users defined in Demantra.                                                                                                                                 |            |
| Tasks              | Specify one or more tasks. See "Specifying a Task".                                                                                                                                                                                                                                                                                             |            |
| Recovery           | <p>Specify what the Workflow Engine should do if the system crashes while performing this step:</p> <p>Ask—the engine follows a Fail-To-Execute procedure for the step.</p> <p>Retry—the engine executes the step again.</p> <p>Continue—the engine continues with the next step.</p> <p>Abort—the engine terminates the workflow instance.</p> | Ask        |
| Check Finish After | Specify how long to wait after starting the step and before first checking to see if the end conditions have been met.                                                                                                                                                                                                                          | 0 seconds  |
| Check Finish Every | The Workflow Engine checks periodically to see whether or not the end conditions have been met. This property specifies how long to wait between two successive checks.                                                                                                                                                                         | 60 seconds |
| Pause              | Specify how long to wait after a step has been completed, before starting the next step.                                                                                                                                                                                                                                                        | 0 seconds  |

| Properties         | Description                                                                                                                                                                                                                                                                          | Default |
|--------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------|
| Manager            | Specify the user who is in charge of or manages the process. In the event of a timeout, this user is notified by email. This notification is in addition to the timeout notification email that is sent to each user whose task has timed out and the timeout step that is executed. |         |
| Timeout>Timer      | Specify a timeout for the step.                                                                                                                                                                                                                                                      | never   |
| Timeout>Alert Time | Specify the alert phase that can occur just before the step times out. During the alert phase, the task due date is displayed in red in the user's My Task module.                                                                                                                   | never   |

## Timeout

A group task times out if one or more of the users in the group do not mark the task as done before the response period has ended. The timeout procedure is as follows:

A user who is responsible for the relevant task receives an email message notification of all the users within the group that have not marked the task as done. For example, this may be the group manager, or a supporting job function.

1. The task stays in the My Tasks module of the users who have not marked it as done.
2. The Workflow Engine continues with an alternative procedure that has been defined within the workflow for this circumstance.

## Notes

If a step ends before the Check Finish After period, or during a Check Finish Every period, then the Workflow Engine still waits for that counter to finish before checking if the step has finished.

In timers, a month is measured as a calendar month.

For easy maintenance and flexibility, it is recommended to use the Group property instead of User whenever possible. Using Group allows you to change the defined user name for a job function within a workflow instance without editing the workflow schema itself.

If any of the specified Group is empty or contains an invalid user name, then this step will Fail-To-Execute.

See also

- For information on tasks, see the Oracle Demantra Demand Management User's Guide.

## Paste Member Step

This kind of step should be used only as a method. It uses the arguments passed from the Copy or Cut step and pastes the level member into the location specified.

The step is used by the Paste workflow and works in conjunction with either the Cut workflow or the Copy workflow. (Note these are default workflows called by menu functions of the same name). When an object has been copied to memory by the Copy function, then this step is used to paste the object to a new location.

The copy-paste functionality works only on General levels (such as Promotion), not item levels or location levels. For example, in Predictive Trade Planning you can copy and paste promotions within a scenario.

However, the cut-paste functionality can work also on members of Promotions.

### End Conditions

This step is completed when the member has been created.

| Properties | Description                                                                                                                                                                                                                                                                                                 | Default |
|------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------|
| Step ID    | Unique identifier for the step.                                                                                                                                                                                                                                                                             | ---     |
| Recovery   | Specify what the WorkflowEngine should do if the system crashes while performing this step: Ask—the engine follows a Fail-To-Execute procedure for the step. Retry—the engine executes the step again. Continue—the engine continues with the next step. Abort—the engine terminates the workflow instance. | Ask     |

---

|       |                                                                                          |           |
|-------|------------------------------------------------------------------------------------------|-----------|
| Pause | Specify how long to wait after a step has been completed, before starting the next step. | 0 seconds |
|-------|------------------------------------------------------------------------------------------|-----------|

---

## Refresh Population Step

Promotions can be defined at an aggregate item or location level. When the items or groups in these aggregate levels change, the Refresh Population Step can be used to refresh the population of the promotion to ensure the promotion data is aligned correctly.

The Refresh Population Step can either be run against:

- A single promotion via a method. Create a workflow containing only the Refresh Population workflow step. Then create a new method called "Refresh Population" on the Promotion level that is linked to the newly created workflow and isn't configured with any input or output arguments. When the Refresh Population method is run against a promotion, it launches the Refresh Population workflow step which refreshes the population data for the promotion.
- All promotions in the system in batch mode. Execute a workflow containing the Refresh Population workstep from the Workflow Manager.

Running the Refresh Population Step refreshes the population data, realigning the data for either a single promotion or all promotions (Promotion\_data table) according to the current promotion population definition(s).

The Refresh Population workflow step does not consider any historical records on a promotion. It only considers current/future records on a given promotion and realigns data in that promotion according to the current promotion group definition. The historical records are left intact in the promotion associated with the original promotion population.

The Refresh Population workflow step can be run against promotions that are in any promotion status except the "Closed" status, when the promotion is in a read-only state.

**Caution:** Define specific item and location levels associated with each promotion. Doing so will ensure that the Refresh Population workflow step adheres to the population definition when run.

| Properties    | Description                                                                                                                                                                                                                                                                                                                                     | Default   |
|---------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------|
| Step ID       | Unique identifier for the step.                                                                                                                                                                                                                                                                                                                 |           |
| User          | Select the user who can run this workflow step. Refresh Population refreshes the population of the promotions in the application according to the security definition of the user specified. This parameter is applicable only when the Refresh Population workflow is executed in batch mode to refresh all promotions in the system.          |           |
| GL Base Level | Select the GL base level to which this workflow applies. This workflow only applies to the Promotion level.                                                                                                                                                                                                                                     | Promotion |
| Recovery      | <p>Specify what the Workflow Engine should do if the system crashes while performing this step:</p> <p>Ask—the engine follows a Fail-To-Execute procedure for the step.</p> <p>Retry—the engine executes the step again.</p> <p>Continue—the engine continues with the next step.</p> <p>Abort—the engine terminates the workflow instance.</p> | Abort     |

| Properties | Description                                                                                                                                                                                                                                                                                                            | Default           |
|------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------|
| Parameters | Select the parameters you want to run during this workflow step. The Refresh_From_Date parameter appears by default. When the value field is left blank, then the data is refreshed for all records within the promotion from this date forward. If desired, you can enter a value to refresh data from a future date. | Refresh_From_Date |

See also

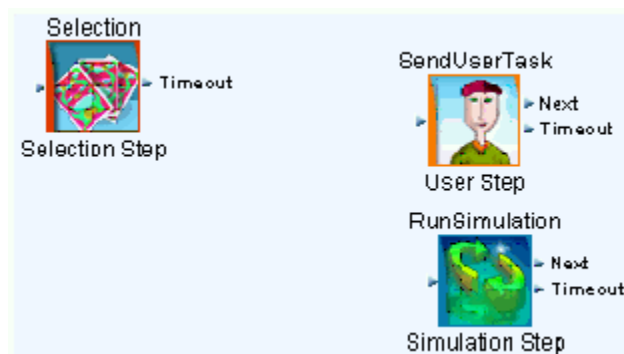
- For information on running the Refresh Population method, see "To reassign a promotion to a different item or location" in the *Oracle Demantra Predictive Trade Planning User's Guide*.

## Selection Step

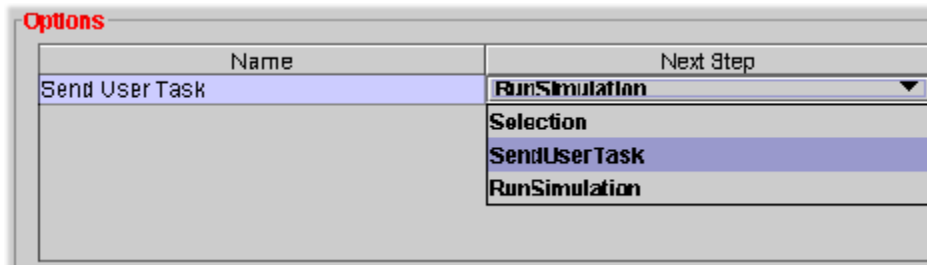
This kind of step sends a selection task to a user. A selection task includes a list of options, each of which specifies a different branch for the workflow to follow. Like other tasks, this task is shown in My Tasks in Collaborator Workbench. This task can also be associated with a worksheet.

### Defining and Linking to Selections

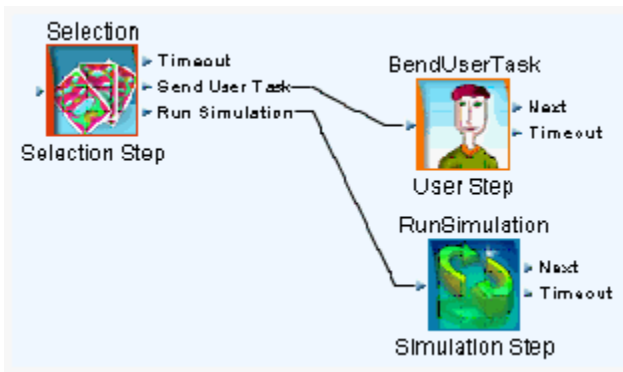
When you first add a Selection step to a workflow, it does not contain any options and therefore does not have any connectors that lead from those options. For example, suppose that you have created a Selection step and two other steps that you want to use as options:



Within the properties of the Selection step, you can add the options. When you add an option, you specify a user-friendly name and you choose the corresponding step from the list of existing steps as follows:



As you add each option, the Workflow Editor automatically creates a connection handle for that option and automatically links that to the appropriate step, as follows:



You may enter as many options as you like into a selection step.

### End Conditions

This step is completed when the user makes a selection and marks the selection task as Done.

### Properties

| Properties  | Description                                                                                                                | Default |
|-------------|----------------------------------------------------------------------------------------------------------------------------|---------|
| Step ID     | Unique identifier for the step.                                                                                            |         |
| Option Name | The Name of the Option. This is the text that shows for each option in the selection list presented by the selection task. |         |



| Properties         | Description                                                                                                                                                                                                                                                                     | Default    |
|--------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------|
| User               | Select the user who should receive this list of options. The list of possible users includes all users defined within this component.                                                                                                                                           |            |
| Task               | This is the task that the SelectionStep sends.                                                                                                                                                                                                                                  |            |
| Recovery           | Specify what the Workflow Engine should do if the system crashes while performing this step:<br><br>Ask—the engine follows a Fail-To-Execute procedure for the step.<br><br>Retry—the engine executes the step again.<br><br>Abort—the engine terminates the workflow instance. | Ask        |
| Check Finish After | Specify how long to wait after starting the step and before first checking to see if the end conditions have been met.                                                                                                                                                          | 0 seconds  |
| Check Finish Every | The Workflow Engine checks periodically to see whether or not the end conditions have been met. This property specifies how long to wait between two successive checks.                                                                                                         | 60 seconds |
| Pause              | Specify how long to wait after a step has been completed, before starting the next step.                                                                                                                                                                                        | 0 seconds  |
| Timeout>Timer      | Specify a timeout for the step.                                                                                                                                                                                                                                                 | never      |

| Properties         | Description                                                                                                                                                        | Default |
|--------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------|
| Timeout>Alert Time | Specify the alert phase that can occur just before the step times out. During the alert phase, the task due date is displayed in red in the user's My Task module. | never   |

## Notes

If a step ends before the Check Finish After period, or during a Check Finish Every period, then the Workflow Engine still waits for that counter to finish before checking if the step has finished.

In timers, a month is measured as a calendar month.

## Simulation Step

This kind of step submits a worksheet to the Simulation Engine and then displays the worksheet and the simulation results to a specific user. The user can either accept or reject the simulation. This kind of step works the same as running a simulation in the user applications.

**Note:** The Simulation Step does not start the Simulation Engine. You must make sure that the engine is running before a step of this kind is launched; otherwise the step will fail. You can start the Simulation Engine by using an Executable Step.

**Note:** You must take care not to initiate the Simulation Engine twice.

**Note:** The Simulation Engine and Analytical Engine cannot run simultaneously. You must take care not to initiate the Simulation Engine if the Analytical Engine is running.

**Note:** Before running the simulation, the Workflow Engine checks to see if a simulation is running or has already run for this worksheet. If a simulation has been scheduled but has not yet run, the Workflow Engine waits until the simulation completes and then continues to the next step in the workflow schema. If a simulation has already run but has not been accepted or rejected, the Workflow Engine rejects it and executes the workflow simulation step.

## End Conditions

This step is completed when the Simulation Engine finishes processing the simulation request.

## Properties

| Properties  | Description                                                                                                                                                                                                                                                                                                                                                                                                         | Default |
|-------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------|
| Step ID     | Unique identifier for the step.                                                                                                                                                                                                                                                                                                                                                                                     |         |
| User        | Select the user who should receive this simulation worksheet. The list of possible users includes all users defined within this component.                                                                                                                                                                                                                                                                          |         |
| Auto Accept | <p>Specify whether to accept the results of the simulation automatically:</p> <p>If Yes, the results of the simulation are saved directly as valid forecast data.</p> <p>If No, the results of the simulation are first made available for review by a user. To review the results of a simulation, the assigned user can open the worksheet from a task. The user can then decide whether or not to save them.</p> | Yes     |
| Query Name  | Select the worksheet on which to run the simulation. The list of worksheets includes all public worksheets and all worksheets that you own.                                                                                                                                                                                                                                                                         |         |

| Properties         | Description                                                                                                                                                                                                                                                                                                                                             | Default    |
|--------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------|
| Recovery           | <p>Specify what the Workflow Engine should do if the system crashes while performing this step:</p> <p>Ask — the engine follows a Fail-To-Execute procedure for the step.</p> <p>Retry — the engine executes the step again.</p> <p>Continue — the engine continues with the next step.</p> <p>Abort — the engine terminates the workflow instance.</p> | Ask        |
| Check Finish After | Specify how long to wait after starting the step and before first checking to see if the end conditions have been met.                                                                                                                                                                                                                                  | 0 seconds  |
| Check Finish Every | The Workflow Engine checks periodically to see whether or not the end conditions have been met. This property specifies how long to wait between two successive checks.                                                                                                                                                                                 | 60 seconds |
| Pause              | Specify how long to wait after a step has been completed, before starting the next step.                                                                                                                                                                                                                                                                | 0 seconds  |
| Timeout>Timer      | Specify a timeout for the step.                                                                                                                                                                                                                                                                                                                         | never      |
| Timeout>Alert Time | Specify the alert phase that can occur just before the step times out. During the alert phase, the task due date is displayed in red in the user's My Task module.                                                                                                                                                                                      | never      |

## Notes

If a step ends before the Check Finish After period, or during a Check Finish Every

period, then the Workflow Engine still waits for that counter to finish before checking if the step has finished.

In timers, a month is measured as a calendar month.

In the event of a timeout or a major system failure, the simulation request remains in the Simulation Engine queue. You must remove it manually from the queue.

See also

- For information on simulation, see the Oracle Demantra Demand Management User's Guide or other user manuals. Also see "Batch and Simulation Modes".

## Stored Procedure Step

This kind of step runs a stored database procedure, such as MDP\_ADD or REBUILD\_INDEXES.

For information on creating database procedures, see the Oracle database documentation.

### End Conditions

This step is completed when the database procedure finishes.

### Properties

| Properties     | Description                                                                                                                             | Default   |
|----------------|-----------------------------------------------------------------------------------------------------------------------------------------|-----------|
| Step ID        | Unique identifier for the step.                                                                                                         |           |
| Pause          | Specify how long to wait after a step has been completed, before starting the next step.                                                | 0 seconds |
| Procedure Name | Name of the stored procedure.                                                                                                           |           |
| Parameters     | Any input parameters that are needed by the database procedure. List the parameters in the order that they are needed by the procedure. |           |

| Properties | Description                                                                                                                                                                                                                                                                                                                                             | Default |
|------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------|
| Recovery   | <p>Specify what the Workflow Engine should do if the system crashes while performing this step:</p> <p>Ask — the engine follows a Fail-To-Execute procedure for the step.</p> <p>Retry — the engine executes the step again.</p> <p>Continue — the engine continues with the next step.</p> <p>Abort — the engine terminates the workflow instance.</p> | Ask     |

see also

- For information on the predefined Demantra database procedures, see "Database Procedures".

## Passing level\_id and member\_id Values

When passing level\_id and member\_id parameter values within a stored procedure step, you must enclose the parameter in # signs. For example, #level\_id# and #member\_id#.

## Transfer Step

This kind of step executes an import or export integration interface. You create integration interfaces within the Business Modeler. See "Series and Level Integration"

**Note:** To execute a file load interface within a workflow, create a batch script that executes the interface, and then use an Executable Step to run the script.

### End Conditions

This step is completed when the interface has been executed.

### Properties

| Properties | Description                                                                                                                                                                                                                                                                                                                              | Default   |
|------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------|
| Step ID    | Unique identifier for the step.                                                                                                                                                                                                                                                                                                          |           |
| Type       | Specify the type of data transfer: Import or Export.                                                                                                                                                                                                                                                                                     |           |
| Profile    | Select the integration interface from the drop-down list. See "Series and Level Integration".                                                                                                                                                                                                                                            |           |
| Pause      | Specify how long to wait after a step has been completed, before starting the next step.                                                                                                                                                                                                                                                 | 0 seconds |
| Recovery   | Specify what the Workflow Engine should do if the system crashes while performing this step:<br><br>Ask—the engine follows a Fail-To-Execute procedure for the step.<br><br>Retry—the engine executes the step again.<br><br>Continue—the engine continues with the next step.<br><br>Abort—the engine terminates the workflow instance. | Ask       |

## Update Data Step

This step is called when data is updated in the worksheet. It is not typically included in user-defined workflows.

### Properties Dialog for Update Data Step

## User Step

This kind of step sends tasks to a user. Tasks are shown in My Tasks in Collaborator Workbench, and a task is usually associated with a worksheet. The purpose of a task is typically to draw attention to exceptions, or to request that the user review and possibly edit the worksheet.

You can also configure the step to automatically send email notification of the new task. **If you do so, make sure that the user has an email address.** You use the Business Modeler to configure email addresses for the users. See "Creating or Modifying a User".

### End Conditions

This step is completed when the user marks all tasks as Done.

### Properties

| Properties | Description                                                                                                                              | Default |
|------------|------------------------------------------------------------------------------------------------------------------------------------------|---------|
| Step ID    | Unique identifier for the step.                                                                                                          |         |
| User       | Select the user who should receive this list of options.<br>The list of possible users includes all users defined within this component. |         |
| Task       | Specify one or more tasks. See "Specifying a Task".                                                                                      |         |



| Properties         | Description                                                                                                                                                                                                                                                                                                                                             | Default    |
|--------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------|
| Recovery           | <p>Specify what the Workflow Engine should do if the system crashes while performing this step:</p> <p>Ask — the engine follows a Fail-To-Execute procedure for the step.</p> <p>Retry — the engine executes the step again.</p> <p>Continue — the engine continues with the next step.</p> <p>Abort — the engine terminates the workflow instance.</p> | Ask        |
| Check Finish After | Specify how long to wait after starting the step and before first checking to see if the end conditions have been met.                                                                                                                                                                                                                                  | 0 seconds  |
| Check Finish Every | The Workflow Engine checks periodically to see whether or not the end conditions have been met. This property specifies how long to wait between two successive checks.                                                                                                                                                                                 | 60 seconds |
| Pause              | Specify how long to wait after a step has been completed, before starting the next step.                                                                                                                                                                                                                                                                | 0 seconds  |
| Timeout>Timer      | Specify a timeout for the step.                                                                                                                                                                                                                                                                                                                         | never      |
| Timeout>Alert Time | Specify the alert phase that can occur just before the step times out. During the alert phase, the task due date is displayed in red in the user's My Task module.                                                                                                                                                                                      | never      |

### Timeout

A user task times out if the user does not mark the task as done before the response

period has ended. The timeout procedure for a user task is as follows:

The user is sent an email notification that the task has timed out.

1. The task is removed from the user's task list.
2. The Workflow Engine continues with an alternative procedure that has been defined within the workflow for this circumstance.

#### Notes

If a step ends before the Check Finish After period, or during a Check Finish Every period, then the Workflow Engine still waits for that counter to finish before checking if the step has finished.

In timers, a month is measured as a calendar month.

See also

- For information on tasks, see the Oracle Demantra Demand Management User's Guide.

## Wait Until Step

This kind of step pauses the workflow until a specific condition is met. When the condition is true, the Workflow Engine continues with the next step in the workflow.

You can specify a condition in either of the following general ways:

- You can instruct the Workflow Engine to look for a specific file and wait until the file is created, or is modified, or reaches a certain size.
- You can specify an SQL query to execute. The Workflow Engine runs repeatedly until it returns a value that is different from the original returned value. With this option, you pause a workflow until, for example, a price changes.

You can specify multiple wait conditions. If you do, they are combined with a logical OR. For example, if you select both Created and Modified, the Workflow Engine waits until either a new file has been created or an existing file has been modified.

#### End Conditions

This step is completed when the condition is met.

#### Properties

| Properties | Description                     | Default |
|------------|---------------------------------|---------|
| Step ID    | Unique identifier for the step. |         |

| Properties | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    | Default |
|------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------|
| File       | <p>Full path and filename for the file to be checked. The path and filename can include wildcards, for example:</p> <p>c:\dat\*.dat</p> <p>The Workflow Engine ignores the case of the path and filename.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |         |
| Wait Until | <p>Specify the file state to wait for:</p> <p>Created—wait until this file is created. If you use a wildcard in the filename, then wait until at least one new file that matches the given name is created.</p> <p>Exists—wait until this file exists. If you use a wildcard in the filename, then wait until at least one new file that matches the given name is created. In contrast to the Created option, this option creates a condition that can be true even for the first time the file is checked.</p> <p>Modified—wait until the timestamp on this file has been changed. If you use a wildcard in the filename, then wait until the timestamp on at least one matching file is changed.</p> <p>Size is bigger than—wait until the file is larger than the given size, in kB. If you use a wildcard in the filename, then wait until at least one of the matching files exceeds the given size.</p> |         |

| Properties         | Description                                                                                                                                                                                                                                                                                                                              | Default    |
|--------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------|
| SQL                | Specify an SQL statement.<br>Every time period, the Workflow Engine will execute this statement. When the result of this statement is different than it was before, the step condition has been met, and the workflow continues.                                                                                                         |            |
| Check Finish Every | The Workflow Engine checks periodically to see whether or not the specified conditions have been met. This property specifies how long to wait between two successive checks.                                                                                                                                                            | 60 seconds |
| Recovery           | Specify what the Workflow Engine should do if the system crashes while performing this step:<br><br>Ask—the engine follows a Fail-To-Execute procedure for the step.<br><br>Retry—the engine executes the step again.<br><br>Continue—the engine continues with the next step.<br><br>Abort—the engine terminates the workflow instance. | Ask        |
| Check Finish After | Specify how long to wait after starting the step and before first checking to see if the end conditions have been met.                                                                                                                                                                                                                   | 0 seconds  |
| Pause              | Specify how long to wait after a step has been completed, before starting the next step.                                                                                                                                                                                                                                                 | 0 seconds  |
| Timeout>Timer      | Specify a timeout for the step.                                                                                                                                                                                                                                                                                                          | never      |

| Properties         | Description                                                                                                                                                        | Default |
|--------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------|
| Timeout>Alert Time | Specify the alert phase that can occur just before the step times out. During the alert phase, the task due date is displayed in red in the user's My Task module. | never   |

#### Note

If a step ends before the Check Finish After period, or during a Check Finish Every period, then the Workflow Engine still waits for that counter to finish before checking if the step has finished.

In timers, a month is measured as a calendar month.

## Worksheet Cache Step

This kind of step automatically refreshes the caches of the specified worksheets in batch mode for the specified users.

Users can also refresh manually.

This step refreshes existing worksheet caches based on the latest source data. This step does not create new worksheet caches.

#### End Conditions

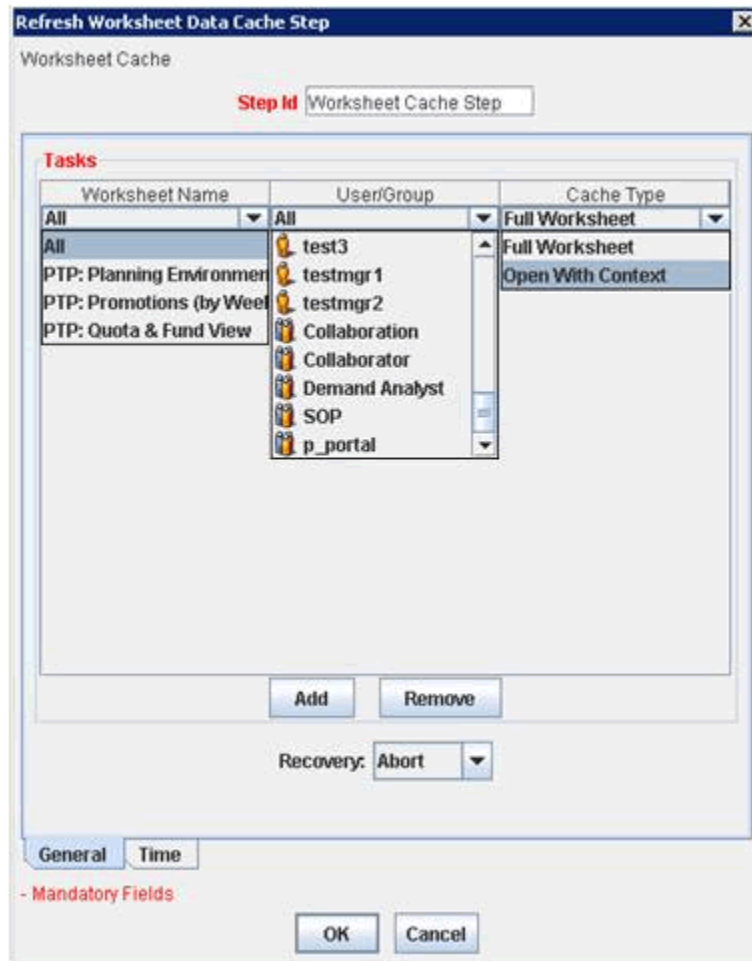
This step is completed when all the specified caches are refreshed.

#### Properties

| Properties | Description                     | Default |
|------------|---------------------------------|---------|
| Step ID    | Unique identifier for the step. |         |

| Properties     | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              | Default |
|----------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------|
| Worksheet Name | <ul style="list-style-type: none"> <li>- Select All to refresh caches for all worksheets that have the <b>Cache Worksheet Data</b> option selected.</li> <li>- Select a specific worksheet to refresh the cache for only that worksheet. Note that only worksheets that have the <b>Cache Worksheet Data</b> option selected are listed.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                      |         |
| User/Group     | <ul style="list-style-type: none"> <li>- Select <b>All</b> to refresh the cache for the selected worksheet (or All worksheets) for all users.</li> <li>- Select a specific User to refresh the cache for the selected worksheet (or All worksheets) for just that User. Note that only Users that have the <code>CAN_CREATE_CW</code> option selected are listed. For details, see Worksheet Caching, page 8-11 &gt; Enabling Worksheet Caching.</li> <li>- Select a User Group to refresh the cache for the selected worksheet (or All worksheets) for all Users that belong to the selected User Group</li> </ul> <p>The drop-down list displays individual Users first followed by User Groups. Users are identified by a single person icon and User Groups are identified by a two person icon.</p> |         |

| Properties         | Description                                                                                                                                                                                                                                          | Default    |
|--------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------|
| Check Finish Every | The Workflow Engine checks periodically to see whether or not the specified conditions have been met. This property specifies how long to wait between two successive checks.                                                                        | 60 seconds |
| Check Finish After | Specify how long to wait after starting the step and before first checking to see if the end conditions have been met.                                                                                                                               | 1 minute   |
| Pause              | Specify how long to wait after a step has been completed, before starting the next step.                                                                                                                                                             | 0 seconds  |
| Timeout>Timer      | Specify a timeout for the step.                                                                                                                                                                                                                      | never      |
| Timeout>Alert Time | Specify the alert phase that can occur just before the step times out.                                                                                                                                                                               | never      |
| Cache Type         | <p>- <b>Select Full Worksheet</b> to refresh the cache for when the worksheet is opened directly (<b>File &gt; Open</b>)</p> <p>- Select <b>Open With Context</b> to refresh the cache for when the worksheet is opened via an <b>Open With</b>.</p> | -          |



## Worksheet Step

The worksheet step runs a worksheet for a specific user and retrieves data for all combinations. It was created as a tool for helping administrators and DBA's tune application server and database performance, by simulating concurrent users opening worksheets. A user can set up several workflows with a set of commonly used worksheets, then launch several of these workflows in parallel to simulate concurrent users. In this way it could be used to load test the system prior to a go-live.

### Properties

| Properties     | Description                                                       | Default |
|----------------|-------------------------------------------------------------------|---------|
| Worksheet Name | Dropdown that lists all worksheets. Select the desired worksheet. | None    |



|          |                                                                                                                                                                                                                                                                                                             |     |
|----------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----|
| Recovery | Specify what the WorkflowEngine should do if the system crashes while performing this step: Ask—the engine follows a Fail-To-Execute procedure for the step. Retry—the engine executes the step again. Continue—the engine continues with the next step. Abort—the engine terminates the workflow instance. | Ask |
| Pause    | Specify how long to wait after a step has been completed, before starting the next step.                                                                                                                                                                                                                    | 0   |

Below is the first page of the dialog that appears when creating this workflow step. Here is where you select the worksheet to call, define the recovery behavior. The Time tab allows you to set the Pause criteria.

Worksheet Step's Properties

Worksheet Step

**Step Id** Step4

**Worksheet Name:** Consensus Forecast Product Category

**Recovery:** Abort

Ask

Abort

Retry

Continue

General Time

- Mandatory Fields

OK Cancel



# Part 4

---

## Configuring Specific Applications



---

## Configuring Predictive Trade Planning

This chapter describes how to configure Promotion Effectiveness, if an existing Demantra implementation is already in place.

This chapter covers the following topics:

- Overview of Promotion Effectiveness
- Overview of the Configuration Process
- Configuring Levels for Promotion Effectiveness
- Setting Parameters
- Configuring the Activity Browser
- Configuring the Application Server
- Configuring Promotion Statuses
- Loading Promotion Data
- Reference: PE Levels
- PTP Data Needs
- Integration in PTP
- Reference: Staging Tables
- Reference: Integration Interfaces
- Configuring the Default Promotion Start and Span
- Changing the Default Retailer Profile
- Configuring the Right-Click "Open With" Option
- Removing Right-Click Menu Options
- Replacing Collaborator Workbench Graphics
- Configuring Promotion Status Behavior
- Re-configuring Series if Syndicated Data Is Not Used

## Overview of Promotion Effectiveness

Promotion Effectiveness is a configurable Web-based product that analyzes the effectiveness of your marketing promotions, in particular trade promotions. In addition to base forecasting and forecasting lift due to promotions, Promotion Effectiveness can analyze the effects of different items on the sales patterns of others.

Promotion Effectiveness uses the same Web client that is used for Demand Management and DSM. For Promotion Effectiveness, the Analytical Engine provides a greater breakdown of details than does the engine for demand planning.

## Overview of the Configuration Process

These steps assume that you have already set up the basic Demantra implementation. This means that your implementation already contains the item levels and location levels that are meaningful in the designated environment.

To configure Promotion Effectiveness, the general steps are as follows:

1. Create the levels required by Promotion Effectiveness and then optionally customize them; see "Configuring Levels for Promotion Effectiveness".
2. Set values of parameters that control the behavior of the Web client, as it relates to Promotion Effectiveness. See "Setting Parameters".
3. Optionally configure the Activity Browser, if the default configuration is not suitable. See "Configuring the Activity Browser".
4. Decide if you are going to use the default promotion life cycle provided by Demantra. Configure this life cycle as described in "Configuring Promotion Statuses".
5. Load promotions and promotion data as described in "Loading Promotion Data".
6. Configure the Analytical Engine for use with Promotion Effectiveness. See "Configuring and Running the Analytical Engine".

**Note:** Deductions and Settlement Management functionality can be disabled for a given user by restricting access to the Settlement Level to 'No Access' in the user's permissions. For details, please see Data Security, page 10-1.

## Configuring Levels for Promotion Effectiveness

The Demantra installer sets up the Promotion Effectiveness (PE) structures by default.

You can customize these levels to some extent. See "Reference: PE Levels".

## Setting Parameters

To configure Promotion Effectiveness, specify values for the following parameters:

| Parameter         | Description                                                                                                                                       |
|-------------------|---------------------------------------------------------------------------------------------------------------------------------------------------|
| ColorCodingLevel  | Specifies the ID of the level that will be used to color code promotions.                                                                         |
| PromoDefaultStart | Specifies the default start date for promotions created within a worksheet: the current date, the last loaded sales date, or the worksheet start. |
| PromoDefaultSpan  | Specifies the default length of time for promotions created within a worksheet, in base time units.                                               |

For additional parameters, see "Engine Parameters"

See also

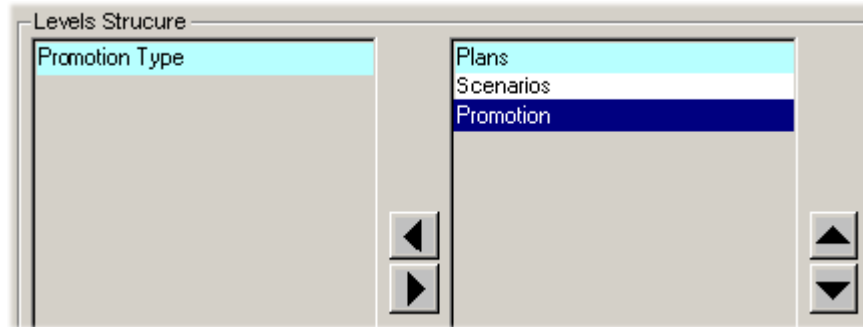
- "Configuring Parameters"

## Configuring the Activity Browser

The `CREATE_PE_STRUCT` procedure configures the Activity Browser in the Activity Details sub tab in worksheets. You can re-configure the Activity Browser as needed.

### To configure the Activity Browser

1. Click Configuration > Configure Levels. Or click the Configure Levels button.  
Business Modeler displays the Configure Levels screen.
2. Right-click a general level and select Open > Activity Browser.



3. For each general level to include in the Activity Browser, click the left arrow to move that general level from the left list to the right list.  
Or drag and drop general levels between the two lists as needed.
4. To specify the order of levels in the right side of the screen, select a level and click the up or down buttons.
5. When you are done, click Finish.

## Configuring the Application Server

**Note:** To configure the Promotion Optimization engine for Tomcat running on Linux you must include the following in the .bash\_profile file:

- export ILJCONFIG="HandleTableSize=1"
- export  
ODMS\_JVM\_LIBRARY\_OVERRIDE=/lib/i386/server/libjvm.so
- export  
LD\_LIBRARY\_PATH=\$LD\_LIBRARY\_PATH:demantr\_install\_directory/optimization/dll

**Note:** To configure the engine for Oracle Application Server, set the above environment variables using Oracle Enterprise Manager.

## Configuring Promotion Statuses

Demantra provides a default set of promotion statuses and structures to support a typical promotion life cycle. The IDs



have default names and specific hardcoded behaviors meanings, as follows:

| <b>promotion.status field</b> | <b>Default status name*</b> | <b>Hardcoded behavior</b>                                                                                                                   |
|-------------------------------|-----------------------------|---------------------------------------------------------------------------------------------------------------------------------------------|
| 1                             | Unplanned                   | Analytical Engine ignores this promotion.<br><br>A user can manually change the promotion status to 1, 2, or 3.                             |
| 2                             | Cancelled                   | A user can manually change the promotion status to 1, 2, or 3.                                                                              |
| 3                             | Planned                     | A user can manually change the promotion status to 1, 2, or 3.                                                                              |
| 4                             | Committed                   | User cannot change the status. (An optional procedure can be used to advance the status; see below.)<br><br>User cannot edit the promotion. |
| 5                             | Executed                    | User can change the status to 6, but cannot otherwise edit the promotion.                                                                   |
| 6                             | Closed                      | User cannot change the status and cannot edit the promotion in other ways.                                                                  |

\*The status names are in the promotion\_status\_lookup table.

## Automatic Advancement of Promotion Status

Demantra provides a tool to automatically advance the status of promotions based on their starting dates. The EXPOSE\_PROMOTIONS procedure iterates through the promotions listed in the promotion table, checks the status field of each, and does the following:

- If the current status is 3 (planned) and if the current date is after the from\_date of the promotion, change the status to 4 (committed).

- If the current status is 4 (committed) and if the current date is after the until\_date of the promotion, change the status to 5 (executed).

You should schedule this procedure to run periodically either within the Workflow Engine.

## Customizing the Promotion Status Behavior

Depending on how suitable the default behavior is, you have several options:

- Give a new name to each status, for example:

| Status ID | Possible Status Names |
|-----------|-----------------------|
| 1         | Unplanned             |
| 2         | Planned               |
| 3         | Committed             |
| 4         | Executed              |
| 5         | Closed                |
| 6         | Canceled              |

This system provides flexibility until promotions are committed.

- If you do not mind the Analytical Engine using all promotions, you can create your own status series and your own procedure to advance the status as needed.
- If you do want the Analytical Engine to ignore unplanned promotions but prefer to use different rules to control promotion editability, you can create a new status series that uses the same database field and update that field in the background. Also write your own procedure to advance the status as needed.

## Loading Promotion Data

To load promotions (which are general levels), create and use an integration interface that includes a level profile. Demantra does not provide a predefined interface for this, because promotions vary in nature.

To load promotional data (promotion-type series), create and use an integration interface that includes a data profile that includes the desired series.

See "Series and Level Integration".

## Reference: PE Levels

The CREATE\_PE\_STRUCT procedure adds the following levels to your database:



**Note:** Demantra provides the Promotion Effectiveness (PE) structures by default. Therefore, the CREATE\_PE\_STRUCT procedure should be run only after creating a new or custom data model from scratch (this is not common or recommended); only in this scenario would these structures not already be in place. To determine whether PE structures exist, search the DB for table names containing %PROMOTION%.

The following sections provide details on these levels:

- Promotion
- Promotion Type
- Scenarios
- Plans

### Promotion

This level contains the details for promotions. This is a general level with the following attributes:

| Attribute            | Column Name | Data Type | Purpose                                                            |
|----------------------|-------------|-----------|--------------------------------------------------------------------|
| Population Attribute | n/a         | n/a       | Allows you to associate the promotion with combinations and dates. |

| Attribute | Column Name | Data Type | Purpose                                                                                                                                                               |
|-----------|-------------|-----------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Status    | status      | Number    | ID of the status of this promotion.<br><br>This is a lookup attribute of type table; it uses the promotion_status_lookup table. See "Configuring Promotion Statuses". |

### Promotion Type

This level contains the default promotion types. This is a general level with no attributes. You can redefine this level as needed.

### Scenarios

This level groups the promotions. This is a general level with the following attributes:

| Attribute | Column Name   | Data Type | Purpose                                                                                                    |
|-----------|---------------|-----------|------------------------------------------------------------------------------------------------------------|
| Name      | SCENARIO_DESC | Character | Name of this scenario.                                                                                     |
| Plans     | PLAN_ID       | Number    | ID of the plan to which this scenario belongs.<br><br>This is a lookup attribute based on the Plans level. |

You can redefine this level as needed.

### Plans

This level groups the scenarios.

This is a general level with one attribute:

| Attribute | Column Name | Data Type | Purpose            |
|-----------|-------------|-----------|--------------------|
| Name      | PLAN_DESC   | Character | Name of this plan. |

You can redefine this level as needed.

## PTP Data Needs

Without going into details of the specific formats (given later), this section lists the data needs for PTP.

### Always Required

Every PTP system must have the following data:

- Basic sales data: the total quantity sold of each SKU at each ship-to location over the course of each week, and the regular retail price paid by the customer.
- The manufacturer's cost of goods (COG) for each item.
- The list price paid by the retailer for each item.
- Information on how each SKU fits into the item hierarchy and information on how each ship-to location fits into the location hierarchies.
- Basic data for historical or future promotions: for each promotion, the start and end dates, the items and locations to which the promotion applies, and the sale price per unit.
- Additional details for the promotions: promotion type, slotting costs, buydown allowance per item, and total vehicle costs.

### Nice to Have

By default, PTP assumes that additional data is also available, but you can reconfigure PTP to work if it is not.

- Syndicated data that includes the following breakdowns:
  - Base sales quantity (items sold if there had been no promotions)
  - Incremental sales quantity (additional items sold because of promotions)
  - Base sales dollars
  - Incremental sales dollars

**Note:** If this data is not available, see "Reconfiguring Series if Syndicated Data Is Not Used".

- Syndicated ACV data, which measures the number of stores that ran each kind of promotion, weighted by store size:

- % ACV ANY PROMO
- % ACV DISP
- % ACV FEAT
- % ACV FEAT&DISPLAY
- % ACV FREQSHOPPER
- % ACV TPR

**Note:** The ACV data is required if promotion data is unavailable.

### Purely Optional

Other data is purely optional:

- Additional details for promotions: consumer overlays, start and end shipping dates, and settlement payment type.
- Additional sales data: number of units shipped from the manufacturer to the retailer.

## Integration in PTP

To understand integration in PTP, it is useful to review how the Demantra platform handles integration.

### Data Loading and Integration in the Platform

Demantra provides the following tools for data loading and integration:

- The Data Model Wizard defines the basic levels and sales series in a data model and creates the `ep_load_sales` procedure, which handles data loading for those levels and series. The wizard also creates a batch script for running that procedure.

The Data Model Wizard does not load promotions or promotion data.

- The Integration Interface Wizard creates integration interfaces that can load promotions and promotion data. You execute the integration interfaces from within Workflow Manager or `aps.bat`.

In both cases, the wizards create staging tables, which usually serve as the starting point for data loading.

These tools are documented in the Part titled: "Basic Configuration".

## Data Loading and Integration in PTP

Because the PTP model is already defined, PTP provides an `ep_load_sales` procedure and integration interfaces that all work with the PTP model. It is not necessary to use the Data Model Wizard or the Integration Interface Wizard. The required staging tables already exist as well.

To facilitate data loading, PTP offers two options.

- Population of staging tables into database, or
- Population of comma delimited files.

Based on the decision made, review the Data Model Wizard as well as Integration Interface Wizard to verify that current settings match selections. Default settings for the Data Model Wizard are for loading through database tables. Promotional information loaded through integration profiles is set as a default for text files.

## Reference: Staging Tables

Internally, PTP uses the following staging tables:

- `BIIO_IMP_PROMO_DATA`
- `BIIO_Promotion`
- `BIIO_Population`

The information here is provided for reference and debugging purposes.

### **BIIO\_IMP\_PROMO\_DATA**

This staging table is used by the Import Promotion Data2 integration interface and has the following structure:

| Field         | Data Type     | Required? | Purpose*                                                                      |
|---------------|---------------|-----------|-------------------------------------------------------------------------------|
| Sdate         | date          |           | Sales date.                                                                   |
| LEVEL1        | varchar2(500) |           | Code of the promotion member to which this data applies.                      |
| PROMO_PRICE   | number(20,10) |           | Sale price per unit at shelf. Used by the Sale Price series.                  |
| SLOTING_SPEND | number(20,10) |           | Planned spend associated with slotting costs. Used by the Slotting \$ series. |

| Field              | Data Type     | Required? | Purpose*                                                                                                                                                                           |
|--------------------|---------------|-----------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| CASE_BUYDOWN       | number(20,10) |           | Buydown allowance per unit, for this promotion. Can be used for off-invoice deductions or billback. Used by the Buydown series.                                                    |
| VEHICLE_SPEND      | number(20,10) |           | Total planned vehicle costs for this promotion. Used by the Loading Veh \$ series.                                                                                                 |
| ALLOWANCE_TY<br>PE | varchar2(50)  |           | Description of the pay type of this promotion, as listed in the pay_type_lookup table. Use the pay_type_desc field, rather than the code or ID field. Used by the Pay Type series. |
| IS_SELF            | number(20,10) |           | Specify as "1" for all records. Used for internal purposes.                                                                                                                        |

\* For details on these series, see "Series".

### BIIO\_Promotion

This staging table is used by the Import Promotion Levels integration interface. It contains all the promotion attributes, of which only a subset are typically imported. This table has the following structure:

| Field                       | Data Type      | Required? | Purpose*                                                                      |
|-----------------------------|----------------|-----------|-------------------------------------------------------------------------------|
| PROMOTION_CODE              | varchar2(500)  | Required  | Unique code for the promotion, for use in Demantra.                           |
| PROMOTION_DESC              | varchar2(2000) | Required  | Description or name of the promotion, for use in Demantra.                    |
| PROMO_INTEG_STATUS<br>_CODE | varchar2(120)  |           | Used for integration to determine whether promotion details have been changed |



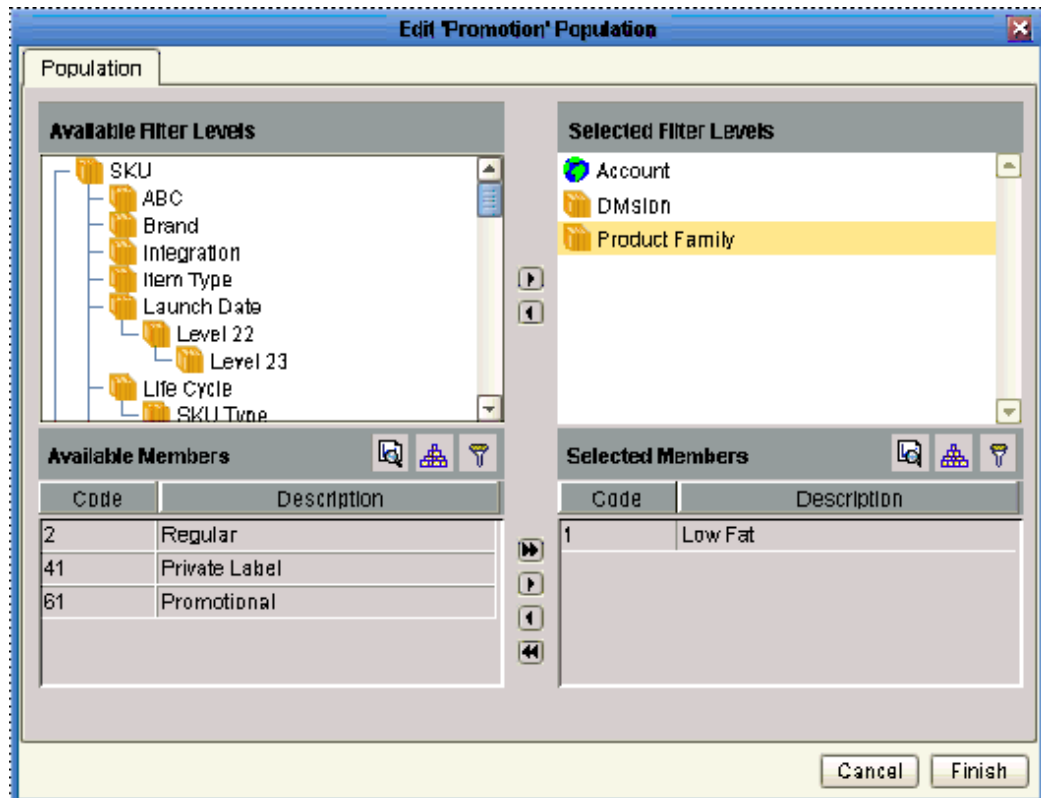
| Field                  | Data Type      | Required? | Purpose*                                                                                           |
|------------------------|----------------|-----------|----------------------------------------------------------------------------------------------------|
| SCENARIO_CODE          | varchar2(500)  | Required  | Unique code for the scenario to which this promotion belongs, as listed in the SCENARIO table.     |
| PROMOTION_TYPE_CODE    | varchar2(500)  | Required  | Unique code for the promotion type, as listed in the PROMOTION_TYPE table.                         |
| PROMOTION_STAT_CODE    | varchar2(120)  | Required  | Unique code for the promotion status, as listed in the promotion_stat table.                       |
| BUY_DOWN               | number(20,10)  |           | Populates the Buydown attribute.                                                                   |
| OPTIMIZATION_GOAL      | varchar2(2000) |           | Populates the Optimization Goal attribute.                                                         |
| OPTIMAL_PRICE_DECREASE | number(20,10)  |           | Populates the Optimal Price Decrease attribute.                                                    |
| STATUS                 | varchar2(30)   |           | Populates the Status attribute.                                                                    |
| APPROVAL               | varchar2(200)  |           | Populates the Approval attribute.                                                                  |
| SHIP_DATE              | date           |           | Populates the Start Ship attribute with the date on which shipments will start for this promotion. |
| CONS_PROMO             | varchar2(50)   |           | Populates the Cons Promo attribute, which indicates any associated consumer overlay.               |
| OPTI_PROMOTION_TYPE_ID | varchar2(2000) |           | Populates the Optimal Type attribute.                                                              |
| MAX_BUDGET             | number(20,10)  |           | Populates the Max Budget attribute. Maximum allowed budget for this promotion.                     |

| <b>Field</b>                | <b>Data Type</b> | <b>Required?</b> | <b>Purpose*</b>                                     |
|-----------------------------|------------------|------------------|-----------------------------------------------------|
| SPEND                       | number(20,10)    |                  | Populates the Optimal Budget attribute.             |
| ROI                         | number(20,10)    |                  | Populates the Return on Investment (ROI) attribute. |
| PROFIT                      | number(20,10)    |                  | Populates the Optimal Profit attribute.             |
| TOTAL_LIFT_U                | number(20,10)    |                  | Populates the Optimal Lift attribute.               |
| TOTAL_LIFT_D                | number(20,10)    |                  | Populates the Optimal Revenue attribute.            |
| PROMOTION_TYPE_ID1          | varchar2(2000)   |                  | Populates the Promotion Type1 attribute.            |
| STATUS_ID1                  | varchar2(30)     |                  | Populates the Promotion Status attribute.           |
| END_SHIP                    | date             |                  | Populates the End Ship attribute.                   |
| METHOD_STATUS               | varchar2(200)    |                  | Populates the method_status attribute.              |
| OPTIMIZATION_STATU<br>S     | varchar2(50)     |                  | Populates the Optimization Status attribute.        |
| PROMOTION_BUDGET            | number(20,10)    |                  | Promotional budget updated by DSM                   |
| MIN_RTL_MARGIN_OV<br>ERRIDE | number(20,10)    |                  | Populates the Min Rtl Margin Override attribute.    |
| FIXED_BUYDOWN_YN            | varchar2(100)    |                  | Populates the Fixed Buydown attribute.              |
| MAX_BUYDOWN                 | number(20,10)    |                  | Populates the Max Buydown attribute.                |

| Field                                               | Data Type     | Required? | Purpose*                                             |
|-----------------------------------------------------|---------------|-----------|------------------------------------------------------|
| OPTIMIZATION_RANGE<br>_START                        | date          |           | Populates the Optimization<br>Range Start attribute. |
| OPTIMIZATION_RANGE<br>_END                          | date          |           | Populates the Optimization<br>Range End attribute.   |
| OPTIMIZATION_COUNT                                  | number(20,10) |           | For internal use only.                               |
| * For details on these attributes, see "Promotion". |               |           |                                                      |

### **BIIO\_Population**

This staging table is used by the Import Promotion Levels integration interface. It describes the population of each promotion. Specifically, it contains the same information as this window:



For each promotion, the table can contain multiple rows. Each row specifies a level and a member of that level, just as the preceding screen does (the previous screen shows that this promotion is associated with the Low Fat member of the Product Family). This table has the following structure:

| Field        | Data Type    | Required? | Purpose                                                          |
|--------------|--------------|-----------|------------------------------------------------------------------|
| LEVEL_MEMBER | varchar2(40) | Required  | Code of the promotion that you are loading.                      |
| FROM_date    | date         | Required  | Start date for this promotion.                                   |
| UNTIL_date   | date         | Required  | End date for this promotion.                                     |
| FILTER_LEVEL | varchar2(50) | Required  | Name of a level, for example "Product Family" or "SKU".          |
| LEVEL_ORDER  | number(15)   | Required  | Use "1" for a location-type level or "2" for an item-type level. |

| Field         | Data Type    | Required? | Purpose                                                       |
|---------------|--------------|-----------|---------------------------------------------------------------|
| FILTER_MEMBER | varchar2(50) | Required  | Description of a member of this level, for example "Low Fat". |

## Reference: Integration Interfaces

The information here is provided for reference and debugging purposes.

Oracle Demantra provides the following integration interfaces.

### Import Promotion Levels

This integration interface is defined as follows:

|                        |                                                                                                                                                                                                                                                                                                                                      |
|------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Type:</b>           | Import                                                                                                                                                                                                                                                                                                                               |
| <b>Description:</b>    | Imports rows from a staging table and adds the new members to the Promotion level. If the new promotions refer to combinations that are not yet present in this database, this interface creates those combinations as well.<br><br>Also imports rows from another staging table, which contains the population of these promotions. |
| <b>Staging Tables:</b> | BIIO_Promotion stores the promotion members. Edit this table before editing BIIO_Population.<br><br>BIIO_Population stores the populations of the promotions.                                                                                                                                                                        |

### Import Promotion Data2

This integration interface is defined as follows:

|                     |                                                                                                             |
|---------------------|-------------------------------------------------------------------------------------------------------------|
| <b>Type:</b>        | Import                                                                                                      |
| <b>Description:</b> | Imports rows from a staging table and updates the promotion series data in the appropriate internal tables. |

## Configuring the Default Promotion Start and Span

### To configure the default promotion start and span:

1. In the Business Modeler, click Parameters > System Parameters.
2. Click the Worksheet tab.
3. Edit the following parameters:

|                   |                                                                                                                                                   |
|-------------------|---------------------------------------------------------------------------------------------------------------------------------------------------|
| PromoDefaultStart | Specifies the default start date for promotions created within a worksheet: the current date, the last loaded sales date, or the worksheet start. |
| PromoDefaultSpan  | Specifies the default length of time for promotions created within a worksheet.                                                                   |

4. Click Save.
5. Click Close.

## Changing the Default Retailer Profile

### Predefined Behavior

For any given retailer, the user can specify attributes or can instead use the default retailer profile. The CopyRetailerDefaults workflow checks for any retailers that use the default profile, and it copies the default details to those retailers. You should run this workflow each time you change a retailer to use the default profile and each time you change the default profile.

### Possible Changes

You can change the default retailer profile, as follows:

1. In the Business Modeler, click Configuration > Configure Levels.
2. Right-click the Retailer level and select Open > General Attributes.

Business Modeler displays the default attributes for the retailer level. Together, these constitute the "default retailer profile."

3. Click a retailer attribute on the left.
4. In the right area, change Default Value.
5. Continue with other attributes as needed.
6. When you are done, click Next and then click Finish.
7. Restart the Application Server to make the changes available to the users.

## Configuring the Right-Click "Open With" Option

By default, when users right-click a promotion, scenario, or retailer within PTP, they can use the Open and Open With menu options to open a worksheet that is filtered to that selection (the Open option opens the default worksheet). This is configurable within the Business Modeler.

### To configure the "Open With" menu option:

1. In the Business Modeler, click Components > Create/Open Component. Or click the Create/Open Component button.
2. Click Oracle PTP, and then click OK.
3. Click Next repeatedly until the Select Component Queries for Levels screen is displayed.

This screen allows you to associate public worksheets with levels.

**Select Level:** City

**Available Queries:**

- 000. Introduction
- 0008. Middle out Enterprise Plan
- 001. Store Plan
- 002. Creating a strategic plan
- 004. New Product Launch
- 005. Analyze historical Statistical perform
- 006. Clustering Report update
- 007. Analyze Historical Plan performance
- 008. Middle out Enterprise Plan
- 009. Store Manager Plan
- 010. Consensus Plan
- 011. Consensus Plan Cross tab analysis
- 012. Budget Exception Analysis
- 013. Lifecycle Management
- 014. Wall Management
- 015. Coupon Program

**Selected Queries:**

**Default Query:**

Total Available Queries: 39      Total Selected Queries: 0

This association is used in two ways:

- Within the Members Browser, a user can use the right-click menu to open any of these associated worksheets directly from a member of the level (via the Open With menu option). In this case, Demantra opens the associated worksheet. The worksheet is filtered to show only data relevant to the member.
- Within the worksheet designer, users can add a sub tab to a worksheet; the sub tab shows details for a given member. The sub tab can display any of the worksheets that are associated with a level included in the main worksheet. The sub tab is filtered to show only data relevant to the member.

4. To associate a worksheet with a level, do the following:
  1. Click the level in the Select Level drop down menu.
  2. Double-click the worksheet in Available Queries list, which moves it to the Selected Queries list.
  3. Move other worksheets from the Available Queries list to the Selected Queries list, as needed.
  4. Decide which worksheet in the Selected Queries list should be the default worksheet for this level. For that worksheet, click the Default check box. When the user right-clicks and selects Open, this is the worksheet that will be used.
5. When you are done on this screen, click OK.



## Removing Right-Click Menu Options

The options on the right-click menu are Oracle methods. You can remove these options if needed; for example, if your system is not using optimization, you might want to remove the optimization options.

### To modify a level method:

1. In the Business Modeler, click Configuration > Configure Methods.  
The system displays a screen showing the existing methods, including all the predefined methods.
2. Optionally click a level name (such as Promotion, Scenario, or Retailer) in the drop-down list at the top of the screen.  
The screen is re-displayed with only the methods associated with that level.
3. To hide this menu option, deselect the Display in menu check box.
4. Click Finish.

## Replacing Collaborator Workbench Graphics

The Web-based Demantra products contain default images that you can replace with your organization's own designs. To do so, just back up the default images and substitute your own image files, giving them the same filenames as listed here.

The graphic files are in the following directory:

Demantra\_root/Collaborator/portal/images

You can replace any of the graphics files in this directory. If you replace the default graphics with other graphics that have the same width and height, those graphics will fit on the page without the need for any further editing.

### Collaborator Workbench Splash Screen

The splash screen uses the graphic collaborator\_splash.gif.

### Collaborator Workbench Login Page

On the login page, the most commonly replaced images are the following:

On the login page, the most commonly replaced images are the following:

- customerLogo.gif
- customerTxt.gif
- customerPics.gif

- collaboratorTitle.gif
- collaboratorTxt.gif



### Collaborator Workbench Main Page

### Collaborator Workbench Main Page

On the main page, the most commonly replaced images are as follows:

- customer\_logo.jpg
- collaborator\_logo.gif



## Configuring Promotion Status Behavior

You can redefine the behavior of the promotion statuses in PTP.

**Caution:** You should not do this unless you are familiar with series definitions and with the Business Modeler.

### Predefined Behavior

Demantra provides predefined promotion statuses (the Status attribute) and behavior,

which PTP uses indirectly. PTP internally uses the Demantra predefined statuses but instead displays its own set of statuses (the Evt Status series).

Specifically the Status series maps the PTP statuses to the internal statuses. To do this mapping, this series has a client expression as follows:

if ( (evt status = 1), 1, if ( (evt status < 7), 2, 6))

This expression checks the value of the Evt Status series and decides the internal status value to which it maps. The resulting value is saved to the update field for the Status series, which is promotion.status. This is the field that controls whether a promotion is editable.

The following table lists the PTP status, the internal hardcoded statuses and their behavior, and explains the added TPMO behavior:

| <b>Evt Status series</b> | <b>Promotion.Status field</b> | <b>Hardcoded behavior</b>                 | <b>Additional Trade Promotions behavior</b>                                                                                                                           |
|--------------------------|-------------------------------|-------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1 (a. Unplanned)         | 1 (Unplanned)                 | Analytical Engine ignores this promotion. | Promotion does not affect fund balances or the forecast.                                                                                                              |
| 2 (b. Planned)           | 2 (Cancelled)                 | .                                         | The cost of the promotion is deducted from the available balance of funds.                                                                                            |
| 3 (c. Approved)          |                               | .                                         |                                                                                                                                                                       |
| 4 (d. Committed)         |                               | .                                         | Via a PTP procedure, PTP takes a snapshot of the current state of the promotion, for use in later analysis. The committed promotions are included in all projections. |
| 5 (e. Partial Paid)      |                               | .                                         | .                                                                                                                                                                     |
| 6 (f. Paid)              |                               | .                                         | .                                                                                                                                                                     |

| <b>Evt Status series</b> | <b>Promotion.Status field</b>                    | <b>Hardcoded behavior</b>                        | <b>Additional Trade Promotions behavior</b> |
|--------------------------|--------------------------------------------------|--------------------------------------------------|---------------------------------------------|
|                          | 3 (Planned)                                      | See "Automatic Advancement of Promotion Status". | .                                           |
|                          | 4 (Committed)                                    |                                                  |                                             |
|                          | 5 (Executed)                                     |                                                  |                                             |
|                          | PTP does not use these hard coded status values. |                                                  |                                             |
| 7 (g. Closed)            | 6 (Closed)                                       | User cannot edit the promotion.                  | .                                           |

## Notes

- Because of the way that PTP handles status, users must save worksheet data and rerun the worksheet whenever they change the status of a promotion. Otherwise, the status change is not reflected in the worksheet series.
- PTP currently includes extraneous status attributes (Event Status and Promotion Status), which you should ignore.
- You should not redefine series unless you are familiar with series definitions and with the Business Modeler.

## Possible Changes

- You can change the drop-down choices of the Evt Status series.
- You can change the client expression of the Evt Status series, to map the statuses differently to the hardcoded statuses.

## Re-configuring Series if Syndicated Data Is Not Used

Demantra uses syndicated data that includes the following breakdowns:

- Base sales quantity (items sold if there had been no promotions)
- Incremental sales quantity (additional items sold because of promotions)
- Base sales dollars

- Incremental sales dollars

If this data is not available, it is necessary to reconfigure some PTP series. You may also need to modify some PTP procedures and triggers.

**Caution:** You should not do this unless you are thoroughly familiar with the Demantra platform.

## Series to Reconfigure

The following series directly use the syndicated breakdowns via client expressions:

- Base Evt \$ Rtl
- Incr Evt Vol
- Incr Evt \$ Rtl

Each of these series has a client expression of the following form:

if past end date = 1, use syndicated data; otherwise, use engine data

(For each promotion, the **Past End Date** series equals 1 if the promotion is past or equals 0 otherwise.) For example, the client expression for **Base Evt \$ Rtl** is as follows:

```
if(past end date = 0,
if(isnull(Base Evt $ Rtl Fut),0, Base Evt $ Rtl Fut),
if(isnull(Base Evt $ Rtl Act),0, Base Evt $ Rtl Act))
```

The **Base Evt \$ Rtl Act** contains the syndicated data.

You can reconfigure these series by rewriting this client expression so that the series always uses engine data. Or you can modify the actuals series to contain different data, depending on your needs.

Also, several series refer directly to the syndicated incremental volume via server expressions. These include BDF Exp Ttl Act, Incr COGS \$ Act, Incr Evt \$ Act, and MDF Exp Ttl Act.

## Data Synchronization

If you reconfigure any of these series, you should understand how PTP maintains the data for these series. PTP loads this data into sales\_data. For performance reasons, this information is needed in the promotion\_data table instead, so PTP uses procedures and triggers to copy this data to that table. The following section gives the details.

## Syndicated Data in PTP

The following table summarizes where the syndicated data is loaded, where that data is synchronized, and what series use this data in the sales\_data and promotion\_data tables, respectively. For completeness, this table lists all the syndicated data, including required data and data that is not synchronized.

| Information                           | In the sales_data table |                    | In the promotion_data table |                     |
|---------------------------------------|-------------------------|--------------------|-----------------------------|---------------------|
|                                       | Field*                  | Series             | Field**                     | Series              |
| Average retail price, always required | item_price              | Avg Rtl sd         | avg_rtl_pd                  | Avg Rtl             |
| Cost of goods, always required        | sdata7                  | COGS sd            | cogs_pd                     | COGS                |
| List price, always required           | sdata8                  | List Price sd      | list_price_pd               | List Price          |
| Shelf price, always required          | shelf_price_sd          | Shelf Price sd     | ed_price                    | Shelf Price         |
| Total volume, always required         | actual_quantity         | Actuals Ttl        |                             |                     |
| Base volume                           | sdata5                  | Actuals Base       |                             |                     |
|                                       | volume_base_ttl***      | Many series        | volume_base_ttl             | Many series         |
| Base dollars                          | BASE_EVT_DOL_RTL        | Base Evt \$ Rtl sd | base_evt_d_rtl_act          | Base Evt \$ Rtl Act |
| Incremental volume                    | sdata6                  | Actuals Incr       | incr_evt_vol_act            | Incr Evt Vol Act    |
| Incremental dollars                   | INCR_EVT_DOL_RTL        | Incr Evt \$ Rtl sd | incr_evt_d_rtl_act          | Incr Evt \$ Rtl Act |
| % ACV ANY PROMO                       | sdata9                  | % ACV ANY PROMO    |                             |                     |
| % ACV DISP                            | sdata10                 | % ACV DISP         |                             |                     |

| Information                                                                 | In the sales_data table |                       | In the promotion_data table |        |
|-----------------------------------------------------------------------------|-------------------------|-----------------------|-----------------------------|--------|
|                                                                             | Field*                  | Series                | Field**                     | Series |
| %ACV FEAT                                                                   | sdata11                 | % ACV FEAT            |                             |        |
| %ACV<br>FEAT&DISPLAY                                                        | sdata12                 | % ACV<br>FEAT&DISPLAY |                             |        |
| %ACV TPR                                                                    | sdata13                 | % ACV TPR             |                             |        |
| %ACV<br>FREQSHOPPER                                                         | sdata14                 | % ACV<br>FREQSHOPPER  |                             |        |
| *These fields in sales_data are loaded.                                     |                         |                       |                             |        |
| **These fields in promotion_data are maintained by procedures and triggers. |                         |                       |                             |        |
| ***This field in sales_data is maintained by procedures and triggers.       |                         |                       |                             |        |

For information on procedures and triggers, see "Procedures".

## Parameters Affecting Goal Function

### Determine Optimization Focus (GOAL\_FOCUS)

- 0=Manufacturer
- 1=Retailer

### Determine Optimization Goal (OPTIMIZATION\_GOAL)

- 0=Revenue
- 1=Profit
- 2=Units

Choosing between these three goal functions will affect the formulation of the expression that the optimization is maximizing. If Units is chosen, optimization will look for valid promotions which will generate the most lift while not exceeding constraints such as maximum budget and minimum margins.

If Revenue is chosen, the expression will incorporate the price paid for each unit sold. The price used will depend on whether the optimization is focused on a retailer or manufacturer.

When choosing Profit the optimized expression will include both incoming revenues and costs associated with the product and promotion.

**Determine Buydown Behavior**

- Fixed
- Calculated



---

## Configuring Promotion Optimization for Predictive Trade Planning

This chapter describes how to configure the Promotion Optimization module. You can skip this chapter until you need to work with Promotion Optimization.

This chapter covers the following topics:

- Overview of the Configuration Process
- Set Up Promotion Optimization Without Using the Installer
- Configuring the Optimization Step
- Other Important Notes

### Overview of the Configuration Process

To configure Promotion Optimization, you use the following general process:

1. First do one of the following:
  - Run the Demantra installer, choosing the option to install Promotion Optimization.
  - Or manually do the same work that the installer does; see "Setting Up PO Without Using the Installer".
2. Then see "Configuring the Optimization Step".
3. Before using Promotion Optimization for the first time, run the Analytical Engine in batch mode so that your machine has access to the engine's cached results. See "Using the Engine Administrator and Running the Engine".

## Set Up Promotion Optimization Without Using the Installer

If you did not use the Demantra installer to set up Promotion Optimization, complete the steps in this section.

### Setting Environment Variables

Promotion Optimization requires the following Windows environment variable be set:

---

|      |                                                                                                                                                                                       |
|------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| PATH | Must include the directory Demantra_root\Collaborator\ <i>virtual_directory</i> \optimization\dll<br><br>For example: F:\Demantra<br>Spectrum\Collaborator\demantara\optimization\dll |
|------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

---

### Registering the Analytical Engine

The installer registers the engine. If you do not use the installer, you may need to register the engine manually.

To do so, double-click the batch file Demantra\_root/Demand Planner/Analytical Engines/bin\RegEngine.bat.

## Configuring the Optimization Step

For Promotion Optimization, the primary configurable element is the Optimization step. Because this requires details about your installation location, you must configure this step manually.

### To configure the optimization step

1. Browse to the following case-sensitive URL:  
`http://server name/virtual directory/workflow/login.jsp`  
A login page appears.
2. Enter the user name *dp* with the password *dp*, and then click Log on.  
**Note:** Only the user of this workflow (*dp*) can make changes to it.
3. In the row for the Call Promotion Optimizer workflow, click Edit.
4. Right-click the Call Promotion Optimizer step, and then choose Properties.
5. In the Parameters section, review the following parameters in case any changes are

necessary. Typically, these parameters do not require modification. This step is required in order to make optimization work in your environment.

| Parameter Name | Purpose                                                                                                                                                                                                                                                                                             |
|----------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| MODEL_PATH     | Complete path and filename of the <i>promoopt.opl</i> file.                                                                                                                                                                                                                                         |
| dbms_type      | The database type, one of the following:<br><br>odbc<br><br>oracle81<br><br>The database type information is automatically received from the application server.                                                                                                                                    |
| dbms_connect   | Database connection information in the form database_user/database_password@databasename . For example: demantra/d@alexish<br><br>The dbms_connect parameter is automatically received from the application server but modifications may be required if the servername and dbname are not the same. |
| JAR_FILE       | Relative Path of OPLtoJava.jar file.                                                                                                                                                                                                                                                                |
| DRIVER_JAR     | Relative Path of ojdbc14.jar file                                                                                                                                                                                                                                                                   |

6. Optionally edit the following parameters.

**Caution:** Do not edit parameters that are not listed in this chapter.

| Parameter Name  | Purpose                                                                                  |
|-----------------|------------------------------------------------------------------------------------------|
| opti_level_item | Name of the item level for optimization. This is case-sensitive.<br><br>For example: SKU |

| Parameter Name         | Purpose                                                                                                                                                                                                                                                   |
|------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| opti_level_location    | <p>Name of the location level for optimization. This is case-sensitive.</p> <p>For example: Ship to</p>                                                                                                                                                   |
| opti_level_promo       | <p>Name of the promotion level for optimization. This is case-sensitive.</p> <p>For example: Promotion</p>                                                                                                                                                |
| promo_max_budget_attr  | Name of the field in the Promotion table that stores the maximum budget.                                                                                                                                                                                  |
| promo_used_budget_attr | Name of the field in the Promotion table to which the method writes the optimized budget.                                                                                                                                                                 |
| opti_value_steps       | Number of variations of the causal factors that the optimizer should try. Use a value from 50 to 500.                                                                                                                                                     |
| MAX_PROMO_ON_PROD_ACC  | Maximum number of concurrent promotions on a given account. Suggested value: 1.                                                                                                                                                                           |
| MAX_LENGTH_OF_PROMO    | Maximum permitted length of any promotion, measured in base time buckets.                                                                                                                                                                                 |
| MIN_RET_MARGIN         | <p>SQL expression that returns the margin that the retailer requires. Do not set equal to 0.</p> <p>Promotion Optimization computes the retailer margin as follows:</p> $(\text{sale price} + \text{buydown}) / \text{list price} - 1$                    |
| MIN_MAN_MARGIN         | <p>SQL expression that returns the minimum margin that the manufacturer requires. Do not set equal to 0.</p> <p>Promotion Optimization computes the manufacturer margin as follows:</p> $(\text{list price} - \text{buydown}) / \text{cost of goods} - 1$ |
| MAX_BUY_DOWN           | Maximum permitted buydown. Use a very large number such as 100000.                                                                                                                                                                                        |

| Parameter Name                | Purpose                                                                                                                                     |
|-------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------|
| RET_CONSUMER_PRICE_EXPRESSION | SQL expression that returns the everyday price seen by the consumer.                                                                        |
| MAN_LIST_PRICE_EXPRESSION     | SQL expression that returns the list price seen by the retailer.                                                                            |
| MAN_COGS_EXPRESSION           | SQL expression that returns the cost of goods to the manufacturer.                                                                          |
| MAN_VEHICLE_COST_EXPRESSION   | SQL expression that returns any fixed costs associated with running the promotion.                                                          |
| FIXED_BUYDOWN_YN              | Specifies whether to use buydown as an input (1) or to calculate the optimal buydown (0). Use 1.                                            |
| BUYDOWN_EXPRESSION            | SQL expression that returns the buydown per unit.                                                                                           |
| MIN_RET_REVENUE               | Minimum retailer revenue for any promotion.                                                                                                 |
| MIN_RET_DEMAND                | Minimum retailer demand (unit count) for any promotion.                                                                                     |
| MIN_MAN_PROFIT                | Minimum manufacturer profit for any promotion.                                                                                              |
| MIN_RET_INC_PROFIT            | Minimum retailer incremental profit for any promotion.                                                                                      |
| MIN_RET_INC_REVENUE           | Minimum retailer incremental revenue for any promotion.                                                                                     |
| GOAL_FOCUS                    | Specifies if Promotion Optimization should optimize from the manufacturer's perspective (0; recommended) or the retailer's perspective (1). |

| Parameter Name           | Purpose                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|--------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| COEFFICIENT_RANGE_FACTOR | <p>Controls the range of possible coefficient values to be searched.</p> <p>For any given causal factor, Promotion Optimization by default tests a discrete set of coefficient values, ranging from 0 to the largest value observed in history. This parameter specifies the additional percentage to add to that range of values.</p> <p>For example, to allow Promotion Optimization to search values 20% larger than historically seen, set this parameter to 0.20.</p> |
| DEFAULT_RET_MARGIN       | SQL expression that returns the default retailer margin.                                                                                                                                                                                                                                                                                                                                                                                                                   |
| DEFAULT_MAN_MARGIN       | SQL expression that returns the default manufacturer margin.                                                                                                                                                                                                                                                                                                                                                                                                               |
| RETAILER_LEVEL_ID        | ID of the level (from group_tables) that corresponds to the retailers.                                                                                                                                                                                                                                                                                                                                                                                                     |
| OPTI_BUYDOWN_COLUMN      | Defines which column the optimized buydown is written to. Refers to a column on the promotion_data table.                                                                                                                                                                                                                                                                                                                                                                  |
| OPTIMIZATION_RANGE_START | SQL expression that returns the start of the date range which is considered by optimization. Expression references promotion table.                                                                                                                                                                                                                                                                                                                                        |
| OPTIMIZATION_RANGE_END   | SQL expression that returns the end of the date range which is considered by optimization. Expression references promotion table.                                                                                                                                                                                                                                                                                                                                          |
| BUYDOWN_EXPRESSION       | SQL expression that returns the actual buydown for promotion being optimized. Expression references promotion_data table. As a default, it incorporates bill back, off invoice and scan values.                                                                                                                                                                                                                                                                            |

## Other Important Notes

The Promotion Optimization module uses certain fields in the Promotion and

promotion\_data tables; do not change the names of any fields.

Also, the Promotion Optimization methods do not check for all the required inputs. In particular, you must make sure that the following information is available before running these methods:

- List price
- Buydown

#### **Additional information regarding degrees of freedom**

The actual possible solutions the optimization engine will evaluate depend on the status of the promotion being optimized. The internal ID of the promotion's status will be compared with the system parameter PromotionOptiStatusThreshold.

- If Promotion Status < PromotionOptiStatusThreshold, the optimizer can modify the promotion's timing, promotion type and values of promotional causals.
- If Promotion Status = PromotionOptiStatusThreshold, the optimizer can modify the promotion type and values of promotional causals.
- If Promotion Status > PromotionOptiStatusThreshold, the optimizer can modify the values of promotional causals only.

Since the comparison is based on whether the status is above or below the parameter, it is strongly recommended that promotion statuses be ordered with increasing internal IDs from least fixed to most fixed promotions.

**Note:** The optimization engine only evaluates the direct effects generated by causal factors. Therefore, optimization results are focused on providing an optimized outcome for the direct effect of the promotion and not the indirect affect. The indirect effects of the promotion can be viewed by accepting the optimized promotion and then executing a simulation.





---

## Configuring Deductions and Settlement Management

This chapter describes how to configure DSM and load an initial set of data.

This chapter covers the following topics:

- Overview of DSM
- Data Flow in DSM
- Overview of the Configuration Process
- Setting Up Database Structures
- Configuring Promotions and Promotion Series
- Identifying Key Promotion Series
- Configuring Settlement Matching
- Configuring Write-Offs
- Loading Initial Data and Creating Possible Matches
- Describing Customer Relationships
- In Case of Problems
- Reference: DSM Levels
- Reference: DSM Series
- Reference: DSM Integration Interfaces
- Reference: Other DSM Tables

### Overview of DSM

Oracle Demantra Deduction and Settlement Management (DSM) is a Web-based, configurable tool to help users at a manufacturing company resolve settlements with customers (usually retailers) who have run promotions, sold products, and now need

reconciliation. Users view the promotional events that DSM provides as possible matches, and then select one and finalize the match. Users can then attach proof of performance for the promotion, approve the match, and request a check to be sent to the customer (if appropriate). Users can also mark a settlement as a duplicate, split a settlement (typically to match only part of it), or deny a settlement.

Often, a third party (a broker) has negotiated the terms. Demantra system may be set up to enable users to collaborate with outside brokers, for example, to acquire extra information if needed.

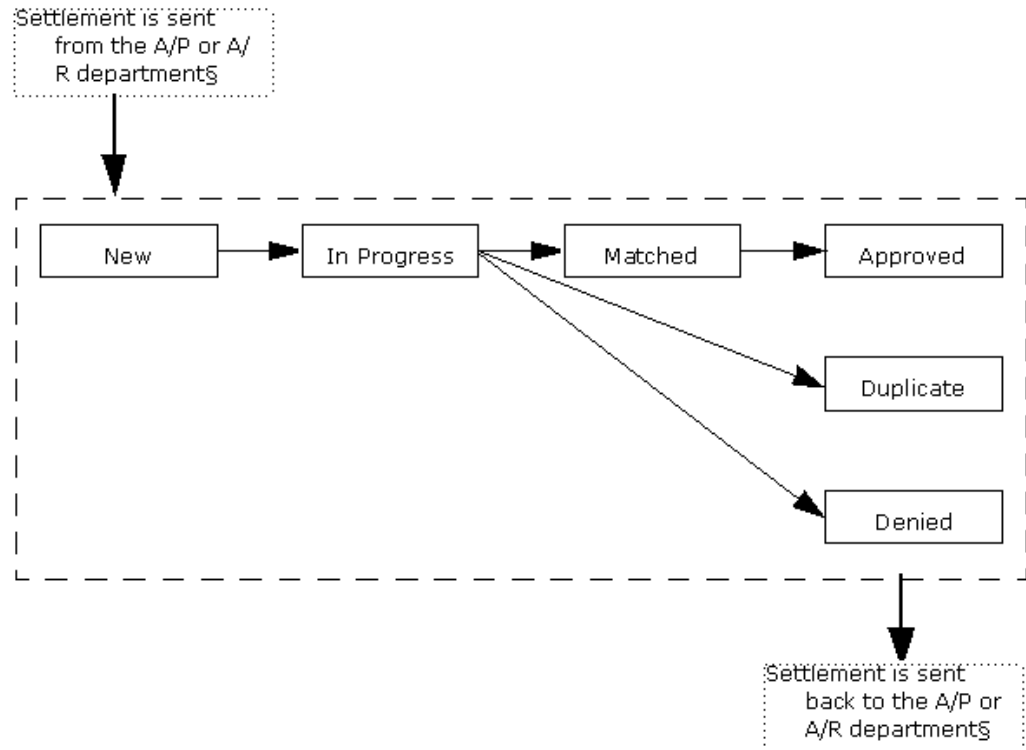
### **Types of Settlements**

DSM organizes settlements into two groups: trade and non-trade. For trade settlements, DSM recognizes three general types of settlements:

- A claim is a request from a customer for payment. In these cases, the customer has run the promotion and is requesting to be reimbursed, based on an agreement with the customer. If a user approves the claim, DSM sends a request to the A/P department to send a check to this customer or to the broker, as applicable.
- A deduction is a short payment on an invoice. In these cases, the customer has run the promotion and has made a short payment on an invoice. By permitting this short payment, the user is reimbursing the customer for running the promotion.
- An off-invoice settlement represents the case where the customer was billed a lower amount (that is, "off invoice") for the products, as compensation for running the promotion. In this kind of settlement, the settlement amount was negotiated ahead of time.

### **User Actions and Settlement Stages**

Within DSM, a settlement can go through the following stages:



In general, claims come into DSM through the accounts payable (A/P) department. Deductions and off-invoice settlements come into DSM through in the accounts receivable (A/R) department. In all cases, the settlement is loaded into DSM automatically.

Then, in most cases, the processing is as follows:

1. When a user receives a new settlement, he or she takes ownership of it. At this point, the settlement is In Progress.
2. Then the user reviews the possible promotions to which the settlement may apply. Oracle Demantra Deductions and Settlement Management displays suitable possible matches.
3. If the user finds a matching promotion, he or she matches the settlement and validate the proof of performance. Usually, the user scans in a document that shows that the promotion was actually run as required, and then uploads that as an attachment to the settlement.
4. The user then approves the settlement. In some organizations, the customer service department (CSD) representative has authority to approve a settlement. In others, a manager does that job.
5. If the settlement is a claim, the user issues a check request to your company's A/P

department, to send a check to the customer or to the broker as needed.

**Note:** Within user security, the user must have New Check Request and Edit Settlement methods enabled for the user to link to or enter a check request.

In other cases, the user may find that the settlement is a duplicate or you may find another reason to deny it, per the company's policies and practices. When the user denies a settlement, the A/P system may enter a chargeback to the customer.

Also, a user can also split the settlement and match only part of it, as appropriate.

### **Customers and Check Recipients**

If a user issues a check request, it is necessary to determine the recipient of that check. The matched event is associated with a particular customer, but the check itself might need to go to a different recipient, depending on the customer's business relationships.

For example, suppose the matched event was associated with Acme Foods. Acme Foods may have created the event with the help of a broker, and it might be appropriate to make the check payable to Acme but send it to the broker, who hand-delivers it as a customer-service gesture.

For another example, the manufacturer might receive an invoice from an indirect customer, who funded and planned the event with Acme (who then ran the event). In this case, it would be suitable to make the check payable to and send it to the indirect customer.

As a result of all this complexity, when an event is matched to a given location A, it is necessary to consider all the other locations that are related to location A in various ways. The user will view these locations and direct the check appropriately.

## **Data Flow in DSM**

DSM works with external systems such as accounts payable and accounts receivable. Ultimately, those systems own most the data; DSM is responsible only for matching settlements to promotions and performing the associated record keeping.

At a high level, the overall flow of data is as follows:

1. Within DSM, an automated workflow imports settlement data from external corporate systems.
2. The same automated workflow then runs database procedures that iterate through the settlements and identify possible matches for each. The procedures write the results to an internal table for later use. Optionally, another database procedure iterates through the settlements and writes off any that fall below a specified threshold.

3. A typical user, who is responsible for a subset of the locations (possibly an account), opens a DSM worksheet and displays settlements associated with those locations.
4. For each unresolved settlement, the worksheet lists possible promotions that could match that settlement. Here, DSM uses the internal table that contains all possible matches.
5. For each settlement, the user does one or more of the following:
  - Linking the settlement to a promotion. If the settlement is a claim, the user can next specify details for a check request.  
  
The same user or another user with higher permission then approves the settlement.
  - Denying a settlement, because it is a duplicate or for another reason.
  - Splitting the settlement, creating a new settlement in the process. A user usually splits a settlement in order to link part of it to a promotion.
6. Within DSM, an automated workflow exports details to the external systems. The details vary by implementation but may include the following, for each settlement:
  - Settlement status, as set by DSM and its users
  - Changes to the G/L (general ledger) code, as set by DSM users
  - Current amount of the settlement, if only part of the settlement was valid

The workflow also exports check requests (as specified within DSM) and chargebacks (if only part of the settlement was valid).

## Overview of the Configuration Process

These steps assume that you have already set up the basic Demantra implementation. This means that your implementation already contains the item levels, location levels, and promotion levels that are meaningful in the designated environment.

To configure DSM, the general steps are as follows:

1. Identify item and location levels associated with settlements. Then run a database script to set up DSM database structures, using that information. See "Setting Up Database Structures".  
  
This script also creates the canned DSM worksheets, all series used in those worksheets, methods (right-click actions) for use in DSM worksheets, all associated workflows, and an integration workflow suitable for use with DSM.
2. Customize your existing promotion level to include a budget, and configure series

to store that data. See "Configuring Promotions and Promotion Series".

3. Indicate how to use your promotion series. See "Identifying Key Promotion Series".
4. Specifying tolerance windows to control how closely to match settlements and promotions. See "Configuring Settlement Matching".
5. Configure the automatic write-off mechanism, if needed. See "Configuring Write-Offs".
6. Load the G/L codes, an initial set of invoices, and an initial set of settlements. See "Loading Initial Data and Creating Possible Matches".
7. Populate two tables that list the types of relationships between retailers and other entities, so that DSM can look up all locations related to the matched location.
8. Optionally customize the DSM worksheets. At a minimum, you may want to create a Promotion Effectiveness worksheet and use that within DSM (as a sub tab worksheet).
9. Customize the integration so that new settlements are loaded periodically. Each time settlements are loaded, be sure to run the DSM procedures in order to create proposed matches.

## Setting Up Database Structures

DSM associates settlements with an item and a location level and is shipped pre-configured to an existing item and location level. If you want to change this information, run the procedure `API_CONFIG_SETTLEMENT`.

Internally, this association is represented in complex tables. The association can be different in different implementations, and performing the steps described below sets up the needed database structures.

**This procedure:**

- Updates the item and location levels with which settlements should be associated (by updating the parameters `SettlementLocationExtension` and `SettlementProductExtension`)
- Upgrades the Settlement Location alias level to point to the new location level
- Ensures that all standard procedures reflect the new settlement levels

**Note:** The parameters `SettlementLocationExtension` and `SettlementProductExtension` are not visible in Business Modeler.

### To set up the DSM database structures:

1. Using PLSQL, query the table SYS\_PARAMS and determine the values of SettlementLocationExtension and SettlementProductExtension. Change these values as necessary.

| Parameter                   | Description                                                                                                                                                                      |
|-----------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| SettlementLocationExtension | Specifies the internal identifier of the location-type level with which settlements should be associated. This generally represents the entity that is being billed or refunded. |
| SettlementProductExtension  | Specifies the internal identifier of the item-type level with which settlements should be associated. This generally represents a promoted product or a product group.           |

2. If you want to change the values of Item and Location levels that are associated with the DSM settlement hierarchy, perform the steps below. Otherwise, go to step 3.
  1. Run the procedure API\_CONFIG\_SETTLEMENT.
  2. Import DSM data.
  3. Verify the following parameters (in the SYS\_PARAMS table) are set correctly for your installation:
    - DSMPromotionBudgetSeries
    - DSMPEShipDateSeries
    - DSMALLShipDateDifference
    - DSMOIPercentDifference
    - DSMOIShipDateDifference
    - DSMOICheckProduct
    - DSMPEOIAmountSeries
    - DSMWriteOffThreshold
3. Run the following procedures:

- POP\_ALL\_MATCH\_PROPOSAL.SQL
- POP\_OI\_MATCH\_PROPOSAL.SQL

These procedures use the values of the parameters listed in step 2 as input. These procedures are described in the "Database Procedures" chapter.

**Tip:** You can run API\_CONFIG\_SETTLEMENT by creating a workflow consisting of a Stored Procedure step. When defining the stored procedure's parameters, enter the Location level ID as the first value and the Item level ID as the second. For example:

**Stored Procedure Step's Properties**

Stored Procedure Step

Step Id: Step1

**Pause**

Years:  Months:  Days:   
Hours:  Minutes:  Seconds:

**Procedure**

Name: API\_CONFIG\_SETTLEMENT

Parameters:

| Parameter |
|-----------|
| 491       |
| 424       |

Add Remove

Recovery: Abort ▼

## Configuring Promotions and Promotion Series

In order to extend your existing promotion levels to work with DSM, do the following:

1. Create series as follows, if you do not yet have series similar to them:
  - A promotion budget series as follows:



---

|                           |                                               |
|---------------------------|-----------------------------------------------|
| <b>Editability:</b>       | Non-editable                                  |
| <b>Data Table:</b>        | Promotion level                               |
| <b>Data Type:</b>         | Numeric                                       |
| <b>Update Field:</b>      | Promotion_budget (for example)                |
| <b>Server Expression:</b> | max(promotion.promotion_budget) (for example) |
| <b>Proportionality:</b>   | Proportional                                  |

---

**Tip:** You might want to create two budget series: one that is editable and the other, non-editable. They both would use the same update field.

- A series to store the monetary off-invoice amounts for the promotions, for cases when that is applicable.

---

|                           |                                         |
|---------------------------|-----------------------------------------|
| <b>Data Table:</b>        | Promotion level                         |
| <b>Data Type:</b>         | Numeric                                 |
| <b>Update Field:</b>      | offinvoice (for example)                |
| <b>Server Expression:</b> | max(promotion.offinvoice) (for example) |
| <b>Proportionality:</b>   | Proportional                            |

---

- A series to store the ending date of each promotion, typically the date on which shipments end. This is the date that you will compare to the settlement dates, for the purpose of matching settlements to promotions.

---

|                     |                 |
|---------------------|-----------------|
| <b>Editability:</b> | Non-editable    |
| <b>Data Table:</b>  | Promotion level |

---

---

|                           |                                                     |
|---------------------------|-----------------------------------------------------|
| <b>Data Type:</b>         | Date                                                |
| <b>Update Field:</b>      | endship (for example)                               |
| <b>Server Expression:</b> | max(promotion.until_date) (for example)             |
| Extra From:               | promotion_dates, promotion (for example)            |
| Extra Where               | promotion.promotion_id=promotion_dates.promotion_id |
| Proportionality:          | Non-proportional                                    |

---

**Note:** This series must have extra from and extra where expressions, or else the POP\_ALL\_MATCH\_PROPOSAL will fail.

- An optional series to store the starting date of each promotion, typically the date on which shipments for the promotion start.

---

|                           |                                                     |
|---------------------------|-----------------------------------------------------|
| <b>Editability:</b>       | Non-editable                                        |
| <b>Data Table:</b>        | Promotion level                                     |
| <b>Data Type:</b>         | Date                                                |
| <b>Update Field:</b>      | startship (for example)                             |
| <b>Server Expression:</b> | max(promotion.from_date) (for example)              |
| Extra From:               | promotion_dates, promotion (for example)            |
| Extra Where               | promotion.promotion_id=promotion_dates.promotion_id |
| Proportionality:          | Non-proportional                                    |

---

2. Make a note of the IDs of these series, as shown in the Business Modeler.

3. Add a budget attribute to your existing promotion level. For Column Name, specify the column that stores the budget series that you just defined.

See also

- "Creating a Series"
- "Adding Attributes to a Level"

## Identifying Key Promotion Series

To connect DSM to your promotions and promotion series, you set the following parameters:

- `DSMPromotionBudgetSeries` should be the ID of the promotion budget series. When a settlement is matched to a promotion, DSM adjusts this budget as appropriate.
- `DSMPEOIAmountSeries` should be the ID of the series that stores the monetary off-invoice amounts for the promotions.
- `DSMPEShipDateSeries` is the ID of the series that stores the appropriate promotion date, to be compared with settlement dates.

## Configuring Settlement Matching

To configure the matching process for DSM, you must specify values for the following additional parameters:

- `DSMAllShipDateDifference`
- `DSMOIPercentDifference`
- `DSMOIShipDateDifference`

The matching process is different for off-invoice settlements than it is for claims and deductions. The following sections provide the details.

### Claims and Deductions

To find promotions to match to a claim or deduction, DSM performs two comparisons:

- DSM compares the promotion date to the settlement date. Only promotions with close enough dates are considered possible matches.

The `DSMAllShipDateDifference` parameter specifies the window of time that Demantra uses to search for a promotion that matches a given settlement. Express this as the number of time buckets between the promotion end date (`DSMPEShipDateSeries`) and the settlement date.

- DSM compares the promotion budget (DSMPromotionBudgetSeries) to the monetary settlement amount. A promotion is a possible match only if its remaining budget is at least as large as the settlement amount.

These parameters are used by the POP\_ALL\_MATCH\_PROPOSAL procedure, which you should execute every time you load settlement data.

### Off-Invoice Settlements

An off-invoice settlement must be handled slightly differently. This kind of settlement can occur only if there was an agreement to bill the customer below the invoice rate. Typically, the settlement amount was decided at that time and less variation is anticipated than with other kinds of settlements. You specify a tolerance window to use when comparing the settlement amount to promotion amounts. For flexibility, you can specify a different tolerance for date comparison as well.

It is expected that each off-invoice settlement will be matched to only one promotion or possibly to a small number of promotions, in contrast to other kinds of settlements.

To find promotions to match to an off-invoice settlement, DSM performs the following comparisons:

- DSM compares the promotion budget to the off-invoice amount. For this comparison, the DSMOIPercentDifference parameter specifies the maximum percent difference (of monetary amount) permitted when matching an off-invoice settlement to possible promotions. The promotion budget is controlled by DSMPromotionBudgetSeries.
- DSM compares the promotion date to the off-invoice date. Only promotions with close enough dates are considered possible matches. You use the DSMOIShipDateDifference parameter to specify the closeness of these dates. The promotion date is controlled by DSMPEShipDateSeries.

These parameters are used by the POP\_OI\_MATCH\_PROPOSAL procedure, which you should execute every time you load settlement data.

## Configuring Write-Offs

If a settlement is below a certain size, DSM can automatically write it off without user intervention. This process changes the settlement status to Write Off. If your implementation requires write-offs, do the following:

1. Create a workflow to automatically run the Java class `com.demantra.workflow.step.CustomWriteOffStep` each time you load settlement data.
2. Set a value for the `DSMWriteOffThreshold` parameter, which specifies the monetary amount below which Demantra automatically writes off a settlement.

## Loading Initial Data and Creating Possible Matches

The recommended way to load an initial set of data is to use the integration interfaces that are provided with DSM. The UPGRADE\_TO\_DSM procedure creates these integration interfaces. After loading the data, you should create the possible matches for use within the DSM worksheets.

This section describes how to quickly get started with an initial set of data; it does not discuss integration or automation in any detail.

The overall procedure is as follows:

1. In the Business Modeler, open each of the import integration interfaces, click Next until you reach the last screen (the preview screen), and click Create in order to create the staging table for that interface.

In this step, you create the following staging tables:

- BIIO\_GL\_Code
  - BIIO\_Settlement\_Import
  - BIIO\_Invoice
2. Load data into these staging tables:
    - For G/L codes, see "G/L Code Import".
    - For settlements, see "Settlement Import".
    - For invoices, see "Invoice Import".
  3. Be sure to commit the changes.
  4. Create and run a workflow or a set of workflows that execute these integration interfaces.

**Note:** Because settlements refer to G/L codes and invoices, you should load G/L codes and invoices first. It is also good practice to check the results of that step before executing the interface to load settlements. You can use the Business Modeler to verify that the G/L, settlement, and invoice levels contain the members that you imported; see "Viewing the Members of a Level".

5. Execute the POP\_ALL\_MATCH\_PROPOSAL and POP\_OI\_MATCH\_PROPOSAL procedures, either from within a workflow or from a SQL command line. Verify that the new tentative matches are written into the proposed\_match table.

6. Optionally create and run a workflow to run the Java class `com.demantra.workflow.step.CustomWriteOffStep`.

See also

"Creating or Editing Workflow Schemas"

## Describing Customer Relationships

DSM uses two tables to describe relationships such as those between a customer and a broker, so that the user can have an appropriate choice of ways to direct the check. You should populate these tables according to the requirements of the implementation.

### To describe the customer relationships:

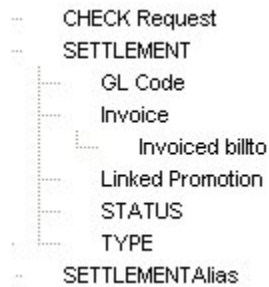
1. Populate the `Customer_Type` table to list all the customer types, such as the following example set:
  - Direct customer
  - Indirect customer
  - Broker
  - Other
2. Second, the `Customer_Type_Relation` table describes all the relationships between the customer locations (specifically, at the location level you have associated with settlements). Each row consists of three fields:
  - `CUSTOMER_LHS` is the ID of a customer (for example, A).
  - `RELATION_TYPE` is the ID of a type of customer, which we use here as a type of relationship (for example, Broker).
  - `CUSTOMER_RHS` is the ID of a related customer (for example, B).

This row means that A is a Broker for B.

## In Case of Problems

See the Oracle Demantra Release Notes for notes on adjustments or corrections to make, in case the `UPGRADE_TO_DSM` procedure has any defects.

## Reference: DSM Levels



You should not modify these levels without consulting Oracle.

**Note:** For technical reasons, the following restrictions apply:

- If you use a settlement level in a worksheet, you cannot use levels from any other hierarchy in that worksheet, either to aggregate or to filter. This means that the only location level you can use is the aliased one that is inside the settlement hierarchy.
- If you use a settlement level in a worksheet, all series in the worksheet must refer to tables used by the settlement hierarchy.

The following sections provide details on these levels:

- Check Request
- Settlement
- GL Code
- Invoice
- Invoiced Billto
- Linked Promotion
- Status
- Type
- SettlementAlias

## Check Request

This level contains the details for check requests, which are exported to the accounting systems that actually perform them. A check request is an instruction to send a check to a customer or designated third party.

This is a general level with the following attributes:

| Attribute         | Column Name          | Data Type | Purpose                                                  |
|-------------------|----------------------|-----------|----------------------------------------------------------|
| Address line 1    | CUST_ADDR_LN1        | Character | Address of this customer, for use in this check request. |
| Address line 2    | CUST_ADDR_LN2        | Character |                                                          |
| Address line 3    | CUST_ADDR_LN3        | Character |                                                          |
| Address line 4    | CUST_ADDR_LN4        | Character |                                                          |
| Amount            | CHECK_REQUEST_AMOUNT | Number    | Monetary amount of the check request.                    |
| BK Address line 1 | BK_ADDR_LN1          | Character | Address of the broker, if applicable.                    |
| BK Address line 2 | BK_ADDR_LN2          | Character |                                                          |
| BK Address line 3 | BK_ADDR_LN3          | Character |                                                          |
| BK Address line 4 | BK_ADDR_LN4          | Character |                                                          |
| BK Attn           | BK_ATTN              | Character | Addressee of the broker.                                 |
| BK City           | BK_CITY              | Character | City of the broker, for use in the address.              |
| BK Company        | BK_COMPANY           | Character | Name of the broker's company, for use in the address.    |
| BK Country        | BK_COUNTRY           | Character | Country of the broker, for use in the address.           |



| Attribute           | Column Name               | Data Type | Purpose                                              |
|---------------------|---------------------------|-----------|------------------------------------------------------|
| BK State            | BK_STATE                  | Character | State of the broker, for use in the address.         |
| BK Zip              | BK_ZIP                    | Character | Postal code of the broker, for use in the address.   |
| Check Request #     | CHECK_REQUEST_NUM         | Number    | Number of the check request.                         |
| Check Requested FOR | CHECK_REQUEST_REASON_ID   | Number    | Reason code associated with this check request.      |
| Customer City       | CUST_CITY                 | Character | City of the customer, for use in the address.        |
| Customer Country    | CUST_COUNTRY              | Character | Country of the customer, for use in the address.     |
| Customer Reference  | CR_CUSTOMER_REFERENCE     | Character |                                                      |
| Customer State      | CUST_STATE                | Character | State of the customer, for use in the address.       |
| Customer Type       | CUSTOMER_TYPE             | Number    |                                                      |
| Customer Zip        | CUST_ZIP                  | Character | Postal code of the customer, for use in the address. |
| Date Issued         | CHECK_REQUEST_DATE_ISSUED | Date      | Date on which the check request was issued.          |
| Date Requested      | CHECK_REQUEST_DATE        | Date      | Date of the check request.                           |

| Attribute         | Column Name        | Data Type | Purpose                                                          |
|-------------------|--------------------|-----------|------------------------------------------------------------------|
| Invoice           | INVOICE_ID         | Number    | ID of the invoice with which this check request is associated.   |
| Mail To Broker    | MAIL_TO_BK         | Number    |                                                                  |
| Name              | CHECK_REQUEST_DESC | Character | Description of the check request.                                |
| Note              | CHECK_REQUEST_NOTE | Character | Note entered when check request was made.                        |
| Payee             | PAYEE              | Character | Person or entity to whom the check should be written.            |
| Promo Description | CR_PROMOTION_ID    | Character | ID of the promotion with which this check request is associated. |
| Settlement ID     | SETTLEMENT_ID      | Number    | ID of the associated settlement.                                 |

For information on importing or exporting this level, see "Check Request Import and Export".

## Settlement

The Settlement level aggregates settlement data. In general, a settlement is an outstanding sum of money that needs to be resolved, related to a promotion.

This is a general level with the following attributes.

| Attribute        | Column Name         | Data Type | Purpose                                                                                                                                                                                            |
|------------------|---------------------|-----------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Account          | t_ep_lr2_EP_ID      | Number    | Description of the customer with which this settlement is associated. This must be the description field as listed in the level table, for the location level that is associated with settlements. |
| Amount To Link   | LINKED_AMOUNT       | Number    | Monetary amount that has been matched to promotions.                                                                                                                                               |
| Cust Check Date  | CUSTOMER_CHECK_DATE | Date      | Applies only to deductions and off-invoice settlements. This is the date of the check from the customer.                                                                                           |
| Customer Check # | CUSTOMER_CHECK_NUM  | Number    | Applies only to deductions and off-invoice settlements. This is the number of the check from the customer for this settlement. Uses the customer's check numbering system.                         |
| Date Posted      | DATE_POSTED         | Date      | Date when the settlement was posted.                                                                                                                                                               |
| Event Product    | promoted_product    | Number    | Code of the associated item, as listed in the Demantra tables.                                                                                                                                     |

| <b>Attribute</b>  | <b>Column Name</b>   | <b>Data Type</b> | <b>Purpose</b>                                                                      |
|-------------------|----------------------|------------------|-------------------------------------------------------------------------------------|
| GL Code           | GL_CODE_ID           | Number           | Code of the associated G/L code, as listed in the Demantra tables.                  |
| Link Date         | LINK_DATE            | Date             | Date on which this settlement was linked to a promotion.                            |
| Linked Promotion  | PROMOTION_ID         | Number           | ID of the associated promotion, as listed in the Demantra tables.                   |
| Method Status     | METHOD_STATUS        | Number           | For internal use only.                                                              |
| Name              | SETTLEMENT_DESC      | Character        | Description of the settlement.                                                      |
| Open Amount       | OPEN_AMOUNT          | Number           | Remaining amount of the settlement that has not yet been matched to any promotions. |
| Related WS        | RELATED_WS           | Number           | Demantra ID of the worksheet that is associated, by default, with settlements.      |
| Settlement #      | SETTLEMENT_NUMBER    | Number           | Number for the settlement, as given in the enterprise systems.                      |
| Settlement Action | SETTLEMENT_ACTION_ID | Number           |                                                                                     |
| Settlement Amount | SETTLEMENT_AMOUNT    | Number           | Total monetary amount of the settlement.                                            |

| Attribute           | Column Name           | Data Type | Purpose                                                                                              |
|---------------------|-----------------------|-----------|------------------------------------------------------------------------------------------------------|
| Settlement Invoice  | INVOICE_ID            | Number    | ID of the associated invoice, as listed in the Demantra tables.                                      |
| Settlement Owner    | SETTLEMENT_OWNER      | Number    | DSM user who has claimed responsibility for this settlement.                                         |
| Settlement Type     | SETTLEMENT_TYPE_ID    | Number    | Type of the settlement. This should be one of the IDs of the Type level; see "Type".                 |
| Split Settlement ID | SPLIT_SETTLEMENT_ID   | Number    | ID of the settlement that was split off from this settlement, if any.                                |
| Status              | SETTLEMENT_STAT_US_ID | Number    | Status of the settlement. This should be one of the IDs of the Status level; see "Status".           |
| Supplier Check #    | SUPPLIER_CHECK_NUM    | Number    | Applies only to claims. Date of the check to the customer.                                           |
| Supplier Check Date | SUPPLIER_CHECK_DATE   | Date      | Applies only to claims. This is the number of the check that reimburses the customer for this claim. |

For information on loading settlements into DSM, see "Settlement Import".

## GL Code

This level contains the G/L codes that you loaded from the other corporate systems.

This is a general level with no attributes.

For information on loading G/L codes, see "G/L Code Import".

## Invoice

This level contains the invoices that you loaded from the other corporate systems.

This is a general level with the following attributes.

| Attribute        | Column Name    | Data Type | Purpose                                             |
|------------------|----------------|-----------|-----------------------------------------------------|
| Invoice #        | INVOICE_NUM    | Number    | Invoice number, as used in the corporate systems.   |
| Invoice Date     | INVOICE_DATE   | Date      | Date of the invoice.                                |
| Invoiced Bill To | t_ep_lr2_EP_ID | Number    | ID of the customer to whom this invoice was issued. |
| Name             | INVOICE_DESC   | Character | Description. Can be identical to the invoice codes. |

For information on loading invoices, see "Invoice Import".

## Invoiced Billto

This level is an alias to the actual location level that you associated with settlements. Demantra creates, maintains, and uses this alias for technical reasons, and you should not make changes to it.

## Linked Promotion

This level is an alias to the lowest promotion level. Demantra creates, maintains, and uses this alias for technical reasons, and you should not change it.

## Status

This level contains the predefined settlement statuses:

- New
- In Progress
- Unapproved

- Approved
- Duplicate
- Denied
- Write Off

These statuses can not be customized. Do not change this level in any way.

## Type

This level contains the predefined settlement types:

- Claim
- Off-invoice
- Deduction
- Non-Trade
- Claim resulting from an original claim split
- Deduction resulting from an original deduction split

These types can not be customized. Do not change this level in any way.

## SettlementAlias

This level is an alias to the settlement level. Demantra creates, maintains, and uses this alias for technical reasons, and you should not change it.

## Reference: DSM Series

For information on the predefined series, see the Oracle Demantra Deduction and Settlement Management User's Guide.

## Reference: DSM Integration Interfaces

The UPGRADE\_TO\_DSM procedure automatically creates integration interfaces to help you import or export the following:

- G/L codes
- Invoices

- Settlements
- Check requests

This section provides details on these integration interfaces.

### G/L Code Import

|                        |                                                                                  |
|------------------------|----------------------------------------------------------------------------------|
| <b>Interface Name:</b> | Reason Code Integration INTERFACE                                                |
| <b>Type:</b>           | Import                                                                           |
| <b>Description:</b>    | Imports rows from a staging table and adds the new members to the GL Code level. |
| <b>Staging Table:</b>  | BIIO_GL_Code                                                                     |

### BIIO\_GL\_Code

This staging table has the following structure:

| Field        | Data Type      | Purpose                                                                                                              |
|--------------|----------------|----------------------------------------------------------------------------------------------------------------------|
| GL_CODE_CODE | varchar2(120)  | <b>Cannot be null.</b> Short version of the general ledger codes, as used in the enterprise system. Example: SPOILED |
| GL_CODE_DESC | varchar2(2000) | <b>Cannot be null.</b> Longer description of the codes. In some cases, these are identical to the codes.             |

### Invoice Import

|                        |                                                                                  |
|------------------------|----------------------------------------------------------------------------------|
| <b>Interface Name:</b> | Invoice Integration INTERFACE                                                    |
| <b>Type:</b>           | Import                                                                           |
| <b>Description:</b>    | Imports rows from a staging table and adds the new members to the Invoice level. |
| <b>Staging Table:</b>  | BIIO_Invoice                                                                     |

### BIIO\_Invoice



This staging table has the following structure:

| Field          | Data Type      | Purpose                                                                                                                                                                                                  |
|----------------|----------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| INVOICE_CODE   | varchar2(120)  | <b>Cannot be null.</b> Unique code for the invoice, for use in Demantra.                                                                                                                                 |
| INVOICE_DESC   | varchar2(2000) | <b>Cannot be null.</b> Description. Can be identical to the invoice codes.                                                                                                                               |
| LR2            | varchar2(120)  | <b>Cannot be null.</b> Code of the customer with which this invoice is associated. This must be the code field as listed in the level table, for the location level that is associated with settlements. |
| INVOICE_NUM    | number(20,10)  | Invoice number, as used in the corporate systems.                                                                                                                                                        |
| INVOICE_DATE   | date           | Date of the invoice.                                                                                                                                                                                     |
| T_EP_LR2_EP_ID | varchar2(2000) | Description of the customer with which this invoice is associated. This must be the description field as listed in the level table, for the location level that is associated with settlements.          |

### Settlement Import

|                        |                                                                                     |
|------------------------|-------------------------------------------------------------------------------------|
| <b>Interface Name:</b> | SETTLEMENT LEVEL import                                                             |
| <b>Type:</b>           | Import                                                                              |
| <b>Description:</b>    | Imports rows from a staging table and adds the new members to the Settlement level. |
| <b>Staging Table:</b>  | BIIO_Settlement_Import                                                              |

### BIIO\_Settlement\_Import

This staging table has the following structure:

| Field                      | Data Type      | Purpose                                                                                                                                                                    |
|----------------------------|----------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| SETTLEMENT_CODE            | varchar2(500)  | <b>Cannot be null.</b> Unique code for the settlement, for use in Demantra.                                                                                                |
| SETTLEMENT_DESC            | varchar2(2000) | <b>Cannot be null.</b> Description of the settlement.                                                                                                                      |
| INVOICE_CODE               | varchar2(120)  | <b>Cannot be null.</b> Code of the associated invoice, as listed in the Demantra tables.                                                                                   |
| GL_CODE_CODE               | varchar2(120)  | <b>Cannot be null.</b> Code of the associated G/L code, as listed in the Demantra tables.                                                                                  |
| SETTLEMENT_STATU<br>S_CODE | varchar2(120)  | <b>Cannot be null.</b> Status of the settlement. This should be one of the codes of the Status level; see "Status".                                                        |
| SETTLEMENT_TYPE_<br>CODE   | varchar2(120)  | <b>Cannot be null.</b> Type of the settlement. This should be one of the codes of the Type level; see "Type".                                                              |
| PROMOTION_CODE             | varchar2(120)  | <b>Cannot be null.</b> Code of the associated promotion, as listed in the Demantra tables.                                                                                 |
| SETTLEMENT_OWNE<br>R       | varchar2(50)   | DSM user who has claimed responsibility for this settlement.                                                                                                               |
| DATE_POSTED                | date           | <b>Should not be null.</b> Date when the settlement was posted.                                                                                                            |
| CUSTOMER_CHECK_<br>NUM     | number(20,10)  | Applies only to deductions and off-invoice settlements. This is the number of the check from the customer for this settlement. Uses the customer's check numbering system. |
| CUSTOMER_CHECK_<br>DATE    | date           | Applies only to deductions and off-invoice settlements. This is the date of the check from the customer.                                                                   |
| SETTLEMENT_AMOU<br>NT      | number(20,10)  | Total monetary amount of the settlement.                                                                                                                                   |

| Field                 | Data Type      | Purpose                                                                                                                                                                                            |
|-----------------------|----------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| SUPPLIER_CHECK_NUMBER | number(20,10)  | Applies only to claims. This is the number of the check that reimburses the customer for this claim.                                                                                               |
| SUPPLIER_CHECK_DATE   | date           | Applies only to claims. Date of the check to the customer.                                                                                                                                         |
| SETTLEMENT_TYPE_ID    | varchar2(2000) | Type of the settlement. This should be one of the IDs of the Type level; see "Type".                                                                                                               |
| SETTLEMENT_ACTION_ID  | varchar2(255)  |                                                                                                                                                                                                    |
| LINKED_AMOUNT         | number(20,10)  | Monetary amount that has been matched to promotions.                                                                                                                                               |
| OPEN_AMOUNT           | number(20,10)  | Remaining amount of the settlement that has not yet been matched to any promotions.                                                                                                                |
| INVOICE_ID            | varchar2(2000) | ID of the associated invoice, as listed in the Demantra tables.                                                                                                                                    |
| T_EP_LR2_EP_ID        | varchar2(2000) | Description of the customer with which this settlement is associated. This must be the description field as listed in the level table, for the location level that is associated with settlements. |
| GL_CODE_ID            | varchar2(2000) | ID of the associated G/L code, as listed in the Demantra tables.                                                                                                                                   |
| SETTLEMENT_STATUS_ID  | varchar2(2000) | Status of the settlement. This should be one of the IDs of the Status level; see "Status".                                                                                                         |
| PROMOTION_ID          | varchar2(2000) | ID of the associated promotion, as listed in the Demantra tables.                                                                                                                                  |
| PROMOTED_PRODUCT      | varchar2(2000) | Description field of the associated item, as listed in the level table for the appropriate level.                                                                                                  |

| <b>Field</b>        | <b>Data Type</b> | <b>Purpose</b>                                                                 |
|---------------------|------------------|--------------------------------------------------------------------------------|
| LINK_DATE           | date             | Date on which this settlement was linked to a promotion.                       |
| METHOD_STATUS       | varchar2(200)    | For internal use only.                                                         |
| SPLIT_SETTLEMENT_ID | number(20,10)    | ID of the settlement that was split off from this settlement, if any.          |
| RELATED_WS          | number(20,10)    | Demantra ID of the worksheet that is associated, by default, with settlements. |
| SETTLEMENT_number   | number(20,10)    | Number for the settlement, as given in the enterprise systems.                 |

### Settlement Export

|                        |                                                                                                                                |
|------------------------|--------------------------------------------------------------------------------------------------------------------------------|
| <b>Interface Name:</b> | SETTLEMENT LEVEL export                                                                                                        |
| <b>Type:</b>           | Export                                                                                                                         |
| <b>Description:</b>    | Exports members of the Settlement level. Performs full export (not incremental) to \TEMP\IntegrationDIR\Export_Settlement.TXT. |
| Staging Table:         | N/A                                                                                                                            |

### Check Request Import and Export

|                        |                                     |
|------------------------|-------------------------------------|
| <b>Interface Name:</b> | CHECK Request Integration INTERFACE |
| <b>Type:</b>           | Import & Export                     |

|                     |                                                                                                                                                                                                                                              |
|---------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description:</b> | <p>When used for import: imports rows from a staging table and adds the new members to the Check Request level.</p> <p>When used for export: performs full export (not incremental) to<br/> \TEMP\IntegrationDIR\Export_CheckRequest.TXT</p> |
| Staging Table:      | BIIO_CheckRequest                                                                                                                                                                                                                            |

### BIIO\_CheckRequest

This staging table has the following structure:

| Field                     | Data Type      | Purpose                                                                        |
|---------------------------|----------------|--------------------------------------------------------------------------------|
| CHECK_REQUEST_CODE        | varchar2(120)  | <b>Cannot be null.</b> Unique code for the check request, for use in Demantra. |
| CHECK_REQUEST_DESCRIPTION | varchar2(2000) | <b>Cannot be null.</b> Description of the check request.                       |
| INVOICE_ID                | varchar2(2000) | ID of the invoice with which this check request is associated.                 |
| CHECK_REQUEST_NUMBER      | number(20,10)  | Number of the check request.                                                   |
| CHECK_REQUEST_AMOUNT      | number(20,10)  | Monetary amount of the check request.                                          |
| CHECK_REQUEST_DATE        | date           | Date of the check request.                                                     |
| CHECK_REQUEST_DATE_ISSUED | date           | Date on which the check request was issued.                                    |
| CHECK_REQUEST_REASON_ID   | varchar2(255)  | Reason code associated with this check request.                                |
| CHECK_REQUEST_NOTE        | varchar2(200)  | Note entered when the check request was created.                               |

| Field                     | Data Type      | Purpose                                                                                 |
|---------------------------|----------------|-----------------------------------------------------------------------------------------|
| CR_CUSTOMER_REFER<br>ENCE | varchar2(200)  |                                                                                         |
| CR_PROMOTION_ID           | varchar2(2000) | ID of the promotion with which this check request is associated.                        |
| CUSTOMER_TYPE             | number(20,10)  | ID of a customer type, from the Customer_Type table. See "Reference: Other DSM Tables". |
| PAYEE                     | varchar2(200)  | Person or entity to whom the check should be written.                                   |
| CUST_ADDR_LN1             | varchar2(200)  | Address of this customer, for use in this check request.                                |
| CUST_ADDR_LN2             | varchar2(200)  |                                                                                         |
| CUST_ADDR_LN3             | varchar2(200)  |                                                                                         |
| CUST_ADDR_LN4             | varchar2(200)  |                                                                                         |
| CUST_CITY                 | varchar2(200)  |                                                                                         |
| CUST_STATE                | varchar2(200)  |                                                                                         |
| CUST_ZIP                  | varchar2(200)  |                                                                                         |
| CUST_COUNTRY              | varchar2(200)  |                                                                                         |
| MAIL_TO_BK                | number(20,10)  |                                                                                         |
| BK_COMPANY                | varchar2(200)  | Name of the associated broker, if any, for use in this check request.                   |
| BK_ATTN                   | varchar2(200)  | Addressee of the broker.                                                                |
| BK_ADDR_LN1               | varchar2(200)  | Address of the broker.                                                                  |
| BK_ADDR_LN2               | varchar2(200)  |                                                                                         |
| BK_ADDR_LN3               | varchar2(200)  |                                                                                         |

| Field         | Data Type     | Purpose                          |
|---------------|---------------|----------------------------------|
| BK_ADDR_LN4   | varchar2(200) |                                  |
| BK_CITY       | varchar2(200) |                                  |
| BK_STATE      | varchar2(200) |                                  |
| BK_ZIP        | varchar2(200) |                                  |
| BK_COUNTRY    | varchar2(200) |                                  |
| SETTLEMENT_ID | number(20,10) | ID of the associated settlement. |

## Reference: Other DSM Tables

DSM uses the following additional tables; see "Describing Customer Relationships".

### Customer\_Type

This table has the following structure:

| Field              | Data Type      | Purpose                                                                        |
|--------------------|----------------|--------------------------------------------------------------------------------|
| CUSTOMER_TYPE_ID   | number(10)     | <b>Cannot be null.</b> Unique ID for the customer type, for use in Demantra.   |
| CUSTOMER_TYPE_CODE | varchar2(240)  | <b>Cannot be null.</b> Unique code for the customer type, for use in Demantra. |
| CUSTOMER_TYPE_DESC | varchar2(1000) | Description of the customer type.                                              |
| FICTIVE_CHILD      | number(10)     | Ignore this field.                                                             |
| IS_FICTIVE         | number(1)      | Ignore this field.                                                             |
| LAST_UPDATE_DATE   | date           | Ignore this field.                                                             |

### Customer\_Type\_Relation

This table has the following structure:

| Field         | Data Type  | Purpose                                                                                                                                             |
|---------------|------------|-----------------------------------------------------------------------------------------------------------------------------------------------------|
| CUSTOMER_LHS  | number(10) | <b>Cannot be null.</b> ID of a customer, specifically a member of the location level with which settlements are associated, in this implementation. |
| RELATION_TYPE | number(10) | <b>Cannot be null.</b> ID of a customer type, from the Customer_Type table.                                                                         |
| CUSTOMER_RHS  | number(10) | <b>Cannot be null.</b> ID of a customer, specifically a member of the location level with which settlements are associated, in this implementation. |

If CUSTOMER\_LHS refers to Customer A, RELATION\_TYPE refers to the Broker type, and CUSTOMER\_RHS refers to Customer B, then this row means that Customer A is a Broker for Customer B.



# Part 5

---

## Other Configuration



---

## Fine Tuning and Scaling Demantra

Typically you adjust parameters to control your solution's global behavior, including various defaults and performance settings. This chapter provides an overview of most of the parameters, grouped into specific areas.

This chapter covers the following topics:

- Basic Parameters
- Application Server
- Collaborator Workbench
- Database
- Date/Time Formats
- Email
- Integration
- Item Aggregation
- Logs and Old Data
- Proport Mechanism
- EngineOutputThreshold
- Simulation
- Solution Branding
- Threading
- Workflow
- Worksheets
- Languages
- Configuring User Productivity Kit (UPK)

## Basic Parameters

For reference, Demantra stores basic configuration information in the following parameters. Unless otherwise noted, you should not change these parameters after going live:

| Parameter                 | Description                                                                                                                                                      |
|---------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| active_forecasts_versions | Specifies how many forecast versions the Demantra database should store.<br><br>You can change this parameter after going live.                                  |
| FIRSTDAYINWEEK            | First day of week to use when binning sales data into base time buckets, in a weekly system. It is not generally safe to change this parameter after going live. |

## Application Server

The APS queue uses the following parameters:

| Parameter              | Description                                                              |
|------------------------|--------------------------------------------------------------------------|
| QueryMechanisimTimeOut | The timeout period for the query notification listener, in milliseconds. |
| StartUpdateQueue       | Specifies whether to start the manual update listener.                   |
| UpdateQueueTimeout     | The timeout period for the manual update listener, in milliseconds.      |

## Collaborator Workbench

The following parameters control Collaborator Workbench. Also see "Solution Branding" for parameters that control Collaborator Workbench titles.

- "General"
- "My Tasks"

- "Who's Online"
- "Content Panes"

## General

The following parameters control Collaborator Workbench in general:

| Parameter                                    | Description                                                                                                                                                                                                                                                                                                           |
|----------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| collaborator.supportURL                      | URL of the Support link, relative to http://server name/virtual directory/portal/. This link is in the upper right corner of Collaborator Workbench.                                                                                                                                                                  |
| collaborator.searchURL                       | URL of the Search link, relative to http://server name/virtual directory/portal/. This link is in the upper right corner of Collaborator Workbench.                                                                                                                                                                   |
| dir.onlineHelp                               | URL of the online help, relative to http://server name/virtual directory/portal/. This link is in the upper right corner of Collaborator Workbench.                                                                                                                                                                   |
| navBarContentProvider.addNewContentLink.Text | Text of the New link, which is shown at the top of the Contents menu.                                                                                                                                                                                                                                                 |
| Server.SessionExpiration                     | This is the time interval (in seconds) between which the current page automatically refreshes itself. If an idle Collaborator Workbench session has expired, the current page will not reflect this until the next refresh event, at which time it will automatically redirect itself to the expiration page.expires. |

Also see "Customizing Demantra Web Pages".

## My Tasks

The following parameter affects the My Tasks pane of Collaborator Workbench:

| Parameter              | Description                                                            |
|------------------------|------------------------------------------------------------------------|
| general.userList.tasks | Specifies whether the My Tasks module displays the Create Task button: |


## Who's Online

The following parameters control the Who's Online pane of Collaborator Workbench:

| Parameter                            | Description                                                     |
|--------------------------------------|-----------------------------------------------------------------|
| general.userList.whoisonline         | Specifies whether the Who's Online module is displayed.         |
| UserListContentProvider.commonTitle  | The title of the Who's Online pane.                             |
| UserTitleContentProvider.TimeToSleep | The time to wait polling user status for the Who's Online pane. |

## Content Panes

The following parameters control the default behavior of graph-type content panes:

| Parameter                 | Description                                                                                                                                                                                                                                                                                                                  |
|---------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Graph.MaxLabelWidth       | Maximum width of labels in graph-type content panes in Collaborator Workbench. If a label is too longer, the last characters are represented by three periods (...).                                                                                                                                                         |
| Legend.MaxLegendItemWidth | Maximum width (in characters) of the legend in a graph-type content pane in Collaborator Workbench. If any lines of the legend are too longer, the last characters of those lines are represented by three periods (...), as follows:<br> |

| Parameter                       | Description                                                                                                                                                                                                                                                                   |
|---------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Query.MaxCombinations           | Maximum number of combinations that can be displayed in a graph-type content pane in Collaborator Workbench, when you display a single series plotted for multiple combinations. The user receives an error if a content pane contains more than this number of combinations. |
| Query.MaxSeries                 | Maximum number of series that can be displayed in a graph-type content pane in Collaborator Workbench. The user receives an error if a content pane contains more than this number of series.                                                                                 |
| Query.TopBottom.MaxCombinations | Maximum number of combinations that can be displayed in a content pane that contains a stacked bar chart or pie chart. The user receives an error if a content pane contains more than this number of combinations.                                                           |

See also

"Email"

"Workflow"

"Worksheets"

## Database

The following parameters control how Demantra connects to and uses the Demantra database.

- "General Database Settings"
- "Database Connections"
- "Oracle Tablespaces"
- "Technical Settings"

For additional parameters that specify *which* database Demantra connects to, see the Oracle Demantra Installation Guide.

## General Database Settings

| Parameter           | Description                                                                                                   |
|---------------------|---------------------------------------------------------------------------------------------------------------|
| DBDateFormat        | Controls the date format used in the database.                                                                |
| LockTimeout         | Specifies the period (in seconds) between killing a database session and releasing the lock for that session. |
| Rebuild_Sales_Table | Specifies whether the REBUILD_TABLES procedure should rebuild the sales_data table. Applies only to Oracle.   |

## Database Connections

The following parameters control Oracle Demantra's database connections:

| Parameter                       | Description                                                                                                                                    |
|---------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------|
| AdditionalConnectionsExpression | <b>New for 7.0)</b> Ignore this parameter for now.                                                                                             |
| DBConnectionTimeout             | The database connection timeout period.                                                                                                        |
| DBIdleTimeOut                   | The connection idle timeout period.<br>Recommended: 300000 (5 minutes)                                                                         |
| MaxDBConnections                | The maximum number of database connections for the Demantra database user.<br><br>Recommended: the number of concurrent users multiplied by 2. |
| MinDBConnections                | The minimum number of database connections for the Demantra database user.                                                                     |

## Oracle Tablespaces

For Oracle databases, Demantra writes to multiple tablespaces, as specified during installation. The tablespace assignments are controlled by parameters, which you can edit through the Business Modeler. Make sure that these parameters refer to tablespaces within the appropriate database user, and make sure each has enough storage.



Additional parameters control the default initial sizes and how much storage is added.

| Parameter                      | Description                                                                                    |
|--------------------------------|------------------------------------------------------------------------------------------------|
| initial_param                  | Default initial size of system tablespaces.                                                    |
| next_param                     | Incremental amount of storage that is added to a tablespace when more space is needed.         |
| tablespace*                    | Tablespace used for the sales table.                                                           |
| indexspace*                    | Database index space that stores the forecast table indexes, as specified during installation. |
| simulationspace*               | Tablespace used for simulation data.                                                           |
| simulationindexspace*          | Tablespace used for simulation index data.                                                     |
| sales_data_engine_index_space* | Tablespace used for the index of sales_data_engine.                                            |
| sales_data_engine_space*       | Tablespace used for sales_data_engine table.                                                   |

\* You set these parameters during installation.

Oracle recommends that you use the standard names for these tablespaces, as documented in the Oracle Demantra Installation Guide. Then it is easier for you to share your database with Oracle Support Services in case of problems.

## Technical Settings

The following parameters should be adjusted only by someone experienced with databases:

| Parameter              | Description                                                                                                                                                                                                                               |
|------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| max_records_for_commit | The number of records that Demantra will insert into the database before performing a COMMIT operation. If you increase this number, the insertion will run more quickly, but you risk losing all uncommitted records in case of a crash. |

| Parameter                                       | Description                                                                                                        |
|-------------------------------------------------|--------------------------------------------------------------------------------------------------------------------|
| oracle_optimization_mode*                       | <b>Oracle only.</b> Optimization mode of the database, either cost-based (most common) or rule-based.              |
| pct_increase_for_analyze                        | Percentage of data increase for a given table, beyond which Demantra automatically increases the table size.       |
| set_rb*                                         | <b>(Oracle 8i only)</b> Set Rollback Segment command. This is database dependent. See your database documentation. |
| *For these parameters, see "Engine Parameters". |                                                                                                                    |

See also

"Integration"

## Date/Time Formats

The following parameters control the formats of date and date/time values throughout Demantra:

| Parameter                 | Description                                                                                                                                                                                           |
|---------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| applicationDateFormat     | The system date format.                                                                                                                                                                               |
| applicationDateTimeFormat | The system date/time format, used where both a date and time are displayed.                                                                                                                           |
| DBDateFormat              | Controls the date format used in the database.                                                                                                                                                        |
| format.dates.longFormat   | Long date format.                                                                                                                                                                                     |
| format.dates.shortFormat  | Short date format, used in the title bar of Collaborator Workbench.                                                                                                                                   |
| InsertDateTimeFormat      | The date-time format that Demantra uses when writing to the database. When you enter dates in a worksheet or import dates, Demantra converts them to this format before writing them to the database. |

## Email

If you are using any of the Demantra Web-based software, Demantra can automatically send email on specific occasions, for example, within workflows. To enable this, first set up an administrator email account on an SMTP server; this account will be the originator of all Demantra's automatic messages. You will probably need the help of the IT department to get this account configured correctly, depending on the network security.

Then use the parameters in this section to specify that email account for use by Demantra.

- "Configuring Demantra Email"
- "Strings Used in Demantra Email"

### Configuring Demantra Email

First, the following parameters specify the email account from which Demantra Web-based software will send email.

| Parameter                | Description                                                                                                                                                                                                                 |
|--------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| mail*                    | Controls whether email is enabled. Can be set with the Business Modeler at: Parameters > System Parameters > Application Server (tab) > Dp Web (tab).                                                                       |
| mail.server*             | SMTP server that is hosting the email application to be used by Demantra. Can be set with the Business Modeler at: Parameters > System Parameters > Application Server (tab) > Collaborator (tab).                          |
| mailAddress*             | Mail address of the designated Demantra administrator. Can be set with the Business Modeler at: Parameters > System Parameters > Application Server (tab) > Dp Web (tab).                                                   |
| mail.strings.from.system | Specifies the title of the sender of Demantra email messages, for example "Demantra Solution Manager". Can be set with the Business Modeler at: Parameters > System Parameters > Application Server (tab) > Workflow (tab). |

| Parameter        | Description                                                                                                                                                                        |
|------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| AuditMailAddress | Mail address of the BCC recipient of Demantra email messages. Can be set with the Business Modeler at: Parameters > System Parameters > Application Server (tab) > Workflow (tab). |

\*These can be set via the Demantra installer or later. See the Oracle Demantra Installation Guide.

### Strings Used in Demantra Email

In addition, the following parameters control the strings used in the email messages that Demantra sends.

| Parameter                          | Description                                                                                     |
|------------------------------------|-------------------------------------------------------------------------------------------------|
| company.name                       | Name of your company; the Workflow Engine uses this string in email when a workflow step fails. |
| mail.strings.internalerror.message | Text of email message sent in case of error.                                                    |
| mail.strings.internalerror.subject | Subject of email message sent in case of error.                                                 |
| mail.strings.from.system           | Message sent in a fail-to-execute task description.                                             |
| mail.strings.processfailuresubject | Message sent when a process is terminated.                                                      |
| mail.strings.processterminated     | String included in recovery email message.                                                      |
| mail.strings.recovery              | Message sent in a fail-to-execute task subject.                                                 |
| mail.strings.taskfailuresubject    | Message sent when a task is timed out.                                                          |
| mail.strings.tasktimedoutsubject   | Message sent when a task is timed out in a group step.                                          |
| mail.strings.timeout.group         | Message sent when a task is timed out in a user step.                                           |

| Parameter                 | Description                                  |
|---------------------------|----------------------------------------------|
| mail.strings.timeout.user | Text of email message sent in case of error. |

## Integration

The following parameters control import and integration in Demantra. These parameters apply only to the core Demantra tools

| Parameter                          | Description                                                                                                                                                                                                                                                                                                                                                                                                                             |
|------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| accumulatedOrUpdate                | For integration, this parameter specifies whether the system adds to the existing data (accumulate) or overwrites the existing data (update).                                                                                                                                                                                                                                                                                           |
| align_sales_data_levels_in_loading | Specifies whether to maintain matrix information (combination information that is time-independent) within the sales_data table. If requested, this adjustment is made when data is added via loading, integration, or other mechanisms.<br><br>If you set this parameter to yes, it is also necessary to rewrite some database procedures. For additional configuration steps, see Part , "Reconfiguring the sales_data_engine Table". |
| ImportBlockSize                    | The number of rows for each commit, used during import.                                                                                                                                                                                                                                                                                                                                                                                 |
| InsertDateTimeFormat               | The date/time format that Demantra uses when writing to the database. When you enter dates in the worksheet wizard, Demantra converts them to this format.                                                                                                                                                                                                                                                                              |
| Insertmissingvalues                | Specifies whether to insert zero values for dates that have null values.                                                                                                                                                                                                                                                                                                                                                                |
| LoadDataStop                       | Specifies whether Demantra should stop loading data when it finds an error in the data.                                                                                                                                                                                                                                                                                                                                                 |
| RunProportInMdp_add                | Specifies whether to call the proport mechanism from the MDP_ADD procedure.                                                                                                                                                                                                                                                                                                                                                             |

| Parameter             | Description                                                                                                                                                                                                   |
|-----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| UpdateAudit           | Specifies whether the MANUALS_INS_INTEGRATION procedure updates the audit tables.                                                                                                                             |
| update_units_by_items | <p>Specifies how to update units for the INSERT_UNITS procedure.</p> <ul style="list-style-type: none"> <li>• By items (faster but less accurate)</li> <li>• By combinations (slower but accurate)</li> </ul> |

## Item Aggregation

For improved performance, you can configure Demantra to aggregate data by items and use that aggregated data whenever possible. In this case, Demantra maintains the branch\_data\_items table in addition to the usual tables. Demantra uses this table whenever possible; it does not use the table whenever you need to view specific locations or filter by location. To configure Demantra in this manner, set the UseItemsAggri parameter.

**Note:** Also be sure the DYNAMIC\_SYNC is scheduled to run periodically to keep the branch\_data\_items table up to date.

## Logs and Old Data

The following parameters control how long Demantra keeps various kinds of historical data:

| Parameter            | Description                                            |
|----------------------|--------------------------------------------------------|
| audit_history_length | Number of months of audit data to keep.                |
| log.history          | The number of days for which workflow history is kept. |

Also see "Logging Messages of the Application Server".

## Proport Mechanism

### Parameters That Control Behavior

| Parameter         | Purpose                                                                                                                                                                                                                                                                                                    |
|-------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| hist_glob_prop    | Specifies the maximum number of base time buckets to use in calculating glob_prop, the running average demand for any given item-location combination.                                                                                                                                                     |
| def_delta         | <p>Specifies the default value for the delta field in the mdp_matrix table. If delta equals null for a given combination, the system uses the value of this parameter instead.</p> <p>In turn, the delta field specifies the month-to-month smoothing of the weekly proportions.</p>                       |
| proport_missing   | Specifies what value to use for dates with null sales (zero or average).                                                                                                                                                                                                                                   |
| proport_threshold | Specifies the number of distinct months needed to compute P1, ... P12 in the usual way.                                                                                                                                                                                                                    |
| proport_spread    | Specifies what value to use for any month that has null data.                                                                                                                                                                                                                                              |
| last_date         | Last date of actual sales, to be used by the Analytical Engine and the proportion mechanism. No dates after this are used towards the forecast or the proportion calculation.                                                                                                                              |
| quantity_form     | Expression that the Analytical Engine uses to select the historical demand from the sales_data table; the result of this expression is the data that the engine uses as input. The default expression transforms negative values to zero and should be modified if business needs require negative demand. |

| Parameter                                       | Purpose                                                                                                                                                                                                                                       |
|-------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| mature_age                                      | Controls the mature_date, which is calculated backwards from the current date using the mature_age parameter. A combination is young (rather than active) if it does not have any non-zero sales data for dates on or before the mature_date. |
| dying_time                                      | If no sales occurred during the length of time specified by dying_time, the combination will be marked as dead (0 forecast will be issued).                                                                                                   |
| *For these parameters, see "Engine Parameters". |                                                                                                                                                                                                                                               |

### Parameters That Affect Performance

| Parameter                                        | Purpose                                                                                                                                                                                                                                                                                                                                                                   |
|--------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| add_zero_combos_to_mdp*                          | If true, add combinations to mdp_matrix even if their historical data consists of zeros. This parameter is used by the proport mechanism.                                                                                                                                                                                                                                 |
| Run_full_matrix_proport                          | <p>Specifies whether to run the proport mechanism on all the item-location combinations.</p> <ul style="list-style-type: none"> <li>• If no (0), run proport only on the combinations that have prop_changes=1.</li> <li>• If yes (1), run proport on all combinations in mdp_matrix.</li> <li>• If 2, run proport on all combinations that have new_member=1.</li> </ul> |
| * For these parameters, see "Engine Parameters". |                                                                                                                                                                                                                                                                                                                                                                           |

### Proport Parallelization

Proport handles large amount of data processing and can require a substantial amount of time and system resources. It is possible to improve run time performance using parallelization and by grouping the Proport process into several iterations.

To do this, define the following parameters in the init\_params table for each engine profile:



| Parameter           | Description                                                                                                                                                                                     |
|---------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ProportParallelJobs | The number of parallel jobs used when running Proport calculations. This parameter's value should not exceed the number of CPUs on the database server.                                         |
| ProportTableLabel   | The name of the level by which the process is broken down. The total number of members in this level is divided into equally sized groups, and one group is processed each time Proport is run. |
| ProportRunsInCycle  | The number of groups that the Proport process is broken down into.                                                                                                                              |

**Example:**

ProportTableLabel = 'Item'

ProportRunsInCycle = 10

ProportParallelJobs = 2

When running Proport with these settings in the example above, processing occurs for all combinations associated with 1/10 of the Item level members. Each execution invokes 2 parallel jobs in the DBMS\_JOB queue. Proport is called ten times before all combinations have been processed.

Use caution when choosing the level by which Proport will be broken into separate tasks. The process will treat each member as equal and will not be aware that some members contain far more data than others. Selecting an inappropriate level could result in an uneven processing time between different session runs.

**Example:**

ProportTableLabel = 'Segment'

ProportRunsInCycle = 5

ProportParallelJobs = 4

If there are 10 segments, then each Proport process will execute on 2 (10/5) segments and use 4 parallel processes. If a specific segment contains a much larger number of items than other segments, then processing of this specific segment require more time to run.

## EngineOutputThreshold

The forecast for certain combinations may not change significantly between runs, this

typically occurs for steady sellers, or extremely slow moving items. Writing out a new forecast in this case would have only marginal benefit to the supply chain. However, it would add cost in:

1. Engine I/O processing
2. The noise of minute variations in the production forecast

The parameter "EngineOutputThreshold" provides control over whether to write the forecast for a combination that has changed very little. Example: If the new value is 3% larger than the old value, and the EngineOutputThreshold is 5%, then the engine will not output the new value, and the old value will remain. In this case the new value would need to be at least 5% larger or smaller than the old value before it replaces the existing forecast value. Note that if either the old value or new value is zero, then the difference will be calculated as 100% (unless they're both zero, in which case the difference would be 0%).

## Simulation

When a user starts a large simulation, it is useful to check the size of that simulation and provide a warning if it will take a long time to run. You may also want to prevent too-large simulations from being run at all.

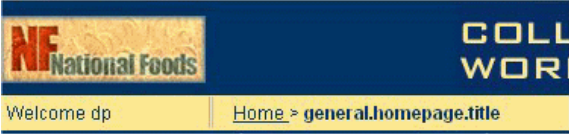

You can configure Demantra to detect large simulations and display a message to the user, to confirm that this is what the user wants to do. You use the following parameters:

| Parameter   | Purpose                                                                                                                                                                                                                                                         |
|-------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| SimWarnSize | Specifies the threshold size of a simulation that is large enough to trigger a warning message to the user. Specify this as a percentage of the total number of combinations.                                                                                   |
| SimMaxSize  | Specifies the threshold size of a simulation that is too large to run. If a user tries to perform a simulation of this size, Demantra displays a message and does not attempt the simulation. Specify this as a percentage of the total number of combinations. |
| MatrixCombs | Indicates the number of combinations currently in the mdp_matrix table.<br><br>This information can be useful in helping you to set SimMaxSize and SimWarnSize.                                                                                                 |

You should run some trial simulations on the solution hardware and set threshold values that are appropriate for the actual users.

## Solution Branding

The following parameters control titles throughout the Demantra solution:

| Parameter              | Description                                                                                                                                                                                           |
|------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| company.name           | Name of your company; the Workflow Engine uses this string in email when a workflow step fails.                                                                                                       |
| general.homepage.title | Title of the Collaborator Workbench home page, as used within the Collaborator Workbench title bar, as follows:<br> |
| general.title.text     | Title of the browser window when it displays Collaborator Workbench. For example:<br>                             |

## Threading

Demantra uses threading mechanisms in multiple places. Threading is a general mechanism that uses system resources more effectively to run several tasks in parallel.

- "Threading for the Attribute Update Mechanism"
- "Threading for the Update Mechanism"
- "Threading for Updating Parallel Values"
- "Threading for Promotion Copy/Paste"
- "Threading for Methods"

- "Threading in the Web Worksheets"
- "Threading in the Business Logic Engine"

### Threading for the Attribute Update Mechanism

This thread pool uses the following parameters:

| Parameter                            | Description                                                                                                               |
|--------------------------------------|---------------------------------------------------------------------------------------------------------------------------|
| threadpool.attributesUpdate.per_comb | Maximum number of threads that a single thread can use.                                                                   |
| threadpool.attributesUpdate.size     | Maximum number of allowed threads for this thread pool. This should be less than MaxDBConnections.                        |
| threadpool.attributesUpdate.time-out | Idle time-out period. This specifies how long (in milliseconds) a thread is left unused before it is ended automatically. |

### Threading for the Update Mechanism

The update mechanism saves data to the database. This thread pool uses the following parameters:

| Parameter            | Description                                                                                                                         |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------|
| MaxUpdateThreads     | Maximum number of allowed threads for the update mechanism. You should set this equal to the number of database server CPUs plus 1. |
| UpdateThreadtime-out | Idle time-out period. This specifies how long (in milliseconds) a thread is left unused before it is ended automatically.           |

### Threading for Updating Parallel Values

This thread pool uses the following parameters:

| Parameter              | Description                                                                                        |
|------------------------|----------------------------------------------------------------------------------------------------|
| threadpool.update.size | Maximum number of allowed threads for this thread pool. This should be less than MaxDBConnections. |

| Parameter                  | Description                                                                                                               |
|----------------------------|---------------------------------------------------------------------------------------------------------------------------|
| threadpool.update.time-out | Idle time-out period. This specifies how long (in milliseconds) a thread is left unused before it is ended automatically. |

### Threading for Promotion Copy/Paste

Another thread pool handles copying and pasting promotions. This thread pool uses the following parameters:

| Parameter                         | Description                                                                                                                          |
|-----------------------------------|--------------------------------------------------------------------------------------------------------------------------------------|
| threadpool.copy_paste.per_process | Maximum number of allowed threads for the copy/paste mechanism in any given process.                                                 |
| threadpool.copy_paste.size        | Maximum number of allowed threads for the copy/paste mechanism. This should be less than MaxDBConnections.                           |
| threadpool.copy_paste.time-out    | Idle time-out period. This specifies how long (in milliseconds) a copy/paste thread is left unused before it is ended automatically. |

### Threading for Methods

Another thread pool handles level methods. This thread pool uses the following parameters:

| Parameter                        | Description                                                                                                                                                       |
|----------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| threadpool.level_method.size     | Maximum number of allowed threads for methods. This should be less than MaxDBConnections.                                                                         |
| threadpool.level_method.time-out | Idle time-out period. This specifies how long (in milliseconds) a method thread is left unused before it is ended automatically. Recommended: 300000 (5 minutes). |

| Parameter                     | Description                                                                                                                                               |
|-------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------|
| threadpool.level_method.block | Specifies how the level methods should access this thread pool, either:<br><br>wait (wait for a free thread)<br><br>abort (do not wait for a free thread) |

### Threading in the Web Worksheets

The Web worksheets also use threading

| Parameter                     | Description                                                                                                                                                                                                                                                                 |
|-------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| threadpool.query_run.size     | Maximum number of allowed threads that Demantra can use to run a Web worksheet. If this number is missing or negative, the worksheet run mechanism does not use threads.<br><br>This should be less than MaxDBConnections. Also be sure to leave room for system processes. |
| threadpool.query_run.time-out | Idle time-out period. This specifies how long (in milliseconds) a worksheet thread is left unused before it is ended automatically.                                                                                                                                         |

### Threading in the Business Logic Engine

The Business Logic Engine uses threading as follows: The thread pool specifies the number of parallel BLE tasks, each of which loads a different combination of the worksheet, runs the calculation engine on it, and saves the data back to the database. The number of threads in the pool is affected by the system resources, mainly the number of CPUs that the machine has (each thread runs on a different CPU). The following parameters control this threading mechanism:

| Parameter         | Description                                                                                                                   |
|-------------------|-------------------------------------------------------------------------------------------------------------------------------|
| BLEThreadPoolSize | Maximum number of allowed threads for the Business Logic Engine.                                                              |
| BLEtime-out       | Idle time-out period. This specifies how long (in milliseconds) a BLE thread is left unused before it is ended automatically. |

## Workflow

The following parameters control the Workflow module:

| Parameter         | Description                                                                                                                                                                                                                                                                                                      |
|-------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| company.name      | Name of your company; the Workflow Engine uses this string in email when a workflow step fails.                                                                                                                                                                                                                  |
| execution.shell   | Applies to the Executable Step. This parameter specifies any prefix that is needed in order to run executable steps. For example, you may need to specify the following for Unix:<br><br>./                                                                                                                      |
| log.history       | The number of days for which workflow history is kept.                                                                                                                                                                                                                                                           |
| server.generalurl | URL for the workflow server, not including the portal/workflow directory.                                                                                                                                                                                                                                        |
| workflow.group    | Comma-separated list of groups whose users are authorized to log into the Workflow Editor. Use the group names as specified in the Business Modeler.<br><br>In order to log into the Workflow Editor, these users also must have System Manager permission level. See "Providing Access to the Workflow Editor". |

See also

"Solution Branding" "Collaborator Workbench" "Email"

## Worksheets

The following parameters affect the Web-based worksheets. They are grouped into several areas:

- "General Worksheet Behavior"
- "Worksheet Performance"
- "Worksheet Designer"

For another way to improve performance, see also "Managing Level Caching".

### General Worksheet Behavior

The following parameters control the default behavior of the Web-based worksheets

| Parameter                            | Description                                                                                                                                                                                                                             |
|--------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| AutoRunMode                          | Specifies whether a worksheet automatically reruns after any change in its definition. This parameter also specifies whether a worksheet is automatically run when it is opened in any way.                                             |
| client.enableOpenNoteWithDoubleClick | <p>Specifies whether users can access the notes dialog box by double-clicking within the worksheet table.</p> <p>In any case, it is always possible to access this dialog box by using the right-click menu, as in Microsoft Excel.</p> |

### Worksheet Performance

The following parameters affect the performance of the Web client:

| Parameter                    | Description                                                                                                                                                |
|------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ClientDDCacheCapacity        | Specifies the maximum number of distinct dropdown lists per worksheet that any client should cache. These caches are cleared when the worksheet is closed. |
| EnableWorksheetCaching       | Enables or disables the worksheet caching feature.                                                                                                         |
| EnableIncrementalLoading     | Enables the Demantra incremental loading feature, for faster worksheet reruns. There is no user impact apart from performance.                             |
| client.JREMaxMemory          | Maximum amount of memory (in MB) that JRE can use. The Web worksheets (Demand Planner Web, Promotion Effectiveness, and Settlement Management) use JRE.    |
| client.MaxCombsLoadChunkSize | Maximum number of combinations to load each time the user chooses to rerun a worksheet.                                                                    |



| Parameter          | Description                                                                                                                                                                                                                                                                                                                                                                                           |
|--------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| UseDateRangeMatrix | <p>Controls whether the system will use new internal data structures to improve the performance of worksheets that include promotions (or other general levels that have population attributes). If you enable this option, the largest benefit occurs in cases where promotions are long (and have many rows of data).</p> <p>The system uses these structures automatically for other purposes.</p> |

### Worksheet Designer

The following parameters control the defaults in the worksheet/content designer

| Parameter                          | Description                                                                                                              |
|------------------------------------|--------------------------------------------------------------------------------------------------------------------------|
| OpenWithContext                    | Specifies the default setting of the Open With Context setting of the worksheet designer.                                |
| client.worksheet.privateAccessType | Specifies the default setting of the public/private option in the worksheet designer.                                    |
| WorksheetDefaultDateChoiceMethod   | Controls the default start date for worksheets, either relative to today or relative to last loaded sales date.          |
| WorksheetDefaultSpan               | Specifies the default length of time for a worksheet, in base time units. Must be a positive, even number, 2 or greater. |
| ManualRefreshAsDefault             | Specifies the default setting of the Refresh Type caching option in the worksheet designer.                              |
| WorksheetCachingAsDefault          | Specifies the default setting of the Cache Worksheet Data check box in the worksheet designer.                           |
| PromoDefaultSpan                   | Specifies the default length of time for promotions created within a worksheet.                                          |

| Parameter                         | Description                                                                                                                                                                                                                                                                                                                                                                                                                             |
|-----------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| PromoDefaultStart                 | <p>Specifies the default start date for promotions created within a worksheet. Use one of the following values:</p> <ul style="list-style-type: none"> <li>• today (0)</li> <li>• last loaded sales date (1)</li> <li>• start date of the worksheet (2)</li> </ul>                                                                                                                                                                      |
| MaxAvailableFilterMembers         | <p>Specifies the maximum number of members that can be retrieved in the worksheet filter screen. If the user selects more members than allowed, a message asks the user to add further filtering.</p> <p>This limit helps to prevent users from creating worksheets with too many members (which can adversely affect performance).</p>                                                                                                 |
| SPF_Enable_Worksheet_Calculations | <p>Automatically calculates planning percentages during worksheet updates. Updates to relevant series trigger BOM tree propagation if set to yes.</p> <p><b>Note:</b> Enabling this parameter may negatively affect worksheet performance. When set to No (disabled), propagation occurs when the user modifies worksheet values. However, the new planning percentages are only displayed after the end user reruns the worksheet.</p> |

See also

"Collaborator Workbench"

## Parallel Hints for Worksheets

Some worksheets access a large amount of data which can cause them render slowly. A parallel hint can be implemented to improve performance for such worksheets. A hint specifies the number of threads used by the worksheet query and can be applied to two parts of the worksheet: generation of the combinations and retrieval of data to be displayed in the worksheet.

**Warning:** Use caution when implementing worksheet hints. When a

large number of users simultaneously access the database, too many hints can overwhelm the database connection pool and substantially decrease performance.

Setting a hint requires manually adding a row to the WORKSHEET\_HINTS table; this task should be performed only by an experienced system or database administrator.

The WORKSHEET\_HINTS table contains the columns in the table below.

| QUERY_ID | USER_ID | CONTEXT_ID | POPULATION_HINT                                                                                                                                                              | DATA_HINT                                                                                                                                              |
|----------|---------|------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------|
| Q        | 0       | 0          | The hint to be used for worksheet Q with any user and any context in the Population SQL                                                                                      | The hint to be used for worksheet Q with any user and any context in the Population SQL                                                                |
| Q        | U       | 0          | The hint to be used for worksheet Q with user U and any context in the Population SQL. For all other users besides U, the generic hint defined above is used, if one exists. | The hint to be used for worksheet Q with user U and any context in the Data SQL. For all other users beyond U, the generic hint defined above is used. |

| QUERY_ID | USER_ID | CONTEXT_ID | POPULATION_HINT                                                                                                                                                                                                           | DATA_HINT                                                                                                                                                                                                        |
|----------|---------|------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Q        | U       | C          | The hint to be used for worksheet Q with user U and context C in the Population SQL. For all other contexts and for the same user U, the above row is used. For all other users, the generic hint is used, if one exists. | The hint to be used for worksheet Q with user U and context C in the Data SQL. For all other contexts and for the same user U, the hint in the row above is used. For all other users, the generic hint is used. |

The POPULATION\_HINT and DATA\_HINT columns contain the actual hint as an SQL string. For example, to specify that the query should use 8 threads when accessing the branch\_data table, this field would be "parallel(branch\_data 8)".

The QUERY\_ID, USER\_ID, and CONTEXT\_ID columns specify the circumstances when this hint would be used.

QUERY\_ID specifies the internal ID of the worksheet for which the hint should be applied.

USER\_ID specifies for which user the hint will apply when opening the worksheet. When this field is set to zero, the hint will apply to all users.

CONTEXT\_ID specifies the level member name; it corresponds to the level member that is selected when a worksheet is opened with the "Open With" menu option. If the CONTEXT\_ID is zero, then the hint will apply to all contexts. If the worksheet is opened without "Open With," this setting is ignored.

## Languages

### Overview

Oracle Demantra is available in multiple languages. At installation, you choose a default language and application objects and platform data appear in that language. For the available languages, see *Oracle Demantra Installation Guide*.

**Important:** For non-ascii languages, see the note regarding text editor capability in the section Logging Messages of the Application Server.

These application objects appear in the default language:

- Series
- Series groups
- Series lookup tables
- Levels
- Level members
- Level attributes
- Level methods
- Worksheets
- Workflows
- Workflow messages

These platform data appear in the default language:

- Button labels
- Tool tips
- Object descriptions
- Menu items
- Error messages
- Printed user's guides and web help; non-user's guides are available only in U.S. English

When you are logging in, Oracle Demantra displays the language for your instance. You cannot change it.

Oracle recommends that if you extend the Oracle Demantra model to include additional series, worksheets, and workflows, do so in the default language.

You can edit worksheet data using the offline worksheet mechanism; however, you cannot edit its definition, for example, the displayed series, the applied filters, and the types of graphs.

When you take a worksheet offline, you see its objects, for example, series, levels, and drop-down menus, in the default language; see *Oracle Demantra User's Guide*.

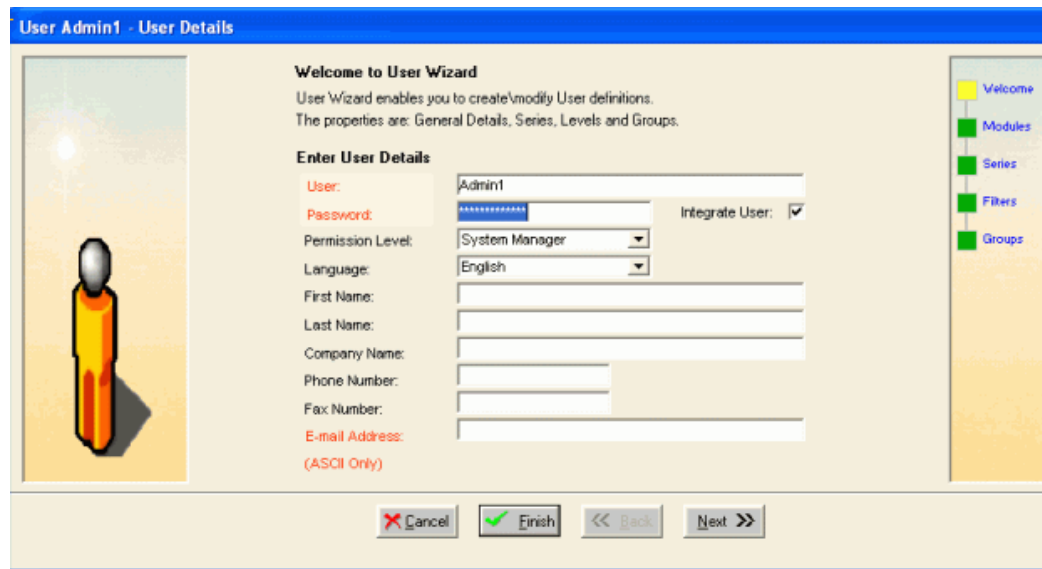
When you export a worksheet to Microsoft Excel, the data appears in the default

language of the user who exported it.

Oracle recommends that you import item master data in the default language. This includes both regular levels, for example, item and location, and general levels, for example, promotions, supply plans, and bills of material. However, there is no requirement that all of your item master data must be in one language.

### ASCII-only Fields in the Desktop Application

Most Business Modeler and Demand Planner fields work with non-ASCII characters, for example, characters in languages, Chinese, Japanese, Korean and Russian. However, some are only available for ASCII characters. Oracle Demantra indicates the fields that are only available for ASCII input by red field titles.



### Date Display

Business Modeler series Display Format results in the same number of digits per date element across locales but the order of date elements and the separator complies with the locale.

For example, you operate in a French locale. This table shows how the Business Modeler choice maps to the Web display.

| Business Modeler Format | Localized Format |
|-------------------------|------------------|
| MM/dd/yyyy              | dd.MM.yyyy       |
| MM/dd/yy                | dd.MM.yy         |
| MM-dd-yyyy              | dd.MM.yyyy       |

| <b>Business Modeler Format</b> | <b>Localized Format</b> |
|--------------------------------|-------------------------|
| dd.MM.yyyy                     | dd.MM.yyyy              |
| dd.MM.yy                       | dd.MM.yy                |
| dd/MM/yyyy                     | dd.MM.yyyy              |
| dd/MM/yy                       | dd.MM.yy                |
| E, MMM. dd yyyy                | E, MMM. dd yyyy         |
| E MM/dd/yyyy                   | E MM.dd.yyyy            |
| E MM-dd-yyyy                   | E MM.dd.yyyy            |
| E dd/MM/yyyy                   | E dd.MM.yyyy            |
| E dd.MM.yyyy                   | E dd.MM.yyyy            |
| MM-dd-yy                       | dd.MM.yy                |

The default language does not typically have an effect on the display of dates, times and numbers.

Oracle Demantra derives the default date display format from the operating system locale. Locale refers to the location where your instance conducts business. For example, you set the locale in operating system Microsoft Windows XP in the Control Panel > Regional and Language Settings.

To refine your date display, use Worksheet Designer, navigate to region Time, click Advanced, and modify the advanced time settings.

### **Number and Currency Display**

Oracle Demantra derives the default number and currency display format from the operating system locale. Locale refers to the location where your instance conducts business. For example, you set the locale in operating system Microsoft Windows XP in the Control Panel > Regional and Language Settings.

The Business Modeler series display format for a number provides a representation of how it is displayed. For example, it provides the standard display formats to reflect the number of decimal places and uses this to determine the need for radix points and thousand separators. Locale controls the delimiters. If a series display property is #,####.##, it displays in:

- North American locales as 1, 234.56
- European locales as either 1.234,56 or 1 234,56

However, determining the currency from the locale may not always be accurate, especially if you report in a currency different from the operating system locale. For example:

- Your default language may be used in multiple areas with different currencies. You have an instance using language Chinese Simplified, but you could report currency in, for example, China Yuan Renminbi (CNY) or Singapore Dollars (SGD).
- If you have a default language that corresponds to one currency, you may want to report it in a different currency. Your language is French (Canada) that corresponds to currency Canadian Dollars (CAD), however, you want to report in United States Dollars (USD)

If your business requires reporting in multiple currencies or a currency different from the locale, Oracle recommends that you:

- Use series titles as a means of clarifying currency, for example, Budget (USD) instead of Budget \$.
- Display the currency without a currency indicator or symbol, for example, Budget (USD) = 2,000 instead of Budget = \$2,000

## Handling Customer-Specific Strings

In order to correctly display customer-specific strings in non-English languages, the operating system client locale, and several database parameters, must be set appropriately. These user-defined strings may include worksheet names, workflow names and messages, and strings for Dynamic Open Link compatibility.

Ensure the following parameters are set as listed below:

- NLS\_LENGTH\_SEMANTICS = CHAR
- NLS\_CHARACTERSET = AL32UTF8
- NLS\_LANG = *language\_territory.characterset*

For details about configuring Oracle to support Globalization, see Oracle Database Installation Guide for Microsoft Windows or the Oracle Database Globalization Support Guide.

## Configuring User Productivity Kit (UPK)

If you have licensed and installed UPK, perform the following to launch UPK from



Oracle Demantra Web help.

1. Install and configure UPK on a Web server. Refer to UPK documentation for details.
2. Launch Business Modeler, and then locate the LaunchUPK parameter.
3. In the Value field, enter a valid URL to launch UPK.  
For example: `http://server name/virtual directory name/index.html`
4. Save the changes.
5. Log in to Demantra, and then click Help from Collaborator Workbench, Workflow Manager, or from within a worksheet.
6. Click the "UPK" link. The Player should launch successfully.



---

## Customizing Demantra Web Pages

This chapter describes how to customize the Demantra Web pages.

This chapter covers the following topics:

- Logging onto the Collaborator Workbench Administrator
- Configuring Menus in Collaborator Workbench
- Running Oracle Executables from Collaborator Menus
- Configuring the Panes
- Specifying Content Pane Security
- Replacing Default Demantra Graphics
- Customizing the Collaborator Workbench Login Page
- Configuring Links in Collaborator Workbench

### Logging onto the Collaborator Workbench Administrator

You use the Collaborator Workbench Administrator to control access to menu items.

#### **To log onto the Collaborator Workbench Administrator:**

1. Open the administration login page:

`http://server name/virtual directory/portal/adminLogin.jsp`

For example:

`http://frodo/demantra/portal/adminLogin.jsp`

2. Enter the user name and password and click Log on.

Demantra displays the Administration page, which includes the following choices:

[Define Menus](#)  
[Define Program Groups](#)  
[Define Program Permissions](#)  
[Define Content Security](#)  
[Default Pane Layout](#)  
  
[Logout](#)  
[Login to Collaborator Workbench](#)  
[Login to Demantra Anywhere](#)

See also

"Customizing Demantra Web Pages"

## Configuring Menus in Collaborator Workbench

You can configure the Planning Applications and Tools and Applications menus, which are in the tool bar at the top of the Collaborator Workbench page.

### To configure Collaborator Workbench menus:

1. Log into the Collaborator Workbench Administrator. See "Logging onto the Collaborator Workbench Administrator".
2. Click Define Menus.

The system displays a page showing the current contents of Planning Applications and Tools and Applications menus:

| "Planning Applications" menu bar  |                     |                                                |
|-----------------------------------|---------------------|------------------------------------------------|
| "Desktop" group                   |                     |                                                |
|                                   | Item Title          | Description                                    |
| <input type="checkbox"/>          | Demand Replenisher  | Demand Replenisher                             |
| <input type="checkbox"/>          | Demand Planner      | Demand planning software                       |
| <input type="checkbox"/>          | Member Management   | Demand planning software - member management   |
| <input type="checkbox"/>          | Chaining Management | Demand planning software - chaining management |
| <input type="checkbox"/>          | Business Modeler    | Business Modeler                               |
| "Tools and Applications" menu bar |                     |                                                |
| "Web Links" group                 |                     |                                                |
|                                   | Item Title          | Description                                    |
| <input type="checkbox"/>          | Demantra Web Site   | Demantra Web Site                              |
| <input type="checkbox"/>          | Workflow            | Workflow                                       |

#### To add a menu item:

1. Within either the Planning Applications or Tools and Applications section, click the Add button.

A page appears prompting you for information about the item to add.

2. In the Item Title field, specify the title of the menu item as it should appear in the menu.
3. For Type, choose one of the following options:

---

|                       |                                                                                                                         |
|-----------------------|-------------------------------------------------------------------------------------------------------------------------|
| Planning Applications | Starts a Demantra desktop product.                                                                                      |
| Program Initiation    | Starts an ordinary executable.                                                                                          |
| Web link              | Opens a Web page.                                                                                                       |
| Encrypted User/PWD    | Starts a product with encrypted user name and password. Do not use for a product installed on a Citrix Metafile server. |
| Special Citrix        | Starts a product installed on the Citrix Metafile server. Sends an encrypted user name and password.                    |

---

4. Complete the rest of the fields as follows:

---

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|-------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Optional description of this menu item.                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Program     | <p>Path and filename of a file to be executed. This field is hidden if the type is Web link.</p> <p>See "Running Oracle Executables from Collaborator Menus" for options.</p>                                                                                                                                                                                                                                                                                                                                       |
| Target      | A URL. This field is visible only if the type is Web link.                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Parameters  | <p>Any command line arguments that the executable file accepts. For example, suppose that the executable is SPM.exe and it accepts arguments to bypass the login, as follows:</p> <p><i>SPM.exe /autologin userid=username pwd=password</i></p> <p>In this case, specify Program as SPM.exe and use the following parameter string:</p> <p><i>/autologin userid=username pwd=password</i></p> <p>For the syntax to run specific Demantra executables, see "Running Oracle Executables from Collaborator Menus".</p> |

---

5. Click OK to close the popup page and save your changes.

#### **To edit a menu item:**

1. Check the check box next to the menu item.
2. Click the Edit button.  
A page appears prompting you for information about the item to change.
3. Complete the fields as in "To add a menu item".
4. Click OK to close the popup page and save your changes.

#### **To delete a menu item:**

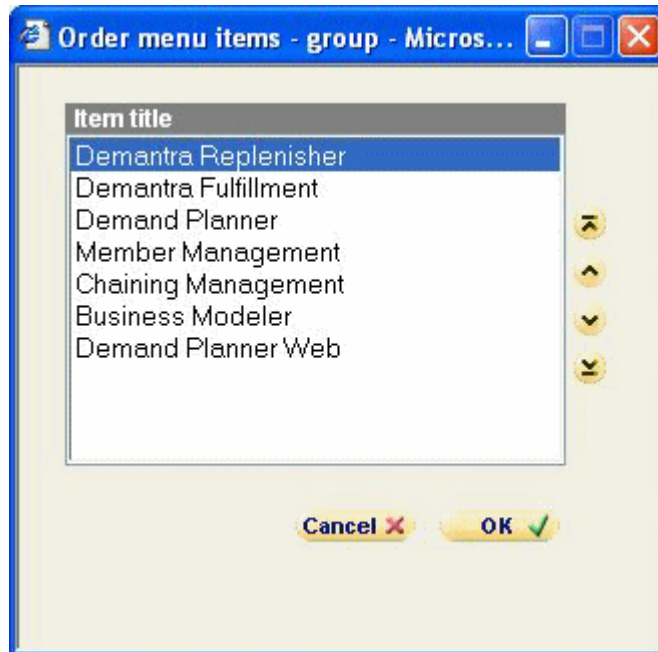
1. Check the check box next to the menu item.
2. Click the Delete button.  
A warning message appears.

3. Click OK to confirm the deletion.

**To change the order of the items in a menu:**

1. In the section corresponding to that menu, click the Order button.

The system displays a popup page where you can change the order of the items.



2. Select an item and click an arrow button to move the item up or down in the list.
3. When you are done, click OK.

See also

"Running Oracle Executables from Collaborator Menus"

## Running Oracle Executables from Collaborator Menus

When you configure the Planning Applications and Tools and Applications menus, you typically add menu items that launch Demantra executables.

This section lists the basic syntax needed in common situations.



| Executable to launch                                  | Settings to use in Menu Item dialog box*                             |                                                                                                                                                                                                                           |
|-------------------------------------------------------|----------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                       | Program                                                              | Parameters/Notes                                                                                                                                                                                                          |
| Member Management                                     | Path and filename of the dp.exe file in your installation            | tools/member management                                                                                                                                                                                                   |
| Chaining Management                                   | Path and filename of the dp.exe file in your installation            | tools/chaining management                                                                                                                                                                                                 |
| Demand Planner, bypassing login screen                | Path and filename of the dp.exe file in your installation            | /autologin userid= <i>user</i> pwd= <i>password</i><br><br>Here <i>user</i> is the user ID and <i>password</i> is the corresponding password.                                                                             |
| Promotion Effectiveness Analytical Engine             | Path and filename of the EngineManager.exe file in your installation | <i>mode profile_ID</i><br><br>Here, <i>mode</i> is either:<br><br>1=batch mode<br><br>99=simulation<br><br>And <i>profile_ID</i> specifies the engine profile to use. For additional parameters, see "Engine Parameters". |
| * In all cases, Type should be Planning Applications. |                                                                      |                                                                                                                                                                                                                           |

For information on how to add menu items to Collaborator Workbench, see "Configuring Menus in Collaborator Workbench."

## Configuring the Panes

### To specify your pane configuration:

1. Log into the Collaborator Workbench Administrator. See "Logging onto the Collaborator Workbench Administrator".

Oracle Demantra displays the Administration page, which includes the following choices:

[Define Menus](#)  
[Define Program Groups](#)  
[Define Program Permissions](#)  
[Define Content Security](#)  
[Default Pane Layout](#)  
  
[Logout](#)  
[Login to Collaborator Workbench](#)  
[Login to Demantra Anywhere](#)

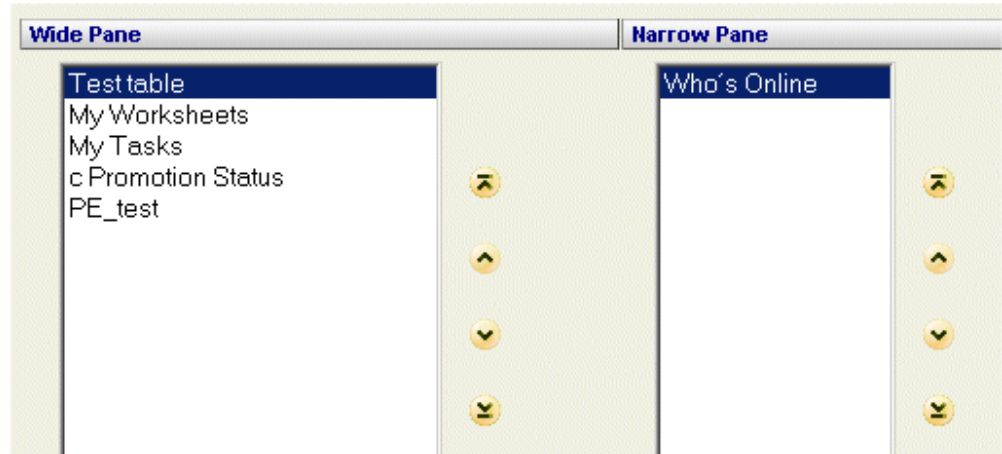
2. Click Default Pane Layout.

The Personalize - Modules page appears. This page contains two lists: one for items that can be displayed in the wide pane and one for items that can be displayed in the narrow pane.

| Wide Pane                                                                                                                                                                                                                                                                                                                                                                                                                  | Narrow Pane                                                                                                                                                                                                                                        |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <p><b>Select All</b></p> <ul style="list-style-type: none"><li><input type="checkbox"/> c. Revenue per Regional Manager</li><li><input checked="" type="checkbox"/> My Worksheets</li><li><input checked="" type="checkbox"/> My Tasks</li><li><input checked="" type="checkbox"/> c Promotion Status</li><li><input checked="" type="checkbox"/> PE_test</li><li><input checked="" type="checkbox"/> Test table</li></ul> | <p><b>Select All</b></p> <ul style="list-style-type: none"><li><input type="checkbox"/> c. 12 Months Revenue by Region</li><li><input checked="" type="checkbox"/> Who's Online</li><li><input type="checkbox"/> c. MY Private Label SKU</li></ul> |

These lists include the following:

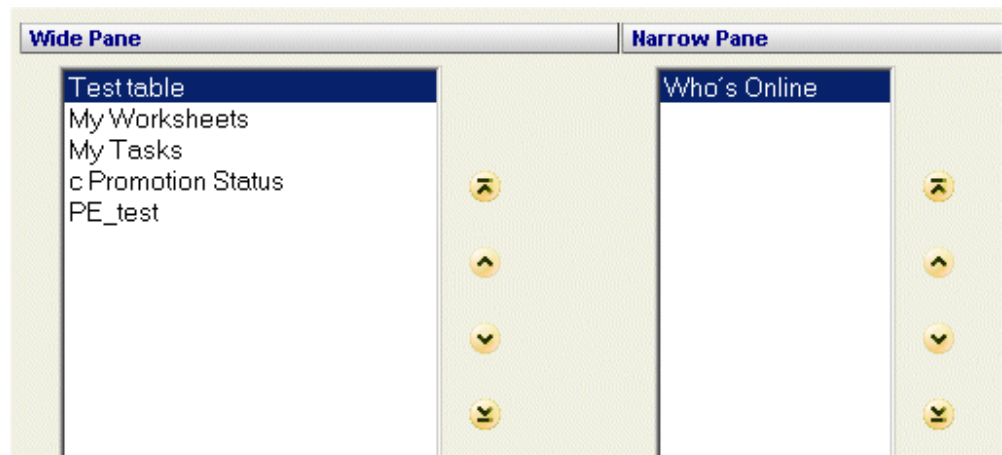
- My Tasks and My Worksheets, which can be displayed only in the wide pane
- Who's Online, which can be displayed only in the narrow pane
- Worksheets that have been defined as content and to which you have access. When a worksheet is defined as content, it is defined as belonging to the wide pane or the narrow pane.



3. In each list, use the check boxes to select or deselect the modules that you want to see.

4. Click Next.

The Personalize - Order page appears. Like the previous page, this page has one list for the wide pane and one for the narrow pane.



5. Select a module and then click the up or down buttons to change its position in the list.

The order here is the order in which these modules are shown in Collaborator Workbench.

6. Click Next.

The next page summarizes your choices. You can return to the previous pages to make further alterations.

- Click Finish to save your changes. Or click Back to go back to the previous pages.

See also

"Specifying Content Pane Security"

## Specifying Content Pane Security

You can control access to the different Collaborator Workbench panes (My Tasks, My Worksheets, and Who's Online).

### To specify access to Collaborator Workbench panes:

- Log into the Collaborator Workbench Administrator. See "Logging onto the Collaborator Workbench Administrator".

The Administration page appears.

- Click Define Content Security.

The system displays a table with one row for each user. Here you specify which panes to make available to each user.

|              | My Tasks                            | My Worksheets                       | Who's Online                        |
|--------------|-------------------------------------|-------------------------------------|-------------------------------------|
| ...          | Select All                          | Select All                          | Select All                          |
| Abby_Rose    | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> |
| Bill         | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> |
| Bill_Feldman | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> |
| dp           | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> |
| ERP          | <input type="checkbox"/>            | <input type="checkbox"/>            | <input type="checkbox"/>            |
| Guy_Cabotini | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> |
| Guy_Vidlov   | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> |
| Jeff_Wilson  | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> |
| Maria_C      | <input type="checkbox"/>            | <input type="checkbox"/>            | <input type="checkbox"/>            |
| Mike_Kim     | <input type="checkbox"/>            | <input type="checkbox"/>            | <input type="checkbox"/>            |
| Misty        | <input type="checkbox"/>            | <input type="checkbox"/>            | <input type="checkbox"/>            |
| Sheron_Crean | <input type="checkbox"/>            | <input type="checkbox"/>            | <input type="checkbox"/>            |

- Do one of the following:
  - Check the check box for a pane to grant user access to the user.
  - Clear the check box for a pane to deny access to the user.
- Click Finish.

See also

"Configuring the Pane Configuration"

## Replacing Default Demantra Graphics

The Web-based Demantra products contain default images that you can replace with

your organization's own designs. To do so, just back up the default images and substitute your own image files, giving them the same filenames as listed here.

The graphic files are in the following directory:

*Demantra\_root/Collaborator/portal/images*

You can replace any of the graphics files in this directory. If you replace the default graphics with other graphics that have the same width and height, those graphics will fit on the page without the need for any further editing. If your graphics files have different dimensions, you may need to edit the corresponding page to accommodate them.

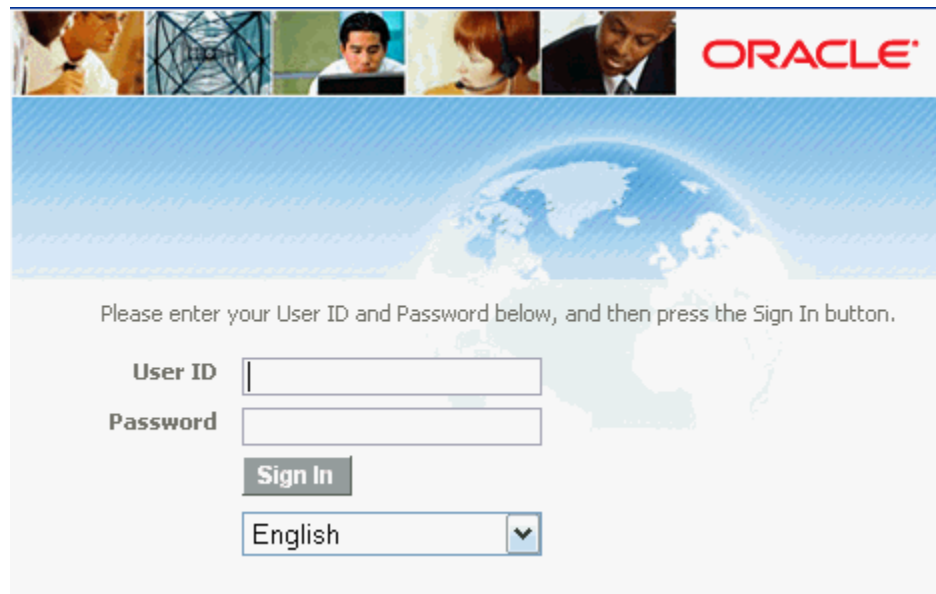
### **Collaborator Workbench Splash Screen**

The splash screen uses the graphic collaborator\_splash.gif.

### **Collaborator Workbench Login Page**

On the login page, the most commonly replaced images are the following:

- customerLogo.gif
- customerTxt.gif
- customerPics.gif
- collaboratorTitle.gif
- collaboratorTxt.gif



### **Collaborator Workbench Main Page**

On the main page, the most commonly replaced images are as follows:

- customer\_logo.jpg
- collaborator\_logo.gif



## Customizing the Collaborator Workbench Login Page

The login page is Demantra\_root/Collaborator/portal/loginpage.jsp

You can edit this page and you can redesign the layout and design as you wish.

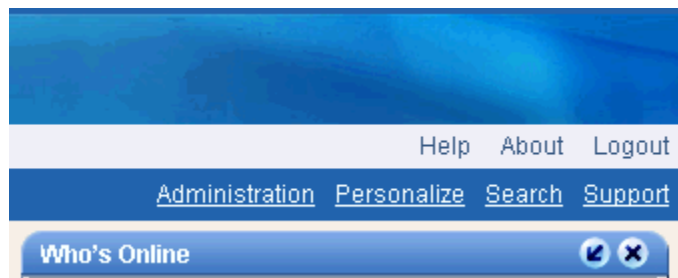
**Caution:** A basic knowledge of HTML is required to perform this task.

However, the following code must be retained, because this provides the login functionality:

```
<TD>
<!-- The login area (username, password, language, login)--> <%@ include
file="loginarea.jsp" %>
\</TD>
```

## Configuring Links in Collaborator Workbench

The main Collaborator Workbench page provides a set of default links, some of which are configurable. These links are located on the second toolbar.



To configure these links, edit the file secondbar.jsp.

**Caution:** A basic knowledge of HTML is required to perform this task.

Links can be added, but the layout of the page must not be changed. Configurable links are marked by href = "#".

Link	Comments
Personalize	Must not be changed. Vital functionality depends on this link
Site Map	This is an empty link that can be customized or removed as required.
Search	This is an empty link that can be customized or removed as required.
Home	This performs a logout and redirects to the login page. This would typically be reconfigured to link to the customers' home page.
Support	This is an empty link that can be customized or removed as required. This would typically be used to provide an email link to the webmaster.

Also see "Collaborator Workbench".





---

## Configuring Rolling Data

This chapter describes how to roll selected data, saving a copy of the current version of that data.

This chapter covers the following topics:

- About Rolling Data
- Creating or Modifying Rolling Data Profiles
- Executing Rolling Data Profiles
- Creating Rolling Data Profile Groups

### About Rolling Data

It is often useful to be able to see older data and possibly compare it with the current data. In the case of forecasts, the Analytical Engine automatically saves older versions. If you need access to older versions of *other* data, however, you must explicitly instruct Demantra to save the data.

To do so, you use the Business Modeler to make a copy of the original data (usually a series) for later use. You define one or more *rolling data profiles*, each of which associates a source series (or a server expression) with a target series.

- The target series must already exist.
- The target series should usually be configured almost the same way as the source series (except for the hint message and so on). See "Creating a New Series Based on an Existing Series."
- The target series must have an update field in the same table as the source series; see "Specifying Data Properties of a Series."
- You can use any kind of series (numeric, date, or string), but the

target and source series must be of the same type.

Then you configure a *rolling data session*, which specifies a set of rolling data profiles to run. The rolling data session specifies which profiles are active.

You can execute the active profiles from within the Business Modeler or from a workflow.

## Creating or Modifying Rolling Data Profiles

### To create or modify a rolling data profile:

1. Select Engine > Rolling Data.

The Rolling Data screen appears.

The screenshot shows the 'Rolling Data' window. On the left, a 'Profiles' list contains 'Rolling Profile Test' with a checkmark. The main area has fields for 'Profile Name' (Rolling Profile Test) and 'Profile Description' (Copy data from test-base to test-archive). Below these are 'Source' and 'Target' sections. The 'Source' section has radio buttons for 'Series' (selected) and 'Server Expression'. The 'Series' dropdown is set to 'test-base' and 'Source's Period' is 'History & Forecast'. The 'Target' section has a 'Series Name' dropdown set to 'test-archive'. At the bottom are buttons: Close, Save, Refresh, Delete, Insert, Configure Session..., and Execute.

2. Do one of the following:
  - To select an existing profile, click the profile in the Profiles list.
  - To create a new profile, click Insert.

The screen displays a check mark next to the profile that you are editing.

3. Type a unique name and an optional description for the profile.
4. In the Source area, do one of the following:
  - Select a source series from the Series drop-down list.

**Note:** If the server expression for the series uses the round function, note that the rounding occurs before the data is rolled forward. In this case, you might have slight errors at aggregated levels.

- Type a server expression into the Server Expression field. In this case, click Verify Expression to check the expression. The expression must be an aggregating SQL expression; see "Syntax of Server Expressions".
5. Specify the period association of this data source by selecting an option from Source's Period.
  6. In the Target area, select a series from the Series Name drop-down box. Make sure that the target series is of the same data type as the source series or the source expression.
  7. Click Save.

**To delete a rolling data profile:**

1. Click the profile in the Profiles list.
2. Click Delete.
3. Click OK to confirm the deletion.

See also

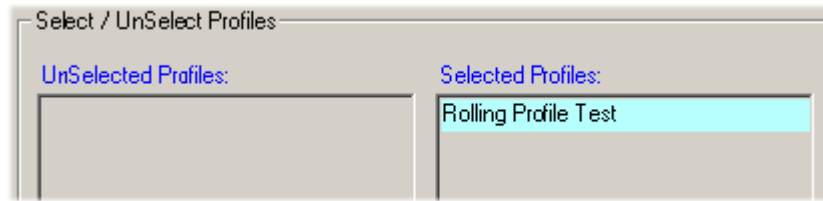
"About Rolling Data"

## Executing Rolling Data Profiles

**To specify the active rolling data profiles:**

1. Select Engine > Rolling Data.
2. Click Configure.

The Configure Rolling Session screen appears.



3. To include a profile to run, drag it from the left list to the right list. Or to exclude a profile from the next session, drag it from the right list to the left list.
4. To change the order in which the profiles should be executed, drag the profiles up or down within the left list as required.
5. Click Save.

### To execute the active rolling data profiles:

You can execute the active profiles in several different ways:

- You can execute the active profiles from within the Business Modeler. To do so, select Engine > Rolling Data. Then click Execute. Demantra immediately starts running the profiles that are currently listed in the Configure Rolling Session window.
- You run the EXECUTE\_PROFILES procedure from within a workflow.
- You can run the INSERT\_UNITS procedure. Depending on the value of the RunInsertUnits engine parameter, this procedure runs the EXECUTE\_PROFILES procedure.

In any case, they are executed in the order in which they are listed in the Configure Rolling Session dialog box. For each active profile, Business Modeler copies the source data into the target series. Data for any given time bucket is copied into the same time bucket in the target series.

See also

"About Rolling Data"

"Preparing the Database"

## Creating Rolling Data Profile Groups

Rolling profiles enable archiving or copying older data and comparing it with current data. (See About Rolling Data, page 45-1). Rolling Profile Groups facilitate data maintenance, by allowing you to organize profiles into business-relevant groups.

To configure a rolling profile group you must insert records into database tables via SQL. Therefore, this should only be performed by an experienced implementer or

database administrator.

### Rolling Profile Group Tables

Two tables are used to establish rolling profile groups and associate them with group member profiles. The ROLLING\_GROUPS table establishes the name and ID of a rolling group, as shown below with sample data:

ROLLING\_GROUPS

ROLLING_GROUP_ID	NAME	DESCRIPTION	IS_ACTIVE	APPLICATION_ID
1	Demand Mgmt	Demand Mgmt profiles.	1	ROLLING_PROFILE:1
2	SPF	SPF profiles	1	ROLLING_PROFILE:2

The ROLLING\_GROUP\_PROFILES table is a one-to-many linking table associating the rolling profile group to one or more rolling profiles, as shown below with sample data.

ROLLING\_GROUP\_PROFILES

ROLLING_GROUP_ID	PROFILE_ID
2	144
2	145
2	156
1	63
1	64

### Stored Procedure: EXECUTE\_PROFILES

The stored procedure EXECUTE\_PROFILES() allows an optional parameter to be passed in specifying the name of a rolling profile group. If no parameter is passed, then the active rolling data profile series is run. (If there is no active rolling data profile series, the following error message is displayed: "There are no active profiles for ROLLING\_UPDATES".)

#### Procedure for configuring a rolling profile group

1. Using an SQL tool for directly accessing the database, insert a new record into the

ROLLING\_GROUPS table.

2. Update the field ROLLING\_GROUP\_ID to a unique value.
3. Enter the group's name into the NAME field ("DM" and "SPF" are pre-seeded examples).
4. These two are the only required fields.
5. Insert a new record into the ROLLING\_GROUP\_PROFILES table for each profile you want to become a member of the group.
6. Update the field ROLLING\_GROUP\_ID with the value of the group's corresponding field of the same name from the previous table.
7. Update the field PROFILE\_ID with the id of the profile you want to add to the group.
8. After these two tables are updated, you must go to the Workflow Editor and create a Workflow step to call the stored procedure. The procedure name will be "execute\_profiles." In the parameters section, add the name of the new rolling profile group.

---

## GL Synchronization

This chapter describes the GL synchronization feature and how to configure this feature.

This chapter covers the following topics:

- Overview

### Overview

The GL synchronization feature allows for the automatic synchronization (copying) of data from a GL source data series to a sales\_data target series. Synchronization is needed to support export of those series which previously had elements of multiple data tables, and to support some configure-to-order calculations. This functionality aggregates data across the gl dimension for each item, location, and sales date combination. The source series server definition determines which aggregation function is used: sum, average, min, max, etc. The aggregated data is then copied into the target sales\_data series.

### GL Synchronization Configuration

1. Make sure to apply the patch for the GL Synchronization feature
2. Configure the synch\_series table with source series, target series and profile ids. The source series and target series columns contain values of the computed\_name field of the computed\_fields table. Information from the source series will be copied into target series.
3. The profile id column holds a number that is used to group series for synchronization. Determining which synchronization profiles can be called is achieved by setting the SynchDefaultProfile system parameter. The following example shows how to configure the synch\_series table:

SOURCE_SERIES	TARGET_SERIES	PROFILE_ID
(sales_data)	(GL or sales_data)	(number)
indirect_lift	indirect_lift_sd	1
direct_lift	direct_lift_sd	1
evt_cost	evt_cost_sd	1
CTO_DEP_DEM_EXIST	dep_demand_exist_sd	2
fcst_consensus	fcst_consensus_copy	3

Note that in this example profile 1 copies series from promotion\_data to sales\_data, profile 2 copies series from cto\_data to sales\_data, and profile 3 copies series from sales\_data to sales\_data. It is possible for a profile's source series to belong to different GL tables.

4. Run the procedure `synch.update_synch_objects` after updating the table `synch_series`. Executing this procedure is required to update objects used for synchronization. Make sure to rerun `synch.update_synch_objects` every time changes are made to the `synch_series` table.
5. No series may appear more than once in the `TARGET_SERIES` column. A series may appear multiple times in the `SOURCE_SERIES` column.

**Note:** It is recommended that the target series is marked as not editable in Business Modeler.

## Synchronization Touchpoints

Synchronization may require a variety of activation points due to the many ways in which data can be modified in Demantra. Below is a partial list of touch points and recommendations on how to execute synchronization. Regardless of the touch point, synchronization will run as a net change. Combinations which do not require synchronization will not have data copied to Sale\_data.

### Batch Synchronization

This will typically be called from a workflow which will include a step calling the procedure `synch_run_batch`.

- Procedure will execute batch synchronization.
  - Calls to the procedure will have syntax `synch.run_batch(profile_id, num_jobs)`



- The `profile_id` parameter controls which synchronization profiles are synched. This parameter is a comma-separated list. If no parameters are passed, the value in parameter `SynchDefaultProfile` will be used instead..
- The `num_jobs` parameter determines how many parallel jobs are created for each GL that synchronized. The default is 4.
- Batch synchronization will execute on all item location combinations for which the relevant data has changed since the last synchronization.

### **Worksheet Synchronization**

This should be added for cases where worksheet updates should trigger a synchronization process.

- Is disabled by setting `SynchEnableAdHoc` parameter to No
- Adds the custom step `com.demantra.workflow.step.SynchUpdateHook` to end of the "update data" workflow
  - The worksheet hook will run GL Synchronization when combinations are updated using manual updates.
  - Synchronization profile will be based on `SynchDefaultProfile` parameter

### **Accept Simulation Synchronization**

- Is triggered when user Accepts Simulation
- Runs a GL Synchronization on data that was changed as a result of simulation
- Is disabled by setting `SynchEnableAdHoc` parameter to No
- The synchronization profile is based on `SynchDefaultProfile` parameter

### **Level Method Synchronization**

Is applied when executing a level method will necessitate synchronization of the information affected by the level member on which method is run.

- Adds a procedure step to the workflow being executed by the method
  - Step follows other steps in workflow
  - Step executes the procedure `synch.post_method_synch`
  - Procedure is passed parameters (`level_id`, `member_id`, `profile_id`)
  - The parameters `level_id`, and `member_id` can be passed from the workflow using `#level_id#` and `#member_id#` notation.

- The `profile_id` parameter determines which profiles the method will run on. If no value is specified, the `sys_param SynchDefaultProfile` will be used.

### **Delete Method Synchronization**

Is applied when synchronization needs to run when members are deleted. If a GL member is deleted, that GL member's population needs to be resynched to `sales_data`.

- Adds a procedure step to run the procedure `synch.pre_delete_synch` before delete step
- The procedure will have the following syntax `pre_delete_synch( level_id, member_id, profile_id )`
- The parameters `level_id`, and `member_id` can be passed from the workflow using `#level_id#` and `#member_id#` notation.
- The `profile_id` parameter determines which profiles the method will run on. If no value is specified, the `sys_param SynchDefaultProfile` will be used.

### **Edit Member Method Synchronization**

Is applied when editing a level member should trigger synchronization. This solution incorporates the solutions used for Level Method Synchronization and Delete Method synchronization

- Adds a procedure step to run the procedure `synch.pre_delete_synch` before delete step
- The procedure will have the following syntax `pre_delete_synch( level_id, member_id, profile_id )`
- Add procedure step to run the procedure `synch.pre_delete_synch` before delete step
- The procedure will have the following syntax `pre_delete_synch( level_id, member_id, profile_id )`
- The parameters `level_id`, and `member_id` can be passed from the workflow using `#level_id#` and `#member_id#` notation.
- The `profile_id` parameter determines which profiles the method will run on. If no value is specified, the `sys_param SynchDefaultProfile` will be used.

### **Duplicate Promotion and Copy/Paste Synchronization**

The synchronization of Duplicate Promotion and Copy/Paste will depend greatly on the levels and members for which this is executed. In cases where these processes require synchronization, custom configuration will be required.

---

## Performing Constraint Profit Optimization

This chapter describes how to use the Constraint Profit Optimizer.

This chapter covers the following topics:

- About Constraint Profit Optimization
- Creating or Modifying an Optimization Profile
- Deleting a Constraint Optimization Profile
- Running a Constraint Optimization Profile

### About Constraint Profit Optimization

Within the Business Modeler, the Constraint Profit Optimizer enables distributors and retailers to make the most effective use of available storage and display space throughout the supply chain. Demand forecasts for specific products are linked to space available, stock on hand, predetermined minimum and maximum levels, and profit scales to calculate the most profitable mix of products.

**Note:** In this example, a retail store is given as an example. However, constraint profit optimization can be applied to any stage of the supply chain, such as warehouses and distribution centers.

To achieve the best plan in a retail environment, it may be necessary to consider the following restraints, in addition to sales data and causal factors:

- Available space in store fixtures
- Available space in the store altogether
- Minimum and maximum facing required for a particular product in the store

The Constraint Profit Optimizer identifies sale opportunities, and fills the existing shelf space with products that have the highest probability of selling.

If the Optimizer discovers that a store has available or badly exploited shelf space, an alert is displayed. For example, if the minimum predetermined quantity for a product in a particular store is 10, but the Optimizer calculates its optimized inventory to be 3, the user is alerted. Similarly, an alert will be displayed if the maximum predetermined quantity for a product is 15 but the optimized inventory is calculated as 20.

The Optimizer considers the profitability of the products (not service levels). For each item, the client must give a figure on a scale of one to ten.

The system uses constraint dimensions (for example display/storage space) and a mix dimensions (for example item).

If a shelf has space which is less than can hold the maximum amounts for each product, so the Optimizer will determine the optimum stocking levels for maximum profit.

When the stock level reaches the maximum number for the most profitable product, the system then starts stocking the next profitable product.

## Creating or Modifying an Optimization Profile

You can create and save any number of optimization profiles.

### **To create or modify an optimization profile:**

1. Select Parameters > Constraint Optimization.

The Constraint Profit Optimization Wizard appears.

2. Do one of the following:

- To create new profile, click the New Optimization Profile button.
- To modify a profile, click the profile button.


The New Optimization Profile - Details screen appears

3. Type a name and optional description for the profile.
4. Click Next.

The Time and Analytical Selections screen appears.

**Time Selection**


Time Mode:
☒ Relative
☐ Fixed

Start Date:


Start After:

Lead:

**Analytical**

Bounded:


Include Stock:
☒ No
☐ Yes

Statistical Model:
☐ Poisson
☒ Normal

Update Review Flag: ☒
Use Prediction Status Constraint: ☒

Created By: dp

Created On: 5/7/2003 12:29:14

Last Modified On: 5/12/2003 14:35:54

5. Complete the fields as follows.

Time Mode	Choose one of the following: <ul style="list-style-type: none"> <li>Relative—specifies the time relative to the date of execution.</li> <li>Fixed—specifies a fixed span of time, starting with the Start Date field and spans the periods specified in the Lead field</li> </ul>
Start Date	Select a date for the optimization process to calculate from.
Start After	Specify the number of time buckets after which to start the optimization relative to execution date. For example, if the number is 2 and the day of execution is today, then the optimization output will be calculated starting from 2 days from today.
Lead	Specify the number of time buckets to use in the optimization process.

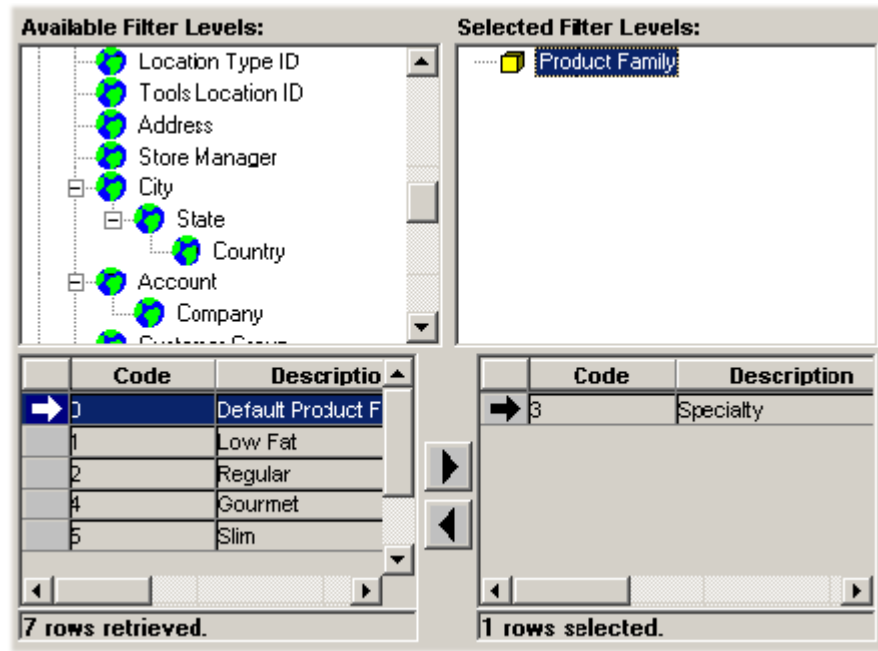
---

Bounding	<p>Choose one of the following:</p> <ul style="list-style-type: none"> <li>• Bounded—Calculates the optimal stock according to both the constraint on space and product boundaries (min and max).</li> <li>• Unbounded—Calculate optimal stock according to the constraint on space only.</li> <li>• Both—Calculates optimal stock according to both options above.</li> </ul>
Include Stock	Specifies whether to include existing stock levels in the calculation.
Statistical Model	<p>Choose one of the following:</p> <ul style="list-style-type: none"> <li>• Normal—Recommended when there is a large amount of data.</li> <li>• Poisson—Recommended when there is a limited amount of data.</li> </ul>
Update Review Flag	If this option is checked, the profile updates the review flags for the combinations. If you are not using these flags, deselect this option for better performance.
Use Prediction Status Constraint	If this option is checked, the profile considers only those combinations that have prediction_status equal to 1 (live combinations).

---

**6. Click Next.**

The Populations screen appears.



It is generally a good idea to specify the scope of the optimization by filtering the data. By doing so, you increase performance; otherwise the optimization process will run on the whole database.

7. Double-click a level in the Available Filter Levels box.

The selected level appears in the Selected Filter Levels box.

The bottom left side of the screen, below Available Filter Levels, now displays members that the optimization will run on.

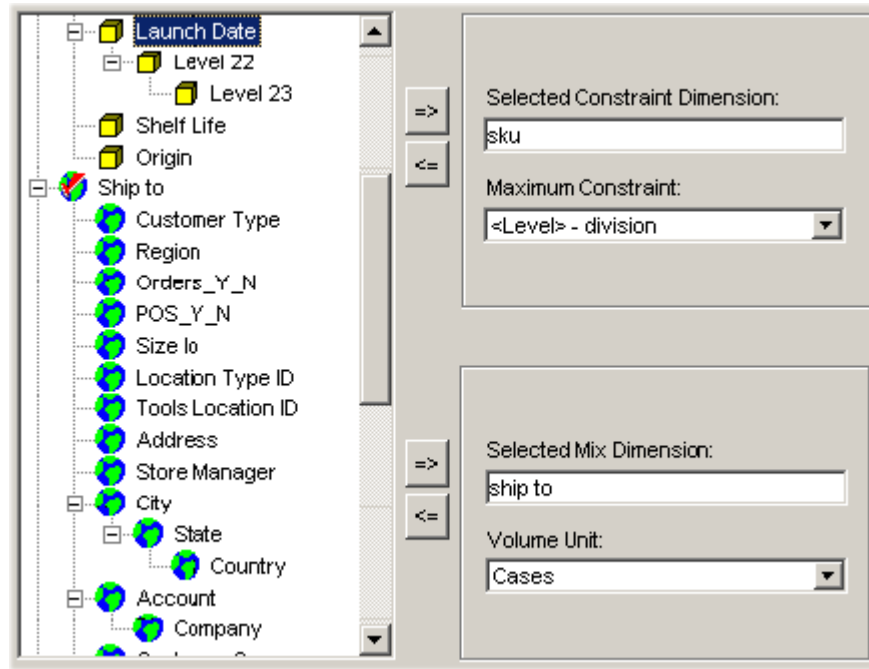
8. Click a member in the list, and then click the right arrow button. Or double-click the member you want to remove from the optimization process.

Business Modeler moves the selected values to the members list under the Selected Filter Levels box.

**Note:** The right list cannot include more than 200 members.

9. Continue filtering the data. When you are done filtering, click Next.

The Base Dimensions screen appears.



10. Complete the fields as follows.

Maximum Constraint	Select a level or mdp_matrix series.
Volume Unit	The unit used to measure the items.

11. Click Next.

The Input/Output Mapping screen appears.



Bounds Series	Minimum Constraint:	Min Amount
	Maximum Constraint:	Max Amount
Profit Scale:		Profit Scale
Tolerance:		Tolerance
Initial Stock:		Stock
Bounded:		Bounded
Unbound:		Unbounded
Review Flag:		Review Flag
Review Amount:		Review Amount
Free Space:		1406
Forecast:		fore_minus_0
Variance:		VARIANCE

12. Here you specify the series containing the input and output data.

Complete the fields (input series) as follows. Be sure to use only numeric series, not string or date series:

---

Bounds Minimum Constraint	The lower boundary constraint
Bounds Maximum Constraint	The upper boundary constraint.
Profit Scale	A customer defined scale of numbers specifying profitability.
Tolerance	Every time an item is added to the shelf, its profitability is reduced. This specifies the lowest boundary of profitability.
Initial Stock	Stock at start of optimization process.

---

13. Complete the field (output series) as follows:

---

Free Space	Available display/storage space.
------------	----------------------------------

---

14. Click Finish. Or to exit without saving, click Exit.

15. Click Execute to execute the optimization profile.

## Deleting a Constraint Optimization Profile

### To delete a constraint profit optimization profile:

1. In Business Modeler, navigate:  
Select Parameters > Constraint Optimization.  
The Constraint Profit Optimization Wizard appears.
2. Click an existing profile.
3. Click Delete.

## Running a Constraint Optimization Profile

### To run a constraint profit optimization profile:

1. In Business Modeler, navigate:  
Select Parameters > Constraint Optimization.  
The Constraint Profit Optimization Wizard appears.
2. Click an existing profile.
3. Click Execute.
  - In the Workflow Manager, execute the optimization programmatically. Specifically, create an executable step (within a workflow) that executes the optimization program.
  - Create a workflow to call the optimization executable, passing a parameter string that is the name of the constraint optimization profile.

**Note:** Depending on your options, the Constraint Profit Optimizer can affect only combinations that have a prediction status equal to 1 (live).

---

## Creating a Standard Error Process

This chapter describes the creation of a standard error process.

This chapter covers the following topics:

- Creating a Standard Error Process

### Creating a Standard Error Process

#### Overview

A standard error process is a database procedure (named `COMPUTE_STD_ERR`) that calculates the standard error between two specified series, at specified aggregation levels. You run this procedure from a Workflow using the Stored Procedure Step that runs `COMPUTE_STD_ERR`. You can use this standard error calculation, for example, to define and maintain the safety stock levels that are needed to meet demand according to a predefined service level and replenishment cycle.

#### Aggregation Levels Used in Standard Error Calculation

When you calculate standard error, you specify two aggregation levels. The standard error is calculated for each combination of those two levels. You can use the following combinations of levels:

- Item level and location level
- Item level and matrix level
- Location level and matrix level
- Matrix level and another matrix level

The target series (which you define to display the result of the calculation) must be defined on the corresponding dimension in the `mdp_matrix` table.

## To create a standard error process:

1. Start the Business Logic Engine user interface
2. Click Safety Stock > Safety Stock Process.

The wizard displays a screen where you specify the aggregation levels where the standard error is to be calculated.

**Item/Location Aggregation Levels:**  
Select one level from Items and one level from Location and Drag It to the box below.

**Items Levels:**

- Demand Class
- Item
- Manufacturer
- Unused
- Forecast User Status
- Item Branch Category 4
- Item Branch Category 1
- Forecast System Status

**Location Levels:**

- General Location Attribute 3
- Customer Category Code 3
- Trading Partner Zone
- Zone
- Bill To
- Account
- Customer
- Customer Class

**Selected Base Levels:**

- Item
- Bill To

<< Back   Cancel   Finish   Next >>





This screen displays two lists of aggregation levels. The left list contains all the item levels and all the matrix levels. The right list contains all the location levels and all the matrix levels.

3. Specify the levels as follows:
  - Drag a level from each list to the Selected Base Levels field at the bottom of the screen.
  - To remove a level from the Selected Base Levels field, right-click it and select Delete Level.
4. Click Next.

The wizard displays a screen where you specify the range of dates for which the standard error is to be calculated.

**Time Range:**

**Time Filtering**

From Date: 08/25/2008  -13  To Date: 11/24/2008  0 


5. Specify dates as follows:


- In the From Date spin box, select the number of base time buckets by which to go back. This is the date from which standard error is to be calculated.
- In the To Date spin box, select the number of base time buckets by which to go back. This is the date to which standard error is to be calculated.


6. Click Next.

The wizard displays a screen where you specify the series to use in the calculation.

**Forecast Series:**

Series Parameter 1: 1292 

Series Parameter 2: 26 

Covariance Series: 153 

Standard error formula:

$$\sqrt{\frac{((\text{Series Parameter 1}) - (\text{Series Parameter 2}))^2}{(\text{Series Parameter 2})^2 / n}}$$

\* n = number of history observations selected

7. Specify the forecast series as follows:

- In Series Parameter 1, select the series to use as Series Parameter 1 in the standard error formula.
- In Series Parameter 2, select the series to use as Series Parameter 2 in the standard error formula.
- In Series Covariance, select the series to use as the standard error calculated for the combination.

8. Click Next.

The wizard displays a screen where you can review your choices.

9. Do one of the following:
  - To change a setting, click Back repeatedly until you reach the screen that has the setting. Then change it as needed.
  - When you are finished, click Finish. The wizard then creates a database procedure named COMPUTE\_STD\_ERR. You must run this procedure from a Workflow (use the Stored Procedure Workflow step).
  - To cancel without saving changes, click Cancel.

## Generating Safety Stock

Once calculated, the Standard Error can be used to drive a simple safety stock calculation in Demantra. A series can combine the standard error with a desired service level and the replenishment time associated with the time and generate a suggested safety stock. An example for such a series in a weekly system would have the expression:

- $\text{consensus plan} * z\_val(\text{service level}) * \text{error standard} * \sqrt{(\text{repl cycle} / 7)}$

where consensus plan, service level, error standard, replenishment cycle are all series.

If the requirement calls for a more complicated safety stock calculation, Oracle Inventory Optimization can be used.

# Part 6

---

## Configuring and Running the Analytical Engine





---

## Introduction to the Analytical Engine

This chapter describes the role the analytical engine plays in the simulating forecasts. It also describes how the analytical engine functions generally and the different engine modes.

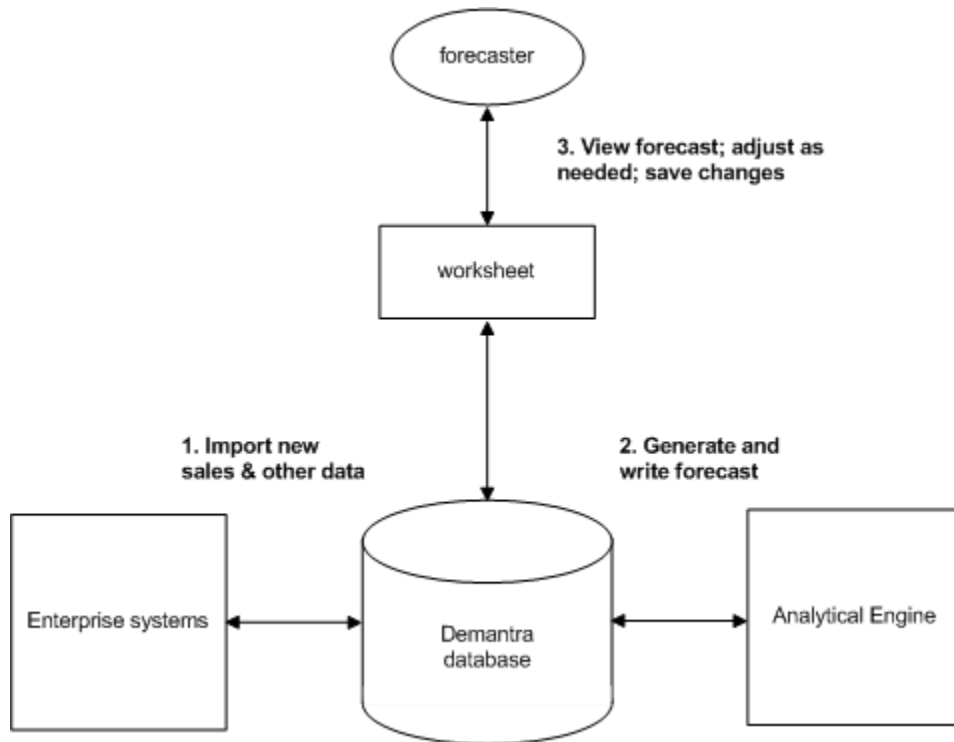
This chapter covers the following topics:

- Overview
- Engine Modes: DP and PE
- What the Engine Does
- Batch and Simulation Modes
- Engine Profiles
- Specifying the Demand Stream in an Engine Profile
- Maintaining Engine Versions
- Illegal Characters in Demantra

### Overview

The Oracle Analytical Engine is an advanced analytical engine capable of multidimensional forecasting with mixed modeling techniques. The system is designed for large-scale installations handling analysis and modeling of tens to hundreds of thousands of different demand patterns.

The following figure shows an overview of how a Demantra solution uses the Analytical Engine:



Within a Demantra solution, the Analytical Engine runs periodically (in the background), reading data from the Demantra database and generating forecast data. The forecaster uses a worksheet to view the forecast and make adjustments, saving those changes to the database. The updated forecast is available to all users with the appropriate authorization.

The preceding figure is not meant to show hardware configuration, which is discussed in the Oracle Demantra Installation Guide, rather than in this manual. You should be aware, however, that the Analytical Engine can be used in a distributed mode. Specifically, your system may include the Distributed Engine, where the Analytical Engine is registered on multiple machines, all with access to the Demantra database. In this mode, the Analytical Engine automatically distributes its work across multiple machines simultaneously. This maximizes processing power and reduces bottlenecks. For more information about the distributed engine, see the Oracle Demantra Installation Guide.

## Engine Modes: DP and PE

Oracle provides two different modes of the Analytical Engine:

- In PE mode, the engine is suitable for use with Promotion Effectiveness.
- In DP mode, the engine is suitable for use in demand planning applications.

## What the Engine Does

The Analytical Engine accesses the database and reads the historical demand and data from the causal factors (such as seasons, price changes, and specific events such as promotions [in the case of Promotion Effectiveness]). It then generates a forecast for all or specific item-location combinations. Wherever possible, it generates the forecast at the lowest possible allowed level (such as SKU-store). If necessary, it aggregates data so that it can generate a forecast at a higher level and split it to the lower level as needed. The forecast tree (which you configure) controls how the Analytical Engine aggregates and splits data when performing this task.

When working on a node of the forecast tree, the Analytical Engine uses a set of engine models, which are mathematical forecasting models. It considers how well each of those models works for that node and it statistically combines the best results, and generates the forecast from that. Advanced users may choose to adjust parameters that control how the individual models work; see "Theoretical Engine Models" for details on the models. Advanced users can also adjust how the Analytical Engine treats different nodes in the forecast tree.

In PE mode, the Analytical Engine also decomposes the forecast into the following:

- The baseline forecast (the forecast that would apply if no promotions were planned for the future)
- Direct effects (uplifts on item-location combinations due to promotions for those combinations).
- Switching effects (positive and negative effects on combinations due to promotions for other combinations)

See also

"Basic Concepts" "Engine Details"

## Batch and Simulation Modes

The Analytical Engine can run in two modes: batch and simulation.

- In batch mode, the Analytical Engine considers all the item-location combinations and generates a forecast for all of them (with a few exceptions, noted in the next chapter). In a typical implementation, the engine automatically runs in batch mode regularly. Batch mode should be run separately of data load, typically after new data is imported.
- In simulation mode, the Analytical Engine considers only a subset of the combinations. In this mode, the engine (called the Simulation Engine) waits for simulation requests and then processes them.

In simulation mode, a user runs a worksheet and submits a simulation request for some or all of the combinations in it. The simulation request is processed in the background but generally fairly soon. When the simulation is done, Demantra alerts the user, who can then accept or reject the results.

In this mode, the user is usually performing a "what if" analysis. That is, he or she has made some changes within the worksheet and then performs the simulation to see whether those changes have the desired effect.

It is also possible to run simulations programmatically from within a workflow.

The Analytical Engine can run in only one mode at a time.

See also

"Comparison of Batch and Simulation Modes"

"Running the Engine from the Engine Administrator"

"Running the Engine from the Start Menu"

## Engine Profiles

Each engine profile is a set of parameters with specific values, causal factors, forecasting models, and a forecast tree. Engine profiles are available to quickly change the functioning of the Analytical Engine dependent on the type of forecasts you want to develop. Oracle Demantra provides some predefined profiles, and you can define additional engine profiles, as needed. When you run the Analytical Engine, you specify the engine profile to use.

The predefined profiles are as follows:

### Base

This engine profile is the standard default Demantra engine.

### Batch

The batch engine profile uses the same forecast tree and causal factors as the Base engine, but the system parameters can be modified individually to reflect a different demand stream or other customization without modifying the Base.

### Booking Forecast

The Booking Forecast engine profile uses the same forecast tree and causal factors as the default Demantra engine, but the system parameters can be modified individually to reflect a different demand stream or other customization without modifying the Base.

## DSR POS Forecast

The DSR POS Forecast engine profile supports the Demand Signal Repository Point of Sale functionality. It uses the same forecast tree and causal factors as the default Demantra engine, but the system parameters can be modified individually to reflect a different demand stream or other customization without modifying the Base.

## Forecast Install Base

This engine supports the forecasting of install base under contract, a service parts forecasting function.

The forecast tree is defined as follows:

Level	Item Level	Location Level
1	Lowest Item	Lowest Location
2	Item	Organization
3	Item	Organization Type
4	Highest Item	Highest Location

The causal factors associated with this engine profile are:

- Constant
- Trend
- Consensus Forecast

All other causal factors are disabled.

## Forecast Spares Demand

This engine profile supports forecasting of spares at an organization. It executes on the data and combination tables used by the Spares general level.

The forecast tree is defined as follows:

Level	Item Level	Location Level
1	Lowest Spares Level	Lowest Location

Level	Item Level	Location Level
2	Latest Revision	Organization
3	Latest Revision	Organization Type
4	Highest Item	Highest Location

The causal factors are defined as follows:

- All existing defaults
- Install Base Under Contract

## Simulation

This engine profile is the standard default simulation engine.

### Simulation Install Base

The Simulation Install Base is a child of the Forecast Install Base engine profile. Should be used when generating a simulation on Install Base.

### Simulation Spares Demand

The Simulation Spares Demand profile is a child of the Forecast Spares Demand engine profile. Should be used when generating a simulation on Spares Demand.

### To create an engine profile:

When you create an engine profile, it is associated with a specific `init_params` table. It must not be the same table used by the other engine profiles. To check which `init_params` tables are in use, you can use the sql command `select * from engine_profiles;`

1. Navigate to System Parameters.

Business Modeler > Parameters > System Parameters.

The System Parameters window appears.

2. Select the Engine tab.

The existing engine profiles are displayed in the Engine Profile drop-down menu.

3. Click New.

The Create Engine Profile dialog box appears.

4. Select the engine profile you would like to use as a base for your new profile, if desired.
5. In the Profile Name field, enter the name of the engine profile.
6. In the Init Params Table Name, enter the init params table to be associated with this engine profile.
7. In the Profile Type field, choose whether your engine profile will be a batch or simulation engine.
8. If you have chosen a simulation engine profile type, select the parent batch profile.
9. Click OK.

## Related Topics

Configuring the Analytical Engine, page 51-xxxix

Engine Parameters, page 58-1

## Specifying the Demand Stream in an Engine Profile

Statistical forecasts are calculated by analyzing a historical data stream to find trends and seasonal patterns in the data, and then projecting those trends and patterns into the future. One of the keys to this process is the historical data stream. This data stream is typically referred to as the *demand stream*. Often there are several options for which data stream to use, with typical examples being shipments, orders, and consumption data.

The decision on which demand stream to use is typically based on:

- Availability of data
- Quality of data
- Business problem being addressed

Sometimes, it is desirable to create forecasts based on historical data taken from more than one demand stream. Some sample scenarios are the following:

- A health food company wants to base its production forecast on orders. Orders are readily available for all stock keeping units (SKUs) and customers. This data serves as the basis for generating a forecast. For key customers, point of sales consumption data is available. This information gives the best insight on true weekly consumption and consumer behavior. For these customers, it is desired to generate a forecast based on the point of sales data. This forecast can be compared with the

order-based forecast as well as allow vendor managed inventory (VMI) relationships. Ad hoc what if analysis (simulation) is required for both data streams.

- A home goods company wishes to generate two forecasts. One forecast is based on shipments, while the second is based on consumer orders. The two forecasts are then compared. Areas with large differences are analyzed for forecasting anomalies. Order data is available weekly, while, due to data collection limitations, shipment information is only available every two weeks. This requires that the forecast generated based on shipments be run every 2 weeks while the order forecast is run weekly. Customer IT best practice requires the system to be able to display 5 versions of the order forecast and 8 versions of the shipment forecast.

Controlling which data series is used as the demand stream by the Oracle Demantra Analytical Engine is done by configuring the `quantity_form` system parameter. The `quantity_form` parameter contains an expression that is used by the engine to retrieve and aggregate demand stream data from the `SALES_DATA` table. Changing the expression results in different demand streams, or different combinations of demand streams, being used as the basis for generating the forecast.

### **To define the `quantity_form` parameter for an engine profile:**

1. Navigate to System Parameters.  
Business Modeler > Parameters > System Parameters.  
The System Parameters window appears.
2. Select the Engine tab
3. Select the forecast from the Engine Profiles drop-down menu.
4. Navigate to the Data Manipulation sub tab.
5. Specify the expression in the Value column for the `quantity_form` profile row. The expression can be simple or complex. Make sure the `quantity_form` expression for a specific engine profile points to database columns containing the historical demand stream desired for the profile.

The default syntax is: `greatest(nvl(pseudo_sale,actual_quantity)*(1 + nvl(demand_fact,0)),0)`. This expression checks for the availability of user overrides stored in the `pseudo_sale` column. If they exist, these overrides supersede the historical sales found in the `actual_quantity` column. Any user defined % increase is then applied.

#### **Example**

- This is an example of a simple expression:  
`actual_quantity`



**Note:** When running the analytical engine the expression will be wrapped by a 'sum' function, therefore aggregation functions should not be explicitly included in the expression.

- This is an example of a more complex expression:

```
nvl(pseudo_sale,nvl(shipments,orders))
```

Here user overrides take precedent. If no user overrides are present, the expression uses shipments where available, otherwise it defaults to orders.

- To use a different demand stream such as booking, enter the following expression:

```
nvl(pseudo_sale,actual_quantity)*(1 + nvl(demand_fact,0))
```

- For a Spares Forecasting demand stream (not based on the SALES\_DATA table), the following expression represents a possible configuration:

```
nvl(nvl(spf_shipment_over, spf_shipment_in),0)
```

6. Click Save.

**Note:** When specifying quantity\_form definitions, it is strongly recommended that the expression be constructed such that it prevents negative values.

## Maintaining Engine Versions

The engine maintains the quantity of most recent engine versions defined in system parameter profile\_forecasts\_versions. For each of these versions, a full complement of columns is kept in SALES\_DATA and PROMOTION\_DATA. If a specific Engine Profile does not have a profile\_forecasts\_versions parameter, then the parameter value found in the system parameter active\_forecast\_versions is inserted into the profile during the engine run.

### To define the number of engine versions maintained:

1. Navigate to system parameters.

Business Modeler > Parameters > System Parameters.

The System Parameters window appears.

2. Select the Engine tab

3. Select the forecast from the Engine Profiles drop-down menu.
4. Navigate to the Time sub tab.
5. The parameter named `profile_forecasts_versions`, located on the Time sub tab controls the number of forecast versions for the given Engine Profile. The default value is 5.

## Illegal Characters in Demantra

Within Demantra, do not use the following special characters:

Single quote (')

Double quote (")

Ampersand (&)

If you use these characters, unexpected results may occur.

---

## Basic Concepts

This chapter introduces the basic concepts involved with configuring the analytical engine.

This chapter covers the following topics:

- Overview of Forecasting
- Causal Factors
- Promotions (PE Mode Only)
- Forecasting Models and the Engine Flow
- The Forecast Tree
- Influence and Switching Effects (PE Mode Only)
- Combination-Specific Settings
- The Forecast Data

### Overview of Forecasting

The Analytical Engine generates a forecast that considers the historical demand and the causal factors.

In this process, the Analytical Engine calculates a set of *coefficients* that describe how each causal factor affects demand for each item-location combination, over time. The Analytical Engine then uses those coefficients, along with future values for the causal factors, to determine the forecast.

You do not see or work with the coefficients directly, but you may find it helpful to see the general equation to which they apply:

$$D = \text{constant} + A1*CF1 + A2*CF2 + A3*CF3 + \dots$$

Where:

- D is the demand for a specific combination.

- constant is the constant demand for that combination, independent of time.
- CF1, CF2, CF3, and so on are the causal factors in the system. Some of them are local and apply just to this combination; others are global. All of them vary with time.
- A1, A2, A3, and so on are the coefficients that the Analytical Engine calculates for this combination. These are the same for all dates.

Demantra uses an equation like this for each combination. The Analytical Engine solves all the equations simultaneously and calculates the coefficients, which it then uses to generate the forecast.

After the forecast is generated the following information may be available:

- Base forecast
- Lift Forecast
- Item node, Location node, and the Level ID for the forecast
- Models used successfully for the forecast
- Models, which the engine attempted to use for the forecast and failed
- How the forecast was generated
- Metrics demonstrating quality of the forecast

## Causal Factors

*Causal factors* provide information about historical events that are expected to recur in the future. Causal factors cause demand to deviate from a trend. More specifically, a causal factor is a time-varying quantity (such as price, season, or day of the week) that affects demand. Demantra requires historical data for causal factors, as well as future data that describes expected occurrences that will affect demand.

**Note:** The Analytical Engine uses multiple theoretical models, and not all of them consider causal factors; see "Forecasting Models and the Engine Flow".

## Types of Causal Factors

Demantra uses the following general types of causal factors:

- *Global causal factors (global factors)* apply to all item-location combinations. For

example, a season is a global causal factor. Most Demantra implementation use global factors. Oracle provides a set of base causal factors; see "Base Causal Factors".

- *Local causal factors* apply to specific item-location combinations. For example, a discount applied to a specific item in a specific sales region is a local causal factor. Price is another local causal factor.

Local causal factors include activities, which are a special kind of local causal factor that supports *activity shape modeling*; see "Activities and Activity Shape Modeling".

- **(For PE mode only)** *Promotional causal factors* apply to specific item-location combinations *and* to specific promotions. Promotional causal factors are available only within Promotion Effectiveness. Promotional causal factors are based on the attributes of the promotions in the system. You can use promotional causal factors to perform *promotional shape modeling*. See "Configuring Promotions and Promotional Causal Factors".

## Base Causal Factors

Demantra provides the following base causal factors. Except for Price, these are all global causals; Price is local:

- Constant
- t (time)
- Causal factors that correspond to the months of the year. The names of these causal factors depend on the time resolution:
  - d1, d2, ..., d12 (if the time resolution is monthly or weekly)
  - m1, m2, ... m12 (if the time resolution is daily)
- Causal factors that correspond to the days of the week (included only if the time resolution is daily): d1, d2, ..., d7
- Price

For these causal factors (except Price), Demantra provides data (for many years in the future) and the correct configuration. You should not edit or delete these causal factors. In the case of Price, you need to make sure that the sales\_data table contains the price information that this causal factor uses.

## Data and Configuration Details

Demantra requires the following information for each causal factor:

- Data for the causal factor for each time bucket, past and future.

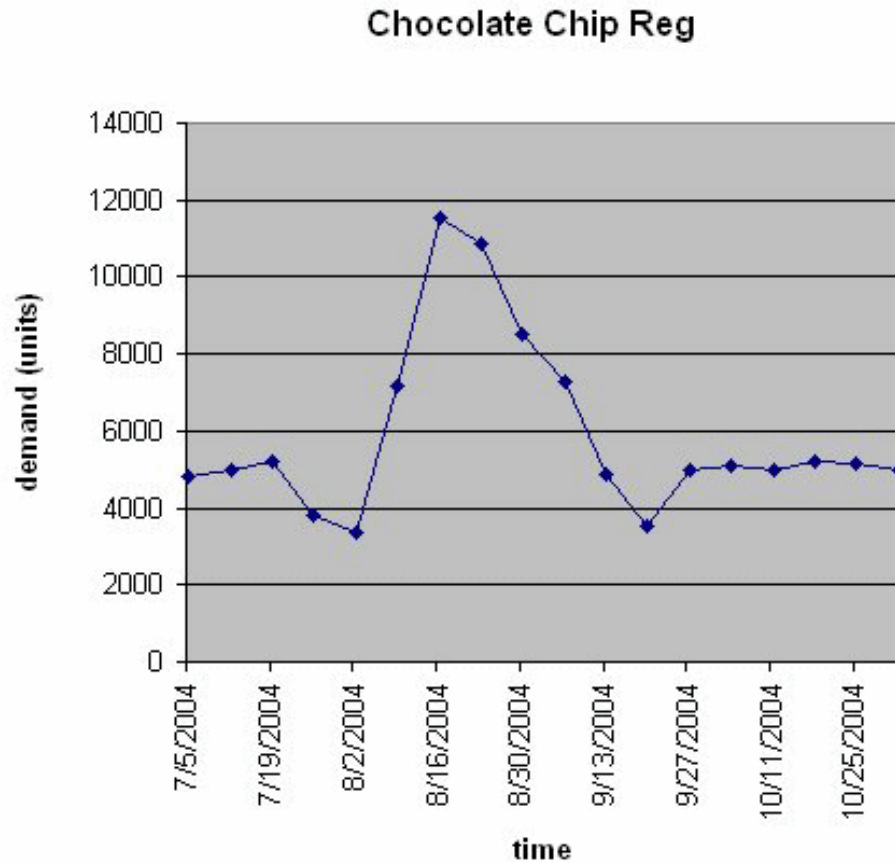
- Configuration details on how the Analytical Engine should use this causal factor. Here you make the causal factor known to the Analytical Engine, and you specify how the engine should use it.

For reference, the following table summarizes where this information is stored:

Causal factor type	Location of data	Configuration details
Global factors	Column in Inputs table	Causal Factors screen of the Forecast Tree Editor
Local causal factors other than activities	Column in sales_data table or SQL expression that aggregates data from that table	
Activities	Column in sales_data table	
Promotional causal factors (For PE mode only)	Aggregation function retrieves data from the promotion_data and promotion tables	Promotional Causal Factors screen of the Forecast Tree Editor

## Activities and Activity Shape Modeling

The Demantra *activity shape modeling* feature helps you easily reapply a demand profile that has a distinct shape over time. For any causal factor, Demantra requires past and future data. In the case of causal factors such as price and seasons, it is a simple process to obtain and load the data. Other causal factors are more difficult to describe mathematically. For example, when you run a promotional activity on a product, you may see a demand curve like the following:



If you plan a future activity that is similar to this historic activity, you would expect it to create similar demand. In general, shape modeling lets you do the following:

- Identify a historic demand curve as a reusable curve
- Create another instance of that curve starting at some future date, creating a new activity

Demantra internally represents the shape as a linear combination of as many as eight Oracle proprietary shapes. Then the Analytical Engine automatically uses this demand shape along with all the other data in the system to determine the forecast.

By default, the Analytical Engine averages the most recent data for a given shape with the stored information about that shape, which is an average of all the past observations of this shape. Users can control this, by forcing the Analytical Engine to rescale the generated shape to align with the recent data. Specifically, the user can indicate the number of buckets for which the shape alignment should occur, starting with the beginning of the shape. Typically the user specifies either 0 (the default) or the length of the shape (to realign the entire shape).

**Note:** Shape modeling capabilities are different in the two engine modes:

- In DP mode, the engine supports only activity shape modeling.
- In PE mode, the engine supports both activity shape modeling *and* promotional shape modeling. See "About Promotion Shape Modeling".

See "Engine Modes: DP and PE".

See also

"Configuring Causal Factors"

"Configuring Promotions and Promotional Causal Factors" (PE only)

## Promotions (PE Mode Only)

A promotion is an occurrence that starts at a specific date, has a certain duration, and has a certain time-varying affect on sales. Specifically, within Promotion Effectiveness, a promotion is associated with one or more item-location combinations (at any aggregation level) for a given time bucket or buckets. A given combination can have multiple promotions at any given time bucket.

As with sales data, promotion data can be imported. Depending on how your system is configured, Promotion Effectiveness may continue to import new promotions or users might create promotions within the Promotion Effectiveness user interface. Promotion Effectiveness displays promotions in the Activity Browser in the worksheets; here users create, edit, and remove promotions.

## Promotion Attributes

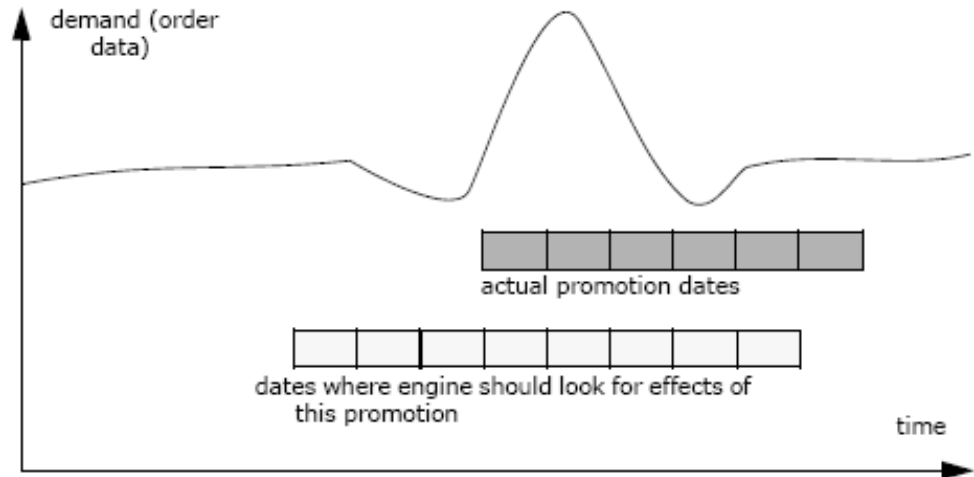
The Analytical Engine does not use the promotions directly. Rather it uses the promotion attributes, such as discount amount, display type, and so on, each of which can have a different effect on demand. The Analytical Engine converts the values of the promotion attributes to promotional causal factors.

During implementation, you specify the attributes that promotions can have, and you specify rules for converting attribute values into causal factors. When users create promotions within Promotion Effectiveness, they specify values for these attributes.

## Promotion Dates

Promotion Effectiveness assumes that a promotion has an effect between its start and end dates, as provided to Demantra, but typically the promotion has an actual effect in a slightly different span of time, as in the following example:





There is often a lag between the demand and the promotion associated with that demand. Typically this lag is larger with order data than with point of sale (POS) data, because retailers place orders further in advance. But there is often a lag even with POS data because customers know about an upcoming promotion and often delay normal purchases until the promotion occurs.

Accordingly, Promotion Effectiveness supports a couple of adjustments:

- First, you can specify an overall shift, which forces the Analytical Engine to shift the promotion dates globally by a specific number of time buckets. In the example above, the shift is -1 bucket.

This shift time applies to all the promotions (but not to other causal factors).

- Second, you can lengthen or "stretch" the span of a promotion by specifying an additional number of time buckets on either end of the promotion. In the preceding example, we added two time buckets to the start of the promotion.

**Note:** Users may want to add lift or other overrides to the promotion. It is important to remember before the Analytical Engine has been run, the database contains records only for the actual promotion dates; these records are created when the promotion is created. So overrides can be entered only on those dates.

After the engine has been run, however, the database has records for the additional dates as well and overrides can then be entered.

## Promotion Hierarchy

For the benefit of users who are creating or managing promotions, you can provide a

hierarchy that helps the users group the promotions. Then, within a worksheet, the Activity Browser can display that hierarchy, as in the following example:



The Analytical Engine ignores the hierarchy itself. For the engine, the important consideration is the promotion attributes, as noted earlier.

## Promotions and Promotion Shape Modeling

In addition to performing shape modeling for activities, the Promotion Effectiveness supports *shape modeling* for promotions. Specifically, you enable shape modeling for individual promotional causal factors, as needed.

As with ordinary activity shape modeling, Demantra internally represents the shape as a linear combination of the shapes. Then the Analytical Engine automatically uses this demand shape along with all the other data in the system to determine the forecast.

## Forecasting Models and the Engine Flow

The Analytical Engine uses a set of theoretical models, each of which evaluates some or all of the data. Most, but not all, of these models use causal factors. The models are documented in "Theoretical Engine Models".

The Analytical Engine follows a specific process of examining the data, checking for outliers and so on, evaluating the usefulness of each theoretical model, and generating the forecast. This process is described in detail in The Forecasting Process, page 57-6.

Demantra supports multiple analytical engine profiles. These engine profiles can be configured to generate forecasts for different scenarios such as service parts planning. These engine profiles are described in detail in Engine Profiles, page 49-4.

Demantra provides parameters to control both the theoretical models and the overall engine flow. See Tuning the Analytical Engine, page 55-xxxii; only advanced users should adjust these parameters.

## The Forecast Tree

In general, forecasting is most accurate when it can be performed at the lowest possible allowed aggregation level. However, sometimes there is not enough data at that level for all combinations. For those combinations, the Analytical Engine aggregates the data to a higher level and tries to generate a forecast there. The purpose of the forecast tree is to organize data for this process.

**Note:** The Analytical Engine also considers flags on different combinations; see "Combination-Specific Settings".

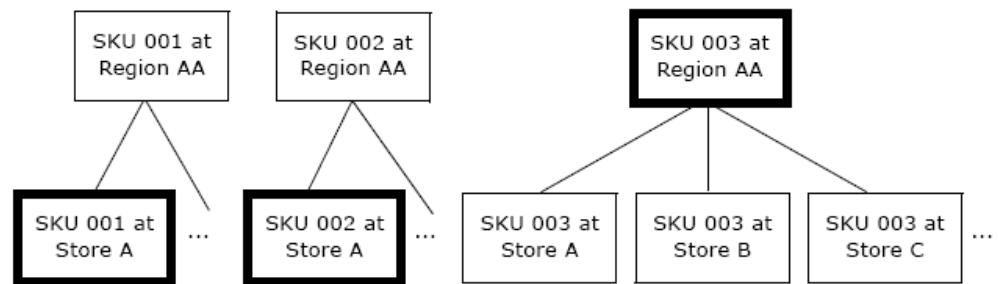
A forecast tree is associated with each of the configured batch engine profiles, thereby providing unique forecasting results for different forecast scenarios such as service parts planning.

As noted in "Levels", you define aggregation levels for use in worksheets. You use some of these levels to build the forecast tree. For PE mode, you also use the forecast tree to define the influence relationships.

## Basics

Whenever the Analytical Engine generates a forecast at an aggregate level, it automatically splits the forecast for the parent node across the child nodes, again using the structure of the forecast tree. The proport mechanism controls how the aggregated forecast is split. For information on tuning proprot, see "Proport Mechanism".

Each node in the forecast tree aggregates both by items and by locations. The following example shows a small part of a forecast tree.



The bold boxes show the nodes at which the Analytical Engine is forecasting.

- In this example, there is enough data at the SKU-store level for SKU 001 and SKU 002; the Analytical Engine can generate a forecast at that level for those SKUs.
- On the other hand, there is less data for SKU 003, so the Analytical Engine aggregates data for that SKU across all the stores in Region AA, generates the forecast for those SKUs at the SKU-region level, and then splits to the store level.

## Accuracy Metrics for Forecasts

While generating the forecasts, the analytical engine also generates the accuracy metrics for the forecast to provide information regarding the quality of the forecast. The analytical engine generates the quality measures for forecasted combinations based on analysis of past forecasts, as the quality of a forecast is largely indeterminable without performing sample tests on the forecast data. Accuracy metrics generated by the

analytical engine are known as in-sample metrics.

In-sample accuracy metrics use the following formulas to judge the quality of the generated forecast for all spares considered in that run:

- **MAPE Mean (Absolute Percentage Error):** Represented by  $\text{mean}(\text{abs}(\text{Series} - \text{Fit})) / \text{mean}(\text{abs}(\text{Series}))$
- **RMSE Root (Mean Squared Error):** Represented by  $\sqrt{\text{sum}(\text{Resid.}^2) / (\text{LengthSeries} - \text{Complexity})}$
- **PBias (Percentage Bias):** Represented by  $\text{sum}(\text{Resid}) / \text{sum}(\text{Series})$
- **Relative\_Error (Relative Error):** Represented by  $\text{median}(\text{abs}(\text{Series.} / \text{Fit} - 1))$

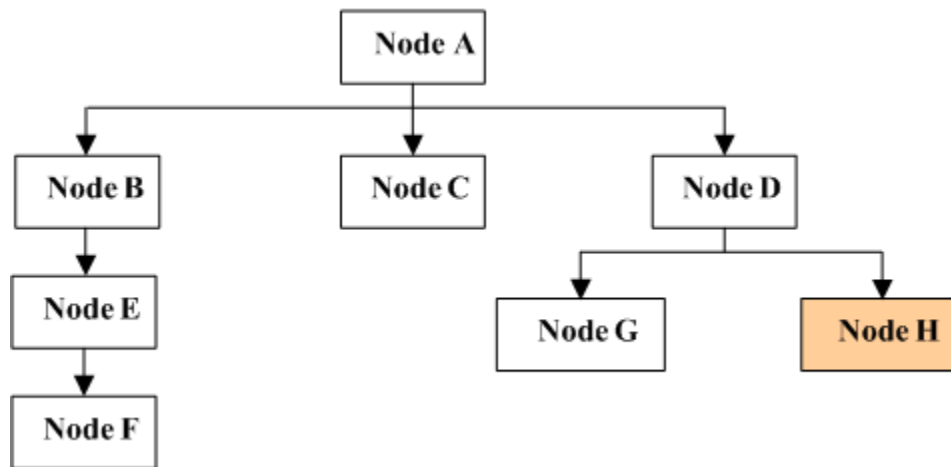
The observations made by the engine using the above-mentioned formulas are stored in the MDP\_MATRIX table.

Additional out-of-sample error calculation can be done using a stand alone error calculation process. For more details refer to Out of Sample Error Calculations, page 50-11.

For a forecast tree, the accuracy metrics allocate down to the lowest level of the tree.

#### Example

The following figure depicts a forecast tree, where each node represents aggregated data, both by items and by locations.



Node F has a forecast at the lowest level. Therefore, all accuracy metrics generated at node F would be assigned to the member data for node F.

Node G has a forecast at the lowest level. Therefore, all accuracy metrics generated at node G would be assigned to the member data for node G.

Node H failed at the lowest level and the forecast eventually is generated at node D. The accuracy metrics from node D should be allocated to all nodes that get a forecast from node D. Node G will get the accuracy metrics from node D, whereas node H will

not receive the same from node D.

## Forecast Tree Example

The following list describes a possible forecast tree.

1. Highest level: all items and all locations, aggregated together
2. All items-Division
3. Brand-Division
4. Brand-Region
5. SKU-Region
6. Lowest level: SKU-Store

## Finding the Effects of Promotions (PE Mode Only)

For PE mode, the forecast tree must also be organized to support how the Analytical Engine identifies the effects of promotions. When you set up the forecast tree, you identify levels in the tree that the Analytical Engine can use in specific ways, as follows:

- The *influence range* level (IRL); see "Influence Ranges".
- The *influence group* level (IGL); see "Influence Groups".
- The lowest promotion level (LPL), which is lower than (or the same as) the IGL. This specifies the level at which promotions can be meaningfully aggregated together. For any node in the LPL, all promotions are assumed to have the same attribute values.

## Out of Sample Error Calculations

Service Parts Forecasting makes use of both in-sample MAPE (see Accuracy Metrics for Forecasts, page 50-9) and out-of-sample MAPE, a calculated metric that occurs later, not during the batch engine run. SPF MAPE (out-of-sample) compares the actual sales data as it becomes available in the system to the forecast for a specified time period, and displays the difference as a percentage. This value is based on the forecast that is sent to Service Parts Planning (determined by SPF Final Forecast) and then archived in Demantra. It is a combination of the analytical forecast, the calculated forecast, and any user overrides.

Formula:  $\text{sum} ( \text{absolute value} | \text{Actual Demand} - \text{Lagged Forecast} | / (\text{Actual Demand}) / \text{Number of Observations}$  For example, a forecast is generated monthly and the total average quantity for each of the first six months of 2010 is 2000 units, resulting in forecast of January of 2000 units Actual sales for January were lower; 1634. Therefore,

this series would display a value of 22.4% (the result of  $\text{abs}(2000-1634)/1634$ ).

Additional example: A forecast is generated weekly. The expected demand for the first week in March is 300, but the actual demand was 346. Therefore, this series would display a value of 13.3% (the result of  $\text{abs}(300-346)/346$ ). If either demand or the forecast is zero (0), then the value would be 0%. If both forecast and demand are zero, then the value would be 100%.

The SPF Calc Forecast Accuracy seeded workflow is used to calculate error and variability associated with Service Parts Forecasting. This workflow can be called ad-hoc when accuracy measures should be generated. The seeded workflow is configured to aggregate information at levels Organization and Latest revision for the last four periods of history. The first three series pairs generate an accuracy measure for the final, analytical and calculated forecast streams by comparing the latest archived forecast with actual usage values. The last series pairs compare the last two archived versions of the final forecast with each other to determine forecast variability. If additional error calculation processes are required, it is recommended that additional steps be added to this workflow or separate workflows be created to call the APPROC\_CALC\_ACCURACY stored procedure.

## **APPROC\_CALC\_ACCURACY Stored Procedure**

Error calculation is called from the procedure APPROC\_CALC\_ACCURACY. The procedure has two parameters as inputs defining the aggregation level at which the calculation is done. Each level defines an aggregation dimension. When calculating error for series on sales\_data, one level from the item and one level from the location dimension should be specified. If the series being compared reside on a General Level data table, one of the levels specified can reside on the General Level. Levels should be specified by the internal ID of the level being referenced. The calculation start and end period are integers specifying which range periods should be aggregated together when calculating the error. These parameters are relative to the current end of history date specified by the max\_sales\_date parameter. A value of 0 would match this date, a value of -1 would be one period prior to this date and a value of 5 would be five periods after this date. For example, if the last four periods of history should participate in the calculations, the values -3 and 0 should be assigned to the start and end parameters respectively. The series groups define the groups of three series associated with the calculation. As defined in the previous formula, the difference between series 1 and series 2 are aggregated to the defined levels and time range and then divided by the total of series one for the same aggregation. The resulting value is written to the third series which is the output series. All three series in a group must reside on the same data model dimension, the first two residing on a data table and the third residing on the combination table associated with the two. For example if series 1 and 2 are associated with the Service Part data table, the third series must be associated with the Service Part Matrix table. Up to five series groups may be defined. Each series group can refer to a different data table and dimension.

Inputs are:

- Calculation Aggregation Level 1
- Calculation Aggregation Level 2
- Calculation Start Period
- Calculation End Period
- Series group 1: Inputs 1, Input 2 and Output
- Series group 2: Inputs 1, Input 2 and Output
- Series group 3: Inputs 1, Input 2 and Output
- Series group 4: Inputs 1, Input 2 and Output
- Series group 5: Inputs 1, Input 2 and Output

## Influence and Switching Effects (PE Mode Only)

To describe how the item-location combinations affect each other, you specify the influence ranges, influence groups, competitive item groups, and competitive location groups.

### Influence Ranges

When you define the forecast tree, you specify the *influence ranges* (IR). To do so, you specify the *influence range level* (IRL); each node within the IRL is an influence range.

Each influence range is a set of combinations that do not interact with combinations within any other IR. The *influence ranges* control how far the Analytical Engine looks for influence when promotions are run. This determines the breadth of the causality. An influence range is a set of item-location combinations that potentially interact with each other but not with combinations of other IRs. Typically each IR represents a different geographical area.

No information is available above the IRL to calculate effects of promotions. Therefore, if for certain nodes the Analytical Engine generates a forecast above the IRL, the forecast for those nodes includes only the baseline forecast.

### Influence Groups

When you define the forecast tree, you also specify the *influence groups* (IG), which are subdivisions of the influence ranges. To do so, you specify the *influence group level* (IGL); each node within the IGL is an influence group.

Each influence group consists of an *item group* and a *location group* with the following behavior:

- An item group (I) is a set of items that relate identically to all other items. In particular, the items within an item group compete in the same way with items of other item groups. These items are interchangeable, as far as promotions are concerned. For example, suppose that an item group is diet colas. A promotion on any diet cola has the same effect on sales of non-diet colas, for example.
- Similarly, a location group (G) is a set of locations that relate identically to all other locations.

Using these definitions, the engine can calculate the following three causal factors for each lowest-level combination:

---

self	Influence caused by promotions on this combination.
own	Influence caused by other combinations within the same IG.
other	Influence caused by all other combinations within the IR.

---

The Analytical Engine uses these causal factors internally to calculate the switching effects caused by promotions.

No information is available above the IGL to calculate switching effects. Therefore, if for certain nodes the Analytical Engine generates a forecast above the IGL, the forecast for those nodes includes only the baseline forecast and the direct effects.

## Competitive Item Groups (CI) and Competitive Location Groups (CL)

Typically, you also define *competitive item groups* (CI) and *competitive location groups* (CL):

- A *competitive item group* (CI) is a set of item groups that compete with each other. For example, diet beverages could be a CI that contains the following three item groups: diet colas, diet fruit juices, other diet beverages. Non-diet beverages could be another CI.
- A *competitive location group* (CL) is a set of location groups that compete with each other.

You do not define these groups directly in the forecast tree. Instead, you set them via parameters. The Analytical Engine does not aggregate data to the CI and CL, so it is not strictly necessary to make them consistent with the rest of the forecast tree; they must of course, be at a higher aggregation level than the item and location groups, respectively.

## Switching Effects

A switching effect occurs when a sale for a given item-location combination affects sales



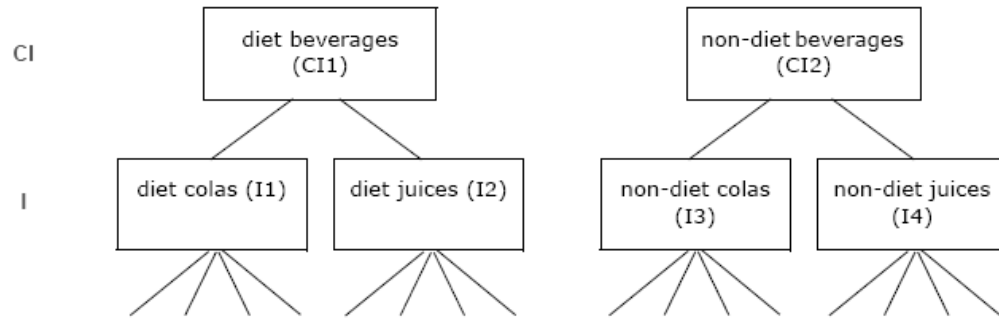
for another item-location combination. Promotion Effectiveness uses the preceding classification system to describe different switching effects. Each effect is associated with relationships between one item-location combination and others.

Effect*	CI	item group (I)	CL	location group (L)
Brand switching (or category switching)	different	<i>different by definition</i>	same	same or different
Channel switching	different	<i>different by definition</i>	different	<i>different by definition</i>
Product switching	same	same	same	same
	same	<i>different</i>	same	same
	same	<i>different</i>	different	<i>different by definition</i>
Store switching	same	same or different	same	different
	same	same	different	<i>different by definition</i>

\*Depending on how you define CI, CL, I, and L, the names of these effects may or may not be appropriate. You can rename these effects by renaming the series that display them.

Notice that if the CI and CL each have only one member, there is no competition, and the only effects that can be seen are product switching and store switching.

For simple example, consider a single store and the following item groups and competitive item groups:



- If a promotion is run for a diet cola (item in I1), that can have the following effects:
- Effect on sales of other diet colas at this store. Because both items are within the same item group, this is a case of product switching.
- Effect on sales of diet juices (I2) at this store. This is another case of product switching. The items are in different item groups but are in the same CI (CI1).
- Effect on sales of non-diet colas (I3) at this store. Because non-diet colas are in a different CI than the diet colas, this is a case of category switching.

## Combination-Specific Settings

The Analytical Engine also considers specific settings that vary from combination to combination, which are stored in the `mdp_matrix` table and which are affected by global parameters. This section provides an overview of the key settings, which are provided to support users who want closer control over the forecast. You can create levels or series that use these settings, as needed. Not all settings are meant to be changed directly.

### Fictive Status

Demantra sets a flag for each combination to indicate whether that combination is real or not. In `mdp_matrix`, the `is_fictive` flag has one of the following values:

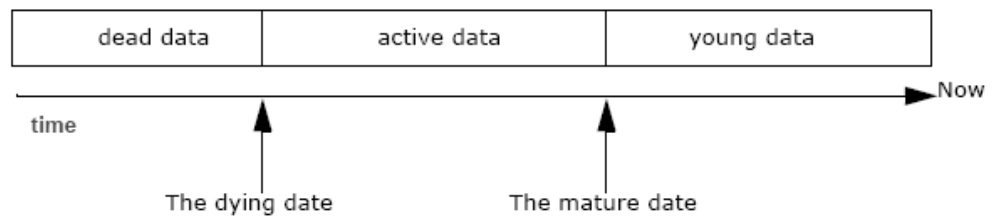
Value	General meaning
1	Combination is fictive (not real). This combination was created via Member Management.
0	Combination is real and it has non zero sales data.

Value	General meaning
2	Combination is real but all sales are zero or null.
3	Errors occurred while loading this combination.

The Analytical Engine does not use this flag directly, and users should not edit it.

## Age

Each combination is either young, live, or dead, depending on the relative age of its sales data. Demantra uses two cutoff dates to determine the age of a combination:



The dying date is controlled by the `dying_time` parameter, and the mature date is controlled by the `mature_age` parameter. Both parameters are global.

Demantra automatically uses these cutoff dates as follows:

- If there are no sales for the combination after the dying date, the combination is considered dead.
- If there are no sales for the combination before the mature date, the combination is considered young.
- Otherwise, the combination is live or active.

See "Engine Parameters".

The parameters used to determine if a combination is active, dead or young are:

- `hist_glob_prop`
- `dying_time`
- `mature_age`

At times there is a business need to override these global definitions. Typically this would be for specific items and locations for which a more or less reactive status change

is desired. For example: The majority of combinations should be deactivated if they have not sold in three months. However, seasonal items require a different setting, and should not be deactivated unless they have not sold for more than a year. These parameters are set globally, but can be overridden individually per combination.

## The User-Controlled Do\_Fore Flag

Demantra provides a combination-specific flag with which advanced end users can control how the Analytical Engine works on individual combinations. This flag is in the `mdp_matrix` table and is called `do_fore`. In order to enable users to set this flag, you generally create an editable series that uses this flag. Users can set this flag to any of the following values:

Value	Meaning
0	The Analytical Engine will ignore this combination
1	The Analytical Engine will consider the combination. This is the default value.
2	The Analytical Engine will create a placeholder forecast for this combination that consists of zero values (which is useful if the user wants to create an override). The engine will otherwise ignore this combination. You typically use this setting for combinations created through Member Management.

The sole purpose of the `do_fore` flag is to give users a way to control the prediction status of the combination, as described next. The `do_fore` flag is not used directly by the Analytical Engine.

## Prediction Status

In `mdp_matrix`, the `prediction_status` indicator of a combination instructs the Analytical Engine how to handle this combination. The following values are possible:

Value	Affect on the Engine	Comments
No Forecast (96)	The Analytical Engine ignores this combination.	For future use; this setting cannot currently be achieved or used.
Create Zero Forecast (97)	The Analytical Engine creates a zero forecast but otherwise ignores the combination.	

Value	Affect on the Engine	Comments
Young (98)	The Analytical Engine creates a zero forecast but otherwise ignores the combination.	Sales for this combination are too new to be used for prediction.
Dead (99)	The Analytical Engine creates a zero forecast but otherwise ignores the combination.	Sales for this combination are not recent enough to be used for prediction.
Live or Active (1)	The Analytical Engine uses this combination for forecasting.	

Demantra automatically sets the prediction\_status indicator and users should not change it.

## How Prediction Status Is Set for Fictive Combinations

For *fictive combinations* (is\_fictive = 1), Demantra automatically sets the prediction status to 98.

## How Prediction Status Is Set for Real Combinations

The proport mechanism considers all real combinations (is\_fictive equal to 0 or 2). It automatically sets the prediction status indicator for each combination based on the age of the combination and the do\_fore setting of the combination. Specifically, it sets the prediction\_status indicator as follows:

	do_fore is 0	do_fore is 1	do_fore is 2
Combination is dead	prediction_status is 99	prediction_status is 99	prediction_status is 97
Combination is young		prediction_status is 98	
Combination is live		prediction_status is 1	

## Aggregation Flags

As noted in "The Forecast Tree", the Analytical Engine sometimes needs to aggregate lowest level data in order to create a more accurate forecast. You can control, per combination, whether this combination should be aggregated to a higher level.

- If data for a specific combination is not aggregated, then that combination is not forecasted.
- Whenever the Analytical Engine aggregates data for any combination during forecasting, the engine also splits some of the forecast to that combination, according to the stored proportions of that combination.

Demantra provides three flags so that you can specify different rules based on the age of the combination. The flags are as follows:

Flag	When used
do_aggri	If combination is live
aggri_98	If combination is young
aggri_99	If combination is dead

Each of these flags (in mdp\_matrix) specifies whether to aggregate data for this combination during forecasting.

## Other Combination-Specific Settings

Demantra provides many parameters that control how the Analytical Engine behaves. By default, these parameters affect all the combinations in the forecast tree. Through the user interfaces, an advanced user can set analytical parameters for individual combinations if needed.

## The Forecast Data

The Analytical Engine writes the current forecast to one of the following fields in sales\_data: Fore\_0, Fore\_1, Fore\_2, and so on.

**Note:** For PE mode, this is the *baseline forecast*.

The Analytical Engine cycles through these columns. Each time, it writes the current forecast into one column (overwriting the oldest forecast). The Analytical Engine then adds a row to the forecast\_history table that describes this forecast and that indicates which column it is stored in.

The number of saved forecasts is specified by the active\_forecasts\_versions parameter.

## Forecast Decomposition (PE Mode Only)

For PE mode, the Analytical Engine also populates the following database fields in promotion\_data to show the effects of the promotions. These fields show the effects of a given promotion on a given combination, at a given date:

Field	Purpose
fore_0_uplift	Direct lift on a combination during the promotion dates, due to a promotion specifically associated with that combination.
fore_0_pre_effect	Direct lift on this combination before the promotion dates, due to a promotion specifically associated with that combination.
fore_0_post_effect	Direct lift on this combination after the promotion dates, due to a promotion specifically associated with that combination.
fore_0_brand	Effects of brand or category switching as described in "Switching Effects".
fore_0_sw_channel	Effects of channel switching on this combination, at this date, due to this promotion.
fore_0_product	Effects of product switching on this combination, at this date, due to this promotion.
fore_0_store	Effects of store switching on this combination, at this date, due to this promotion.

For the benefit of the users, you create series that use these data fields. Be sure to provide series names that make sense to the users and that are appropriate for the business.

## Forecast Versions (for Batch Runs)

As noted earlier, Demantra keeps a number of previous forecasts (as specified by the active\_forecasts\_versions system parameter). The most recent batch forecast is numbered 0, the previous one is numbered 1, and so on. When the Analytical Engine

generates a new forecast, it moves the previous ones to different columns in the database. See "Engine Parameters".

Each series you create is implicitly or explicitly associated with a specific forecast version or multiple forecast versions. Typically, the large majority of series are associated with the most recent forecast, but it is often useful to configure some series to capture information associated with a previous forecast, or to compare multiple forecasts.

**Note:** If you need to display present and past versions of other data, you can configure and run *rolling data sessions*, which copy data from one series to another as specified. See "Configuring Rolling Data".

For information on `active_forecasts_versions`, see "Non-Engine Parameters".



---

## Configuring the Analytical Engine

This chapter describes how to configure the analytical engine. It also introduces guidelines for configuring the forecast tree, causal factors, and the configure to order (CTO) feature.

This chapter covers the following topics:

- Configuring the Engine
- General Data Requirements
- Structure and Requirements of the Forecast Tree
- Split Forecast by Series
- Configuring SALES\_DATA node-splitting
- Guidelines for the Forecast Tree
- Guidelines for Causal Factors
- Setting Up Configure to Order

### Configuring the Engine

To configure the Analytical Engine, you generally perform the following tasks:

Engine mode	Task	Tool used	For information
Both	Loading demand data	Data Model Wizard or the Integration Interface Wizard	"Using the Data Model Wizard"  "Series and Level Integration"

Engine mode	Task	Tool used	For information
Both	Configuring the forecast tree	Forecast Tree Editor, in the Business Modeler	"Configuring the Forecast Tree"
PE mode only	Configuring influence behavior	Parameter user interface, in the Business Modeler	"Defining Influence and Competition (PE Mode Only)"
Both	Modifying tables to store causal factor data; loading data	Business Modeler or third-party database tool	"Configuring Causal Factors"
Both	Configuring the causal factors	Forecast Tree Editor, in the Business Modeler	
PE mode only	Configuring the promotional causal factors	Forecast Tree Editor, in the Business Modeler	"Configuring Promotions and Promotional Causal Factors"
Both	Creating engine profiles and adjusting engine parameters as needed	Parameter user interface, in the Business Modeler	"Tuning the Analytical Engine"
Both	Running the Analytical Engine and checking the results	Engine Administrator (separate desktop user interface)	"Using the Engine Administrator and Running the Engine"

## General Data Requirements

All the sales and causal factor data should be as complete as possible. In particular, if you do not have complete causal factor data, you may have problems like the following:

- If a causal factor does not have values for future dates, it may not have a desired effect on forecasts. For example, if the Analytical Engine has learned that changes in price have an impact on sales, and the price causal factor is not extended into the future, this implies that the future price is zero. In this case, there will be a shift in the forecast values (presumably upwards: free items "sell" well). To overcome this problem, the fill-causals method can be used by checking the fill-causal option for that causal factor.

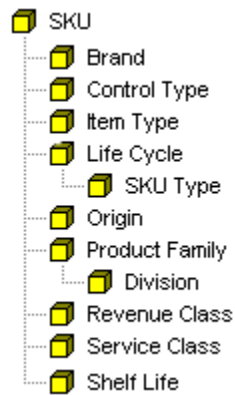
- Likewise, if the historical data is not long enough to learn the influence of all seasonal causal factors, the forecast for a missing seasonal period (for example month) may have an unexpected jump.

In general, point-of-sale (POS) data is preferable to orders. POS data is continuous, in contrast to order data, which is more sporadic, and it is easier to generate forecasts from POS data.

If you are using the Analytical Engine in PE mode, note that it is also hard to detect switching effects in order data, and the lags between promotion dates and their effects are more variable.

## Structure and Requirements of the Forecast Tree

Within the forecast tree, all item levels of the tree must belong to the same item hierarchy of the same item dimension, and all location levels must belong to the location hierarchy of the same location dimension. For example, consider the following set of item levels:



Here, the **SKU** dimension includes nine hierarchies. If you include the **Life Cycle** level in the forecast tree, that means that the only other levels you can include are **SKU** and **SKU Type**. (A true implementation would have many more levels with more nesting.)

A given level can be included in multiple, adjacent levels of the forecast tree. For example, if the lowest forecast level consists of SKUs and stores, the next level above that could be the SKUs and regions. See "Forecast Tree Example".

After you specify the levels in the forecast tree, you must indicate which levels the Analytical Engine should use for various purposes, during the forecasting process. Specifically, you must indicate the following levels in the forecast tree:

Engine mode	Level	Description	Requirements
Both	Highest fictive level (HFL)	Level at which data is completely aggregated. Includes the item HFL and the location HFL.	Created automatically.
PE mode only	<i>Influence range level</i> (IRL)	Defines the influence ranges. Each node of this level is a different influence range. Typically each IR represents a different geographical area.	<p>Must be above the influence group level (IGL).</p> <p>This is usually above the maximum forecast level.</p> <p>Oracle recommends that it is at least two levels above the IGL.</p>
Both	Maximum forecast level	Highest aggregation level at which the Analytical Engine runs.	Must be at or above the minimum forecast level.
PE mode only	Influence group level (IGL)	Defines the influence groups. Each node of this level is a different influence group.	<p>Must be at or above the lowest promotion level (LPL).</p> <p>Must be consistent with the item groups and location groups that you define in "Defining Influence and Competition (PE Mode Only)".</p> <p>Oracle recommends that it is two levels above the LPL.</p>
Both	Minimum forecast level	Lowest aggregation level at which the Analytical Engine runs.	
PE mode only	<i>lowest promotion level</i> (LPL)	Lowest level at which promotions can have different attribute values from other.	Must be at or below the minimum forecast level.

**Note:** General levels are also valid in the forecast tree. For more

information, see "Levels".

## Split Forecast by Series

Demantra time dependant information (including sales and forecast data) is always stored at the lowest possible item, location, and time granularity. And since the forecast engine is typically run at a higher level, forecast allocation or "splitting" must occur. There are two ways the engine can split higher level forecast to the lowest levels. The "matrix proportion" method splits node data according to calculated monthly proportions. These proportions are based on average historical monthly sales. If the sales history for Item A and Item B, as a percentage at a given location, has averaged 70/30 (70% Item A, 30% Item B), then by the matrix proportion method, when forecasted together in aggregate, the engine will allocate a future forecast for these items using the same proportion. However, there are circumstances where this method is not appropriate. Some items may have no sales history, or require more granular and varying allocation rules than the monthly proportions would allow. If new item models are replacements of existing models, a sales manager may want to modify the forecast allocation from what those previous sales histories would dictate. For example, the new Item B may have significant improvements over the old one, and the expectation is that a split between the two would instead be 40/60. The PromotionSeries parameter allows users to modify forecast allocation rules for this kind of situation. It allows you to use a "series-based proportion" method for splitting higher level forecast. The user must first choose which series will be used as provided by the forecast allocation logic. In addition, users must explicitly specify which combinations are to use the Series Based proportions. In cases where an aggregated forecast has both Matrix and Series based proportions, Matrix based proportions will occur.

## Configuring SALES\_DATA node-splitting

1. Open the Business Modeler and go to the Engine > Proport subtab.
2. Set the PromotionSeries to the internal name of a series that will be used for apportioning node-splitting
3. Open a worksheet that includes the population for which a series-based split is desired, and add it to the series called Engine Proportion Method
4. Modify the series "Engine Proportion Method" from "Matrix Based Proportions" to "Series Based Proportions"
5. Save Updates

## Guidelines for the Forecast Tree

When creating a forecast tree, it is important to consider the following guidelines.

- The forecast tree should include an appropriate number of levels that can be forecasted.

The forecast tree should contain 3 to 6 levels on which the engine can traverse and forecast. This number does not include any levels below the minimum forecast level and does not include the HFL.

- The forecast levels should be meaningful to the business.

The levels of the forecast tree need to have meaningfully changing data sets per level in order to be effective. A move from level to level should substantially increase the amount of data that is being analyzed by the Analytical Engine while maintaining an aggregation method that makes sense from a business perspective. A good guideline is to have each parent node aggregate between 3 to 12 lower level nodes (on average).

- The minimum and maximum forecast levels should contain reasonable and relevant data.

The minimum forecast level should have enough data to facilitate a forecast desirable by the customer. For instance, if exponential smoothing is not desired, then try to ensure that the lowest level has a long enough sales history for a non-exponential smoothing model to be active.

The maximum forecast level should still be disaggregated enough to maintain some data granularity. As a general rule, none of the maximum forecast level nodes should contain more than five percent of the total data set; this means the maximum forecast level should have at least 20 nodes, and perhaps more.

- It is useful for the forecast tree to include the level on which accuracy is measured, if possible.

Accuracy is often measured at a specific level. Often the best results can be seen if the forecast is also done at this level. This is not always true or possible but the option should be seriously considered.

- The TLMO (the level just below the top level, called top level minus one), affects performance, because it is the level for which the Analytical Engine generates the sales\_data\_engine table. (In the case of the Distributed Engine, Demantra creates multiple, temporary versions of this table.) As a consequence:

- When you are using the Distributed Engine, each engine task (distributed process) receives one or more nodes of the TLMO. In order to take best advantage of the distributed mode, it is advisable for the TLMO to have many nodes and to ensure that none of them contains too many lowest level

combinations.

- If the nodes of the TLMO are comparatively small, the Analytical Engine generates sales\_data\_engine more quickly, which reduces run time.
- If the nodes of the TLMO are comparatively small, simulation can run more quickly, for *two* reasons: because the Analytical Engine searches smaller amounts of data and because sales\_data\_engine is generated more quickly.
- When you plan the forecast tree for PE mode, consider how you will set the LPL, IGL, and IRL. It is generally good to have a large number of influence ranges, each of which has a relatively small number of influence groups. Because the effect of promotions cannot be evaluated above the IRL, that means the IRL should be a fairly high level in the tree. To minimize the number of influence groups per influence range, the IGL should be fairly close to IRL.

## Guidelines for Causal Factors

- It is important to avoid introducing too many causal factors, for mathematical reasons. For a given combination, if Demantra has more causal factors than sales data points, then it is mathematically impossible to calculate the coefficients for that combination. And as you approach the mathematical limits, the computation becomes progressively more difficult.

It is desirable to have a ratio of about 3 to 5 data points per causal factor. For mathematical reasons, you must have at least 2 more data points than causal factors for any given combination.

For example, in a monthly system, if you have two years' worth of data, that represents about 24 data points (maximum) for any combination. It would be desirable to have no more than 8 causal factors for any combination.

It is useful to count up the causal factors you plan to use and to discard any that are not truly needed, if the count is too high. Remember that you typically need the base causal factors (see "Base Causal Factors") in addition to any other causal factors you add, so be sure to include those in your count.

- Using either shape modeling feature adds causal factors, so consider carefully when to use these features. When you model a causal factor as a shape, the data for that causal factor is replaced by as many as eight shapes internally. Each internal shape is a causal factor. You can limit the number of shapes that the Analytical Engine uses internally.
- Shape modeling generally requires continuous data: POS data rather than orders. Each shape that you model should also be multiple time buckets in length; otherwise, there is no real shape to use.

- The causal factors should not be co-linear; that is, they should not have a significant degree of dependence on each other. If the causal factors are co-linear, that introduces numerical instability and the Analytical Engine can produce unreliable results.

## Additional Guidelines for PE Mode Only

- Note that when you transpose a promotional causal factor (such as a qualitative attribute), that creates additional causal factors (one for each value of the attribute). See "How the Analytical Engine Uses Promotions".
- As you create promotional causal factors, consider maintenance issues. You may not have complete control over the original location and form of the promotional data, and you may need to write procedures to maintain the promotional tables that the Analytical Engine uses.
- Pay attention to the order in which the Analytical Engine processes the promotional causal factors, which can have an impact on performance. For example, if you need to filter data, it is better to do so earlier rather than later. See "How the Analytical Engine Uses Promotions".

## Setting Up Configure to Order

Information about the functional aspects of configure to order is in *Oracle Demantra Demand Management User's Guide*.

In addition to performing the standard Oracle Demantra Demand Management setup, perform these additional steps to set up Configure to Order.

Review and update calculation parameters:

- CTO\_HISTORY\_PERIODS
- CTO\_PLANNING\_PERCENTAGE\_CHOICE

Review and update user profile options:

- MSD\_DEM: Include Dependent Demand
- MSD\_DEM Explode Demand Method
- MSD\_DEM: Two-Level Planning

Review and update administrator profile options MSD\_DEM: Calculate Planning Percentage.

If standalone running desired, develop procedure for initiating workflow CTO Download Existing Planning Pcts.



Review upload forecast processes, select an appropriate one, and develop a procedure for initiating it.

Change worksheet filters to appropriate values after implementation.

### **Calculation Parameters**

CTO\_HISTORY\_PERIODS specifies the number of historical periods to average for the history planning percentage calculation. The default is 0. If set to 0, then no historical dependent demand will be generated. It is recommended to keep the planning percentage calculation period shorter than 52 weeks as planning percentages from an average of 52 or even 26 weeks ago may be very different than current values. If options generally have demand every week, a calculation span of 26, 13, or even 8 periods may be more accurate. If options demand will be fairly sparse, setting the parameter to 52 or 26 weeks may be appropriate.

CTO\_PLANNING\_PERCENTAGE\_CHOICE specifies the planning percentage to use when series Planning Percentage Choice calculates the dependent information in Forecast Dependent Demand.

- The default is Existing; that means to use the downloaded planning percentages from Oracle e-Business Suite that are in series Planning Percentages - Existing.
- The other option is to have Oracle Demantra calculate planning percentages based on history and forecast of options.

### **Profile Options**

MSD\_DEM: Include Dependent Demand: Use this profile option to specify whether the collections process should include configure to order structures, demand, and history. Valid values are:

- Yes: Include configure to order structures, demand, and history
- - No: Do not include configure to order structures, demand, and history. This is the default

MSD\_DEM Explode Demand Method: Use this profile option to specify where to calculate dependent demand and derive planning percentages. Valid values are

- Organization Specific: Calculate dependent demand and derive planning percentages at each organization.
- Global: Calculate planning percentages at the global level and apply to all organization. The default is Organization Specific.

MSD\_DEM: Two-Level Planning: Use this profile option to specify that way that the collections process should collect family members and their sales histories. Valid values are:

- Exclude family members with forecast control 'None'. This is the default. Collect:

- Only the product family members and their sales history from the organization specified in profile option MSD\_DEM: Master Organization for which the forecast control is set to Consume or Consume and derive.
- The master organization product families from the organization specified in profile, MSD\_DEM: Master Organization for which the forecast control is set to Consume or Consume and Derive and the planning method for the product family is set to other than Not Planned.
- All the items and their sales history--even if they are not related to a product family--for all the enabled organizations for which the forecast control is Consume or Consume and Derive. These items are rolled up to the pseudo-product family Other.
- Collect all family members and their sales histories. Collect:
  - The same entities as setting Exclude family members with forecast control 'None'.
  - All the product family members and their sales history from the organization specified in profile option MSD\_DEM: Master Organization) regardless of the forecast control, as long as the product family forecast control is Consume or Consume and Derive and the planning method for product family and all of its members are set to other than Not Planned.

MSD\_DEM: Calculate Planning Percentage: Use this profile option to specify upon what entities to calculate planning percentages. This occurs when system parameter CTO\_PLANNING\_PERCENTAGE\_CHOICE instructs Oracle Demantra to calculate planning percentages.

Do not change the setting of this profile option after the initial download or invalid combinations may result:

- The typical reason for changing this value is to correct a setup error.
- In that case, delete the invalid combinations and re-download. See Implementation Considerations.

The default is Null. Valid vales are:

- Yes for Consume & Derive Options and Option Classes: Include the sales history of options and option classes
- Yes for Consume & Derive Options only: Include the sales history of options to model and removes option classes. Use it for assemble to order situations.
- Yes for all the Options and Option Classes: Include the sales history of options and option classes including optional components of forecastable product families that

have Forecast Control None.

**Note:** It is recommended to set this parameter to Yes, for All Options and Option Classes before running shipping and booking history for Configure to Order.

### Collections - e-Business Suite To Demantra

The Collect Shipment and Booking History process includes configure to order structures.

Profile option MSD\_DEM Include Dependent Demand has a value of Yes.

There is an additional process Collect BOM. It brings this information to Oracle Demantra:

- Item
- Parent Item
- Base Model
- Organization
- BOM Eff Start Date
- BOM Eff End Date
- Plng Pct- Existing

### Calculating Forecast and Dependent Demand in Demantra

Run the archive workflow to archive the prior period forecasts.

CTO Calculation runs as part of the Forecast Calculation & Approval workflow.

The CTO Forecast Calculation process generates the forecast for the history.

CTO Calculate Planning Percentages process calculates planning percentages. Its parameters determine whether planning percentages based on an average of history are calculated

- CTO\_HISTORY\_PERIODS
- CTO\_PLANNING\_PERCENTAGE\_CHOICE

**Note:** You can improve performance of the Calculate Dependent Demand workflow using the parameters listed in the table below. These parameters control the number of jobs that run and how many jobs run at the same time. To define these parameters, open the

Calculate Dependent Demand workflow for editing and then open the Calculate Dependent Demand stored procedure step. In the Properties page, click Add for each parameter you want to define and simply enter a value. Row 1 corresponds to Parameter 1, row 2 to Parameter 2, and so on. To accept the default value for a parameter, leave the row null. These parameters can be used with either Oracle 10g or 11g.

Parameter #	Description	Default Value
1	Number of slices/data partitions. Leave null (the default) to set equal to the number of base models.	null
2	Maximum number of processes (jobs) to run at a time	4
3	Sleep interval (i.e. how often the system checks the queue,) This value is in seconds.	10
4	Log level. Valid values are between 0 and 5 where 5 provides the most data.	0

### Calculating Forecast for Line of Business in Oracle Demantra

Configure the line of business population by setting Business Modeler parameters:

- LOB Population Level
- LOB Population Members

Calculate Dependent Demand for LOB is run as a step in Line of Business - Forecast workflow. It runs:

- CTO Forecast Calculation for LOB process: Generates the forecast for the history series
- CTO Calculate Planning Percentages for LOB: Calculates planning percentages. Its parameters determine whether planning percentages based on an average of history are calculated
  - CTO\_HISTORY\_PERIODS

- CTO\_PLANNING\_PERCENTAGE\_CHOICE

### **Download - e-Business Suite To Demantra**

The EBS Full Download and Import Integration Profiles process brings configure to order data into Oracle Demantra. It:

- Brings history streams, level members, and bill structures.
- Downloads planning percentages from the source using additional workflow CTO Download Existing Planning Pcts. You can also run this workflow by itself between full downloads.

### **Upload – Models and Options**

The uploads concern consensus forecast and dependent demand data (information for models and their options)

To initiate these uploads, log into Workflow Manager.

CTO Upload Global Forecast - Demand Class, Week

- Output Levels: Item, DC, Week
- Series: Consensus Forecast, Demand Priority, MAPE CTO
- Data Profiles: CTO Global Fcst-DC,Week
- Levels in the Data Profile: Item, Demand Class

CTO Upload Global Forecast - Demand Class, Week

- Output Levels: Item, DC, Week
- Series: Final Forecast Dependent Demand, CTO Parent Demand, Final Plng Pct Aggregate, Dep Demand - Demand Priority, MAPE CTO
- Data Profiles: CTO Global Dependent Demand-DC,Week
- Levels in the Data Profile: Item, Demand Class, Parent Item, Base Model

CTO Upload Global Forecast - Zone, Demand Class, Week

- Output Levels: Item, Zone, DC, Week
- Series: Consensus Forecast, Demand Priority, MAPE CTO
- Data Profiles: CTO Global Fcst-Zone,DC,Week
- Levels in the Data Profile: Item, Zone, Demand Class

CTO Upload Global Forecast - Zone, Demand Class, Week

- Output Levels: Item, Zone, DC, Week
- Series: Final Forecast Dependent Demand, CTO Parent Demand, Final Plng Pct Aggregate, Dep Demand - Demand Priority, MAPE CTO
- Data Profiles: CTO Global Dependent Demand-Zone,DC,Week
- Levels in the Data Profile: Item, Zone, Demand Class, Parent Item, Base Model

CTO Upload Global Forecast - Zone, Week

- Output Levels: Item, Zone, Week
- Series: Consensus Forecast, Demand Priority, MAPE CTO
- Data Profiles: CTO Global Fcst-Zone,Week
- Levels in the Data Profile: Item, Zone

CTO Upload Global Forecast - Zone, Week

- Output Levels: Item, Zone, Week
- Series: Final Forecast Dependent Demand, CTO Parent Demand, Final Plng Pct Aggregate, Dep Demand - Demand Priority, MAPE CTO
- Data Profiles: CTO Global Dependent Demand-Zone,Week
- Levels in the Data Profile: Item, Zone, Parent Item, Base Model

CTO Upload Local Forecast - Org, Demand Class, Week

- Output Levels: Item, Org, DC, Week
- Series: Consensus Forecast, Demand Priority, MAPE CTO
- Data Profiles: CTO Local Fcst-Org,DC,Week
- Levels in the Data Profile: Item, Organization, Demand Class

CTO Upload Local Forecast - Org, Demand Class, Week

- Output Levels: Item, Org, DC, Week
- Series: Consensus Forecast, Demand Priority, MAPE CTO
- Data Profiles: CTO Local Fcst-Org,DC,Week
- Levels in the Data Profile: Item, Organization, Demand Class

CTO Upload Local Forecast - Org, Demand Class, Week

- Output Levels: Item, Org, DC, Week
- Series: Final Forecast Dependent Demand, CTO Parent Demand, Final Plng Pct Aggregate, Dep Demand - Demand Priority, MAPE CTO
- Data Profiles: CTO Local Dependent Demand-Org,DC,Week
- Levels in the Data Profile: Item, Organization, Demand Class, Parent Item, Base Model

#### CTO Upload Local Forecast - Org, Week

- Output Levels: Item, Org, Week
- Series: Consensus Forecast, Demand Priority, MAPE CTO
- Data Profiles: CTO Local Fcst-Org,Week
- Levels in the Data Profile: Item, Organization

#### CTO Upload Local Forecast - Org, Week

- Output Levels: Item, Org, Week
- Series: Consensus Forecast, Demand Priority, MAPE CTO
- Data Profiles: CTO Local Fcst-Org,Week
- Levels in the Data Profile: Item, Organization

#### CTO Upload Local Forecast - Org, Week

- Output Levels: Item, Org, Week
- Series: Final Forecast Dependent Demand, CTO Parent Demand, Final Plng Pct Aggregate, Dep Demand - Demand Priority, MAPE CTO
- Data Profiles: CTO Local Dependent Demand-Org,Week
- Levels in the Data Profile: Item, Organization, Parent Item, Base Model

#### CTO Upload Consensus Forecast - Global, Period

- Output Levels Item, Period :
- Series: Consensus Forecast
- Data Profiles: CTO Consensus Fcst-Global,445
- Levels in the Data Profile: Item

#### CTO Upload Consensus Forecast - Global, Period

- Output Levels: Item, Period
- Series: Final Forecast Dependent Demand, CTO Parent Demand, Final Plng Pct Aggregate
- Data Profiles: CTO Final Planning Percentage - Global (445)
- Levels in the Data Profile: Item, Parent Item, Base Model

#### CTO Upload Consensus Forecast - Org, Period

- Output Levels: Item, Org, Period
- Series: Consensus Forecast
- Data Profiles: CTO Consensus Fcst-Org,445
- Levels in the Data Profile: Item, Organization

#### CTO Upload Consensus Forecast - Org, Period

- Output Levels: Item, Org, Period
- Series: Final Forecast Dependent Demand, CTO Parent Demand, Final Plng Pct Aggregate
- Data Profiles: CTO Final Planning Percentage - Local (445)
- Levels in the Data Profile: Item, Organization, Parent Item, Base Model

#### CTO Upload Consensus Forecast - Org, Week

- Output Levels: Item, Org, Week
- Series: Consensus Forecast
- Data Profiles: CTO Consensus Fcst-Org,Week
- Levels in the Data Profile Item, Organization :

#### CTO Upload Consensus Forecast - Org, Week

- Output Levels: Item, Org, Week
- Series: Final Forecast Dependent Demand, CTO Parent Demand, Final Plng Pct Aggregate
- Data Profiles: CTO Final Planning Percentage - Local (Week)



- Levels in the Data Profile: Item, Organization, Parent Item, Base Model

CTO Upload Consensus Forecast - Zone, Period

- Output Levels: Item, Zone, Period
- Series: Consensus Forecast
- Data Profiles: CTO Consensus Fcst-Zone,445
- Levels in the Data Profile: Item, Zone

CTO Upload Consensus Forecast - Zone, Period

- Output Levels: Item, Zone, Period
- Series: Final Forecast Dependent Demand, CTO Parent Demand, Final Plng Pct Aggregate
- Data Profiles: CTO Final Planning Percentage - Zone,445
- Levels in the Data Profile: Item, Zone, Parent Item, Base Model

CTO Upload Consensus Forecast - Zone, Week

- Output Levels: Item, Zone, Week
- Series: Consensus Forecast
- Data Profiles: CTO Consensus Fcst-Zone,Week
- Levels in the Data Profile Item, Zone :

CTO Upload Consensus Forecast - Zone, Week

- Output Levels: Item, Zone, Week
- Series: Final Forecast Dependent Demand, CTO Parent Demand, Final Plng Pct Aggregate
- Data Profiles: CTO Final Planning Percentage - Zone,Week
- Levels in the Data Profile: Item, Zone, Parent Item, Base Model

### **Upload – Product Families and Children**

The uploads concern product family & item level consensus forecasts and product family planning ratios (information for product families and their children).

To initiate these uploads, log into Workflow Manager.

CTO Upload Product Family Global Forecast - DC, Week

- Output Levels: Item, PF, DC, Week
- Series: Final Forecast, Demand Priority, Mean Absolute Pct Err
- Data Profiles: CTO Global Fcst-PF,DC

CTO Upload Product Family Global Forecast - Zone, DC, Week

- Output Levels: Item, PF, Zone, DC, Week
- Series: Final Forecast, Demand Priority, Mean Absolute Pct Err
- Data Profiles: CTO Global Fcst-PF,Zone,DC

CTO Upload Product Family Global Forecast - Zone, Week

- Output Levels: Item, PF, Zone, Week
- Series: Final Forecast, Demand Priority, Mean Absolute Pct Err
- Data Profiles: CTO Global Fcst-PF,Zone

CTO Upload Product Family Local Forecast - Org, DC, Week

- Output Levels: Item, PF, Org, DC, Week
- Series: Final Forecast, Demand Priority, Mean Absolute Pct Err
- Data Profiles: CTO Local Fcst-PF,DC

CTO Upload Product Family Local Forecast - Org, Week

- Output Levels: Item, PF, Org, Week
- Series: Final Forecast, Demand Priority, Mean Absolute Pct Err
- Data Profiles: CTO Local Fcst-PF

CTO Upload Product Family Consensus Forecast - DC, Week

- Output Levels: Item, PF, DC, Week
- Series: Consensus Forecast
- Data Profiles: CTO Consensus Fcst-PF,DC

CTO Upload Product Family Consensus Forecast - Org, DC, Week

- Output Levels: Item, PF, Org, DC, Week
- Series: Consensus Forecast

- Data Profiles: CTO Consensus Fcst-PF,Org,DC

CTO Upload Product Family Consensus Forecast - Org, Week

- Output Levels: Item, PF, Org, Week

- Series: Consensus Forecast

- Data Profiles: CTO Consensus Fcst-PF,Org

CTO Upload Product Family Consensus Forecast - Zone, DC, Week

- Output Levels: Item, PF, Zone, DC, Week

- Series: Consensus Forecast

- Data Profiles: CTO Consensus Fcst-PF,Zone,DC

CTO Upload Product Family Consensus Forecast - Zone, Week

- Output Levels: Item, PF, Zone, Week

- Series: Consensus Forecast

- Data Profiles: CTO Consensus Fcst-PF,Zone

### **Other Configure to Order Workflows**

In order to export the MAPE CTO series that contains forecast accuracy data for options, the Administrator must:

- Edit the Export workflow and add a step to kick off the MAPE CTO calculation procedure
- Edit the Export Integration Profile used in the workflow and replace the MAPE series with the MAPE CTO series

These are configure to order workflows that are not part of Oracle Demantra Demand Management workflows:

- CTO Download Existing Planning Pcts: Downloads existing planning percentages to Demantra configure to order and is called by process EBS Full Download. It calls Import CTO Base Model, Import CTO Level, and Import CTO Data
- CTO Calculation: Calculates the dependent demand and planning percentage calculation processes. It is called by the Forecast Calculation & Approval workflow.
- CTO Calculation for LOB : Calculates the dependent demand and planning percentage for lines of business. It is called by the Line of Business Forecast Workflow.

- Import CTO Option Price: Downloads the options' prices from the source. It is called by EBS Price List Download workflow.

### **Implementation Considerations**

The series Final Plng Pct, is a client expression and is accurate at the lowest level, for example, Item-Org. When viewing worksheets at levels higher than the lowest level, use the series Final Plng Pct – Aggregate.

The concurrent program Purge CTO Data deletes all data from the configure to order general level related tables. However the items are not deleted from the Oracle Demantra Demand Management-related tables. This program, only available as a Request, is used when an undesired option was selected from Profile MSD\_DEM: Calculate Planning Percentage option and a download run. Run the purge is run-- the profile option value changes to your desired option-- and re-execute the download.

Run the workflow CTO Calcs after the forecast is calculated either as part of a workflow or manually as the dependent demand calculations are done after the forecast has been generated. This workflow runs automatically by the seeded workflows for configure to order. However, be aware of the workflow run order in case you have customizations.

In a configure to order worksheet, if an option class, for example Harddrive, is filtered out of the worksheet, it's children are filtered out also, for example 120gig Harddrive, 200gig Harddrive.

If an option class is not in a worksheet but the independent demand of its base model is changed, the changes are propagated to the option class and its children.

If an option class is in the worksheet but its children are not, changes to the option classes dependent demand are propagated to its children.

If an option class is the child of another option class and its parent's dependent demand is changed, the changes must be propagated to its children

In worksheets, you can turn a Summary Row in the worksheet by configure to order level on and off.

You can use subtab notes at the configure to order level.

## **Recurring Item with Multiple Parents in a BOM**

In Demantra 7.3 and earlier, during Collections, if the same Option Class-Item combination is present more than once in the BOM of a Base Model, the Options under the second occurrence of the Option Class are aggregated appropriately under the first occurrence and then deleted. This second occurrence of the Option Class remains to preserve its planning percentage to the 2nd parent.

In this case, the planning percentages of the options under the first occurrence do not accurately reflect the ratio of the options to their parent. There are two solutions to this:

1. Calculate historical planning percentages which will accurately reflect the demand of the options, and choose 'History' for the Plng Pct Choice option for these options

2. The planning pct for the options under the first occurrence of Option Class 1 can also be overridden so that the planning percentages reflect the correct ratio.



**Before Collections 7.3**

	Plng Pct	Dep. Dem.	Ind. Dem.
ATO Model 1			100
Option Class 1	100%	100	
- Option 1	25%	25	
- Option 2	75%	75	
Option Class 3	50%	50	
Option Class 1	40%	20	
- Option 1	25%	5	
- Option 2	75%	15	
Option Class 2	50%	25	
- Option 3	40%	10	
- Option 4	60%	15	

**After Collections 7.3**

	Plng Pct	Dep. Dem.	Ind. Dem.
ATO Model 1			100
Option Class 1	100%	100	
- Option 1	25%	30	
- Option 2	75%	90	
Option Class 3	50%	50	
Option Class 1	40%	20	
Option Class 2	50%	25	
- Option 3	40%	10	
- Option 4	60%	15	

In Demantra version 7.3.0.1, the item is displayed the same wherever it occurs in the tree, with the correct data for each occurrence.

1.

**Before Collections 7.3.0.1**

	Plng Pct	Dep. Dem.	Ind. Dem.
ATO Model 1			100
Option Class 1	100%	100	
- Option 1	25%	25	
- Option 2	75%	75	
Option Class 3	50%	50	
Option Class 1	40%	20	
- Option 1	25%	5	
- Option 2	75%	15	
Option Class 2	50%	25	
- Option 3	40%	10	
- Option 4	60%	15	

**After Collections 7.3.0.1**

	Plng Pct	Dep. Dem.	Ind. Dem.
ATO Model 1			100
Option Class 1	100%	100	
- Option 1	25%	25	
- Option 2	75%	75	
Option Class 3	50%	50	
Option Class 1	40%	20	
- Option 1	25%	5	
- Option 2	75%	15	
Option Class 2	50%	25	
- Option 3	40%	10	
- Option 4	60%	15	

## Importing CTO Data from a non-Oracle System

This section provides an example of how to import data from a third party system into the Oracle Demantra staging tables and describes how the data will appear after it is imported.

### Initial Data

Consider the following BOM structure:

#### ATO Model 1

- **Option Class 1**
  - Option A
  - Option B

#### Option Class 3

- **Option Class 1**

- Option A

- Option B

- **Option Class 2**

- Option C

- Option D

This structure is represented in the table below.

Base Model	CTO Parent	CTO Child	DM item	Parent Item	Demand Type
ATO Model 1	ATO Model 1	ATO Model 1	ATO Model 1	ATO Model 1	Base Model
ATO Model 1	ATO Model 1	Option Class 1	Option Class 1	ATO Model 1	Option Class
ATO Model 1	Option Class 1	Option A	Option A	Option Class 1	Option
ATO Model 1	Option Class 1	Option B	Option B	Option Class 1	Option
ATO Model 1	ATO Model 1	Option Class 3	Option Class 3	ATO Model 1	Option Class
ATO Model 1	Option Class 3	Option Class 1	Option Class 1	Option Class 3	Option Class
ATO Model 1	Option Class 1	Option A	Option A	Option Class 1	Option
ATO Model 1	Option Class 1	Option B	Option B	Option Class 1	Option
ATO Model 1	Option Class 3	Option Class 2	Option Class 2	Option Class 3	Option Class
ATO Model 1	Option Class 2	Option C	Option C	Option Class 2	Option

Base Model	CTO Parent	CTO Child	DM item	Parent Item	Demand Type
ATO Model 1	Option Class 2	Option D	Option D	Option Class 2	Option

Tables 1 and 2 below show the workflows, integration interfaces, and integration profiles that are used to import the levels and data into the Demantra staging tables (BIIO\_CTO%).

Table 1: Level Integration

Data Element	Level	Workflow	Integration Interface	Integration Profile	Attribute	Integration (Staging) Table	Integration Table Column
Base Model Code	Base Model	Import CTO Base Model	CTO	IMPORT_CTO_B ASE_MO DEL	Member Description	BIIO_CTO_BASE_MODEL	T_EP_CTO_BASE_MODEL_DESC
Base Model Desc	Base Model	Import CTO Base Model	CTO	IMPORT_CTO_B ASE_MO DEL	Member Code	BIIO_CTO_BASE_MODEL	T_EP_CTO_BASE_MODEL_CODE
CTO Child Code	CTO Child	Import CTO Child	CTO	IMPORT_CTO_C HILD	Member Description	BIIO_CTO_CHIL D	T_EP_CTO_CHIL D_DESC
CTO Child Desc	CTO Child	Import CTO Child	CTO	IMPORT_CTO_C HILD	Member Code	BIIO_CTO_CHIL D	T_EP_CTO_CHIL D_CODE
DM Item Code Attribute ( this the real DM item associated with this child)	CTO Child	Import CTO Child	CTO	IMPORT_CTO_C HILD	Item	BIIO_CTO_CHIL D	T_EP_ITEM_ID

Data Element	Level	Workflow	Integration Interface	Integration Profile	Attribute	Integration (Staging) Table	Integration Table Column
CTO Code (Internal)	CTO	Import CTO Level	CTO	IMPORT_CTO_LEVEL	Member Code	BIIO_CTO_LEVEL	T_EP_CTO_CODE
CTO Desc (Internal)	CTO	Import CTO Level	CTO	IMPORT_CTO_LEVEL	Member Description	BIIO_CTO_LEVEL	T_EP_CTO_DESC
Base Model Code	CTO	Import CTO Level	CTO	IMPORT_CTO_LEVEL	Base Model	BIIO_CTO_LEVEL	T_EP_CTO_BASE_MODEL_CODE
CTO Parent Code	CTO	Import CTO Level	CTO	IMPORT_CTO_LEVEL	t_ep_cto_parent	BIIO_CTO_LEVEL	T_EP_CTO_PARENT_ID
CTO Child Code	CTO	Import CTO Level	CTO	IMPORT_CTO_LEVEL	t_ep_cto_child	BIIO_CTO_LEVEL	T_EP_CTO_CHILD_CODE
Demand Type Code	CTO	Import CTO Level	CTO	IMPORT_CTO_LEVEL	Demand Type	BIIO_CTO_LEVEL	T_EP_CTO_DEMAND_TYPE_CODE
Default Quantity per Parent (Internal EBS)	CTO	Import CTO Level	CTO	IMPORT_CTO_LEVEL	Quantity Per Parent	BIIO_CTO_LEVEL	CTO_QUAN_PER_PAR
Default Optional Flag (Internal EBS)	CTO	Import CTO Level	CTO	IMPORT_CTO_LEVEL	Is optional	BIIO_CTO_LEVEL	IS_OPTIONAL



Data Element	Level	Workflow	Integration Interface	Integration Profile	Attribute	Integration (Staging) Table	Integration Table Column
Parent Item Code (the actual Parent DM item associated with this item)	CTO	Import CTO Level	CTO	IMPORT_CTO_LEVEL	Parent Item	BIIO_CTO_LEVEL	ITEM
Planning PCT	CTO	Import CTO Level	CTO	IMPORT_CTO_LEVEL	Planning PCT	BIIO_CTO_LEVEL	PLANNING_PCT
CTO Code (internal)	CTO	Import CTO Level	CTO	IMPORT_CTO_LEVEL	Member Code	BIIO_CTO_POPULATION	LEVEL_MEMBER
BOM Start Date	CTO	Import CTO Level	CTO	IMPORT_CTO_LEVEL	Start Date	BIIO_CTO_POPULATION	FROM_DATE
BOM End Date	CTO	Import CTO Level	CTO	IMPORT_CTO_LEVEL	End Date	BIIO_CTO_POPULATION	UNTIL_DATE
Filter Level	CTO	Import CTO Level	CTO	IMPORT_CTO_LEVEL	Level Name	BIIO_CTO_POPULATION	FILTER_LEVEL
Level Order	CTO	Import CTO Level	CTO	IMPORT_CTO_LEVEL	Level Order	BIIO_CTO_POPULATION	LEVEL_ORDER
Filter Member	CTO	Import CTO Level	CTO	IMPORT_CTO_LEVEL	Member Code	BIIO_CTO_POPULATION	FILTER_MEMBER

Table 2: Data Integration

Data Element	Level	Workflow	Integration Interface	Integration Profile	Attribute	Integration (Staging) Table	Integration Table Column
Sales Date	CTO	Import CTO Data	CTO	IMPORT_CTO_DATA	Sales Date	BIIO_CTO_DATA	SDATE
CTO Code	CTO	Import CTO Data	CTO	IMPORT_CTO_DATA	CTO	BIIO_CTO_DATA	LEVEL1
Item Code	CTO	Import CTO Data	CTO	IMPORT_CTO_DATA	Item	BIIO_CTO_DATA	LEVEL2
Demand Class Code	CTO	Import CTO Data	CTO	IMPORT_CTO_DATA	Demand Class	BIIO_CTO_DATA	LEVEL3
Organization Code	CTO	Import CTO Data	CTO	IMPORT_CTO_DATA	Organization	BIIO_CTO_DATA	LEVEL4
Site Code	CTO	Import CTO Data	CTO	IMPORT_CTO_DATA	Site	BIIO_CTO_DATA	LEVEL5
Sales Channel Code	CTO	Import CTO Data	CTO	IMPORT_CTO_DATA	Sales Channel	BIIO_CTO_DATA	LEVEL6
Series	CTO	Import CTO Data	CTO	IMPORT_CTO_DATA	Dependent Booking - Book Items - Book Date	BIIO_CTO_DATA	EBS_BH_BOOK_QTY_BD_DEP

Data Element	Level	Workflow	Integration Interface	Integration Profile	Attribute	Integration (Staging) Table	Integration Table Column
Series	CTO	Import CTO Data	CTO	IMPORT_CTO_DATA	Dependent Booking - Book Items - Req Date	BIIO_CTO_DATA	EBS_BH_BOOK_QTY_RD_DEP
Series	CTO	Import CTO Data	CTO	IMPORT_CTO_DATA	Dependent Booking - Req Items - Book Date	BIIO_CTO_DATA	EBS_BH_REQ_QTY_RD_DEP
Series	CTO	Import CTO Data	CTO	IMPORT_CTO_DATA	Dependent Booking - Req Items - Req Date	BIIO_CTO_DATA	EBS_BH_REQ_QTY_RD_DEP
Series	CTO	Import CTO Data	CTO	IMPORT_CTO_DATA	Dependent History	BIIO_CTO_DATA	ACTUAL_QUANTITY_DEP
Series	CTO	Import CTO Data	CTO	IMPORT_CTO_DATA	Dependent Shipping - Req Items - Req Date	BIIO_CTO_DATA	EBS_SH_REQ_QTY_RD_DEP
Series	CTO	Import CTO Data	CTO	IMPORT_CTO_DATA	Dependent Shipping - Ship Items - Req Date	BIIO_CTO_DATA	EBS_SH_SHIP_QTY_RD_DEP

Data Element	Level	Workflow	Integration Interface	Integration Profile	Attribute	Integration (Staging) Table	Integration Table Column
Series	CTO	Import CTO Data	CTO	IMPORT_CTO_DATA	Dependent Shipping - Ship Items - Ship Date	BIIO_CTO_DATA	EBS_SHIP_QTY_SD_DEP
Series	CTO	Import CTO Data	CTO	IMPORT_CTO_DATA	Plng Pct Existing	BIIO_CTO_DATA	CTO_PLN_PCT
Sales Date	CTO	Import CTO Option Price	CTO	IMPORT_CTO_OPTION_PRICE	Sales Date	BIIO_CTO_OPTION_PRICE	SDATE
Item Code	CTO	Import CTO Option Price	CTO	IMPORT_CTO_OPTION_PRICE	Item	BIIO_CTO_OPTION_PRICE	LEVEL1
Series	CTO	Import CTO Option Price	CTO	IMPORT_CTO_OPTION_PRICE	Option Price	BIIO_CTO_OPTION_PRICE	OPTION_PRICE

### Procedure for Importing

Before importing CTO data, load all item, location, and sales data via the EP\_LOAD process. Refer to "EP\_LOAD, page 14-48" in the Demantra Demand Management to EBS Integration chapter.

After loading data into the Demantra staging tables, run the following workflows in the order specified to import data into the Demantra CTO application tables (T\_EP\_CTO%):

1. Import CTO Base Model
2. Import CTO Child
3. Import CTO Level
4. Import CTO Data

5. Import CTO Option Price

Resulting Data

The tables below provide an example of how the data will appear in the Demantra application tables after running the Import CTO workflows.

**Note:** Only the database tables and columns that are relevant to importing CTO data are shown here.

---

**Level: CTO**

**Table: T\_EP\_CTO**

**Column: T\_EP\_CTO\_CODE**

---

ATO Model 1 | ATO Model 1 | ATO Model 1

ATO Model 1 | ATO Model 1 | Option Class 1 | ATO Model 1 | ATO Model 1

ATO Model 1 | Option Class 1 | ATO Model 1 | ATO Model 1 | Option A

ATO Model 1 | Option Class 1 | ATO Model 1 | ATO Model 1 | Option B

ATO Model 1 | ATO Model 1 | Option Class 3

ATO Model 1 | Option Class 3 | Option Class 1 | Option Class 3 | ATO Model 1

ATO Model 1 | Option Class 1 | Option Class 3 | ATO Model 1 | Option A

ATO Model 1 | Option Class 1 | Option Class 3 | ATO Model 1 | Option B

ATO Model 1 | Option Class 3 | Option Class 2

ATO Model 1 | Option Class 2 | Option C

ATO Model 1 | Option Class 2 | Option D

---

---

**Level: Base Model**

**Table: T\_EP\_CTO\_BASE\_MODEL**

**Column: T\_EP\_CTO\_BASE\_MODEL\_CODE**

---

ATO Model 1

---

---

**Level: CTO Parent**

**Synonym: T\_EP\_CTO\_PARENT**

**Column: T\_EP\_CTO\_CHILD\_CODE**

---

ATO Model 1

Option Class 1 | ATO Model 1 | ATO Model 1

Option Class 3

Option Class 1 | Option Class 3 | ATO Model 1

Option Class 2

---

---

**Level: CTO Child**

**Synonym: T\_EP\_CTO\_CHILD**

**Column: T\_EP\_CTO\_CHILD\_CODE**

---

ATO Model 1

Option Class 1 | ATO Model 1 | ATO Model 1

Option A

Option B

Option Class 3

Option Class 1 | Option Class 3 | ATO Model 1

---

---

**Level: CTO Child**

**Synonym: T\_EP\_CTO\_CHILD**

**Column: T\_EP\_CTO\_CHILD\_CODE**

---

Option Class 2

Option C

Option D

---

---

**Level: Parent Item**

**Synonym: T\_EP\_CTO\_PARENT\_ITEM**

**Column: ITEM**

---

ATO Model 1

Option Class 1

Option Class 3

Option Class 2

---

---

**Level: Demand Type**

**Synonym: T\_EP\_CTO\_DEMAND\_TYPE**

**Column: T\_EP\_CTO\_DEMAND\_TYPE\_CODE**

---

Base Model

Option Class

Option

---

CTO Level Population

Table: BIIO\_CTO\_POPULATION

LEVEL\_MEMBER: T\_EP\_CTO\_CODE

FILTER\_LEVEL: Population Item and Location Level names

FILTER\_MEMBER: Population Item and Location Members

**Note:** Be sure to specify all lowest-level dimensions for both item and location. Also, this is a sample row for a Base Model; all CTO combinations should have a population entry for all dimensions of Item and Location.

	LEVEL_MEMBER (Member Code)	FROM_DATE (Start Date)	UNTIL_DATE (End Date)	FILTER_LEVEL (Level Name)	LEVEL_ORDER (Level Order)	FILTER_MEMBER (Member Code)
Location Entry:	ATO Model 1   ATO Model 1   ATO Model 1	10/4/2010	10/3/2011	Organization	1	ORG1
Item Entry:	ATO Model 1   ATO Model 1   ATO Model 1	10/4/2010	10/3/2011	Item	2	ATO Model 1

#### Additional information

- A CTO node (combination) represents the relationships between Base Model, CTO Parent, CTO Child and Item. If the BOM varies by Demand Class or other Location dimensions, then Oracle recommends that you include dimensions such as Demand Class, ORG, Site and Sales Channel. Use the default "N/A" for any dimensions that you do not use.
- To support multi-parent BOM structures, it is important to generate unique codes for CTO Child Code and CTO Code (Internal). This is done by concatenating the internal codes for the full CTO branch.

The concatenated codes for branches of the BOM structure shown at the beginning of this document are listed below as an example.

- ATO Model 1 | ATO Model 1 | ATO Model 1
- ATO Model 1 | ATO Model 1 | Option Class 1 | ATO Model 1 | ATO Model 1



- ATO Model 1 | Option Class 1 | ATO Model 1 | ATO Model 1 | Option A
- (etc)



---

## Configuring the Forecast Tree

This chapter describes how to configure the forecast tree. In the case of PE mode, it also describes how to configure the influence relationships, and competition among the combinations.

This chapter covers the following topics:

- Configuring the Forecast Tree
- Pooled Time Series
- Defining Influence and Competition (PE Mode Only)
- Defining the Forecast Tree for Service Parts Planning Supersessions
- Specifying Additional Parameters

### Configuring the Forecast Tree

**Caution:** The Promotion Optimization engine uses levels in the Oracle Demantra forecast tree, and uses their names rather than their internal identifiers. This means that if you change the name of a level, you must rebuild the forecast tree to make sure that Promotion Optimization can find the level (because the forecast tree is not automatically synchronized with the level definitions).

If you are not using Promotion Optimization, you would need to rebuild the forecast tree only if you remove a level or add a new level that you want to include in the forecast tree.

See also

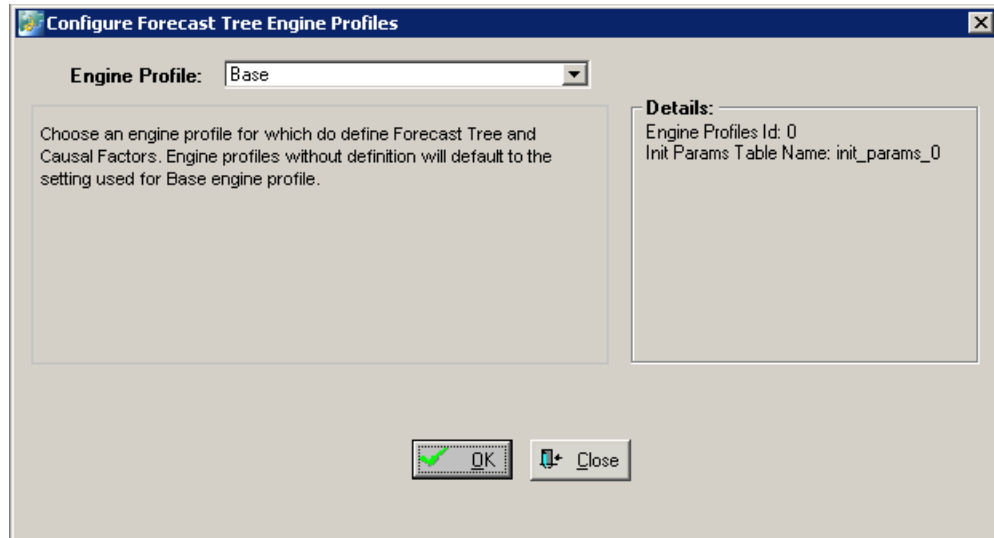
"Basic Concepts"

"Guidelines for the Forecast Tree"

### To configure the forecast tree:

1. Click Engine > Forecast Tree. Or click the Forecast Tree button.

The Configure Forecast Tree Engine Profiles dialog box appears.



2. Select the engine profile for the forecast tree you want to modify and click OK.

The Forecast Tree Editor displays lists of all the item and location levels that you have created in the system.

**Note:** General levels can also be selected in the forecast tree instead of an item or location level. For example, in the case of service parts planning, the general level "Lowest spf Level" and "SPF Latest Revision" are selected as item levels. For general levels to be available in forecast tree configuration, the engine profile must refer to a general level data table. The engine parameter EngDimDef\_ItemOrLoc determines whether the general levels will appear in the item or location dimension of the forecast tree.

**Forecast Tree (page 1)**

**Items**

**Levels for Item Groups**

Item Level	Join Field
Item Package	prod_att1_EP_ID
SKU	prd_att2_EP_ID
ABC Classification	att3_prd_EP_ID
Supplier	att4_prd_EP_ID
Channel	famatt1_EP_ID
Flavor	prodfamily_EP_ID
Item Type	prodgrp_EP_ID
Product	product_EP_ID

**Forecast Order for Items**

Order	Item Level
1	Lowest Item Level
2	Highest Fictive Level

**Location**

**Levels for Location Groups**

Location Level	Join Field
Demand Loc Type	dl_type_EP_ID
Demand Loc Country	dl_country_EP_ID
Demand Loc State	dl_state_EP_ID
Demand Loc City	dl_city_EP_ID
Demand Loc Org	dl_org_EP_ID
Demand Location	dem_loc_EP_ID
Demand Type	dem_type_EP_ID

**Forecast Order for Location**

Order	Location Level
1	Lowest Location Level
2	Highest Fictive Level

**Buttons:** Add >, < Remove, Add All >>, << Remove All, Exit, Save, Next >>

You use this dialog box to select the item levels and location levels to include in the forecast tree.

**Note:** As you select item and location levels in the following steps, add levels from the lowest level to the highest. Business Modeler automatically adds the highest fictive level to each list.

You can have different number of elements in these two lists.

3. Select the item levels to be included in the forecast tree. To do so, use the two lists at the top of the dialog box. Use any of the following techniques:
  - In the left list, double-click a row.
  - Click a row and then click Add.
  - Click Add All to transfer all items.
4. Select the location levels to be included in the forecast tree. Use the two lists at the bottom of the forecast tree, and use any of the methods described in the previous step.
5. When you have finished selecting levels, click Save.
6. Click Next.

The Forecast Tree Editor displays a dialog box that you use to build the forecast tree

itself.

Forecast Level	Item Order	Item Level	Location Order	Location Level
1		Lowest Item Level	1	Lowest Location Level
2	2	Highest Fictive Level	2	Highest Fictive Level

Add  
Delete  
Save  
Retrieve

<< Back   Exit   Next >>

In this dialog box, each row corresponds to a level in the forecast tree. In turn, a level in the forecast tree consists of one item level and one location level.

**Note:** As you build the forecast tree, add levels from the lowest level to the highest. Business Modeler automatically adds the HFL, if you do not do so explicitly.

7. To create a level in the forecast tree, do the following:
  1. Click Add.
  2. In the drop down list in the Item Order column, select an item level.
  3. In the drop down list in the Location Order column, select a location level.
8. Add more levels to the forecast tree as needed, and then click Save.
9. Click Exit or click Next.

If your system includes Promotion Effectiveness, the Forecast Tree Promotion Levels screen appears. This screen displays the forecast levels as created in the previous screen.

Forecast Level	Item Level	Location Level	Promotion Level Type
1	Lowest Item Level	Lowest Location Level	
2	Lowest Item Level	Account	
3	Family	Account	
4	Highest Fictive Level	Highest Fictive Level	

10. (PE mode only) On this screen, specify the following:

- Level to use as the lowest promotional level (LPL). This is the lowest aggregation level the Analytical Engine will consider when evaluating the effects of promotions.
- Level that defines the influence groups. This is the influence group level (IGL). This indirectly specifies the item groups and location groups.
- Level that defines the influence ranges. This is the influence range level (IRL).
- If the system includes modules AFDM, PTP or TPO, an additional screen is available. This screen controls whether the engine simply aggregates data when forecasting at higher levels or whether it groups aggregated data nodes into longer time series.
- As a default, the Forecast Detail and Forecast Range should be set to the same levels as those in the dialog with title "Forecast Tree (page 2)." For additional information regarding modifications of this screen see Pooled Time Series Below.

For example, in the row that should corresponds to the influence range level, select Influence Range from the drop down list in Promotion Level Type.

**Note:** To establish the LPL and IGL at the same level, select the option Lowest Promotion Level & Influence Group.

11. Do one of the following:

- Click Next. Business Modeler next displays the Causal Factors dialog box; see "Configuring Global and Local Causal Factors".
- Click Exit. You can return later to configure causal factors.

See also

"Guidelines for the Forecast Tree"

## Pooled Time Series

When the forecast tree calculation encounters a node for which it cannot generate a forecast, it traverses up to the next highest forecast tree node to generate the forecast. The engine then allocates an appropriate value back down, according to a proportional algorithm. For example, if a particular product/store did not have enough historical data, then the forecast engine would aggregate data at a higher level, such as product/region, generate a forecast at this level, then allocate a proportional amount of that back down to the product/store node.

Pooling a times series is a way of supplying the engine with more information than what would have been available without summing several nodes together. Instead of aggregating the data at a higher level node, it concatenates, or "pools" the data, allowing it to evaluate all the more granular data points together. The figures below show an example of these different types of data:

Item	Store	Date											
		1/12/2007	18/2/2007	1/6/2007	1/22/2007	1/29/2007	2/6/2007	2/12/2007	2/19/2007	2/26/2007	3/6/2007	3/12/2007	
AS-17-5	121A	99	89	120	20	107	114	84	230	80	108		
AS-17-6	121 B	92	90	65	165	118	83	87	10	75	115		
AS-17-8	122 C	84	117	103	140	105	74	106	40	110	120		

		Date										
Item	Store	1/1/2007	1/8/2007	1/15/2007	1/22/2007	1/29/2007	2/5/2007	2/12/2007	2/19/2007	2/26/2007	3/5/2007	3/12/2007
Az-175	All	276	297	267	325	330	271	277	300	265	342	

[illegible]

- Forecast the initial demand for a new product in an existing store (with no sales history) based on how other products pooled together at the same store
- Forecast the initial demand for existing products in a new store, based on sales at similar stores
- Better capture the lifecycle of short lifecycle products.
- Help capture the impact of a promotion type that is new to a product by leveraging its historical impact on similar products
- Capture overlapping product introductions, phase out and allocation complexities within a product family
- Better forecast new products with overlapping life-cycles

## 52-6 Oracle Demantra Implementation Guide



time series analysis is to increase the chance for utilizing data relating to irregular occurrences, or "events." Events can be simple actions, such as a change of a price at a store, or more rare events such as natural disasters. Each event may cause an increase or decrease in demand, and the pooled time series calculation is better able to reflect this when evaluating historical demand.

**Note:** For combinations with sufficient history it is still recommended the combination be forecasted independently and not in a pooled manner. Pooled information may average the behavior across several combinations and be less accurate than focusing on the data of a specific combination.

## Configuring the Forecast Tree for Pooled Time Series

When configuring the forecast tree for Pooled Time Series, an additional configuration must be made in the forecast tree. For each forecast tree level, a Forecast Detail and a Forecast Range must be defined. Forecast Detail is the data aggregation to be used during forecast generation. For example, if a region has three stores, when setting detail to level region, the forecast node value will be the aggregation of the three stores.

The Forecast Range defines how Forecast Detail nodes are pooled together to form larger data sets with additional information. A Forecast Range can be set to the same value as the Forecast Detail. In this case, data is aggregated to the Detail node and a forecast is generated for that node independently. If Forecast Range is set higher than Forecast Detail, more than one Detail node will be pooled and concatenated together, and then forecasted together. If the above region detail level is associated with a range level of country, then all aggregated region information will be pooled together and forecasted rather than each region-based node independently.

In the example below, we see that the Item/Site detail level is associated with two range levels. The first forecast tree level (where Range and Detail are set the same: Item/Site -- Item/Site) means that when the forecast calculation is trying to generate a forecast for the Item/Site detail level, it will generate a forecast at each Item/Site node individually. However, if the forecast fails at a specific Item/Site node, then the next forecast tree level will be used. At this next level, Range is greater than Detail (Item/Site – Item/Customer). This tells the forecast calculation to generate a forecast for Item/Site using the all the Item/Sites in the range of Item/Customer.

The following replaces Step 8 in the procedure above.

8. Click Next.

1. Click Add.
2. In the drop down list under Detail Levels, select an item and location level.
3. In the drop down list under Range Levels, select an item and location level.

Detail Levels		Range Levels	
Forecast Strategy	Item and Location	Item and Location	Lowest Location Level
1	Lowest Item Level	Lowest Item Level	Lowest Location Level
2	Item	Item	Site
3	Item	Item	Retailer
4	Promotion Group	Promotion Group	Retailer
5	Promotion Group	Promotion Group	Corporate
6	Highest Fictive Level	Highest Fictive Level	Highest Fictive Level

## Defining Influence and Competition (PE Mode Only)

To describe how the item-location combinations affect each other, you specify the following information:

- The level of the forecast tree to use as the IRL; each node within the IRL is an influence range.
- The level of the forecast tree to use as the IGL; each node within the IGL is an influence group. This indirectly specifies the item groups (I) and location groups (L).
- The level of the forecast tree to use as the LPL.
- The levels to use as the competition item groups (CI) and the competition location groups (CL). You specify these via parameters.

For the first three tasks, see "Configuring the Forecast Tree". To define the CI and CL, do the following:

1. For each level you create, Business Modeler creates a row in the group\_tables table for each level. Make a note of the level ID of the levels that you want to use as the CI and CL.
2. Navigate to the Parameters > System Parameters > Engine > Shell parameters. Each

value should be a level ID as given in the group\_tables table.

Parameter	Purpose
COMPETITION_ITEM	Specify the level whose members are the competitive item groups.
COMPETITION_LOCATION	Specify the level whose members are the competitive location groups.

The CI and CL should be consistent with the item groups and location groups. Specifically, any lowest level items within a given item group must belong to the same competitive item group. The easiest way to follow this rule is to set the CI equal to an item level that is higher than I and that is within the same hierarchy. A similar rule applies for the locations.

See also

"Switching Effects"

"Guidelines for the Forecast Tree"

## Defining the Forecast Tree for Service Parts Planning Supersessions

Service Parts Forecasting supports the superseding of old parts with new parts, known as supersessions in EBS Service Parts Planning. The Forecast Spares Demand engine profile has been defined to use this functionality. In particular, the Forecast Spares Demand engine profile forecasts on the t\_ep\_spf\_data table instead of SALES\_DATA. In addition, service parts forecasting also refers to two engine parameters to configure the forecast. They are:

- **GLPropSuperSessionMethod**: Defines the method general level proportions use to allocate proportions during supersessions. When set to the default for each period, proportions are allocated completely to the member with the latest starting date. If set to All Active Revisions for each period, proportions are allocated equally among all active members.
- **EngKeyDef\_Supersession**: Key used to aggregate members belonging to the same supersession set. When set to the same value as EngKeyDefPK, the proportion calculates proportions for each lowest-level member processed by the engine individually and no special handling of supersessions is done. If set above the level defined by EngKeyDefPK, then calculation of proportions is done at this aggregated level considered the supersession and all underlying combinations receive the same proportional values.

For more information about these engine parameters, see Analytical Engine Parameters,

page 58-1.

For information about how the split at supersession is done as part of the proportion, see Proport When Forecasting on General Levels, page 11-11.

## Specifying Additional Parameters

Use the Business Modeler user interface to set the following additional engine parameters, if needed:

Parameter	Purpose
max_fore_level	<p>The maximum level on the forecast tree at which a forecast may be produced. Upon failure at this level, the NAIVE model will be used, if enabled.</p> <p>For PE mode:</p> <ul style="list-style-type: none"><li>• This level is usually below the IRL.</li><li>• Sometimes the natural top forecast level does not make a good choice of IRL, and a more aggregated level would be better for the IRL. This new level may be too high for forecasting, but it is useful for calculating indirect effects. In such a case, set max_fore_level to the highest level to use for forecasting, and the IRL to the higher level.</li></ul>
min_fore_level	<p>Minimum forecast level that the engine will forecast. From that level down, the engine will split the forecast using the precalculated proportions in the mdp_matrix table.</p> <p>The engine does not necessarily create the forecast at this level. If the results are not good at this level (for a portion of the forecast tree), the Analytical Engine moves to a higher node of the forecast tree, creates a forecast there, and splits down to the minimum forecast level. As before, the engine splits using the precalculated proportions in the mdp_matrix table.</p> <p>For PE mode, this level must be at or above the LPL.</p>

For information on these parameters, see "Engine Parameters".

The Forecast Tree Editor displays a dialog box that you use to build the forecast tree itself. The Forecast Tree Editor displays lists of all the item and location levels that you have created in the system.

---

## Configuring Causal Factors

This chapter describes how to create causal factors, configure them, and populate them with data. It also describes the predefined causal factors provided by Demantra.

This chapter covers the following topics:

- Notes About Causal Factors
- Creating a Global Factor
- Creating a Local Causal Factor
- Configuring Global and Local Causal Factors
- About Activity Shape Modeling
- Enabling Activity Shape Modeling
- Deleting a Causal Factor

### Notes About Causal Factors

For each causal factor, you must provide data for all time buckets, both historical and in the future. Depending on the type of causal factor, this data is stored in different locations in the database. Causal factors are associated with a specific batch engine profile.

Causal factor type	Location of data	How to edit the table
Global factors	Column in Inputs table	Business Modeler Third-party database tool

Causal factor type	Location of data	How to edit the table
Local causal factors other than activities	Column in the sales_data table or SQL expression that aggregates data from that table	Third-party database tool
Activities	Column in the sales_data table	Third-party database tool

## Causal Factors and Engine Models

The Analytical Engine uses a set of theoretical models, each of which evaluates some or all of the data. When you configure a causal factor, you specify the following flags to specify which models should consider that causal factor:

Flag *	Meaning
short	For use by the short models (BWINT, IREGR, LOGREGR, LOGISTIC, and REGR). These models use all causal factors that they are given.
long	For use by the long models (ARLOGISTIC, CMREGR, ELOG, ICMREGR, and MRIDGE). These models examine all the causal factors they are given, but choose the ones that give the best results.
non seasonal	For use by the non seasonal models (ARIX and ARX). The only causal factors that should be flagged as non seasonal are ones that are not a predictable function of time. For example, price varies with time, but randomly, so price should be flagged as non seasonal.
multiplicative group 1	For use only by the DMULT model. If you are using this model, each causal factor should use one of these flags.
multiplicative group 2	Typically you place causal factors that vary on a daily basis into one group and place all others in the other group. No causal factor should be in both groups. See "Theoretical Engine Models".
* Name of flag as displayed in the Causal Factors screen.	

Not all models use these flags. Models not listed here do not use causal factors.

## Typical Flags for Causal Factors

Typically you initially flag causal factors as follows:

	Causal Factor	Short	Long	Non-Seasonal	Multiplicative Group 1	Multiplicative Group 2
base (predefined) causal factors	CONSTANT	yes	yes	yes	no	no
	t	yes	yes	no	yes	no
	d1, ... d12* or m1, ... m12**	yes	yes	no	no	yes
	d1, ... d7**	yes	yes	no	yes	no
	price	yes	yes	yes	yes	no
your added causal factors	If factor is a predictable function of time	usually not	yes	no	if factor varies by day	if factor varies by month
	If factor is not a predictable function of time	usually not	yes	yes	no	no

\*Included only if time resolution is monthly or weekly.

\*\*Included only if time resolution is daily.

**Important:** In many cases, these flags have to be adjusted. Contact Oracle for assistance.

## Creating a Global Factor

A global causal factor has time-varying data that applies in the same way to all items and locations.

**To create a global causal factor:**

1. Do one of the following:
  - Go into the database and add a column to the Inputs table.
  - Create the global causal factor within the Business Modeler user interface, as follows:
    1. Click Data Model > Global Factors > Options to access the global factor user interface.
    2. Click Data Model > Global Factors > New Factor. The New Factor dialog box appears.
    3. Type in the factor name.
    4. Click Add New Factor.
    5. Click Cancel to close the dialog box. The Business Modeler adds a new column to the Inputs table.
2. Load data into the new column by using a script, a database tool, or the Business Modeler.

**To use the Business Modeler, do the following**

3. Click Data Model > Global Factors > Options.
4. Click Data Model > Choose Factor. Or click the Choose Factor button.
5. The Choose Factor dialog box appears.
6. Check the check box for each of the causal factors you wish to view. Make sure that Date is selected so that you can see the dates along with the causal factor data.
7. Click OK.
8. Click Data Model > Global Factors > View. Or click the Create View button.

Business Modeler displays a table that shows the value of each global factor over time. This table displays one row for each base time bucket in the planning horizon. Each column corresponds to one global factor.



Date	T	Winter	Summer	January	February	March	April	May	June
01/01/95	1	1	0	1	0	0	0	0	0
01/02/95	2	1	0	0	1	0	0	0	0
01/03/95	3	0	0	0	0	1	0	0	0
01/04/95	4	0	0	0	0	0	1	0	0
01/05/95	5	0	0	0	0	0	0	1	0
01/06/95	6	0	1	0	0	0	0	0	1
01/07/95	7	0	1	0	0	0	0	0	0
01/08/95	8	0	1	0	0	0	0	0	0
01/09/95	9	0	1	0	0	0	0	0	0
01/10/95	10	0	0	0	0	0	0	0	0
01/11/95	11	1	0	0	0	0	0	0	0
01/12/95	12	1	0	0	0	0	0	0	0

9. Select the cell or cells to be edited. The editable cells are colored white. When selected, the cells turn yellow.
10. Click Data Model > Global Factors > Edit Data. Or click the Edit Data button.  
The Edit Data dialog box appears.
11. Type the number required and click OK.  
The data appears in each highlighted cell.
12. Click Save to save your changes.
13. Click Cancel to close the dialog box.
14. Configure the global factor as described in "Configuring Global and Local Causal Factors".

See also

Creating a Local Causal Factor, page 53-5

"Base Causal Factors"

## Creating a Local Causal Factor

A local causal factor has time-varying data that is potentially different for each item-location combination.

**To create data for a local causal factor:**

1. If the sales\_data table does not include a column that contains the data you want to use as a causal factor, go into the database and add the desired column.
2. Load data into the new column by using a script or by a database tool.
3. Configure the new causal factor as described in "Configuring Global and Local Causal Factors".

See also

"Creating a Global Factor"

**Setting up the price causal factor:**

The predefined price causal factor uses the field item\_price in the sales\_data table. You should make sure that this data is available.

**Transpose Function**

This allows one causal to be converted into several causals using a transpose function, with different transpose values in different periods resulting in engine accepting multiple causals instead of one. Transpose function should only refer to sales\_data and mdp\_matrix tables. Transpose is done after information is aggregated and it is important to ensure underlying values are consistent.

## Configuring Global and Local Causal Factors

Here you provide information about how the Analytical Engine should use each global and local causal factor.

**To configure a causal factor:**

1. Click Engine > Forecast Tree. Or click the Forecast Tree button.
2. Select the batch engine profile for which you want to configure causal factors and click OK.

**Note:** The simulation engine profiles inherit the causal factor settings from the parent batch engine profile.

3. Click Next repeatedly until you reach the Causal Factors dialog box.
4. If the causal factor is not yet listed here, do the following:
  1. Click Add.

A new line is displayed.

2. Describe the new causal factor by specifying the following:

---

Factor Name	<p>Depends on the type of causal factor:</p> <ul style="list-style-type: none"><li>• For a global factor: name of an existing column in the Inputs table.</li><li>• For a local causal factor, this can be the name of an existing column in the sales_data table. The factor name can also just be a name; in this case, you must specify an expression in the Local Function field.</li><li>• For an activity: name of a column in the sales_data table. Business Modeler adds this column automatically if it does not yet exist.</li></ul>
Factor Type	<p>Choose one of the following:</p> <ul style="list-style-type: none"><li>• global</li><li>• local</li><li>• activity (a special kind of local causal factor that supports shape modeling)</li></ul> <p>Do not use the event choice, which is an older implementation of the more general local choice. The price option is useful only for the predefined price causal factor.</p>

---

3. Specify how the Analytical Engine should use the causal factor. To do so, specify the following values:

---

Short	Usually you enable this check box only for the following global causal factors: Constant, t, d1, ... d12. See "Typical Flags for Causal Factors".
Long	Usually you enable this check box for all causal factors. See "Typical Flags for Causal Factors".

---

---

Multiplicative Group 1	<p>Enable this check box to include this causal factor in the first multiplicative group for use by the DMULT model; you should enable this check box for at least one causal factor.</p> <p>Typically you place causal factors that vary on a daily basis into one group and place all others in the other group. No causal factor should be in both groups. see "Theoretical Engine Models".</p> <p>This setting affects only the DMULT model.</p>
Multiplicative Group 2	<p>Enable this check box to include this causal factor in the second multiplicative group for use by the DMULT model; you should enable this check box for at least one causal factor.</p> <p>This setting affects only the DMULT model.</p>
Non Seasonal	<p>Enable this check box if the data associated with this causal factor is not known to be a predictable function of time. For example, price varies with time, but randomly, so price should be flagged as non seasonal. See "Typical Flags for Causal Factors".</p>
Fill Causals	<p>Specifies whether Demantra should interpolate when values are missing for a date. The missing local causal factor will receive the average of its nearest two non-missing neighbors.</p> <p>For example:</p> <ul style="list-style-type: none"> <li>• If the causal values are 1, missing, and 2, then Demantra replaces the missing value with 1.5.</li> <li>• If the causal values are 1, missing, missing, missing, and 2, then Demantra replaces each missing value with 1.5.</li> </ul>
Shape Indicator	<p><b>Only for activities.</b> Specifies whether Demantra should perform shape analysis and calculations on this activity.</p>

---

---

Omit Seasonal

**Only for activities.** This option specifies whether to nullify values of the global seasonal causal factors for the time buckets during which the causal factor occurs. Specifically this refers to the causal factors d1—d12 or d1—d7 and m1—m12. For example, if you have monthly data and you omit seasonal effects for a given causal factor Promo1, that means that Demantra switches off the causal factors for the duration of Promo1.

By omitting seasonal effects, you enable Demantra to capture the shape more clearly for the analysis. This option is suitable only if you expect the effect of this causal factor to be much stronger than the seasonal effects.

If causal factors overlap each other, then Demantra gives precedence to the causal factor that you have flagged to omit seasonal effects.

Local Function

**Only for local causal factors.** An SQL expression that describes how to aggregate causal factor data from the lowest level data. Use one of the following SQL aggregating functions:

- Min
- Max
- Sum
- Avg

Within the expression, refer to the name of the causal factor (the column name in which the causal factor is stored).

Within the expression, you can also refer to fields in the mdp\_matrix table.

You can also include tokens of the form #FORE@<Version>#. See "Server Expression Functions and Operators".

---

4. Add comments to the Comments field, if desired.
5. Click Validate to check the validity of the configuration.

6. Click Save to save changes.

7. Do one of the following:

- Click Next.

If you created any activities, Business Modeler displays a message indicating the name of the series that it automatically creates for each activity.

For PE mode, then the Business Modeler displays the Promotional Causal Factors dialog box. See "Configuring Promotional Causal Factors".

- Click Exit.

See also

"Configuring Promotional Causal Factors"

"Base Causal Factors"

## About Activity Shape Modeling

In shape modeling, you capture the profile of the demand over the duration of a promotion. The Analytical Engine models the overall demand as a linear combination of Oracle proprietary shapes, as many as eight shapes; this information replaces the normal causal factor that would have been used instead. The Analytical Engine calculates the coefficients for each shape, for each relevant combination.

Remember that when you enable shape modeling for a causal factor, the single causal factor is replaced by up to eight causal factors. To keep the number of causal factors down, you can specify the maximum number of shapes permitted for activity shape modeling.

**Note:** Shape modeling capabilities are different in the two engine modes:

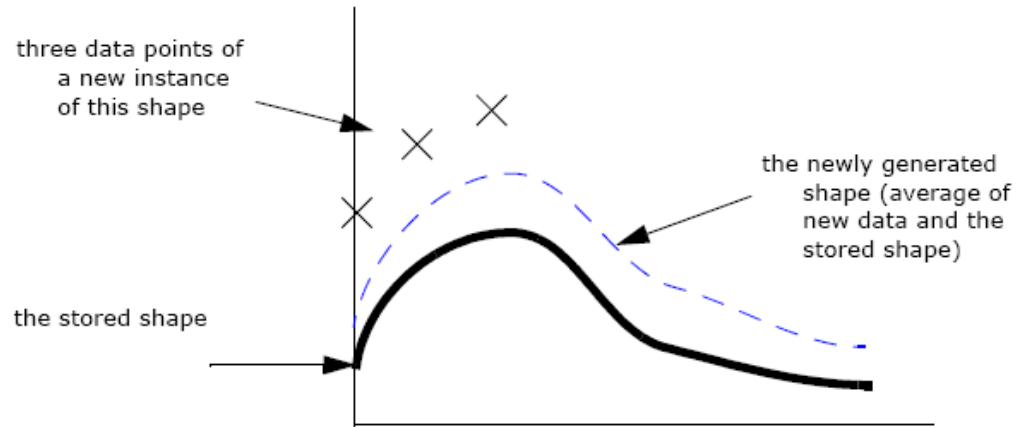
- In DP mode, the engine supports only activity shape modeling.
- In PE mode, the engine supports both activity shape modeling *and* promotional shape modeling. See "About Promotion Shape Modeling".

See "Engine Modes: DP and PE".

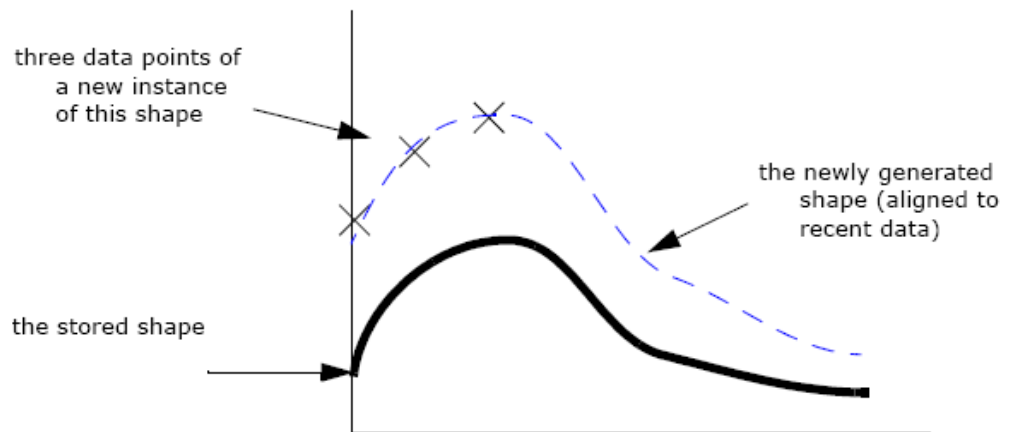
## Shape Alignment

Each stored shape is an average of the past instances of that particular shape. It is important to understand that the stored *information* consists of both the shape and the actual amplitude of the curve.

When the Analytical Engine observes the beginning of a new instance of a given shape, it is necessary to decide how to set the amplitude of the new curve that it generates. By default, the engine assumes that the amplitude of the stored shape should be taken into consideration. Therefore, when the Analytical Engine generates the new shape, it averages the new data together with the stored shape, as follows:



The default behavior is appropriate when the history contains many instances of a given shape. When the shape is new to the system, however, it is more appropriate to force the Analytical Engine to re-scale the generated shape so that it aligns with the most recent observations:



To force this realignment you use the QAD (quantity alignment duration) series associated with the shape. This series specifies the number of time buckets during which this alignment should occur, starting with the beginning of the shape. If you need to align the shape, you generally should align the entire shape; that is, you set the series equal to the expected length of the shape.

## Samples of Activity Shape Modeling

To see samples, use the UPGRADE\_TO\_SHAPE\_MODELLING procedure, which does the following:

- It creates two sample activity causal factors: Product\_launch and Price\_change.
- It creates four series for the benefit of end users:
  - Price\_change
  - Price\_change\_QAD
  - Product\_launch
  - Product\_launch\_QAD

See also

"Enabling Activity Shape Modeling"

## Enabling Activity Shape Modeling

### To enable activity shape modeling:

1. For each specific shape you want to represent, create the causal factor data, as described in "Creating a Local Causal Factor".
2. Configure this causal factor as type Activity, as described in "Configuring Global and Local Causal Factors".

When you configure this causal factor as an activity (named, for example, Product Launch), the Business Modeler automatically creates two series that constitute the user interface for the activity. These series are as follows:

Generic name / Example name	Purpose
<i>Causal-factor-name/</i> <b>ProductLaunch</b>	Lets the user indicate the start and duration of the activity associated with a specific combination. Within this series, for each date, the user chooses "Start" or "Active" from a drop-down menu to specify the promotion start and continuation dates. The default is "None," meaning no promotion. The user identifies past activities and marks where future activities will occur.



Generic name / Example name	Purpose
<i>Causal-factor-name_QAD/</i> <b>ProductLaunch_QAD</b>	<p>Controls whether the Analytical Engine re-scales the generated shape to align with the amplitude of the most recent observed instance of this shape, for a given combination.</p> <p>By default, this is zero, and the Analytical Engine <i>averages</i> the most recent data with the stored shape, which is an average of all the past observations of this shape.</p> <p>When the shape is "new" to the system, the user should set ProductLaunch_QAD equal to the typical length of the activity, so that the new data takes precedence.</p>

3. Add these series to a worksheet at the appropriate aggregation level.
4. Edit the *Causal-factor-name* series to identify when the activity occurred and when it will occur and save the changes.
5. If appropriate, use the *Causal-factor-name\_QAD* series to control whether to realign the shape. Edit the series and save the changes.
6. To specify the maximum number of Oracle proprietary shapes that the Analytical Engine can use for activity shape modeling, set the NumShapes parameter.
7. Run the Analytical Engine as usual.

See also

"About Activity Shape Modeling"

## Deleting a Causal Factor

### To delete a causal factor:

1. Click Engine > Forecast Tree. Or click the Forecast Tree button.
2. Select the Engine Profile that contains the causal factor you want to delete and click OK.
3. Click Next repeatedly until you reach the Causal Factors dialog box or (PE mode only) the Promotional Causal Factors dialog box.
4. Click the causal factor you want to delete.

5. Click Delete.
6. Business Modeler asks for confirmation. Click Yes or No.

See also

"Configuring Global and Local Causal Factors"

"Configuring Promotional Causal Factors"

---

## Configuring Promotions and Promotional Causal Factors

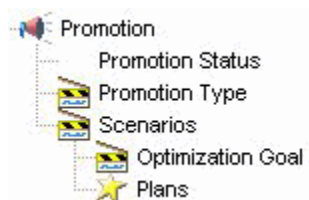
This chapter describes how to configure promotions and promotional causal factors in the Business Modeler.

This chapter covers the following topics:

- Base Behavior
- Customizing the Promotion Levels
- Loading Historical Promotions
- How the Analytical Engine Uses Promotions
- Configuring Promotional Causal Factors
- Adjusting the Promotion Dates
- About Promotion Shape Modeling
- Enabling Promotion Shape Modeling

### Base Behavior

The Demantra installer automatically defines the following required promotion levels:



These levels have the following purposes:

Level	Purpose	Permitted Customization
Promotion	<p>Defines the promotions themselves. This level must define all the possible attributes that can be associated with promotions.</p> <p>This level must include the Population attribute, which specifies the item-location combinations and the time span with which the promotion is associated.</p>	Add attributes only.
Promotion Status	<p>Controls the following:</p> <p>Whether the promotion is used in forecasting.</p> <p>Whether users can edit the promotion.</p> <p>How DSM uses the promotion.</p> <p>See "Promotion Status".</p>	
Promotion Type		
Scenarios	Provides optional organizational structure, for the benefit of users, particularly within a worksheet.	Any change is allowed.
Optimization Goal	For use by the Promotion Optimization module only; see the Oracle Demantra Release Notes.	
Plans	Provides optional organizational structure, for the benefit of users, particularly within a worksheet.	Any change is allowed.

## Promotion Status

The values of Promotion Status are as follows:

Status	Meaning		
	In forecasts	In worksheets	In DSM
Unplanned	The Analytical Engine does not consider this promotion.	Users can edit the promotion.	DSM considers this promotion to be uncommitted.
Planned	The Analytical Engine does consider this promotion.		
Committed		The user who committed this promotion can edit it.	DSM considers this promotion to be committed
Running		Nobody can edit the promotion.	The promotion is currently running.
Unmatched			The promotion has ended but has not yet been matched to an invoice.
Matched			The promotion has ended and has been matched to an invoice.

For information on DSM, see the *Oracle Demantra Deduction and Settlement Management User's Guide*.

## Customizing the Promotion Levels

You typically customize the promotion levels by adding attributes, although other changes are also permitted; see "Base Behavior". To customize the promotion levels, you use the Business Modeler.

You should also configure the Activity Browser of the worksheets, which displays a hierarchy of promotions. The Activity Browser has the same structure for all worksheets. To configure it, you use the Business Modeler.

For information, see "Configuring the Activity Browser"

## Loading Historical Promotions

To load historical promotions, use the Integration Interface Wizard, described in "Series and Level Integration". Create and execute an integration interface that loads both the

promotion members (via a level profile) and any promotion data (via a data profile).

## How the Analytical Engine Uses Promotions

The Analytical Engine does not directly use the promotions for forecasting. Instead, it converts their attributes to promotional causal factors, which it then converts to normal causal factors.

In this process, it uses the configuration information that you provide in the Business Modeler. For each promotional causal factor, the key options are as follows:

- Column Name Expression
- Filter
- Transpose by Column
- Merge Function
- Aggregation Function

It is important to understand how the Analytical Engine uses these options. The following sections describe how the Analytical Engine starts with promotions and converts them to causal factors. For more details on the engine flow, see "Engine Details".

These options use expressions that refer to promotion data. Note that these expressions can refer only to the tables that are used by the levels within the hierarchy of the analytical general level (promotion). For example: `promotion_data` or `promotion`.

For information on setting these options, see "Configuring Promotional Causal Factors".

## Kinds of Attributes

In this discussion, it is useful to consider the general kinds of promotional attributes that the Analytical Engine can use:

- *Quantitative attributes* such as discount. These attributes have numeric values that the Analytical Engine can use in their present form. The Analytical Engine assumes that the effects of these attributes is correlated with the value of the attribute. For example, if `discount1` is larger than `discount2`, then `discount1` has a larger effect on demand than `discount2`.

The Analytical Engine does not assume that the correlation has a positive sense.

- *Boolean attributes*, where the attribute either has a value or does not have a value.
- *Qualitative attributes*, where the attribute can have one value from a given set of values. The set of values is unordered, which means that even if the values are

numeric, there is no intrinsic meaning in the relative sizes. For example, you might use numeric color codes 4 and 5, but color 5 does not have a larger effect on demand than color 4.

Demantra converts this kind of attribute into a set of unrelated causal factors. For example, color code 4 is one causal factor and color code 5 is another. For any given promotion, this causal factor either has a value or does not have a value.

## Step 1: Aggregate Promotion Attributes to the LPL

The lowest promotion level (LPL) is a level in the forecast tree. Specifically, it is the lowest level at which promotions can have different attribute values from other, and it must be at or below the minimum forecast level.

The Analytical Engine retrieves the promotional causal factor data and aggregates it to the LPL, as specified by the Column Name Expression. You use an aggregating expression like the following example:

```
max(promotion_data.discount)
```

**Note:** As with all the options discussed here, the expression can refer only to tables that are used by the levels within the hierarchy of the analytical general level (promotion). For example: promotion\_data or promotion.

As a general rule, an expression that uses the max function is probably appropriate in most cases, because promotions should have the same attribute values below the LPL, by definition.

## Step 2: Applying Filters

Sometimes you need to convert one set of promotional causal factor data into multiple causal factors. To do so, you use a Filter expression, an aggregating expression that evaluates to true or false. The promotional causal factor uses only the data for which the expression is true. You typically create multiple promotional causal factors, each with a different filter expression that uses a different part of the source data.

For example, consider the following promotional data. This table contains one row for each promotion for a given combination and time bucket. (For simplicity, the table shows only one combination, one time bucket, and three promotions.) The promotion\_data table shows values of attributes (promo\_type and discount) associated with those promotions.

Item	Location	Date	Promotion	Promo_type	Discount
100	333	1	214	1	15

Item	Location	Date	Promotion	Promo_type	Discount
100	333	1	296	2	5
100	333	1	340	3	10

Suppose that we have configured the following promotional causal factors:

Factor Name	Column Name Expression	
Special Discount	max(promotion_data.discount)	max(promotion_data.promo_type=3)
Discount	max(promotion_data.discount)	max(promotion_data.promo_type<3)

Internally, Demantra would convert the promotion attributes to the following promotional causal factors:

Item	Location	Date	Promotion	Special Discount	Discount
100	333	1	214	0	15
100	333	1	296	0	5
100	333	1	340	10	0

### Step 3: Transposing Promotion Attributes

Next, the Analytical Engine considers the Transpose by Column setting, which you use for qualitative promotion attributes. This setting converts a single promotion attribute into multiple causal factors. For quantitative or Boolean attributes, specify 0, which means that Promotion Effectiveness can use the attributes as casual factors in their present form.

For example, suppose that promotions use different "delivery types," which correspond to different mechanisms such as circulars, extra product samples, coupons, and so on. Each of these mechanisms might have a different affect on sales. Suppose we have the following example data in the promotion\_data table:



Item	Location	Date	Promotion	Delivery_type
150	344	1	214	4
150	344	1	296	5
150	344	1	340	6

Because delivery\_type is a qualitative attribute, it is generally appropriate to transpose it. We could configure a Delivery Type promotional causal factor, as follows:

Factor Name	Column Name Expression	Filter	Transpose by Column
Delivery Type	max(promotion_data.Delivery_type)	null	max(promotion_data.Delivery_type)

Note that Transpose by Column must be an aggregating expression.

Using this configuration data, Demantra would internally convert the preceding promotion attributes into the following set of promotional causal factors:

Item	Location	Date	Promotion	Delivery Type(4)	Delivery Type(5)	Delivery Type(6)
150	344	1	214	4	0	0
150	344	1	296	0	5	0
150	344	1	340	0	0	6

You may want to transpose by a promotion attribute (as in this example) or by members of a level in the promotion hierarchy.

**Note:** If you use the members of a level to transpose an attribute, be sure to first filter out the default member (which has an ID of 0) of that level.

## Step 4: Merging Across Promotions

Next the Analytical Engine uses the Merge Function setting, which describes how to

merge promotional causal factors that occur on the same date at the same item-location combination (thus merging across all the promotions for that combination and date).

For Merge Function, you can choose one of the functions provided by Business Modeler.

The way that you merge depends upon the meaning of the data in the promotional causal factor. For example, if you have multiple discounts on the same date, you would want to merge them by the compound rule (so that 10% and 20% are merged to 28%).

## Step 5: Aggregating Attributes within the IGL

The influence group level (IGL) is another level in the forecast tree. The Analytical Engine uses this level to simplify the computational problem. It creates the following three historical promotional causal factors for each node in the forecast tree:

---

self	Influence on this node caused by attributes on this node
own	Influence on this node caused by other nodes within the same IG
other	Influence on this node caused by all other IGs within the IR

---

In this last step, the Analytical Engine uses the Aggregation Function option, which describes how to aggregate the promotional causal factor to the IGL.

**Note:** The Analytical Engine uses the same option whenever it needs to aggregate to higher levels for forecasting purposes.

For Aggregation Function, you can choose one of the functions provided by Business Modeler.

## Configuring Promotional Causal Factors

This section describes how to configure promotional causal factors. You can do most of the work within the Business Modeler, but it is necessary to go into the database for the final steps.

**Note:** Causals participating in Optimization require these additional settings

- Opti\_Causal\_Type
  - 0=Price Decrease \$

- 1=Price Discount %
- 2=Spent Not Used?
- 3=Boolean
- Opti\_Causal\_Output= The promotion table column containing optimized causal result.
- Opti\_Transpose\_Output=The promotion table column containing optimized causal transpose result.

**Note:** The engine only evaluates causals with direct effect. Causals with indirect effects are not evaluated.

#### **Analytical Recommendations**

- Limited Number of Causals
- Quantitative or Boolean Causals only
- Qualitative Causals supported through Transpose

#### **To configure promotional causal factors:**

1. Click Engine > Forecast Tree. Or click the Forecast Tree button.
2. Click Next repeatedly until you reach the Promotional Causal Factors screen.  
Each row in this screen specifies a promotional causal factor.
3. To add a new promotional causal factor for Promotion Effectiveness, click Add.  
A new line is added.
4. Describe the promotional causal factor by specifying the following:

---

Factor Name	Name of the promotional causal factor. This name should consist only of alphanumeric characters.
-------------	--------------------------------------------------------------------------------------------------

---

---

Column Name Expression	<p>An expression that retrieves the causal factor (promotional attribute) data and aggregates it to the LPL. For example:</p> <pre>max(promotion_type.is_ad)</pre> <p>An expression that uses the max function is probably appropriate in all cases, because promotions should have the same attribute values below the LPL, by definition. See "Step 1: Aggregate Promotion Attributes to the LPL".</p>
Filter	<p>An aggregating expression that returns the true or false value. This expression filters out promotion data that should not be used for this promotional causal factor; that is, the promotional causal factor uses only the rows for which this expression returns true. See "Step 2: Applying Filters".</p>
Transpose by Column	<p>An aggregating expression that returns the values by which the data is to be transposed. You usually transpose an attribute only if it is qualitative. To avoid transposing, use the value 0.</p> <p>See "Step 3: Transposing Promotion Attributes".</p> <p><b>Note:</b> If you use the members of a level to transpose an attribute, be sure to first filter out the default member (which has an ID of 0).</p>
Merge Function	<p>Specifies how Oracle Demantra should internally merge promotions of the same kind that apply to the same item, location, and time. Click one of the following:</p> <ul style="list-style-type: none"> <li>Compound (Use only for numeric causal factors. All values must be greater than or equal to 0 and less than 1; otherwise, this function throws an error.)</li> <li>WAVR (Weighted average. Use only for numeric causal factors. If you use this option, also specify Merge Function Column.)</li> <li>Boolean (Use for boolean causal factors or for transposed causal factors.)</li> </ul> <p>See "Step 4: Merging Across Promotions".</p>
Merge Function Column	<p>Applies only if you select WAVR for the merge function. Specifies the weights to use when performing a weighted average.</p>

---

---

The preceding expressions can refer only to the tables that can be logically linked to levels within the hierarchy of the analytical general level (promotion). For example: promotion\_data or promotion

Causal From Expression	Specifies which tables the causal will be referencing. Should include all tables referenced in Column Name Expression, Filter and Transpose.
Causal Where Expression	Should only be used if Causal From Expression is populated. Specifies logical links between tables defined in Causal From Expression to promotion and promotion_data tables. For example promotion_data.promotion_type_id = promotion_type.promotion_type_id and promotion_data.promotion_id=promotion.promotion_id
Aggregation Function	<p>Specifies how Demantra should internally aggregate this promotional causal factor across combinations, whenever it is necessary do so. Click one of the following:</p> <ul style="list-style-type: none"><li>• WAVR (Weighted Average. For the weights, Demantra uses the stored proportions of the combinations)</li><li>• Boolean (Typically you use this if you have transposed a causal factor.)</li><li>• Sum</li></ul> <p>See "Step 5: Aggregating Attributes within the IGL".</p>
Priority	Ignore this field.
Influence	<p>Specifies the effect of this promotional causal factor on other members. Click one of the following options:</p> <ul style="list-style-type: none"><li>• Has effect on other members (this causal factor can affect other combinations, in addition to the combinations with which it is associated)</li><li>• Has direct effect only (this causal factor affects only the specific combinations with which it is associated)</li><li>• Has only indirect effect (this causal factor affects only the combinations with which it is not associated)</li></ul>

---

5. Describe how the Analytical Engine should use this promotional causal factor. To do so, specify the following values:

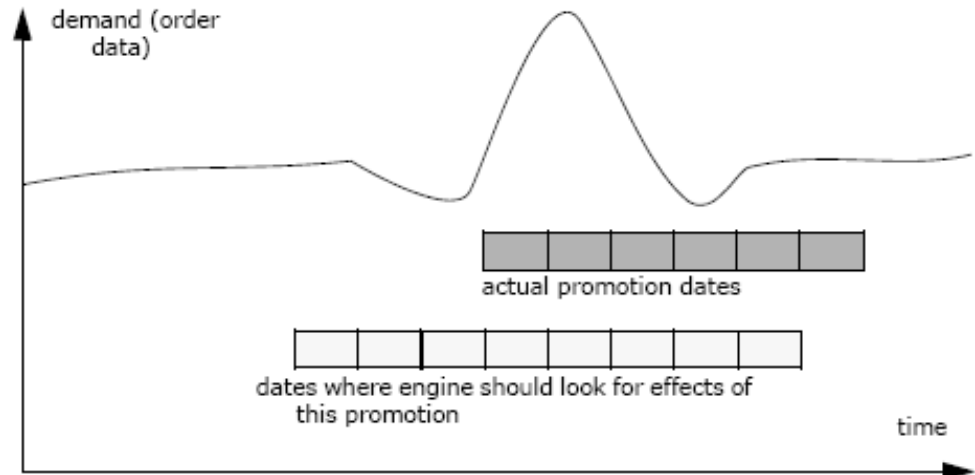
Short	Usually you enable this check box only for the following global causal factors: Constant, t, d1, ... d12. See "Causal Factors and Engine Models".
Long	Usually you enable this check box for all causal factors. See "Causal Factors and Engine Models".
Multiplicative Group 1, Multiplicative Group 2	Ignore these options, which do not affect promotional causal factors.
Non Seasonal	Enable this check box if the data associated with this causal factor is not known to be a predictable function of time. For example, price varies with time, but randomly, so price should be flagged as non seasonal. See "Causal Factors and Engine Models".
Self Shape Indicator	Enable this check box if this promotion causal factor should be represented as a shape. See "About Promotion Shape Modeling".
IG Shape Indicator	Enable this check box if this promotion causal factor should be represented via shape modeling when it is aggregated to the IGL.
Omit Seasonal	<p>Enable this check box if you want to nullify values of the global seasonal causal factors for the time buckets during which the causal factor occurs. Specifically this refers to the causal factors d1—d12 or d1—d7 and m1—m12. For example, if you have monthly data and you omit seasonal effects for a given causal factor Promo1, that means that Demantra switches off the causal factors for the duration of Promo1.</p> <p>By omitting seasonal effects, you enable Demantra to capture the promotion shape more clearly for the shape analysis. This option is suitable only if you expect the effect of this causal factor to be much stronger than the seasonal effects.</p> <p>If causal factors overlap each other, then Demantra gives precedence to the causal factor that you have flagged to omit seasonal effects.</p>
Num Shapes	Specify the maximum number of allowed shape causal factors for the engine to use for a given node in the forecast tree, for this promotional causal factor. Use an integer from 0 to 8, inclusive.

6. Click Validate to check the validity of the configuration.
7. Click Save.

8. Now you can return to previous dialog boxes to make further changes. Or click Finish to exit.

## Adjusting the Promotion Dates

By default, Promotion Effectiveness assumes that a promotion has an effect between its start and end dates, as provided to Oracle Demantra. Typically the promotion has an actual effect in a slightly different span of time, as in the following example:



You can adjust the dates used by the Analytical Engine in two complementary ways:

- To specify the number of days to add to the sales date for any given promotion, set the ShiftDynPromoDate, which is typically an expression that varies by promotion and that returns a negative number. In the preceding example, ShiftDynPromoDate equals -1 for the promotion shown here.

**Note:** Alternatively, to specify an overall shift in time for all promotions, set the ShiftPromoCausals parameter.

- To stretch a promotion by adding time buckets to the beginning or end, do the following.
  - Decide which attribute or attributes have pre and post-promotional effects.
  - Enable shape modeling for those promotional causal factors; see "About Promotion Shape Modeling".
  - For those promotional causal factors, set Num Shapes equal to 1.

- The user must change the `pre_effect` and `post_effect` settings of the combination, which default to zero. These settings (in `mdp_matrix`) specify the number of buckets to search backwards and forwards outside the promotion dates. In the preceding example, we set `pre_effect` equal to 2.

Typically you also set the `ShiftPromoMaxValue` parameter, to make sure that you adjust the dates of promotions in the near future (rather than adjusting only historical promotions).

See also

"How the Analytical Engine Uses Promotions"

"Configuring Global and Local Causal Factors"

"Deleting a Causal Factor"

## About Promotion Shape Modeling

In shape modeling, you capture the profile of the demand over the duration of a promotion. The Analytical Engine models the overall demand as a linear combination of Oracle proprietary shapes, as many as eight shapes; this information replaces the normal causal factor that would have been used instead. The Analytical Engine automatically associates a different shape with each value of the promotional attribute that uses shape modeling. The Analytical Engine calculates the coefficients for each shape, for each relevant combination.

**Note:** The feature described here is available *in addition to* activity shape modeling; see "About Activity Shape Modeling".

## When to Enable Shape Modeling

You should enable shape modeling only if the following are all true:

- The demand data is continuous (point-of-sale data rather than order data).
- The typical length of a promotion is more than one time bucket.
- You need to search for pre and post effects of promotions.

## Other Considerations

Remember that when you enable shape modeling for a promotional causal factor, the single promotion causal factor is replaced by up to eight causal factors. If the promotional causal factor is transposed, that adds even more causal factors: up to eight for each column that the transpose creates.



To keep the number of causal factors down, you can specify the maximum number of shapes permitted for any given promotional causal factor.

## Enabling Promotion Shape Modeling

### **To enable promotion shape modeling:**

1. Identify the promotional causal factors that you want to represent as shapes.
2. On the Promotional Causal Factors screen, make sure to check the Self Shape Indicator option for each of those promotional causal factors.
3. Consider also setting the following options.
  - IG Shape Indicator
  - Omit Seasonal
  - Num Shapes



---

## Tuning the Analytical Engine

It is usually necessary to adjust some parameters to configure the Analytical Engine correctly before running it the first time. Other adjustments can be made later to optimize the behavior and performance.

This chapter covers the following topics:

- Editing Engine Parameters
- Creating or Renaming Engine Profiles
- Tuning Analytics
- Tuning Performance
- Reconfiguring the sales\_data\_engine Table
- Enabling Engine Models Globally
- Configuring the Engine Mode
- Advanced Analytics (Nodal Tuning)

### Editing Engine Parameters

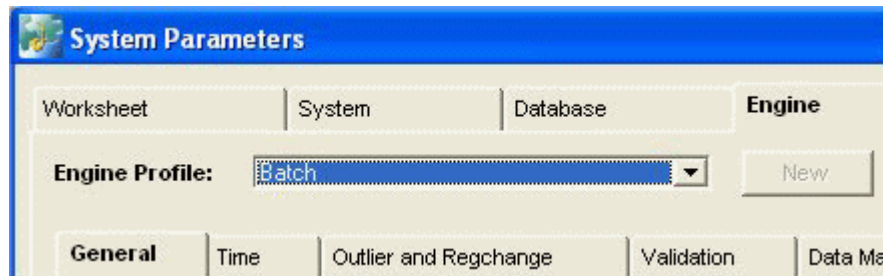
To tune the Analytical Engine, you modify values of two types of engine parameters:

- Global parameters that apply to the engine or to most or all of the forecasting models. For convenience, you define *engine profiles*, which are sets of engine parameters with specific values. Demantra provides some predefined profiles for different purposes, and you can define additional engine profiles, as needed.
- Parameters that apply to specific forecast models.

#### **To edit the global engine parameters:**

1. Log onto the Business Modeler.

2. Click Parameters > System Parameters. The System Parameters dialog box appears.
3. Click the Engine tab.



4. From the Engine Profile drop-down, select the engine profile whose parameter settings you want to adjust.
5. Find the parameter of interest. The dialog box provides find, sort, and filter capabilities to help you with this. See "Engine Parameters".
6. To change the value of the parameter, click the Value field for that parameter. Type the new value or select a value from the drop-down menu.
7. Click Save to save your changes to this profile.
8. Click Close.

#### **To edit specific model parameters:**

To edit most model-specific parameters, you must work directly within the Demantra database. For information on the parameters and their locations in the database, see "Engine Parameters".

See also

"Creating or Renaming Engine Profiles"

"Tuning Analytics"

"Enabling Engine Models Globally"

## **Creating or Renaming Engine Profiles**

#### **To create or rename an engine profile:**

1. Log onto the Business Modeler.
2. Click Parameters > System Parameters. The System Parameters dialog box appears.

3. Click the Engine tab.
4. Do one of the following:
  - To rename an existing profile, click the profile from the Engine Profiles list and then click Edit.
  - To create a new profile, click New.

5. Enter a (new) name for the profile.

See also

"Editing Engine Parameters"

"Tuning Analytics"

6. When creating a new Engine Profile, determine whether it is to be a batch or a simulation profile. A simulation profile must be attached to a parent batch forecast, because the simulation is stored in the `sim_val` column matching the batch parent profile.

#### **Example**

For example, a simulation with the batch parent profile ID of 3 is stored in the `sim_val_3` column. The internal profile ID can be found in `ENGINE_PROFILES` table.

7. If this is a simulation profile, select the Simulation Engine Profile check box.
8. If the new profile is a simulation profile, select the Parent batch Profile from the drop-down menu.
9. Click Ok.

## **Tuning Analytics**

For basic parameters related to the forecast tree, see "Specifying Additional Parameters". For information on all parameters (including default values), see "Engine Parameters".

## **Analytical Parameters**

The following parameters control analytics:

Parameter	Purpose
UseNonNegRegr	<p>Specifies whether to allow negative coefficients. Most of the models use this parameter.</p> <p>In cases with multiple, possibly co-varying causal factors, the Analytical Engine sometimes finds a solution that includes a large positive coefficient for one causal and a large negative coefficient for another causal factor, so that they nearly cancel one another.</p> <p>Mathematically, this solution may be good. But a negative coefficient means that the demand acts in the opposite sense to the causal factor; that is, demand drops when the causal factor increases. And a negative coefficient does not make sense in the vast majority of cases. This means that it is generally good practice to disable negative coefficients.</p>
ShapeSign	Specifies the signs for the shape causal factors when using them in non negative regression.
NumShapes	Specifies the maximum number of allowed shape causal factors for the engine to use for a given node in the forecast tree. Use an integer from 0 to 8, inclusive. This applies to activity shape modeling (rather than to promotional shape modeling).
CannibalizationIgnore	Controls whether the Analytical Engine will calculate switching effects (cannibalization). You can use this parameter to switch off that calculation in order to check that the Analytical Engine is calculating the basic lift appropriately.

## Parameters Related to Promotional Causal Factors (PE Mode Only)

The following parameters are related to promotional causal factors:

Parameter	Purpose
PromotionStartDate	Earliest date for which promotion data can be considered reliable.

Parameter	Purpose
ShiftDynPromoDate	<p>SQL expression that returns the number of days to <i>add</i> to the sales date for any given promotion; typically this is a negative number. If this expression is null, then the default promotion dates are used.</p> <p>Alternatively, to specify an overall shift in time for all promotions, set the ShiftPromoCausals parameter.</p>
See "Adjusting the Promotion Dates".	

## Parameters Related to Validation (PE Mode Only)

The Analytical Engine applies different forecasting models to each node of the forecast tree, calculates the uplift for each node, and uses that uplift to check whether the model is appropriate for that node. If not, the model is not used for the node.

The Analytical Engine can discard a model for a given node for either of two reasons:

- The model generated an uplift that was beyond the upper allowed bound, as specified by the UpperUpliftBound parameter.
- The model generated too many exceptional uplifts. An uplift is considered "exceptional" if it exceeds the lower bound specified by the LowerUpliftBound parameter. The AllowableExceptions parameter controls how many exceptional uplifts are permitted.

## Parameters Related to Output (PE Mode Only)

The following parameters control the output of Promotion Effectiveness forecast values:

Parameter	Purpose
NormalizeResults	Specifies whether to normalize the historical engine results so that the observed baseline values are preserved. If you normalize the engine results, note that the Analytical Engine writes these results to different fields in promotion_data than it does otherwise. See "Key Tables".

Parameter	Purpose
WriteMissingDatesUplift	Specifies whether to write uplifts for dates that are missing from sales_data. If you specify no, then the Analytical Engine writes uplifts only for dates that already have sales. However, the uplifts will not necessarily add up to the total uplift.
UpliftThresholdValue	Specifies a threshold for uplift values. If the Analytical Engine calculates uplift values below this threshold, those values are dropped rather than being written to the database.
UpliftThresholdMethod	Specifies whether the previous threshold is expressed as an absolute value or as a percentage of baseline.

See also

"Editing Engine Parameters"

"Creating or Renaming Engine Profiles"

## Tuning Performance

To improve the performance of the Analytical Engine, check the settings of the following parameters. To access these parameters in Business Modeler, click Parameters > System Parameters and then click the Database tab.

### Basic Engine Parameters for Performance

The following engine parameters are critical to good performance. Make sure they are set appropriately for your configuration.

Parameter	Purpose
min_fore_level	Minimum forecast level that the engine will forecast. For PE, this must be at or above the lowest promotional level (LPL). Make sure this is defined appropriately for your forecast tree.
start_new_run	Specifies whether to start a new Analytical Engine run or to perform an engine recovery. Use yes or prompt.



Parameter	Purpose
node_forecast_details	Specifies whether the Analytical Engine should write forecast data for each node (the NODE_FORECAST table), before splitting to lower levels. Writing to this table slows the engine, so you should switch off this option unless you have tested that the impact is acceptable.
WriteIntermediateResults	Specifies whether to enable the advanced analytics function, which is available only on the desktop. Make sure this option is off unless you have tested that it does not interfere unduly with performance.
BulkLoaderBlockSize	Specifies the minimum number of rows that Analytical Engine loads at one time, when writing to the database. The larger this is, the more quickly the data is loaded, but there is greater risk if the database connection is lost.  For a high-volume system, use 20,000.
BulkLoaderEnableRecovery	Specifies whether Oracle Bulk Loader should perform recovery after a lost database connection. For a high-volume system, use 0.

## Parameters That Can Speed Performance

The following parameters can help the Analytical Engine run more quickly by omitting processing steps. You should change these only if you are sure that doing so will not cause problems.

Parameter	Purpose
ForecastGenerationHorizon	Specifies what historical fit data the engine will write to the database. If this parameter is 0, the engine writes the forecast only. If this parameter is a positive integer N, the engine writes the last N historical fit values.

Parameter	Purpose
ResetForeVals	<p>Controls the method of clearing current forecast values for the forecast version currently being generated. If set to Yes (default), then all combinations with prediction status of 97, 98, or 99, fore = 2, will get null forecast values and active combinations will be overwritten by the new forecast. If set to No, then the existing forecast for inactive combinations will not be cleared. If set to All, then all combinations will have their forecast cleared regardless of prediction status. Note that setting this parameter to 'All' may substantially increase engine run time. If you want to reset the forecast outside the engine date range, set this option to All.</p> <p>The individual engines perform this function during the run. Each engine produces a list of the inactive nodes for the branch/simulation it is processing and adds special rows for the bulk loader. Procedures ProcessTempSaleTable and ProcessTempPromoTable perform the resetting. They update the configured data table and the PROMTION_DATA table. Engine parameter DBHintInitialForeClean applies only to the functionality of Engine Manager parameter DeleteIsSelfRows.</p>
RunInsertUnits	<p>Specifies whether the Analytical Engine calls the INSERT_UNITS procedure at the start of an engine run. This procedure makes sure the engine has rows to write into when generating the forecast.</p> <p>For information on all procedures, see "Database Procedures".</p>
BatchRunMode	<p><b>Applies to PE mode, and applies to both batch run and simulation run.</b> Specifies the kind of forecasting to do:</p> <ul style="list-style-type: none"> <li>• 0=run the forecast against only the learning (estimation)</li> <li>• 1=run the promotion forecast (the normal setting)</li> <li>• 2=perform an estimation and promotion forecast run (fast simulation; this option uses previously cached data)</li> </ul> <p>For options 0 and 2, the Analytical Engine performs fewer scans. (For details on the engine flow, see "Promotion Effectiveness Engine Phases".)</p>

Parameter	Purpose
align_sales_data_levels_in_loading	<p>Specifies whether to adjust the sales_data table for direct use by the engine (instead of the sales_data_engine table).</p> <ul style="list-style-type: none"> <li>0=no (do not adjust the sales_data table for direct use by the engine)</li> <li>1=yes (adjust the sales_data table)</li> </ul> <p>For information on this parameter, see "Non-Engine Parameters".</p>
start_date	<p>Beginning of historical data used by the engine. Used together with parameter HistoryLength. If left at default 01/01/1995, may require the engine to find first period of real history in historical demand. For larger environments this can add significant time to the engine run. It is strongly recommended this parameter be reset to beginning actual date where history begins.</p>

The engine divider uses Fast Divider functionality. The engine uses the ENGINE\_BRANCH\_LIST table to determine the actual branch, not the BRANCH\_ID column of the configured combination table. Each time the engine processes a branch, it updates the BRANCH\_ID column with the actual allocation.

## Database Partitioning for the Engine

You can partition the database so that the Analytical Engine can access data more rapidly. Specifically, you can place different parts of the sales\_data, mdp\_matrix, and promotion\_data tables on different partitions, so that each partition corresponds to a potentially different item and/or location.

The overall process is as follows:

1. Create the partitions and move rows to them as needed. This is beyond the scope of this documentation.
2. To partition only by item, choose a database column that you can use to subdivide the records by item. This column must exist in the sales\_data, mdp\_matrix, and (in the case of Promotion Effectiveness) promotion\_data tables and must have the same name in each of these tables.

For example, it might be suitable to partition by brand. The brand information is available in mdp\_matrix as (for example) the t\_ep\_p2a\_ep\_id field. You would have to replicate this column to the sales\_data and promotion\_data tables as well, perhaps by a database trigger.

Similarly, to partition only by location, choose a database column that you can use to subdivide the records by location.

To partition by item and by location, choose a database column that you can use to subdivide the records by item and another column that subdivides them by location.

3. Set the following parameters so that the Analytical Engine can find the partition on which any combination resides:

Parameter	Purpose
PartitionColumnItem	Specifies the name of the column that partitions the data by item.
PartitionColumnLoc	Specifies the name of the column that partitions the data by location.

## Other Database Considerations

Pay attention to the indexes of sales\_data and mdp\_matrix tables.

Also, for Oracle databases, Demantra writes to multiple tablespaces, as specified during installation. The tablespace assignments are controlled by parameters, which you can edit through the Business Modeler. Make sure that these parameters refer to tablespaces within the appropriate database user, and make sure each has enough storage.

**Note:** Oracle recommends that you use the standard names for these tablespaces, as documented in the Oracle Demantra Installation Guide. Then it is easier for you to share your database with Oracle Support Services in case of problems.

Additional parameters control the default initial sizes and how much storage is added.

Parameter	Description
initial_param	Default initial size of system tablespaces.
next_param	Incremental amount of storage that is added to a tablespace when more space is needed.
tablespace	Tablespace used for the sales table.

Parameter	Description
indexspace	Database index space that stores the forecast table indexes, as specified during installation.
simulationspace	Tablespace used for simulation data.
simulationindexspace	Tablespace used for simulation index data.
sales_data_engine_index_space	Tablespace used for the index of sales_data_engine.
sales_data_engine_space	Tablespace used for sales_data_engine table.
*For information on these parameters, see "Non-Engine Parameters".	

## Reconfiguring the sales\_data\_engine Table

The Analytical Engine creates and uses a table (or view) called sales\_data\_engine. You can control how the Analytical Engine does this, in order to improve performance.

- You can adjust the sales\_data table for direct use by the Analytical Engine, so that the sales\_data\_engine table is not needed.
- Normally, the Analytical Engine internally creates the sales\_data\_engine table for its own use, and creating this table can be time-consuming. You can speed up the engine by configuring it to use the sales\_data table instead of the sales\_data\_engine table.
- Normally, when the Analytical Engine runs, it joins sales\_data\_engine (or its synonym) with the mdp\_matrix table. This is not always necessary, and you can prevent this join to speed up the Analytical Engine.

The following table lists the key parameters and some typical settings:

Parameter	Description	Normal batch run	Normal simulati on	Faster engine run*	Fast simulatio n*
align_sales_data_levels_in_loading**	<p>Specifies whether to adjust the sales_data table for direct use by the engine (instead of the sales_data_engine table).</p> <ul style="list-style-type: none"> <li>• 0=no (do not adjust sales_data)</li> <li>• 1=yes</li> </ul>	0	0	1	1
SdeCreateSwitch	<p>Specifies the type of logic to use in order to create the sales_data_engine table.</p> <ul style="list-style-type: none"> <li>• 0=Use internal engine logic</li> <li>• 1=Use external logic, as specified by stored @ procedures, create_process_template, create_object and drop_object @ procedures. These procedures may be modified by consultants.</li> </ul>	0	0	1	1

Parameter	Description	Normal batch run	Normal simulation	Faster engine run*	Fast simulation*
SdeAnalyzeSwitch	<p>Specifies the type of logic to use in order to analyze the sales_data_engine table.</p> <ul style="list-style-type: none"> <li>0=no. The engine assumes analyze is already performed and creates indexes as part of external procedures.</li> <li>1=yes. The engine analyzes and creates indexes. This is the default value.</li> </ul>	1	1	0	0
SdeCreateJoin	<p>Specifies whether the Analytical Engine should join sales_data_engine (or its synonym) and mdp_matrix during its run.</p> <ul style="list-style-type: none"> <li>0=no (do not join these tables)</li> <li>1=yes (join these tables; this is the default)</li> </ul>	0	0	0	1

\*See "Additional Steps". Also note that fast simulation forecasts future uplift only. \*\*For information on this parameter, see "Non-Engine Parameters".

### Additional Steps:

1. Configure the forecast tree as you normally would. See "Configuring the Forecast Tree".
2. In the database, create a synonym for sales\_data. The name of synonym should be sales\_data\_engine or whatever synonym you plan to use.

3. Rewrite the following database procedures as needed:
  - create\_process\_temp\_table
  - create\_object
  - drop\_object
4. Consult Demantra for assistance.
5. Test that you have configured the engine correctly.
  1. Add new records to sales\_data, in any of the following ways: by loading via the Data Model Wizard, by running integration, or by chaining.
  2. Run the engine.
  3. Check the sales\_data table and make sure of the following:
    - This table should have a column for every level in the forecast tree,
    - This table should have a column named do\_aggri.
    - This table should have should include non-null data in these columns for at least some of the records.

## Enabling Engine Models Globally

Demantra provides a set of theoretical engine models that the Analytical Engine uses when it creates a forecast. Usually you do not make changes, but you can specify which models to use, as well as set basic parameters for each model.

**Caution:** Only advanced users should make these changes.

When the Analytical Engine runs, it may use a subset of these models on any particular combination. The engine tests each model for applicability; see "The Forecasting Process".

**Note:** Optimization can only use linear generalized coefficients

### To enable models for the Analytical Engine to use:

1. Log onto the Business Modeler.
2. Click Engine > Model Library.



The following dialog box appears.

Bayesian Model Manager

Engine Profile: Base

Model Name	Active	Minimum/maximum history periods to apply model	
		Min Len	Max Len
ARLogistic	<input type="checkbox"/>	52	99999
Auto and Linear Regression (ARX)	<input checked="" type="checkbox"/>	4	99999
Combined Transformation Model (elog)	<input checked="" type="checkbox"/>	52	99999
Croston For Intermittent	<input checked="" type="checkbox"/>	4	99999
DailyMultiplicativeRegression	<input type="checkbox"/>	52	99999
Holt	<input checked="" type="checkbox"/>	2	51
Integrated Auto and Linear Regression (ARIX)	<input type="checkbox"/>	4	99999
Integrated Causal Exponential Model (BIVint)	<input checked="" type="checkbox"/>	52	99999
Logistic	<input type="checkbox"/>	52	99999
Modified Ridge Regression	<input type="checkbox"/>	52	99999
Multiplicative Monte Carlo Regression (CMReg)	<input checked="" type="checkbox"/>	52	99999
Multiplicative Monte Carlo Regression FOR Intermittent (ICMRegr)	<input checked="" type="checkbox"/>	52	99999
Regression	<input checked="" type="checkbox"/>	52	99999
Regression FOR Intermittent	<input checked="" type="checkbox"/>	52	99999

Restore Defaults Save Close

3. Select the batch engine profile to be associated with the model library configurations.
4. For each model, do the following:
  - To enable the Analytical Engine to use this model, make sure the Active check box is checked. For details on these models, see "Theoretical Engine Models". Note that not all models are supported with any given Analytical Engine.
  - The other two settings control the minimum and maximum number of non zero observations that a combination must have in order for the Analytical Engine to consider using this model for this combination. To specify these values, type integers into the Min Len and Max Len fields.

**Note:** Min Len must be equal to or greater than the number of causal factors in the forecast, except for the HOLT and FCROST models, which do not use causal factors.

5. Click Save and then click Close.

## Configuring the Engine Mode

Oracle provides two different modes of the Analytical Engine:

- The DP mode is for use with Demand Planner or other planning products.
- The PE mode is for use with Promotion Effectiveness.

### To specify the engine mode

The RUNMODE parameter specifies the mode of the Analytical Engine to use:

- Use 1 to specify the PE mode.
- Use 0 to specify the DP mode.

If you use this setting, also be sure that you have defined the forecast tree appropriately. In particular, make sure that the LPL (PROMO\_AGGG\_LEVEL) is the same as the minimum forecast level. To set this, use the forecast tree editor in the Business Modeler.

See also

"Troubleshooting"

## Advanced Analytics (Nodal Tuning)

Normally, the Analytical Engine uses the same options for every node in the forecast tree, but you can make certain adjustments for individual nodes, if necessary. This task is recommended only for advanced users in conjunction with Oracle Support.

Of the models you specify for a given node, when the Analytical Engine runs, it may use a subset of these models, as described in "The Forecasting Process". The Analytical Engine indicates (in the models column of mdp\_matrix) the models that it used.

### To enable advanced analytics :

1. Set the usemodelspernode and UseParamsPerNode parameters to yes.
2. Then for each node of the forecast tree, you can specify engine models and engine parameters for different nodes in the forecast tree. To do so, you use the Analytics window.

---

## Using the Engine Administrator and Running the Engine

Before you run the Analytical Engine for the first time, it is useful to ensure that you have configured it correctly. This chapter describes how to administer the analytical engine.

This chapter covers the following topics:

- Before Running the Analytical Engine
- General Notes about Running the Analytical Engine
- Registering the Analytical Engine
- Starting the Engine Administrator
- Configuring Engine Settings
- Running the Engine from the Engine Administrator
- Running the Engine from the Start Menu
- Running the Engine from the Command Line
- Troubleshooting
- Viewing the Engine Log
- Examining Engine Results
- Running the Engine in Recovery Mode
- Stopping the Engine

### Before Running the Analytical Engine

Before you run the Analytical Engine for the first time, it is useful to ensure that you have configured it correctly.

- Make sure that you have installed the correct version of the Analytical Engine and

that you have set the RUNMODE parameter correctly; see "Configuring the Engine Mode".

- Make sure the engine is registered on all the machines where you want to use it. See "Registering the Analytical Engine."
- Make sure that you have enough (and not too many) observations for every node in your forecast tree, as needed by the engine models you plan to use.

If a node is left with no suitable model, the Analytical Engine will not forecast on that node. Instead it will forecast at a higher level, if possible.

- Various configurable fields contain parts of SQL queries used by the engine during run and may fail the engine if configured incorrectly. Common reasons for failures are misspellings, references to non existent columns, or using functions or syntax not compatible with the database server.

To check all your engine-related SQL, check the following tables:

- In the Init\_Params\_\* tables, check the parameters quantity\_form and UpTime.  
The default expression for quantity\_form transforms negative values to zero. This is considered best practice, and should be modified only if there is a direct need for the analytical engine to see negative demand and for negative proportions to be calculated. Care must be taken when enabling negative proportions, as allocation using negative proportions may result in violation of business rules
- In the causal\_factors table, the Local\_Funct column uses SQL.
- (For PE mode) In the m3\_causal\_factors table, many settings here use SQL.
- Make sure that the database is configured correctly, specifically the table extents. Also, if you have loaded the Demantra schema from a dump file, make sure that the current database contains table spaces with the same names as in the original database.
- Determine the best number of branches for the Analytical Engine to use for the current forecast tree. See "Viewing Branch Information".

## General Notes about Running the Analytical Engine

- The first engine run takes longer than later runs. This is because the Analytical Engine must set up internal tables on its initial run. You can reduce the length of time of the first engine run; see "Reconfiguring the sales\_data\_engine Table".
- You cannot run the Analytical Engine in batch mode unless the Business Logic

Engine is closed and no simulation is running.

- On older Windows NT versions, you may get errors on running the Analytical Engine (2K DLL missing). To solve this, run the file mdac\_typ.exe before running the Analytical Engine. (This executable is located in the bin directory under the Analytical Engine directory)

Also see

"Introduction to the Analytical Engine"

## Registering the Analytical Engine

The installer registers the Analytical Engine for you, but in case of problems, you can register the engine manually. To do so, run the batch file Demantra\_root/Demand Planner/Analytical Engines/bin\RegEngine.bat.

If your system includes the Distributed Engine, you can run the engine on multiple machines. You do not need to install it on every machine; you need only to register it on them. These machines must also be running the appropriate database client software, so that they can communicate with the Demantra database.

For information on choosing the machines to run the engine, see "Choosing Machines to Run the Engine".

## Starting the Engine Administrator

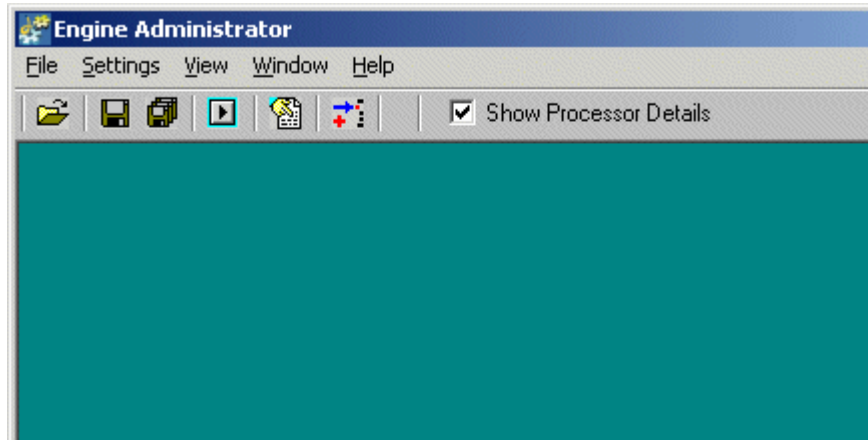
The Engine Administrator is a desktop-based user interface that lets you perform the following general tasks:

- Specify settings for the Analytical Engine. You save the settings in an XML file for convenience. You can open and use settings files that you have previously saved.
- Run the Analytical Engine itself.
- View the engine log.

### To start the Engine Administrator:

1. On the Start menu, click Programs.
2. Click Demantra > Demantra Spectrum release > Engine Administrator.

The Engine Administrator screen appears.



3. Here you can do the following:
  - Specify settings for the Analytical Engine. You save the settings in an XML file for convenience.
  - Run the Analytical Engine itself.
  - View the engine log.

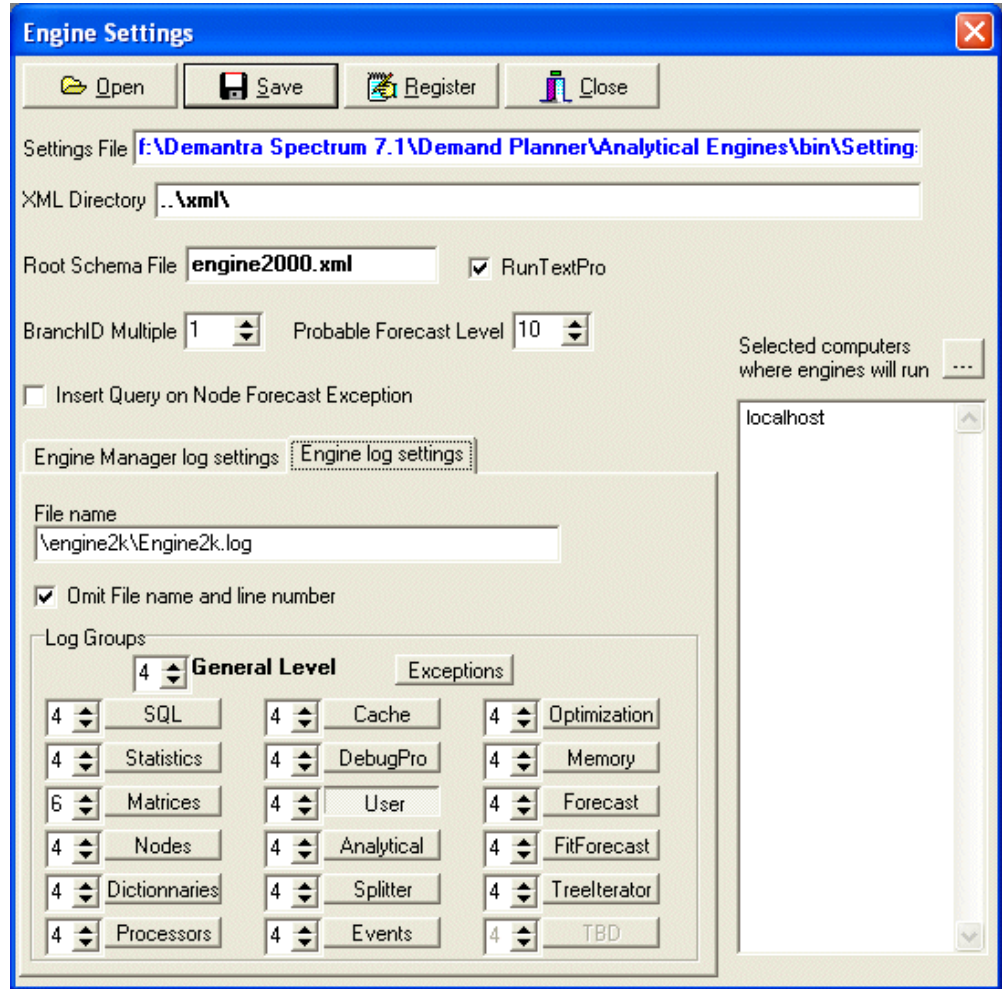
## Configuring Engine Settings

Engine configuration settings are edited in the Engine Settings window and saved in the file named settings.xml. When the engine starts, it reads the settings from this file.

### To open the Engine Settings window:

1. Start the Engine Administrator.
2. Click Settings > Configure Engine Settings. Or click the Configure Engine Settings button.

The Engine Settings window appears.



### To load settings:

1. Click Open.
2. Select settings.xml from the bin directory of the Analytical Engine.  
The Settings File field displays the location of settings.xml.
3. Complete the fields as needed; see "Engine Settings".
4. Optionally choose different machine(s) to run the engine; see "Choosing Machines to Run the Engine".
5. To save your settings, click Save.
6. To register your settings, click Register.

## Engine Settings:

You can configure the following settings.

Setting	Meaning
Run TextPro	If this option is checked then a database preprocessing step will take place where SQL statements (sql_def) will be prepared and parameters will be initialized.
BranchID Multiple	<p>Specifies how to divide the forecast tree, as described in "Engine Components" and "Viewing Branch Information".</p> <p>The Engine Manager divides the forecast tree into smaller tasks, by updating one column in the mdp_matrix table that links each node with a task ID. The number of the tasks is the number of engine servers that were initialized successfully, multiplied by the number in the BranchID Multiple field.</p>
Selected computers where engines will run	<p>The Engine Manager tries to create and initialize all the engine servers specified in this list.</p> <p>You can choose one or more machines on which the Analytical Engine has been installed. These machines must also be running the appropriate database client software, so that they can communicate with the Demantra database.</p> <p>In order to run the Analytical Engine on multiple machines, your system must include the Distributed Engine.</p>
Probable Forecast Level	Insert Query on Node Forecast Exception

### Engine Manager Log Settings

File name	Path and filename of the log file that will record errors from the Engine Manager.
Output Target	<p>Select either stdout or File.</p> <p>If you choose stdout, the output is sent to the Log File window. In this case, you can still save the log to a file, from the Run Engine window.</p>
Log Groups	Specifies the types of errors to log.

### Engine Log Settings



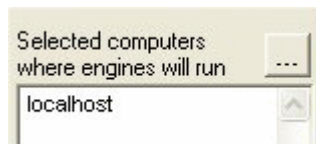
Setting	Meaning
File name	Path and filename of the log file that will record errors from the engine itself.
Log Groups	Specifies the types of errors to log.

### Choosing Machines to Run the Engine:

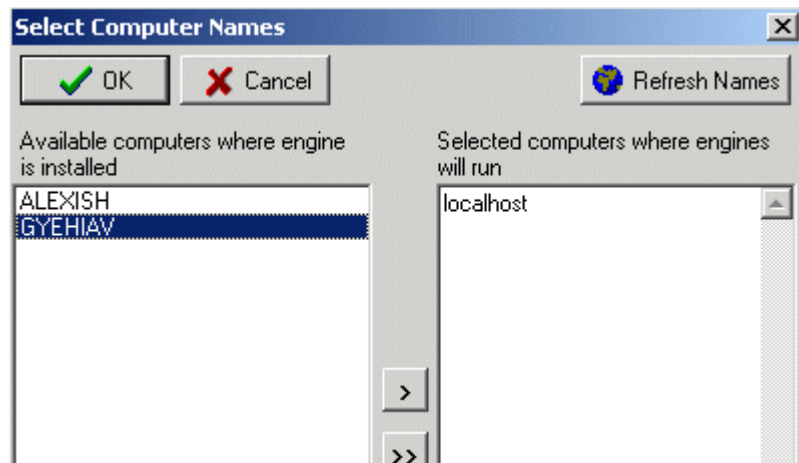
If your system includes the Distributed Engine, you can run the engine on multiple machines; see "Registering the Analytical Engine". Otherwise, Oracle Demantra runs the Analytical Engine on only one machine.

#### To choose the machines to run the engine:

1. Click the button next to the Selected computers field, shown here.



The Engine Administrator displays the following screen:



The left list shows the machines on which the Analytical Engine has been installed. The right list shows the machines where the Analytical Engine will run.

2. Click the arrow buttons to move machine names to and from the right list.
3. Click OK.

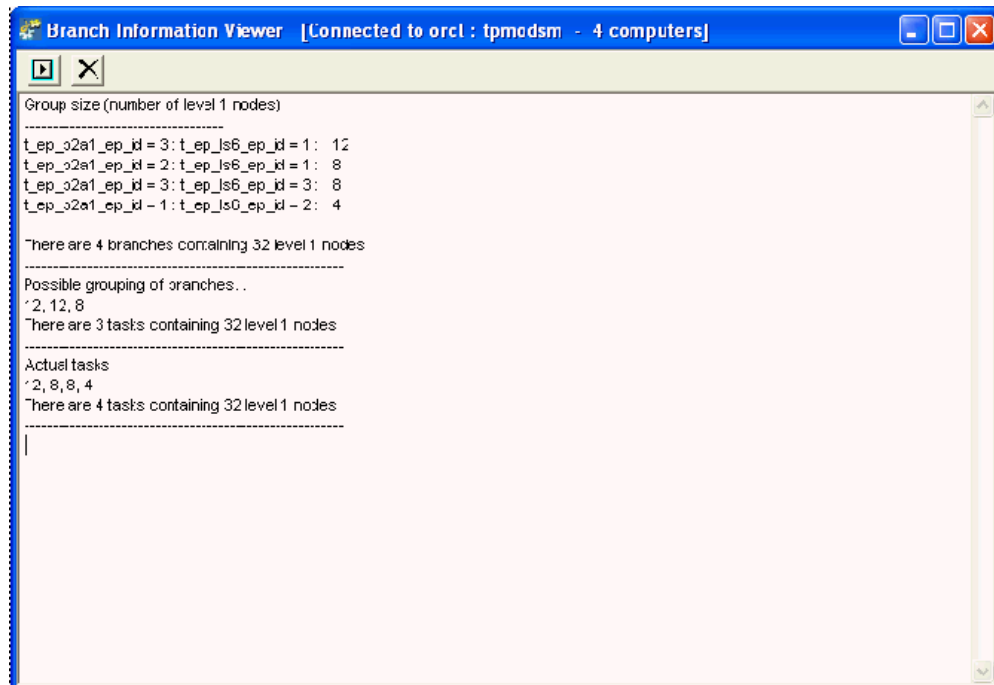
### Viewing Branch Information:

Internally, the Analytical Engine divides up each forecast tree into multiple branches and sends each branch to a different engine server. On one hand, it is useful to declare many branches, so that each engine will have less work. On the other hand, it is useless to declare more tasks than the forecast tree can possibly include. Therefore it is useful to understand what the branches of a given forecast tree would be like.

### To view branch information for a given forecast tree:

1. Click View > Branch Information.

The Branch Information Viewer window appears.



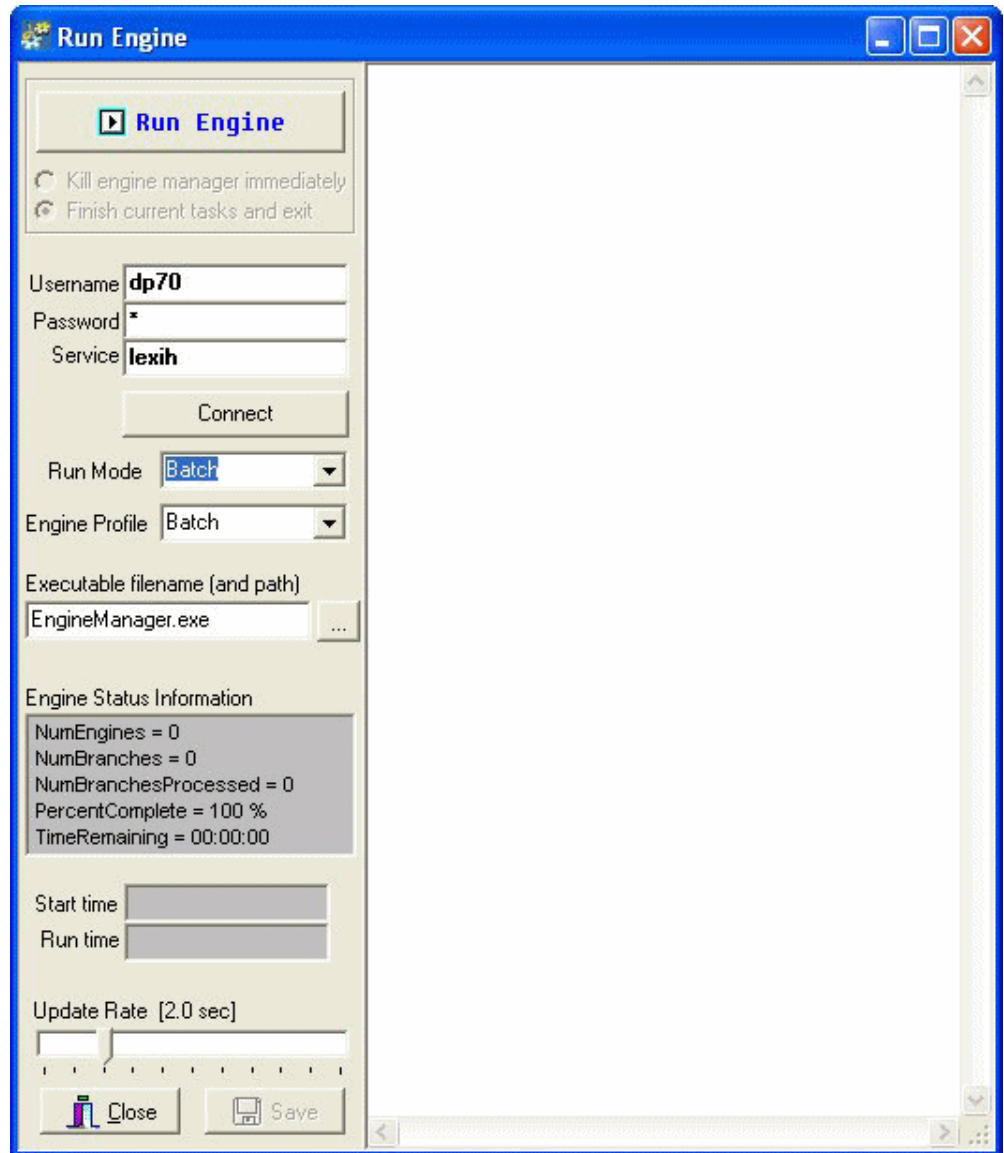
## Running the Engine from the Engine Administrator

Before running the Analytical Engine, make sure that there are no old engine processes running in the chosen host machines.

### To run the engine:

1. Start the Engine Administrator.
2. Click File > Run Engine. Or click the Run Engine button.

The Run Engine window appears.



3. Complete the fields Username, Password, and Service.
4. Click Connect.
5. Change the Run Mode to Batch or Simulation.
6. For Engine Profile, choose an engine profile, which specifies the parameter settings to use during this engine run.
7. Enter EngineManager.exe in the Executable Filename (and path) field. This executable resides within the bin directory of the Analytical Engine.

8. Click Run Engine.

The Engine Status Information area displays the following:

- Number of engines running
- Number of branches (you use multiple branches when you have a large database)
- Number of branches processed
- Percent complete
- Start time of the engine run, time remaining (if any), and end time (if finished)

The white text area in the right side of the Run Engine window displays the log.

9. To adjust the rate at which the log area is filled with data, drag the Update Rate slide to the left (slower) or to the right (faster).
10. The log shown in the text area can be saved to a file by clicking Save to file.

### To stop the engine run:

1. Under Terminate Engine Manager, select one of the following:
  - Finish current tasks and exit. This clears the queue of branch IDs (the messages sent by the Engine Manager to the engine servers). This allows the engine servers to complete their current processes. When they do not receive any further branch IDs, they shut down in an organized fashion. This is the recommended shutdown method in most circumstances. It is possible to continue processing at a later time.
  - Kill Engine Manager Immediately. This stops the Engine Manager immediately. The engine servers will continue for a while but when they notice that the Engine Manager has stopped, they will cease operation. Use this method when a quick shutdown is required, such as during debugging.

**Note:** Do not restart the Engine Manager if one or more engine servers are still running. Wait until they stop or close them using Windows Task Manager.

2. Click Stop.

See also

"Running the Engine from the Start Menu"

"Running the Engine from the Command Line"

## Running the Engine from the Start Menu

### To run the Analytical Engine from the Start menu:

1. To run the engine in batch mode: on the Start menu, click Programs. Then click Demantra > Demantra Spectrum release > Analytical Engine.
2. To run the engine in simulation mode: on the Start menu, click Programs. Then click Demantra > Demantra Spectrum release > Simulation Engine.

See also

"Running the Engine from the Command Line"

"Running the Engine from the Engine Administrator"

## Running the Engine from the Command Line

### To run Analytical Engine from the DOS window:

It is useful to be able to run the Analytical Engine from the command line, which you may want to do from within a workflow, for example.

1. In a DOS window, change to the following directory: Demantra\_root/Demand Planner/Analytical Engines/bin
2. In this directory, enter one of the following commands:
  - To run the Analytical Engine with a specific engine profile: EngineManager.exe *mode profile*
  - To run the Analytical Engine on Oracle with a different database: EngineManager.exe *TnsName UserID password mode profile*
3. Notice that the parameters are order-dependent; this means that you must use one of the syntax variants given here, with all the parameters as shown for that variant. These parameters have the following meanings:

---

TnsName	The server TnsName
User ID	The DB identifier.
Password	The DB password

---

---

Mode	Integer that indicates the engine profile to run: <ul style="list-style-type: none"> <li>• 1=Batch mode</li> <li>• 2=Single Node</li> <li>• 3=Full Divider</li> <li>• 99=Simulation</li> </ul>
Profile ID	Engine Profile ID to use (from the ENGINE_PROFILES table)

---

**Tip:** You can create a Collaborator Workbench menu item that runs these commands; see "Configuring Links in Collaborator Workbench".

See also

"Running the Engine from the Start Menu"

"Running the Engine from the Engine Administrator"

## Troubleshooting

This section contains tips that address specific error conditions that you could encounter:

- If the Analytical Engine fails to run, see the list in "Before Running the Analytical Engine".
- If the engine failed while running an SQL statement, check the following logs:
  - manager.log
  - engine2k.log

Find the offending SQL and try running it within a third-party database tool to identify the problem.

If the engine iterator failed, resulting in the error "node not found in map," that indicates a problem in the mdp\_matrix table. Usually, this means that you need to set the align\_sales\_data\_levels\_in\_loading parameter to true and then run the MDP\_ADD procedure. (For information on this parameter, see "Non-Engine Parameters".)

- If the Analytical Engine run does not finish and gives a message saying that it is stacked at some node or that it "does not have a usable number of observations," this means that the mdp\_matrix table is not in a good state. To correct the problem, run the MDP\_ADD procedure.
- If the Engine Administrator displays the message "Can not initialize caches," that may mean that your database is too large for the given number of branches. Reconfigure the engine to run on more branches and try running it again. See "Running the Engine from the Engine Administrator".
- If the Analytical Engine fails or generates errors when processing large amounts of data, use the MaxEngMemory parameter. As each engine task is completed, the Engine Manager evaluates the amount of memory currently used by all analytical engine processes on the machine. If this amount (expressed in megabytes) exceeds the value of MaxEngMemory, the engine instance is stopped and a new instance is initiated. For example, if this parameter is set to 250, a new engine instance is initiated at the end of each task that consumes more than 250 MB of memory.
- **PE only:** If you receive a message like "ERROR Node not found in map", that means that something is wrong with synchronization between sales\_data and mdp\_matrix. To correct the problem, truncate mdp\_matrix and run the MDP\_ADD procedure.
- **Oracle only:** If the Analytical Engine takes an unreasonably long amount of time to create the sales\_data\_engine or the promotion\_data table, make sure that you have done an analyze table on these tables.
- **Oracle only:** If you receive a message such as "Description: ORA-00959: tablespace 'TS\_SALES\_DATA' does not exist," that typically means the dump file you installed refers to different table spaces than you have in the current database. Reassign the Demantra table spaces by changing the parameters that control them:
  - indexspace
  - sales\_data\_engine\_index\_space
  - sales\_data\_engine\_space
  - simulationindexspace
  - simulationspace
  - tablespace

For information on these parameters, see "Non-Engine Parameters".

## Validating Input Parameters

Validating engine and model input parameters is used to identify the source of errors caused by configuration issues and errors. This streamlines and shortens the troubleshooting process and reduces the need for support.

- **Parameters:**
  1. The Analytical Engine loads the 'Parameters' data from the PARAMETERS table.
  2. The engine then loads the 'Parameters' data from 'Parameters Daily.xml', 'Parameters Monthly.xml', or 'Parameters Weekly.xml' depending on 'timeunit'.
- **InitParams:**
  1. The Analytical Engine loads the 'InitParams' data from INIT\_PARAM\_0 table.
  2. Then the engine loads the 'InitParameters' data from 'Init Params 0 Daily.xml', 'Init Params 0 Monthly.xml', or 'Init Params 0 Weekly.xml' values.

The Analytical Engine loops through parameters from xml, validates them against the database parameters, fixes the collected parameters, or adds the missing parameters in the database.

The validation rules are configurable. If they belong to the current run, you can specify the parameter group, and the restrictions by which the parameters are compared.

If any of the input parameters fails the validation, the system replaces the erroneous parameters with the default value if the restriction does not contain '?'. Otherwise, the system simply generates a warning message to inform the user of the erroneous input parameter.

**Note:** Demantra supports only the following type "double" validations for parameters:

- 1 - All the groups - always validate
- 2 - DP batch
- 3 - PE batch
- 4 - DP simulation
- 5 - PE simulation



### Example 1

```
<Entry>
 <Key argument="AllowNegative"/>
 <Value type="double" argument="0"/>
 <Validate group="1" restrict="=1,=0"/>
</Entry>
```

The above-mentioned validation means "Allow Negative" parameter of type "double" with default value "0". The validation belongs to group "1" thereby run during all engine runs and its value can either be "1" or "0"

### Example 2

```
<Entry>
 <Key argument="lead"/>
 <Value type="double" argument="52"/>
 <Validate group="3,4" restrict=">0,?<=100"/>
</Entry>
```

The above-mentioned validation means "lead" parameter of type "double" with default value "52" belongs to group 3 and 4, for which the value must be greater than "0" and less or equal to "100". The "?" means that it is not mandatory to fix the parameter if it is greater than "100". If under 0 the parameter would warn the user and replace the value with 52 while if greater than 100 a warning will be generated but not override would occur.

### Example 3

```
<Entry>
 <Key argument="PROMO_AGGR_LEVEL"/>
 <Value type="double"/>
 <Validate group="3,5" restrict=""/>
</Entry>
```

The above-mentioned validation means "PROMO\_AGGR\_LEVEL" parameter of type double with no default parameter belongs to group 3 and 4, and the validation is done through custom function. The engine will quit running if the validation fails

To add the custom function to the process, you should add your function to *..\Common\Util\Validation Functions.cpp*.

Then add the name and address of this function to the array of function pointers, so that the application can execute this function dynamically:

```
m_mPoint2Function["PROMO_AGGR_LEVEL"]=
PromoAggrLevel;m_mPoint2Function["PROMO_AGGR_LEVEL"]= PromoAggrLevel;
```

## Viewing the Engine Log

The log viewer helps you debug the engine run. The log for the Analytical Engine appears in a text file in the directory Demantra\_root/Demand Planner/Analytical Engines/bin.

### To open the log file viewer:

1. Start the Engine Administrator.

2. Click the View log file button.

#### **To view a log file as it is:**

1. Click the Open with Tree View button.
2. When the Processors check box is chosen in Log Groups, you can view the log file with processors tree assistance. If you click on a processor in the right side of the Log File window, you are brought to the corresponding line in the log file.

## **Examining Engine Results**

This section contains assorted tips on viewing and understanding the engine results from a more technical point of view.

### **Seeing What Level the Forecasting Was Done**

When forecasting, the Analytical Engine writes information to the mdp\_matrix table to indicate where it performs the forecast. For each combination, it writes this information to the following columns:

- level\_id is the strategy in the forecast tree where the forecast for this combination was generated. Strategy includes data aggregation level referred to as detail node and possible pooling of detail nodes into longer time series referred to as range.
- item\_node is the item member in that detail level.
- loc\_node is the location member in that detail level.

### **Seeing if Any Nodes Were Not Forecasted**

To see if any nodes failed to receive a forecast, run the following SQL:

```
SELECT level_id, COUNT(*) FROM MDP_MATRIX WHERE prediction_status=1
GROUP BY level_id
```

Explanation: At the start of the run, the engine iterates through all forecastable nodes and sets their level\_id to the fictive level. As it forecasts the nodes, it resets the level\_id back to normal. At the end of the run, if you have nodes with a level\_id = fictive level, those nodes did not get a forecast.

Possible reasons:

- The forecast tree might not be well formed.
- There might not be any models that can work on at the Top Forecast Level.
- There might be nodes that do not have the correct number of observations for the

models.

- Naive forecasting might be off; see "Forecast Failure".

## Writing Intermediate Results

In a batch run, the Analytical Engine can write intermediate results to the database, to help you determine the source of a problem. To enable this, set the `WriteIntermediateResults` parameter to yes (1) and then run the engine. When this flag is enabled, the Analytical Engine writes intermediate results to the `INTERM_RESULTS` table.

**Warning:** Use this feature only with help of Oracle consulting. This feature may greatly inflate the engine run time.

You can also configure the engine to write forecast data for each node, before splitting to lower levels. This data is written to the `NODE_FORECAST` table, which includes information on how each model was used for that node. To enable this, set the `node_forecast_details` parameter to forecast is written with model details (1) before running the engine.

To edit these parameters, use the Business Modeler.

## Running the Engine in Recovery Mode

Internally, the Analytical Engine records information to indicate its current processing stage. As a result, if the previous engine run did not complete, you can run recovery, and the Analytical Engine will continue from where it was interrupted.

### To run the engine in recovery mode:

1. In the Business Modeler, set the `start_new_run` parameter to either No or Prompt.
2. Start the Analytical Engine as described in "Running the Engine from the Engine Administrator" or "Running the Engine from the Start Menu".

## Stopping the Engine

Normally the Analytical Engine stops on its own when it has completed processing.

If you are automating processes, you may want to make sure that the Analytical Engine is not running, before starting it again.

In the directory `Demantra_root/Demand Planner/Analytical Engines/bin`, there is a batch file that you can use to kill the engine manager (and therefore the engine as well). This is called `KillEngine.bat`.

**Tip:** After killing the Analytical Engine, it is advisable to wait about 10 seconds before starting a new one.

---

## Engine Details

This chapter provides details on the Analytical Engine, for the benefit of advanced users.

This chapter covers the following topics:

- Preparing the Database
- Promotion Effectiveness Engine Phases
- The Forecasting Process
- Comparison of Batch and Simulation Modes
- Engine Components and High-Level Flow
- Details of the Distributed Engine

### Preparing the Database

At the start of an engine run, the Analytical Engine prepares the database, to make sure that the appropriate tables contain rows into which the Analytical Engine can write results. To do so, the Analytical Engine calls the `INSERT_UNITS` procedure, which is controlled by the `RunInsertUnits` parameter and can do several things, depending on the value of that parameter:

- Makes sure the engine has rows to write into when generating the forecast. In particular, for *all non-dead* combinations, this procedure does the following:
  1. Checks to see if the database contains records for this combination for all dates in the span of time from `max_sales_date` to `max_sales_date + lead`.
  2. For any dates when the combination does not have records, this procedure inserts records with zero sales, into which the Analytical Engine can then write the forecast.
  3. Records with dates in the past are ignored.

- Runs the EXECUTE\_PROFILES procedure, which executes the active rolling data profiles.

## Additional Details for PE Mode

For Promotion Effectiveness, if the DeleteIsSelfRows parameter is 1, the Analytical Engine also performs a cleaning step. In this step, it removes unneeded rows from the promotion\_data, which otherwise can grow to an unreasonable size. (If this table contained a row for every item, every location, every promotion, and every date, performance would suffer.) Specifically, the Analytical Engine deletes rows that have is\_self is 0 and that have zero lift values (details below).

In some cases, users may enter override values, and the Analytical Engine should not delete rows that contain those values. The DeleteIsSelfCondition parameter specifies other fields in promotion\_data that should be checked before this cleaning occurs. The Analytical Engine deletes only the rows that have is\_self is 0 and zero values for all of the following fields: uplift, pre and post-effect, switching effects, and the field or fields specified by DeleteIsSelfCondition.

## Promotion Effectiveness Engine Phases

In PE mode, the Analytical Engine runs in multiple phases (the last of which actually generates the forecast), and it caches data at critical points, for better performance. The earlier phases map the promotion attributes internally into causal factors, so that they can be used in the same way as the other causal factors.

This section describes these engine phases.

## Global Preparations

This phase uses the following settings from the Promotional Causal Factor screen; see "Configuring Promotional Causal Factors":

---

Column Name Expression	An expression that retrieves and aggregates the promotion attribute.
Filter	An aggregating expression that returns the true or false value, filtering the source data of this promotional causal factor. You can use this expression to create multiple causal factors from a single set of source data.

---

When the Analytical Engine runs, the first step is to perform the following global preparations:

- Create the promotion\_data\_engine table, which is analogous to the

sales\_data\_engine used in demand planning.

- In memory, aggregate the promotion attribute data to the lowest promotional level, as defined in the forecast tree. Here the Analytical Engine uses the Column Name Expression option.
- Apply filters as defined by the Filter option.

## Initial Phase

This phase uses the following settings from the Promotional Causal Factor screen; see "Configuring Promotional Causal Factors":

---

Transpose by Column	Optionally converts a qualitative promotion attribute into multiple unrelated causal factors.
Merge Function	Specifies how Demantra should internally merge promotions of the same kind that apply to the same item, location, and time.
Aggregation Function	Specifies how Demantra should internally aggregate the promotional causal factor above the LPL.

---

After making global preparations, the Analytical Engine performs the first scan of the forecast tree, as follows:

1. Read from the database and load the forecast tree into memory.
2. Calculate the absolute and relative addressing within each influence group, for internal use. In this step, the Analytical Engine uses the COMPETITION\_ITEM and COMPETITION\_LOCATION parameter settings.
3. Creating promotional causal factors at the LPL. In this step, the engine does the following:
  - Transpose the promotion attributes, according to the Transpose by Column option.
  - Merge the attributes across promotions, according to the Merge Function option.
  - Cache the data for nodes of this level.
4. Creating promotional causal factors at the IGL. In this step, the engine does the following:

- Aggregate the promotional causal factors within each IG, according to the Aggregation Function field. (If a given promotional causal factor is represented by shapes, those shapes are summed instead.)
  - Cache the data for the IGs.
5. Cache the data for the IRs.

## Learning Phase

After the first scan of the forecast tree, the Analytical Engine performs the learning phase, which consists of the following steps:

1. Iterate through the forecast tree, starting at the minimum forecast level.
2. Create the following three historical promotional causal factors for each node in the forecast tree:

---

self	Influence on this node caused by promotions on this node
own	Influence on this node caused by other nodes within the same IG
other	Influence on this node caused by all IGs within the IR

---

3. Perform processing to clean up historical data, as specified by various parameters:
  - CutTailZeros
  - ShiftPromoCausals
  - PromotionStartDate
4. Combine the promotional causal factors with the baseline causal factors.
5. Estimate the fit for baseline and promotion coefficients (self, own, and other). If necessary, discard groups of causal factors for specific combinations.
6. Separately validate the fits for baseline and uplifts.
7. Perform the baseline forecast. This forecast represents the sales without any promotions.
8. Validate the baseline forecast.
9. For any node where the promotion coefficients were validated, partition the uplifts



to the promotion attributes that caused them, taking into account the attribute values.

10. Split the baseline and promotional uplifts to the LPL. For lifts, the splitting mechanism does not use the proposit mechanism; instead it considers the attribute values, as appropriate. For baseline, proposit is used as usual.
11. Decompose the promotional uplifts. In this step, the Analytical Engine associates the uplifts with the specific promotions, rather than the attributes.
12. Compact the promotional uplifts for each combination (combining effects of different promotions).The direct and cannibalization effects are treated separately.
13. For past data, split the fit uplifts to the lowest forecast level (using the normal proposit mechanism) and write them to the database.
14. For past data, split the baseline fit and forecast to the lowest forecast level and write them to the database. This step also uses the normal proposit mechanism.
15. Cache the forecast level node data.
16. Cache the IDs of relevant forecast nodes to the database.

## Promotion Forecast Phase

After the learning phase, the Analytical Engine performs the promotion forecast phase, which consists of the following steps:

1. Iterate the forecast tree, this time only on relevant nodes.
2. Load the forecast node data from the cache.
3. From the cached data, create the future promotional causal factors (self, own, and other) for each node in the forecast tree.
4. Complete the coefficients for future promotional causal factors.
5. Combine the promotional causal factors with the baseline causal factors.
6. Generate the promotional forecast. See "The Forecasting Process".
7. Validate the uplifts. (The baseline has already been validated.)
8. Partition the uplifts, as in the learning phase.
9. Split the baseline and promotional uplifts to the LPL, as in the learning phase.
10. Decompose the promotional uplifts.

11. Compact the promotional uplifts.
12. Split the forecast uplift series to the lowest forecast level and write them to the database.

## The Forecasting Process

This section describes the overall forecasting process.

**Note:** For PE mode, this section describes the process that is performed within the final phase of the engine run; see "Promotion Forecast Phase"

The topics here are as follows:

- Summary of the Forecasting Process
- Preprocessing
- Estimation
- Fit and Residuals
- Validation of Fit
- Causal Factor Testing (Envelope function)
- Forecast
- Engine Split for Future Forecasting
- Validation of Forecast
- Bayesian Blending
- Adjustment
- Forecast Failure
- Intermittent Flow

## Summary of the Forecasting Process

The preprocessing module performs the following functions:

1. Cutting leading zeros.
2. (PE mode only) Checking to see whether this node is a promotional node, that is, a

combination that has promotions.

3. Deciding whether the node should be treated by the intermittent flow module.

- (PE mode) First, the node is classified as either promotional or non-promotional, based on whether it has any associated promotions. If the node is promotional, no checking is done for intermittency. If the node is non-promotional, the node is then checked for sparse data; if the node has sparse data, it is flagged for use by the intermittent flow module.

**Note:** In later processing, promotional nodes are treated differently from non-promotional nodes in two other ways:

- The ARIX and ARX models are never used on promotional nodes.
  - The HOLT model is used on promotional nodes only if no other models can be used.
- (DP mode) If the node has sparse data, it is flagged for use by the intermittent flow module.

4. Treating missing values.

5. Performing preliminary outlier and regime change detection.

6. Removing obvious (gross) outliers, if requested. (This feature is not recommended for use with the engine in PE mode.)

7. Transforming data for use in specific models.

After preprocessing, if appropriate (see Step 3, above), the node is now treated by the Intermittent flow module, which uses special model types; see "Intermittent Flow".

Otherwise, the Analytical Engine applies and tests models as follows:

1. Checking that the number of data points exceeds the number of causal factors by at least two. This is done to ensure that no overfitting will occur, and so that coefficients for all causal factors can be determined.

The check is valid only for models IREG, LOG, BWINT, and DMULT. If a model fails this check, it is rejected and a message is written to the log.

2. Estimation. Statistical algorithms are implemented to data and their parameters are calculated.

3. Fit and residuals calculation. The fit reflects the ability of the model to reproduce the actual historical data. The residuals describe the deviation of the fit from the

actual data. The results are used later, in the *Bayesian blending method*.

Then residual outliers are removed, if this option is requested.

4. To check the ability of a model to mimic the actual series, a fit validation is performed (if enabled by the EnableFitValidation parameter). In fit validation, the residuals undergo multiple statistical tests.
5. *Forecast* performs identical calculation to Fit, only for the future period, lead.
6. For a given model, if the forecasting is occurring at the highest forecast level, the Analytical Engine applies a more liberal treatment of models. During forecast validation, models undergo three tests:
  - A test for an unusual zigzag-type jump.
  - A test for abnormal divergence of forecast relative to fit (this is done by building a funnel-shaped envelope and ensuring that the forecast is confined entirely within it).
  - A statistical comparison of forecast and fit means.

Forecast validation is performed only if it is enabled (via the EnableForecastValidation parameter).

7. If at this stage there are no valid models, the time series will be treated by the forecast\_failure procedure, where either the control will be passed over to the shell and data accumulated to the next level on the forecast tree, or, if we are already at the top forecast level, the HOLT model will be attempted, if it has not been tried previously as a regular model (and obviously failed). If it has, or if it fails this time, the NAIVE model is fitted (if enabled by the NaiveEnable parameter).
8. On the other hand, if there are valid models, the Analytical Engine applies the *Bayesian blending method*. This combines the results of all the models, taking two factors into account:
  - The variance of the residuals for each model
  - The complexity of each model (models that use more causal factors can be overfitted and thus should receive less weighting).
9. It may be necessary to adjust it to pick up the recent trend. The EnableAdjustment parameter directs the flow to the adjustment processor, where trend adjustment is performed, using a set of user-specified parameters.

## Preprocessing

The preprocessing stage consists of the following steps:

1. Removing leading zeros. If a series begins with leading zeros, that part of data may be omitted. This is controlled by the CutTailZeros parameter.
2. Intermittency detection and processing. Before checking a series for intermittency, its trailing zeros are temporarily truncated.
  - If there are not enough remaining non zero elements (as measured by the TooFew parameter), the forecast failure module is activated.
  - Otherwise, the IntermitCriterion parameter is checked. This parameter specifies the minimum percentage of zero data points that a series must have to be considered intermittent.
3. Missing values treatment. The Analytical Engine checks the parameter FillParameter. Depending on this parameter null values are replaced by zeros or by the method specified by the FillMethod parameter, which supports the following choices
  - Filling in values by linear interpolation of nearest neighbors.
  - Omitting the values, at the same time adjusting the time scale of causal factors and trends of the Holt procedure. This is useful if you do not want these values not to be accounted for in the estimation procedures. Furthermore, this is the only way to have exact zero "forecasts" in time points where it is known that no demand is expected, like holidays and vacations. Be careful to mark these time points by means of the UpTime parameter.
4. Preliminary outlier detection (if outlier detection is enabled, via the detect\_outlier parameter). Outliers are "unusual" data points, that may distort the result of the forecasting process and lead to erroneous decisions. Detecting them is a nontrivial problem. Often what seems to be an outlier turns out to be a result of expected behavior. Even more frequent are cases in which seemingly sound data are in reality outliers.

**Note:** Outlier detection should be used cautiously with the engine in PE mode. You should not use *gross* outlier detection at all in this mode.

If outlier detection is overused, the engine discards promotions and cannot learn from them. Future promotions will then have no lift.

- The MinLengthForDetect parameter specifies the minimum number of data points needed to perform outlier detection (the default is a year's worth of data).
- Demantra computes a range of "normal" values and counts the number of data

points that lie outside that range. If a relatively small number of data points lie outside the range, they are considered outliers and are discarded. On the other hand, if a relatively large number of data points lie outside the range, then Demantra considers all of them to be real data points, and does not discard any of them as outliers. The `OutliersPercent` parameter controls the threshold for this test.

5. Preliminary outlier handling, of only obvious (gross) outliers. This step is performed only if gross outlier handling is enabled via the `GrossRemove` parameter. The `OutlierStdError` parameter controls the sensitivity of the gross outliers detection. The smaller the value, the more aggressively the procedure will detect outliers.

**Note:** At this stage, only the gross outliers are removed. Other outliers are retained, because they may later be attributed to assignable causes, which will be revealed only at the model building stage.

Gross outlier detection is not recommended for use with the engine in PE mode.

6. Gross outliers are permanently filled by linear interpolation.
7. Preliminary regime change detection (if enabled by the `detect_cp` parameter). In the preliminary stage, this procedure finds points of change in the level or trend. The `RegimeThreshold` parameter controls the sensitivity of detection regime change. The smaller the value, the more aggressively the procedure will detect regime changes.

**Note:** There is no outlier or regime change detection for intermittent data.

8. If `TrendPreEstimation` is yes (1), the Analytical Engine performs trend detection.

**Note:** If you have disabled negative regression (via `UseNonNegRegr`), then it is difficult for the Analytical Engine to detect downward trends. In such cases, you should enable trend detection via `TrendPreEstimation`.

Trend detection works as follows. The history is divided into two segments: the long segment, which is followed by the short segment. The short segment is assumed to have a trend. Demantra automatically generates a new trend causal factor for each segment (by fitting to the general shapes of those segments) and passes those new causal factors to the engine, to replace the existing trend causals.

You can specify the following settings to control the specific behavior:

- First, the TrendPeriod parameter specifies the boundary between the long segment and the short segment. This parameter specifies this boundary in terms of latest, most recent time buckets.
- The TrendDampPeriod and TrendDampStep parameters specify how this trend should be dampened (toward the future), which is useful particularly with an upward trend (which, when extrapolated, would give unrealistic values). The TrendDampPeriod parameter specifies a block of time (as a number of time periods) over which the residual dampening is applied. Dampening is not applied for the last historical block, and is applied in an exponential manner on previous historical blocks. The size of the dampening depends on parameter TrendDampStep. The TrendDampStep parameter specifies the dampening factor, which is applied n times to the nth block of time. The result is exponential dampening.
- The TrendModelForShort parameter specifies which engine model to use in order to generate the trend causal factor in the short segment (either REGR or HOLT).
- The TrendOutlierRatio and TrendShortRatio parameters specify how to treat points found as outliers during trend pre-estimation. Each of these is a numeric weight to apply to the outliers. The TrendOutlierRatio parameter controls the weighting of outliers in the long segment, and the TrendShortRatio controls the weighting of outliers in the short segment.

#### 9. Data transformations for use in specific models.

## Estimation

The Analytical Engine uses different estimation procedures for each engine model. See "Theoretical Engine Models".

If UseWeightedRegression is yes (1), then the Analytical Engine applies a weight to each observation when fitting each model. The OBS\_ERROR\_STD field (in sales\_data) specifies the weights for each observation; the default value is 1.

## Fit and Residuals

Fit and residual procedures are also model-specific. They calculate values fitted by the model to historical data and evaluate the residuals. Non-positive fitted values are set to zero (depending on the setting of the AllowNegative parameter).

For the logarithmic models (LOG and ELOG), the operation of antilog, to convert results back to original metric, must consider the form of the expectation of a lognormal variable. To use this corrected conversion, activate the LogCorrection parameter.

The Analytical Engine sorts the residuals by size and removes the largest residuals. The parameter `RemoveResidOutlier` specifies how many residuals to remove, as a percentage of the total number of residuals.

## Validation of Fit

Although fit validation is model-specific, it is activated globally by the parameter `EnableFitValidation`.

This procedure consists of the following steps:

1. **Outliers.** Check the influence of outliers on the residuals. The `Quantile` parameter specifies a standard normal percentile for detecting outliers at a prescribed significance level. If an outlier affects the residuals, no further validation is needed, and we proceed to the problem correction stage. Otherwise, the Analytical Engine tests the goodness of fit.
2. **Valid\_fit.** Here a battery of four statistical tests are performed. Failure of one of them leads to rejection of fit validity.
  - `Mean_check` is a test for comparison of means of fitted and actual data. The `MeanRelativeDistance` parameter is the maximum MAPE (Mean Absolute Percentage Error) allowed in a model that is `MeanValid`.
  - `Std_check` is a test for comparison of standard deviations of two parts of the residuals. The division into parts (earlier and later) is controlled by the `TestPeriod` parameter. The `StdRatio` parameter is the maximum allowed ratio of the standard deviation of the later part to the standard deviation of the earlier part.
  - `Bjtest` is the Bera-Jarque test for normality of residuals. Normal distribution of errors is a desired feature, assuring randomness, independency and lack of bias in the errors, thus indicating that the model was successful in catching and removing all systematic variability in data.
  - Finally, residuals are checked for presence of large deviations, by comparing them to a multiple of standard deviation, as specified by the `DeviationFactor` parameter.
3. If fit validation fails, the following occurs:
  1. Detect outliers.
  2. Replace the outlying values by values calculated by linear interpolation.
  3. Refit. Re-estimation of model parameters for the series corrected for outliers, recalculation of fit and residuals, followed by revalidation.



## Causal Factor Testing (Envelope Function)

For some of the engine models (CMREGR, ELOG, LOG, MRIDGE, and REGR), Demantra can choose random sets of causal factors, which it then tests. Demantra can then either use the set of causal factors that gives the best result or use a mix of causal factors.

This operation is known as the *envelope* function, because it is performed as an envelope around the main engine flow. This operation is controlled by the UseEnvelope parameter, which can equal any of the following:

- 0 (Do not use the envelope function).
- 1 (Use the envelope function on five groups of causal factors: base plus direct and the four switching groups).
- 2 (Use the envelope function on the causal factor groups defined in Estimation\_groups table)

Additional parameters further control the behavior for specific engine models:

- ENVELOPE\_RESET\_SEED specifies whether to reset the randomization seed for the envelope function, which evaluates different sets of causal factors for different engine models.
- ENVELOPE\_CHAIN\_LENGTH specifies the number of variations of causal factors to try, for each model.
- BestOrMix specifies whether to use the best set of causal factors (0) or to use a mix of the causal factors (1).

## Forecast

The forecast is calculated in almost the same way as the fit; see "Fit and Residuals". The key difference is that the Analytical Engine does not analyze causal factors when computing the forecast. Instead, the engine uses its learning, combined with the future values of the causal factors. The lead parameter specifies the length of time (in the future) for which the forecast is generated. If negative values are disallowed, the Analytical Engine sets them to zero.

## Validation of Forecast

At this point, the forecast is validated. The purpose of this validation is to avoid abnormalities in the projected result of a model. The validation is identical for all models, except HOLT, which does not use it. The EnableForecastValidation parameter controls the applicability of forecast validation.

Forecast validation includes three tests:

1. Jump test. This test detects up-and-down or down-and-up zigzag-like jumps. The magnitude of upward jumps is controlled by the Quantile parameter. The larger the value of this parameter, the more liberal is the jump test.
2. Envelope test. This test spreads a funnel-like envelope over the forecast. The shape of the envelope is a function of the behavior of the underlying time series. There is no external control over the sensitivity of envelope test.
3. Mean test is a test on means of the forecast and the later part of the time series of length given by the test\_samp\_len parameter.

The ForecastMeanRelativeDistance parameter controls the sensitivity of forecast validation. The larger its value, the more liberal is the test.

## Bayesian Blending

First, the Analytical Engine checks the setting of the DetectModelOutliers parameter, which specifies whether to detect model outliers for each forecast node. A model outlier is an engine model that does not give good enough results for that node. The ModelValidationBound parameter controls the sensitivity of the test, which proceeds on each node as follows:

1. For each model, a Demantra proprietary algorithm computes an index that indicates the goodness of fit for that model at that node. Small values are considered good.
2. The Analytical Engine sorts these indexes by value and computes the difference in value between each successive pair of indexes.
3. If none of these differences are greater than the value of ModelValidationBound (whose default is 0.2), the Analytical Engine considers *all* the models good enough and does not look for outliers.
4. If any of the differences are greater than ModelValidationBound, then the Analytical Engine fits a line through the indexes and uses it to determine which models to discard. Any models with points that lie too far above the line are discarded.

For each forecast node, the Analytical Engine discards any model outliers and then combines the results for all models using the Bayesian blending method. This combines the results of all the models, taking two factors into account:

- The variance of the residuals for each model.
- The complexity of each model (models that use more causal factors can be overfitted and thus should receive less weighting).

It is often necessary to enhance models that perform better on most recent historical

data, as opposed to models that show close fit to the remote history. This is achieved by assigning decaying weights to residuals, so that recent residuals have greater weights than the remote ones. The DampStep parameter specifies the rate of weights decay, and the DampPeriod parameter specifies the number of periods in which the residuals will receive the same weights. The dampening of weights is done between each successive period, so that the result is exponential decay.

## Adjustment

In the adjustment phase, the Analytical Engine performs a final tuning of the forecast, enabling the user to adjust the forecast to the recent trend in the historical data. Not recommended, unless it is known that a change in trend happened recently, which is likely to be missed by the models. The following parameters are used for adjustment:

- EnableAdjustment enables the adjustment.
- TrendPeriod specifies the period for trend estimation; if zero then no adjustment will be made.
- DownTrend (a value from 0 to 1, inclusive) specifies the degree of descending trend adjustment.
- UpTrend (a value from 0 to 1, inclusive) specifies the degree of ascending trend adjustment.
- PercentOfZeros specifies the maximum percent of zero values in the estimation part to enable trend adjustment.

## Forecast Failure

If all participating models fail one of the preceding validations, the control is transferred to the engine shell in order to aggregate to the next level on the forecast tree.

If the model HOLT has not been previously applied at the last level and if there are enough data points, then HOLT is attempted. (HOLT is usually configured for short time series, less than one season). One can optimize its parameters by requesting *Optimally*. The model follows the usual path of estimation, fit and residuals calculation, fit validation, forecast calculation and forecast validation.

If HOLT fails, or if it has been used on this level before, or if there are very little data, an attempt is made to obtain a last resort forecast. Here, the parameter NaiveEnable controls the choice of how to proceed; this parameter has one of the following values:

- no (0): Do not enable either NAIVE or Moving Average models. Do not generate a forecast.
- yes (1): Enable use of the NAIVE model.

- 2 or higher: Enable use of the Moving Average model. In this case, the setting of NaiveEnable specifies the number of recent time buckets to use in calculating the moving average.

If you are using the Analytical Engine in PE mode, note that the NAIVE and Moving Average models do not generate any lift.

## Intermittent Flow

First:

- For PE mode, if a given node has an associated promotion, no checking is done for intermittent data. If it does not have a promotion and if it has as intermittent (sparse) data, it is treated by the Intermittent flow module, which uses special model types.
- For DP mode, if a node has intermittent (sparse) data, it is treated by the Intermittent flow module, which uses special model types.

In the intermittent flow module, the Engine Administrator handles series that were found to be intermittent at the preprocessing stage, according to the IntermittentCriterion parameter. Basically, it has many common features with the main flow.

In contrast to the case with non-intermittent models, if there are too many causal factors in comparison with the length of time series, a warning message will be issued, but the model will still be estimated.

The fit validation of intermittent models is simplified and brought down to a comparison of means.

No real forecast validation is done for intermittent models.

If there is a decline in demand at the end of the historical period, then the engine will update the fit after the last spike in history accordingly. To control the intensity of the forecast, you use the IntUpdate parameter.

If the final result is asked for in the form of spikes (as specified by the need\_spread parameter), the unspread processor is activated.

The Analytical Engine can run with a minimal set of causal factors. There is no prerequisite for causals in both global and local causal groups. If no global or local causal factors are available, then the Constant global causal factor is used. If the constant causal factor is set to 0, the model could fail with the following message:

"Constant should be chosen for both groups of causals. This is strongly recommended for estimation results, unless sales should be zero for particular time."

The Analytical Engine adheres to the following steps for each causal driven model:

1. Before launching the model, the Analytical Engine builds the matrix ModelGroupCausal from local, global, and PE causal factors. The causal factors are

stored in the GroupCausal matrix, and the Analytical Engine picks up only those rows that belong to given model.

2. If no causal factors are available, the model fails with the message "No Causals Available".
3. If the number of available causal factors is more than the number of data points for the forecasted combination, the model fails with the message "Does not have a usable number of observations (too few or too many)."

## Comparison of Batch and Simulation Modes

For reference, this section compares how the Analytical Engine runs in batch mode and in simulation mode.

### Batch Mode Characteristics

In a batch run, the Analytical Engine does the following:

1. Traverses a large forecast tree, described in a database. Each node in this tree represents a time-based data series that is a subject to forecast.
2. Performs statistical model calculations on a large subset of the data series (tree nodes). The order of the processing the nodes is important, and is derived from the forecast tree, defined by a few business rules and performance limitations. The forecast tree is traversed using a recursive tree scan.
3. Writes the processed data series to the forecast table in the database.
4. Runs a database procedure that scans and updates the forecast table.

### Simulation Mode Characteristics

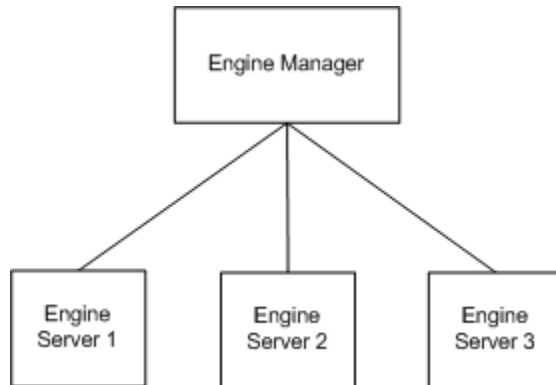
In a simulation run, the Analytical Engine performs 'what if' scenarios, in which some of the forecast data is changed or different models are run to see how this influences the final results. The four steps related to the batch engine run are also applied here, but on a much smaller section of the forecast tree. The number of data series modeled is much smaller compared to a batch engine run.

## Engine Components and High-Level Flow

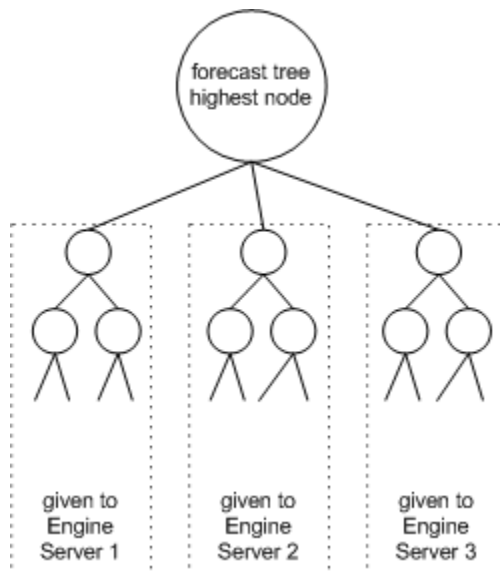
At a higher level, it can be useful to understand how the Analytical Engine divides and processes its work.

## Engine Components

Internally, the Analytical Engine consists of one Engine Manager and multiple engine servers.



The engine server scans a portion of the forecast tree, and sends the output to the report mechanism. The engine server masks the mdp\_matrix table and processes only the nodes that are in the part of the tree relevant to its task. The ID of the task is received from the Engine Manager, which is responsible for dividing the forecast tree into smaller sub trees (called tasks).



The Engine Manager is responsible for controlling the run as a whole. Communication between the various engine modules is achieved by using the COM protocol.

## Engine Components and Batch Run

The following steps describe the responsibilities of each component during a batch run

of the Analytical Engine.

1. The Engine Manager creates and initializes the engine servers. Initialization includes the following steps:
  - The Engine Manager passes a callback interface to the engine servers. The engine servers will use this interface in order to make requests for new tasks to process, or to return status completion information to the Engine Manager.
  - The Engine Manager passes the database settings and all other settings to the engine servers.
  - The engine servers connect to the database and load parameters.
  - The engine servers initialize themselves using the xml schema files and request the Engine Manager for tasks to process.
2. The Engine Manager checks if the run is a recovery run or a new run, and acts accordingly. If it is a recovery run, the Engine Manager retrieves unfinished tasks. If it is a new run, the Engine Manager resets the mdp\_matrix table and allocates a forecast column. The Engine Manager divides the forecast tree into smaller tasks by updating one column in mdp\_matrix that links each node with a task ID. The number of the tasks that the Engine Manager attempts to create is the number of engine servers that were initialized successfully, multiplied by a configurable factor.
3. The Engine Manager executes all the engine servers and waits for them to return a final completion status.
4. When an engine server is executing, it uses the Engine Manager callback interface in order to get task IDs to process (pull approach). The data flow between the Engine Manager and the engine servers is very low volume, containing only settings, task IDs and statuses. The data that flows between the engine servers and the database includes the sales (input) and forecasted (output) data (very high volume), forecast tree configuration information, database parameters, and certain other information.
5. The engine server uses the task ID to create a sales\_data\_engine table (or view) with the records for that task and then scans the forecast tree, operating select and update queries on the mdp\_matrix table. During the processing of a task, an engine server filters mdp\_matrix according to the task ID and operates only the subtree relating to that task. It uses two threads, one for scanning the tree and performing calculations, and one for the proport mechanism.
6. When the engine server gets a null task ID from the Engine Manager, it knows that no more task IDs are available, and it sends a completion notification to the Engine Manager.

7. When the Engine Manager has received a completion status indicator from all the engine servers, it updates the run status, executes the post process procedure, and the engine run is completed.

## Details of the Distributed Engine

Your system may include the Distributed Engine, which is a mode in which the Analytical Engine automatically distributes its work across multiple machines simultaneously.

**Note:** For the Distributed Engine to work, the Analytical Engine must be registered on multiple machines, all of which have database client software in order to access the Demantra database.

The Distributed Engine drastically shortens the run time for a single batch engine run by processing the engine tasks in parallel, on *different machines*, for improved engine processing time. Also, multiple simulation requests can be handled simultaneously.

In a batch run, the Distributed Engine starts by reading a settings file that lists the machines on the network where the Analytical Engine is installed. The Engine Manager tries to instantiate an engine server on the machines in this list. Processing then continues with Step 1.



---

## Engine Parameters

This chapter describes the Analytical Engine parameters that you can see in Business Modeler and lists their default values, if any.

This chapter covers the following topics:

- About Engine Parameters
- Analytical Engine Parameters

### About Engine Parameters

For each parameter, this chapter indicates which engine variations that parameter can be used with. This chapter also indicates which parameters can be used with nodal tuning. Some of the Promotion Effectiveness (PE) parameters are useful only if your system also includes Promotion Optimization.

Oracle provides two different modes for the Analytical Engine:

- In PE mode, the engine is suitable for use with Promotion Effectiveness.
- In DP mode, the engine is suitable for use in demand planning applications.

As indicated, most parameters are visible to all users; a few are visible only if you log in as the owner of the component.

See also

"Theoretical Engine Models"

### Analytical Engine Parameters

Parameter	Location	Default	Engine Mode*	Details	Tuning
<b>A</b>					
add_zero_combos_to_mdp	Engine > Data Manipulation	yes	Both**	<p><b>Visible only to owner.</b> Specifies the Proportional mechanism handles combinations whose historical data consists of zeros. Use one of the following values:</p> <ul style="list-style-type: none"> <li>yes: Add these combinations to mdp_matrix even if their historical data consists of zeros.</li> <li>no: Do not add these combinations.</li> </ul>	
AllowableExceptions	Engine > Validation	10	PE only	<p><b>Visible only to owner.</b> Specifies the permissible amount of exceptional uplifts, as a percentage of total number of uplifts. The LowerUpliftBound parameter controls the threshold for exceptional uplifts.</p> <p>The engine discards a model (for a given forecast node) in either of two cases:</p> <ul style="list-style-type: none"> <li>If the model generates too many exceptional uplifts (as specified by the LowerUpliftBound and AllowableExceptions parameters).</li> <li>If any uplift exceeds the bound given by the UpperUpliftBound parameter.</li> </ul>	Global setting only

Parameter	Location	Default	Engine Mode*	Details	Tuning
AllowNegative	Engine > Adjustment	no	Both	<p>This parameter is used by the fit and residuals module of the Analytical Engine. Use one of the following values:</p> <ul style="list-style-type: none"> <li>• yes: Negative values of fit and forecast are allowed.</li> <li>• no: Any non-positive fitted and forecasted values are set to zero.</li> </ul>	Can be tuned by node
AnalyzeMdp	Engine > Shell	Full analyze	Both	<p><b>Visible only to owner.</b> Specifies how to analyze the mdp_matrix table after the Engine Manager divides the forecast tree into tasks. Use one of the following values:</p> <ul style="list-style-type: none"> <li>• 5 columns analyze: Enable a partial analysis using the five most important fields: prediction_status, prop_changes, branch_id, do_aggri, and do_fore.</li> <li>• Full analyze: Enable a full analysis.</li> <li>• No analyze: Disable the analysis.</li> </ul> <p><b>Note:</b> The branch_id field is for internal use only.</p>	Global setting only

Parameter	Location	Default	Engine Mode*	Details	Tuning
AverageHorizon	Engine > Data Manipulation	12 for monthly 52 for weekly 7 for daily	PE only	<p><b>Applies only to Promotion Optimization; parameter is visible only to owner.</b></p> <p>Specifies the length of time to be used in calculating the average baseline forecast. This window of time starts at the date given by the StartAverage parameter.</p> <p>For information on configuring Promotion Optimization, see "Configuring Promotion Optimization for PTP".</p>	Global setting only
B					
BatchRunMode	Engine > Shell	estimation and forecast run	PE only	<p>Specifies the kind of forecasting to do:</p> <ul style="list-style-type: none"> <li>run the forecast against only the learning (0; estimation)</li> <li>run the promotion forecast (1; recommended setting)</li> <li>estimation and promotion forecast run (2; fast simulation), using previously cached data. If no cached data is found, the Analytical Engine gives a message and calculates the needed data.</li> </ul> <p>This parameter applies to both batch run and simulation run.</p>	Global setting only

Parameter	Location	Default	Engine Mode*	Details	Tuning
BottomCoefficientLevel	Engine > Data Manipulation	1	PE only	<p><b>Applies only to Promotion Optimization; parameter is visible only to owner.</b></p> <p>Specifies the lowest forecast tree level for which the Analytical Engine will calculate coefficients. Use any forecast tree level between the lowest promotional level and the InfluenceRangeLevel, inclusive.</p> <p>For information on configuring Promotion Optimization, see "Configuring Promotion Optimization for PTP".</p>	Global setting only
BulkLoaderBlockSize	Engine > Shell		Both	<p><b>Oracle only; visible only to owner.</b> Specifies the minimum amount of number of rows that the Analytical Engine loads at one time, when writing to the database. The larger this is, the more quickly the data is loaded, but there is greater risk if the database connection is lost. Use a value between 100 and 100,000.</p>	Global setting only
BulkLoaderEnabledRecovery	Engine > Shell		Both	<p>Specifies whether Oracle Bulk Loader should perform recovery after a lost database connection. Oracle Bulk Loader is used by the Analytical Engine.</p>	Global setting only
C					

Parameter	Location	Default	Engine Mode*	Details	Tuning
CachePath		Null	Both	<p>Specifies the path to the directory into which the Analytical Engine should write its caching files. This can be any of the following:</p> <ul style="list-style-type: none"> <li>• A relative path (relative to Demantra_root/Demand Planner/Analytical Engines/bin).</li> <li>• An absolute path.</li> <li>• Null. In this case, the Analytical Engine creates its caches in Demantra_root/Demand Planner/Analytical Engines/bin/cache.</li> </ul> <p>You should create the directory manually if it does not yet exist.</p>	Global setting only
CalcOptimizationInput	Engine > Data Manipulation	no	PE only	<p><b>Applies only to Promotion Optimization; parameter is visible only to owner.</b></p> <p>Specifies whether the Analytical Engine should calculate inputs needed for Promotion Optimization. Use one of the following values:</p> <ul style="list-style-type: none"> <li>• yes (1): See "Configuring Promotion Optimization for PTP". Make sure to set the IS_OPTIMIZATION flag equal to 1 for at least one of the linear engine models.</li> <li>• no (0)</li> </ul>	Global setting only

Parameter	Location	Default	Engine Mode*	Details	Tuning
cannibalism	Engine > Data Manipulation		Both**	<p>Specifies the default values for aggri_98 and aggri_99, which are combination-specific fields.</p> <p>If equal to 0 or 1, the defaults for both fields are 1.</p> <p>If equal to 2, the default for aggri_98 is 1, and the default for aggri_99 is 0.</p>	Global setting only
CannibalizationIgnore	Engine > Data Manipulation		PE only	<p>Controls whether the Analytical Engine will calculate switching effects (cannibalization). You can use this parameter to easily switch off that calculation when needed, for example, when running specific simulations.</p>	Global setting only

Parameter	Location	Default	Engine Mode*	Details	Tuning
COMPETITI ON_ITEM	Engine > Shell		PE only	<p><b>Visible only to owner.</b> Specifies the level (from the group_tables table) that defines the competitive item (CI) groups. Each node of this level represents a different item group.</p> <p>The CI should be consistent with the item groups (I). Specifically, two items within a given item group must also belong to the same competitive item group. The easiest way to follow this rule is to set the CI equal to an item level that is higher than I and that is within the same hierarchy. A similar rule applies for the locations.</p> <p><b>Note:</b> You specify the item groups indirectly when you specify the IGL in the forecast tree. see "Configuring the Forecast Tree".</p>	Global setting only
COMPETITI ON_LOCATI ON	Engine > Shell		PE only	<p><b>Visible only to owner.</b> Specifies the level (from the group_tables table) that defines the competitive location (CL) groups.</p> <p>See the notes for COMPETITION_ITEM.</p>	Global setting only



Parameter	Location	Default	Engine Mode*	Details	Tuning
CutTailZeros	Engine > Data Manipulation	yes	Both	<p><b>Visible only to owner.</b> Specifies how the preprocessing module (of the Analytical Engine) should handle series that start with zero values. Use one of the following values:</p> <ul style="list-style-type: none"> <li>• yes: Delete the leading zeros.</li> <li>• no: Retain them as actual zero values.</li> </ul>	Can be tuned by node
D					
DampPeriod	Engine > General	0	Both	<p>This parameter is used by the Bayesian blending module of the Analytical Engine. It specifies the length of periods in which the residuals will receive the same weights. The dampening of weights is done between each successive period.</p> <p>This parameter lets you put greater weight on models that perform better on most recent historical data, as opposed to models that show close fit to the remote history.</p>	Can be tuned by node
DampStep	Engine > General	0	Both	<p>This parameter is used by the Bayesian blending module of the Analytical Engine. It specifies the rate of weights decay. By setting this parameter to 0 (or 1), you set all weights to be equal to 1 (equal weights).</p>	Can be tuned by node

Parameter	Location	Default	Engine Mode*	Details	Tuning
def_delta	Engine > Proport	0.75	Both**	<p>Specifies the default value for the delta field in the mdp_matrix table. If delta equals null for a given combination, the system uses the value of this parameter instead.</p> <p>All new combinations created through data loading, member management, and/or chaining will have a null value in their delta column, thus indicating that they will also take the default delta value from this parameter.</p> <p>In turn, the delta field is used in the proposit calculation as in the following example:</p> <p><math>P1 = glob\_prop * delta + (monthly\ demand) * (1 - delta)</math></p>	Global setting only

Parameter	Location	Default	Engine Mode*	Details	Tuning
DeleteIsSelfRows			PE only	<p>Specifies whether the Analytical Engine deletes unneeded promotion_data records. Use one of the following values:</p> <ul style="list-style-type: none"> <li>0 means that the Analytical Engine does not delete records in promotion_data.</li> <li>1 means that the Analytical Engine deletes unneeded records.</li> </ul> <p>A record is considered unneeded if all the following conditions are true:</p> <p>It is flagged as is_self = 0</p> <p>All lifts (uplift, pre and post effect, and switching effects) equal 0</p> <p>The condition specified by DeleteIsSelfCondition is true</p> <p>Also see "Is_Self".</p>	Global setting only
DeleteIsSelfCondition			PE only	<p>Specifies an additional true/false condition that must be met to delete unneeded records in promotion_data. Used only if DeleteIsSelfRows is 1.</p> <p>This parameter is used as an SQL extra where clause. The Analytical Engine uses it to restrict the deletion.</p>	Global setting only

Parameter	Location	Default	Engine Mode*	Details	Tuning
detect_cp	Engine > Outlier and Regchange	yes	Both	<p>This parameter is used by the preprocessing module of the Analytical Engine. Use one of the following values:</p> <ul style="list-style-type: none"> <li>• yes: The engine should attempt to detect a regime change in the level or trend. If it finds a change point, it performs the analysis on the leveled out series. The threshold for change points is controlled by the RegimeThreshold parameter.</li> <li>• no: The engine should not attempt to detect change points. The RegimeThreshold parameter is ignored.</li> </ul>	Can be tuned by node
detect_outlier	Engine > Outlier and Regchange	yes	Both	<p>This parameter is used by the preprocessing module of the Analytical Engine. Use one of the following values:</p> <ul style="list-style-type: none"> <li>• yes: The engine should attempt to detect outliers. If it finds outliers, it considers them in the analysis.</li> <li>• no: The engine should not attempt to detect outliers.</li> </ul> <p>Also see GrossRemove. To disable all outlier detection, both these parameters must be switched off.</p>	Can be tuned by node

Parameter	Location	Default	Engine Mode*	Details	Tuning
DetectModel Outliers			Both	<b>Visible only to owner.</b> Specifies whether to check for outlier models for each forecast node. Outlier models are models that do not fit well enough. The sensitivity of the test is controlled by the ModelValidationBound parameter.	Global setting only
DeviationFactor	Engine > Validation	5	Both	<b>Visible only to owner.</b> This parameter is used by the fit validation module of the Analytical Engine, and it controls the sensitivity of one of the fit validation tests. In this test, residuals are checked for presence of large deviations, as specified by DeviationFactor. This parameter specifies the maximum number of standard deviations that the residuals are allowed to attain. A model is rejected if it fails this test.	Can be tuned by node

Parameter	Location	Default	Engine Mode*	Details	Tuning
DownTrend	Engine > Adjustment	0.2	Both	<p>This parameter is used by the adjustment module of the Analytical Engine, if that module is enabled (via EnableAdjustment). It controls the forecast adjustment for downward trend. Specifically, it specifies the amount by which the forecast is rotated to align with recent trend in data.</p> <p>Use a value from 0 to 1, inclusive.</p> <p>Enabling adjustment is not recommended, unless it is known that a change in trend happened recently, which is likely to be missed by the models.</p>	Can be tuned by node
dying_time	Engine > Proport	0.5 season (1 season in media)	Both**	<p>If no sales occurred during the length of time specified by this parameter, the combination is marked as dead. See prediction_status.</p>	Global setting only
E					

Parameter	Location	Default	Engine Mode*	Details	Tuning
EnableAdjustment	Engine > Adjustment	no	Both	<p>This parameter controls the adjustment module of the Analytical Engine. Use one of the following values:</p> <ul style="list-style-type: none"> <li>• yes: Enable the adjustment module, which performs a final tuning of the forecast, adjusting the forecast to the recent trend in the historical data.</li> <li>• no: This is the recommended setting, unless you are sure that a change in trend happened recently, which is likely to be missed by the models.</li> </ul>	Can be tuned by node
EnableFitValidation	Engine > Validation	yes	Both	<p><b>Visible only to owner.</b> This parameter controls the fit validation module of the Analytical Engine. Use one of the following values:</p> <ul style="list-style-type: none"> <li>• yes: Perform a normal validation for the fit.</li> <li>• no: Perform only a weak validation.</li> </ul>	Can be tuned by node

Parameter	Location	Default	Engine Mode*	Details	Tuning
EnableForecastValidation	Engine > Validation	yes	Both	<p><b>Visible only to owner.</b> This parameter is used by the forecast validation module of the Analytical Engine. Use one of the following values:</p> <ul style="list-style-type: none"> <li>• yes: Perform a normal validation for the forecast.</li> <li>• no: Perform only a weak validation.</li> </ul>	Can be tuned by node
EnableModifiedVariance	Engine > General	no	Both	<p><b>Visible only to owner.</b> This parameter is used by the fit validation module of the Analytical Engine. Use one of the following values:</p> <ul style="list-style-type: none"> <li>• yes: Perform the modified variance, which specifies how the variance is calculated in determining weights for Bayesian blending.</li> <li>• no</li> </ul>	Can be tuned by node



Parameter	Location	Default	Engine Mode*	Details	Tuning
EnableSimGLFilter	Engine > General	yes	PE only	<p><b>Visible only to owner.</b> Specifies whether simulation should respect or ignore any general-level filtering applied to the worksheet. Use one of the following values:</p> <ul style="list-style-type: none"> <li>yes: Respect the general level filter and run the simulation only on combinations in the worksheet and only on the general level members that is included in the filter. This option ignores, for example, any other general level members associated with those combinations.</li> </ul> <p>This setting should be used for fast simulations only. If used on promotions or scenarios, only the selected member will receive a regeneration of uplift. All other members—even if they would normally interact with each other—will be excluded. If learning is run using this setting, there is a very good chance that engine results will be wrong due to inclusion of only a part of history.</p> <ul style="list-style-type: none"> <li>no: Ignore the general level filter and potentially run the simulation on combinations that are not included in the worksheet. This is the previous behavior.</li> </ul>	Global setting only

Parameter	Location	Default	Engine Mode*	Details	Tuning
				This parameter has no effect if the worksheet is not filtered by a general level.	
EngDimDef_ItemOrLoc	Engine > Tables	none	DM only	Determines which forecast tree dimension, item or location displays additional levels. The default is that no additional levels are shown.	Global Setting only
EngineOutputThreshold	Engine > General	0	Both	see Fine Tuning and Scaling Demantra -- EngineOutputThreshold	Global Setting only
EngKeyDef_Supersession	Engine > Proport	item_id , location_id	DM only	Key used to aggregate members belonging to the same supersession set. When set to the same value as EngKeyDefPK, the proportion calculates proportions for each lowest-level member processed by the engine individually and no special handling of supersessions is done. If set above the level defined by EngKeyDefPK, then calculation of proportions is done at this aggregated level considered the supersession and all underlying combinations receive the same proportional values.	Global Setting only
EngKeyDefPK	Engine > Tables	item_id , location_id	DM only	Defines the primary key of the combination and data tables selected for this analytical engine profile.	Global Setting only
EngTabDef_HistoryForecast	Engine > Tables	SALES _DATA	DM only	Table that holds historical demand and into which the forecast to be written.	Global Setting only

Parameter	Location	Default	Engine Mode*	Details	Tuning
EngTabDef_Inputs	Engine > Tables	INPUTS	DM only	Table that contains time definitions for the chosen engine profile. This parameter should not be modified.	Global Setting only
EngTabDef_Matrix	Engine > Tables	MDP_MATRIX	DM only	Table that holds the combinations available for the chosen engine profile.	Global Setting only
EngTabDef_Parameters	Engine > Tables	PARAMETERS	DM only	Table that holds the analytical model parameters for chosen engine profiles. This parameter defines the method that the supersession forecast is allocated to general level members during the supersession forecast. When set to the default for each period, proportions are allocated completely to the member with the latest starting date. If set to All Active Revisions for each period, proportions are allocated equally among all active members.	Global Setting only
F					

Parameter	Location	Default	Engine Mode*	Details	Tuning
FillMethod	Engine > Data Manipulation	linear interpolation	Both	<p>This parameter is used by the preprocessing module of the Analytical Engine (if FillParameter equals 1). The FillMethod parameter specifies how to fill any null (missing) values. Use one of the following values:</p> <ul style="list-style-type: none"> <li>linear interpolation: Fill in values by linear interpolation of nearest neighbors.</li> <li>omitting missing values: Omit the null values and adjust the time scale of causal factors and trends of the Holt procedure; also see the UpTime parameter.</li> <li>This parameter is ignored if FillParameter equals 0.</li> </ul>	Can be tuned by node
FillParameter	Engine > Data Manipulation	0	Both	<p>This parameter is used by the preprocessing module of the Analytical Engine. It specifies how to handle null (missing) values. Use one of the following values:</p> <ul style="list-style-type: none"> <li>yes:</li> <li>no:</li> <li>If equal to 0, null values are replaced by zeros and FillMethod is ignored.</li> <li>If equal to 1, null values are filled as specified by FillMethod.</li> </ul>	Can be tuned by node

Parameter	Location	Default	Engine Mode*	Details	Tuning
ForecastGenerationHorizon	Engine > Time	0	Both	Specifies what historical fit data the engine will write to the database. If this parameter is 0, the engine writes the forecast only. If this parameter is a positive integer N, the engine writes the last N historical fit values.	Global setting only
ForecastMeanRelativeDistance	Engine > Validation	3.5	Both	<b>Visible only to owner.</b> This parameter is used by the forecast module of the Analytical Engine. It specifies the sensitivity of forecast validation. The smaller the value, the stricter the test.	Can be tuned by node
G					
GLPropSuperSessionMethod	Engine > Proport	latest revision	DM only	Defines the method general level proportions use to allocate proportions during supersessions. When set to the default for each period, proportions are allocated completely to the member with the latest starting date. If set to All Active Revisions for each period, proportions are allocated equally among all active members.	Global setting only

Parameter	Location	Default	Engine Mode*	Details	Tuning
GrossRemove	Engine > Outlier and Regchange	yes	Both	<p>This parameter is used by the preprocessing module of the Analytical Engine. Use one of the following values:</p> <ul style="list-style-type: none"> <li>yes: The engine should process gross outliers. Enable this feature only if there is a clear reason to remove obviously unreasonable values. The threshold for gross outliers is controlled by the OutlierStdError parameter.</li> <li>no</li> </ul>	Can be tuned by node
H					
HighestSquaring	Engine > Validation	4	Both	<p><b>Visible only to owner.</b> This parameter is used by the fit validation module of the Analytical Engine. It specifies the number of residual standard deviations, beyond which the residuals participate in the sum of squares calculation in their absolute value, rather than squared.</p>	Can be tuned by node
hist_glob_prop	Engine > Proport	1 season	Both**	<p>Maximum number of base time buckets to use in calculating glob_prop, the running average demand for any given item-location combination. This parameter is used by the proport mechanism.</p>	Global setting only

Parameter	Location	Default	Engine Mode*	Details	Tuning
HistoryLength	Engine > Time	0	Both	The number of base time buckets to consider for fit estimation and for the proportion mechanism. Must be a non-negative integer. If equal to 0, the length of the history is set by the start_date parameter instead.	Can be tuned by node
I					
InfluenceGroupLevel	Engine > Shell		PE only	<b>Read-only.</b> Specifies which level (from the group_tables table) is used as the influence group level of the forecast tree. To specify this parameter, you use the Forecast Tree Editor within the Business Modeler.	Global setting only
InfluenceRangeLevel	Engine > Shell		PE only	<b>Read-only.</b> Specifies which level (from the group_tables table) is used as the influence range level of the forecast tree. To specify this parameter, you use the Forecast Tree Editor within the Business Modeler.	Global setting only

Parameter	Location	Default	Engine Mode*	Details	Tuning
IntermitCriterion	Engine > General	99	Both	<p>This parameter is used by the preprocessing and intermittent flow modules of the Analytical Engine. It specifies the minimum percentage of zero data points that a series must have to be considered intermittent.</p> <p>In this test, leading zeros may or may not be considered (depending on the setting of CutTailZeros). Trailing zeros are ignored in either case.</p> <p>In the extreme case where this parameter equals 0, all series are treated as intermittent.</p>	Can be tuned by node
IntUpdate	Engine > Adjustment	0.5	Both	<p>This parameter is used by the intermittent flow module of the Analytical Engine. It specifies the degree to which the Analytical Engine will update the fit after the last spike in history, in the case where there is decline in demand at the end of historical period.</p> <p>Use a number between 0 and 1, inclusive.</p> <p>The value 1 means that the change in fit is to be carried forward fully to the forecast.</p> <p>On the other extreme, the value 0 means that no change is to be applied.</p> <p>A value between 0 and 1 will be used as a weight for combining past and updated behavior.</p>	Can be tuned by node
L					



Parameter	Location	Default	Engine Mode*	Details	Tuning
last_date	Engine > Time	1/1/1900	Both	Last date of actual sales, to be used by the Analytical Engine and the proposit mechanism. No dates after this are used towards the forecast or the proposit calculation. If this parameter equals 1/1/1900, the system instead uses last_date_backup.	Global setting only
last_date_backup	Engine > Time		Both	Specifies a backup value to use for the last sales date, in case last_date is 1/1/1900.  Sometimes, when you load sales data, you need to change this parameter so that you can ignore a recent subset of history.  The proposit mechanism makes sure that this parameter is never later than max_sales_date.  See "max_sales_date".	Global setting only
lead	Engine > Time	12 for monthly data, 52 for yearly, 30 for daily	Both	The number of base time buckets to predict. The Analytical Engine generates a forecast for the base time buckets in the span from max_sales_date to max_sales_date + lead.  See "max_sales_date".	Global setting only

Parameter	Location	Default	Engine Mode*	Details	Tuning
LogCorrection	Engine > General	no	Both	<p>This parameter is used by the fit and forecast modules of the Analytical Engine. The issue is that logarithmic models use log-transformed demand data, which can give inaccurate results if that transformed data is near to 1 in value. In such a case, you may want to use this parameter to make an internal adjustment. Use one of the following values:</p> <p>yes: Use correct form of the expectation of a lognormal variable.</p> <p>no: Do not perform the log correction.</p>	Can be tuned by node

Parameter	Location	Default	Engine Mode*	Details	Tuning
LogLevel				<p>Controls the amount of detail that is written into the Analytical Engine log. Use one of the following values:</p> <ol style="list-style-type: none"> <li>1. Critical</li> <li>2. Error</li> <li>3. Warning</li> <li>4. Message</li> </ol> <p><b>Note:</b> This corresponds to the amount of detail that the log has contained in past releases.</p> <ol style="list-style-type: none"> <li>5. Info</li> <li>6. Detail</li> </ol> <p>This setting applies to all log groups chosen through the Engine Administrator:</p>	Global setting only

Parameter	Location	Default	Engine Mode*	Details	Tuning
LowerUpliftBound	Engine > Validation	3	PE only	<p><b>Visible only to owner.</b> Specifies the limit beyond which an uplift value is considered "exceptional." This limit is specified as a proportion of baseline. For each model, the Analytical Engine compares the absolute value of the uplift, divided by baseline, to this parameter.</p> <p>The engine discards a model (for a given forecast node) in either of two cases:</p> <ul style="list-style-type: none"> <li>• If the model generates too many exceptional uplifts (as specified by the LowerUpliftBound and AllowableExceptionsparameters).</li> <li>• If any uplift exceeds the bound given by the UpperUpliftBound parameter.</li> </ul>	Can be tuned by Node
M					
mature_age	Engine > Data Manipulation	2	Both**	<p>Controls the mature_date of each combination, which is calculated backwards from the current date using the mature_age parameter.</p> <p>A combination is young (rather than active) if it does not have any non-zero sales data on or before the mature_date.</p> <p>See prediction_status.</p>	Global setting only

Parameter	Location	Default	Engine Mode*	Details	Tuning
max_accept_num				<p>Maximum absolute value that is permitted for the forecast results. If the Analytical Engine generates a result larger than this in absolute value, it substitutes this maximum (or minimum, if applicable).</p> <p><b>Tip:</b> Make sure the forecast columns are large enough to accommodate a number of this size, and be sure to account for a possible negative sign. Errors occur if the Analytical Engine cannot write the forecast because the database columns are not large enough.</p>	Global setting only
max_fore_level	Engine > Shell	Level just below the highest fictive level	Both	<p>The maximum level on the forecast tree at which a forecast may be produced. Upon failure at this level, the NAIVE model will be used (if NaiveEnable is yes).</p> <p>In Promotion Effectiveness, this must be at or below the influence range level (IRL); see InfluenceRangeLevel.</p>	Global setting only

Parameter	Location	Default	Engine Mode*	Details	Tuning
MaxEngMemory	Engine > Proport	100	Both	<p>Specifies the maximum amount of system memory usage (in megabytes) at the end of a task, before an engine is re-initiated.</p> <p>Use this parameter to prevent the analytical engine from failing or generating errors when processing large amounts of data.</p> <p>For machines that run multiple engines, ensure that each engine has access to the amount of memory specified.</p>	Global setting only
MeanRelativeDistance	Engine > Validation	0.5	Both	<p><b>Visible only to owner.</b> This parameter is used by the fit validation module of the Analytical Engine, and it controls the sensitivity of one of the fit validation tests. A model is rejected if its MAPE (Mean Absolute Percentage Value) is greater than this threshold.</p> <p>The smaller the value, the stricter is the validation.</p>	Can be tuned by node
min_fore_level	Engine > Shell	1	Both	<p>Minimum forecast level that the engine will forecast. From that level down, the engine will split the forecast using the precalculated proportions in the mdp_matrix table.</p> <p>For PE, this must be at or above the lowest promotional level (LPL).</p>	Can be tuned by node

Parameter	Location	Default	Engine Mode*	Details	Tuning
MinLengthForDetect	Engine > Outlier and Regchange	12 for monthly data, 52 for weekly, 14 for daily	Both	This parameter is used by the preprocessing module of the Analytical Engine. It specifies the minimum number of data points needed in order for the engine to try to detect outliers and regime changes.	Can be tuned by node
ModelValidationBound		0.2	Both	Specifies the sensitivity of the test used to detect "outlier" models for a given node. Outlier models are models that do not fit well enough. This parameter is used only if model outlier detection is enabled (via the DetectModelOutliers parameter.)	Global setting only
N					
NaiveEnable	Engine > General	yes	Both	<p>Specifies what to do at the highest forecast level, upon failure of all models. Use one of the following values:</p> <ul style="list-style-type: none"> <li>no (0): Do not enable either NAIVE or Moving Average models. Do not generate a forecast.</li> <li>yes (1): Enable use of the NAIVE model.</li> <li>2 or higher: Enable use of the Moving Average model. In this case, the setting of NaiveEnable specifies the number of recent time buckets to use in calculating the moving average.</li> </ul>	Can be tuned by node

Parameter	Location	Default	Engine Mode*	Details	Tuning
need_spread	Engine > Adjustment	produce continuous forecast	Both	<p>This parameter is used by the intermittent flow module of the Analytical Engine, and it controls whether the final result should be given in the form of spikes. Use one of the following values:</p> <p>produce forecast with spikes</p> <p>produce continuous forecast</p> <p>This applies only to intermittent models.</p>	Can be tuned by node
node_forecast_details	Engine > Shell	forecast is written to node_forecast_q	Both	<p><b>Visible only to owner.</b> Specifies whether the Analytical Engine should write forecast data for each node, before splitting to lower levels. Use one of the following values:</p> <ul style="list-style-type: none"> <li>forecast is written with model details (1): The Analytical Engine writes intermediate forecast data for each node, to the NODE_FORECAST table. The table includes information on how each model was used for that node. The Analytical Engine will run more slowly because of the additional work in writing to this table.</li> <li>forecast is written to node_forecast_q (0): The Analytical Engine writes the forecast as usual.</li> </ul>	Global setting only



Parameter	Location	Default	Engine Mode*	Details	Tuning
NonNegRegr MaxTolMult				<p>Specifies the maximal multiplier to be used in order to increase the tolerance value in nonnegative regression. When you disable negative coefficients (via UseNonNegRegr) and are unable to acquire a solution, it may be helpful to increase this tolerance.</p> <p>Recommended value range: 30 - 2000</p> <p>Default value: 30</p>	Global setting only
NormalizationFactor	Engine > Data Manipulation	0	PE only	<p><b>Parameter is visible only to owner.</b> Specifies the degree of normalization to perform, if NormalizeResults is yes. Use a number from 0 to 1, inclusive. The ends of this range have the following meanings:</p> <ul style="list-style-type: none"> <li>• 1 means preserve the baseline fit. In this case, all residuals are added to the uplift.</li> <li>• 0 means that both the baseline and uplift are modified according to the normalization algorithm. This is the recommended setting.</li> </ul> <p>This normalization is applied only to historical data (where the baseline is known).</p>	Global setting only

Parameter	Location	Default	Engine Mode*	Details	Tuning
NormalizeResults	Engine > Data Manipulation	no	PE only	<p><b>Parameter is visible only to owner.</b> Specifies whether to normalize the historical engine results so that the observed baseline values are preserved. Use one of the following values:</p> <ul style="list-style-type: none"> <li>yes (1): Normalize historical engine results so that the observed baseline values are preserved. In this case, the Analytical Engine writes these results into the columns <code>fore_a_normal</code>, etc. This setting is recommended for use when historical analysis is of importance. Will cause Base + Lift to exactly match demand (<code>quantity_form</code>). The results are written in different column from base and lift to enable ease of comparison. No normalized results are available for future dates, because of the lack of normalization number; this potentially makes the connection between historical and future forecast not smooth.</li> <li>no (0): Do not perform normalization.</li> </ul>	Global setting only

Parameter	Location	Default	Engine Mode*	Details	Tuning
NumShapes	Engine > Validation	8	Both	<b>Parameter is visible only to owner.</b> Specifies the maximum number of allowed shape causal factors for the engine to use for a given node in the forecast tree. Use an integer from 0 to 8, inclusive. Applies to activity shape modeling (rather than to promotional shape modeling).	Global setting only
O					
oracle_optimization_mode	Engine > Shell	cost	Both	<b>Oracle only; visible only to owner.</b> Optimization mode of the database.	Global setting only
OutliersPercentage	Engine > Outlier and Regchange	25	Both	This parameter is used by the preprocessing module of the Analytical Engine. A set of points, suspicious as outlying, will be regarded as such only if its size does not exceed this given percentage of data.	Can be tuned by node
OutlierStdError	Engine > Outlier and Regchange	2.5	Both	<p>This parameter is used by the preprocessing module of the Analytical Engine, if gross outlier processing is enabled (via the GrossRemove parameter).</p> <p>The OutlierStdError parameter specifies the sensitivity of gross outlier detection. The greater this value, the less sensitive (more liberal) is detection of gross outliers. The value 0 is not allowed.</p>	Can be tuned by Node
P					

Parameter	Location	Default	Engine Mode*	Details	Tuning
PartitionColumnItem			Both	<p>Specifies the name of the column that partitions the data by item. This column must exist in sales_data, mdp_matrix, and (for Promotion Effectiveness) promotion_data.</p> <p>If this is null, data is not partitioned by item.</p> <p>See "Database Partitioning for the Engine".</p>	Global setting only
PartitionColumnLoc			Both	<p>Specifies the name of the column that partitions the data by location. This column must exist in sales_data, mdp_matrix, and (for Promotion Effectiveness) promotion_data.</p> <p>If this is null, data is not partitioned by location.</p> <p>See "Database Partitioning for the Engine".</p>	Global setting only
PercentOfZeros	Engine > Adjustment	0.2	Both	<p>This parameter is used by the adjustment module of the Analytical Engine, if that module is enabled (via the EnableAdjustment parameter). It specifies the maximum fraction of zero values in data beyond which no forecast adjustment is performed. Use 0.2 for 20 percent, for example.</p> <p>Enabling adjustment is not recommended, unless it is known that a change in trend happened recently, which is likely to be missed by the models.</p>	Can be tuned by node

Parameter	Location	Default	Engine Mode*	Details	Tuning
PopulationExtraFilter	Engine> Shell	Null	Both	<p>This parameter can be used to apply an extra filter on the combination population processed by the engine. This can help support incremental engine runs as well as ensure only desired combinations are forecasted.</p> <p>The default setting is null, applying no additional filter. An expected filter is on the MDP_MATRIX table with the synonym M. This filter will not require specifying the table for this table, and should only include the additional WHERE clause used by the filter.</p> <p>Example: WHERE PREDICTION_STATUS=1</p> <p>Filters including other tables must specify those other tables and join then to MDP_MATRIX. The syntax for the additional tables must begin with a comma, followed by the table name. It is not recommended that tables with a time dimension be used, including SALES_DATA. Care should be taken when defining columns for the filter, as they may exist in multiple tables, causing SQL errors. It is strongly recommended that any column referenced in the filter be prefaced by the table from which it is being retrieved.</p> <p>Example: ... ITEMS I WHERE I.ITEM_ID=M.ITEM_ID AND I.ITEM_ID &lt; 100</p>	Global Setting Only

Parameter	Location	Default	Engine Mode*	Details	Tuning
				<p>Example: ...</p> <p>T_EP_FRANCHISE T1, T_EP_RETAILER T2 WHERE T1.FRANCHISE_ID=M.FRAN CHISE_ID AND T2.RETAILER_ID=M.RETAIL ER_ID AND T1.FRANCHISE_CODE &lt; 100 AND T2.RETAILER_CLASS &gt; 5</p>	
PROMO_AG GR_LEVEL	Engine > Shell		PE only	<b>Read-only.</b> Specifies which level is used as the lowest promotional level of the forecast tree. To specify this parameter, you use the Forecast Tree Editor within the Business Modeler.	Global setting only
PromotionStartDate	Engine > Time		PE only	<b>Parameter is visible only to owner.</b> Earliest date for which promotion data can be considered reliable. The Analytical Engine ignores any promotion data before this date. This parameter applies only to combinations that have promotions.	Global setting only
PromotionSeries	Engine > Proport	null	Globally Defined	see chapter Configuring the Analytical Engine, section Split Forecast by Series	Global Setting only

Parameter	Location	Default	Engine Mode*	Details	Tuning
proport_missing	Engine > Proport	treated as zero observations	Both**	<p>Specifies how missing dates are treated. Use one of the following values:</p> <ul style="list-style-type: none"> <li>treated as zero observations: The missing dates are set equal to zero. That is, suppose that you have three months worth of data as follows: 30, null, 60. If <code>proport_missing</code> equals 0, the average of these three months is calculated as 30 (or <math>[30+0+60]/3</math>)</li> <li>treated as missing: The missing dates are assumed to have average values. Using the previous example, if <code>proport_missing</code> equals 1, the average of these three months is calculated as 45 (or <math>[30+60]/2</math>). This is mathematically equivalent to assuming that the missing month has average sales (45).</li> </ul>	Global setting only

Parameter	Location	Default	Engine Mode*	Details	Tuning
proport_spread	Engine > Proport	receive 0 proportions/global proportions	Both**	<p>Specifies how months that are missing from historical data are filled. Use one of the following values:</p> <ul style="list-style-type: none"> <li>• receive zero proportions: For each missing month, set the proportions equal to 0.</li> <li>• receive global proportions: For each missing month, set the proportions equal to glob_prop. In this case, Demantra checks the value of the proport_missing parameter and then does the following: <ul style="list-style-type: none"> <li>• If proport_missing equals 0, then missing months receive glob_prop*delta.</li> <li>• If proport_missing equals 1, then missing months receive the rolling average (glob_prop).</li> </ul> </li> <li>• receive 0 proportions/global proportions: For missing months that would have occurred after the first sale for this combination, assign 0 proportions. For months that could not occur in the range of first sale- end of sales, use glob_prop. In this case, for months that could not</li> </ul>	Global setting only



Parameter	Location	Default	Engine Mode*	Details	Tuning
				<p>have been included, Demantra checks the value of the <code>proport_missing</code> parameter and then does the following:</p> <ul style="list-style-type: none"> <li>• If <code>proport_missing</code> does not equal 1, then missing months receive the rolling average (<code>glob_prop</code>).</li> <li>• If <code>proport_missing</code> equals 1, then missing months receive <code>glob_prop*delta</code>.</li> </ul>	

Parameter	Location	Default	Engine Mode*	Details	Tuning
proport_threshold	Engine > Proport	0	Both**	<p>Specifies how many different months of the year must include data in order for Demantra to calculate proportions for the individual months (P1 - P12, PW1-PW6, etc.). Use any integer from 0 to 12, 24, inclusive.</p> <p>For each combination, the number of unique observable buckets is found (having 3 different observations of January counts as only one month).</p> <p>If not enough months have non-null values, Demantra checks the value of the proport_missing parameter and then does the following:</p> <p>If proport_missing equals 0, then missing months receive glob_prop*delta.</p> <p>If proport_missing equals 1, then missing months receive the rolling average (glob_prop).</p>	Global setting only
ProportParallelJobs	Engine > Proport	1.00	Both**	The number of parallel jobs used when running Proport calculations. This parameter's value should not exceed the number of CPUs on the database server.	Global setting only
ProportRunIntervalCycle	Engine > Proport	1.00	Both**	The number of groups that the Proport process is broken down into.	Global setting only

Parameter	Location	Default	Engine Mode*	Details	Tuning
ProportTable Label	Engine > Proport		Both**	The name of the level by which the process is broken down. The total number of members in this level is divided into equally sized groups, and one group is processed each time Proport is run.	Global setting only
Q					
Quantile	Engine > Outlier and Regchange	2.5	Both	<b>Visible only to owner.</b> This parameter is used by the validations module of the Analytical Engine, when checking the influence of outliers. It specifies a standard normal percentile for detecting outliers at a prescribed significance level.	Can be tuned by Node
quantity_for m	Engine > Data Manipulation	See details.	Both	<p><b>Visible only to owner.</b> Expression that the Analytical Engine uses to compose the historical demand from the sales_data table; the result of this expression is the data that the engine receives as input.</p> <p>This expression should return 0, null, or a numeric quantity for any date. A date with 0 is treated as if there were no sales. A date with null is treated as a missing date; in this case, the system can interpolate a value or just ignore the date.</p> <p>On Oracle, the default is as follows:</p> $\text{nvl}(\text{pseudo\_sale}, \text{actual\_quantity}) * (1 + \text{nvl}(\text{demand\_fact}, 0)) + \text{nvl}(\text{demand\_lift}, 0)$	Global setting only

Parameter	Location	Default	Engine Mode*	Details	Tuning
R					
RegimeThreshold	Engine > Outlier and Regchange	5	Both	<p>This parameter is used by the preprocessing module of the Analytical Engine. It specifies the sensitivity of regime change detection. The smaller the value, the more aggressively the engine will detect regime changes.</p> <p>This parameter is used only if regime change is enabled (via the detect_cp parameter).</p>	Can be tuned by node
RemoveResidualOutlier		0	Both	Specifies the percentage of residuals (by number) to remove before validating the fit. The residuals are sorted by size and the largest residuals are removed.	Can be tuned by node
ResetForecast	Engine > Shell	yes	Both	<p><b>Visible only to owner.</b> Specifies whether the engine should clear out previous forecast data before generating the forecast. Use one of the following values:</p> <ul style="list-style-type: none"> <li>yes: Demantra clears the previous forecast for all combinations with prediction status equal to 99. (The other combinations are left alone, because the engine will overwrite their forecast anyway.)</li> <li>no: Demantra does not clear out the previous forecast. This is less ideal but runs more quickly.</li> </ul>	Global setting only

Parameter	Location	Default	Engine Mode*	Details	Tuning
resetmat	Engine > Shell	yes	Both	<b>Visible only to owner.</b> Use one of the following values: <ul style="list-style-type: none"> <li>yes: Reset loc_node, item_id, and location_id in mdp_matrix.</li> <li>no: Do not reset these fields.</li> </ul>	Global setting only
RunFastDivider	Engine> Shell	Yes - Divide into engine tasks using lookup table	Both	This parameter is used to control whether division of forecasted combinations is done by updates to a lookup table or to MDP_MATRIX directly. When set to Yes (default), updates to the lookup table can generate substantial performance benefits for large scale environments. Setting of No will update BRANCH_ID column in MDP_MATRIX, which will take more time but is easier to query in order to determine which branch a specific combination belongs to.	Global Setting Only
RunPartialDivider	Engine > Shell	No	Both	No = before every new forecast, reallocate all combinations to tasks. Yes = only allocate new combinations to tasks	Global Setting only

Parameter	Location	Default	Engine Mode*	Details	Tuning
R unInsertUnits			Both	<p>Specifies the behavior of the INSERT_UNITS procedure, which Demantra calls at the start of an engine run. This procedure makes sure the engine has rows to write into when generating the forecast. This parameter also controls whether Demantra runs the active rolling data profiles when it runs this procedure. Use one of the following values:</p> <ul style="list-style-type: none"> <li>• 0 means that Demantra does not insert rows and does not execute the rolling data profiles.</li> <li>• 1 means that Demantra insert rows and executes the active data profiles (by running the EXECUTE_PROFILES procedure).</li> <li>• 2 means that Demantra does not insert rows, but does execute the active data profiles.</li> </ul>	Global setting only

Parameter	Location	Default	Engine Mode*	Details	Tuning
RUNMODE			Not applicable	<p><b>Read-only; parameter is visible only to owner.</b></p> <p>Specifies which version of the Analytical Engine to use.</p> <ul style="list-style-type: none"> <li>• Use 1 to specify the Promotion Effectiveness version.</li> <li>• Use 0 to run the engine in demand planning mode. This mode will not generate promotional lift. If you use this setting make sure that the LPL is the same as the minimum forecast level.</li> </ul>	Global setting only
RunPartialDiver	Engine> Shell	No - Assign new Branch to every combination participating in forecast	Both	<p>This parameter is used to control whether all combinations are assigned a branch at the beginning of a new engine run. Setting of No (default) will assign a branch ID to all combinations taking part in the engine run. Setting of Yes will assign Branch ID only to nodes that currently do not have a branch ID. Note: Setting value to No can improve engine run performance but may result in unbalanced engine tasks over time.</p>	Global Setting Only
S					

Parameter	Location	Default	Engine Mode*	Details	Tuning
SdeAnalyzeS witch	Engine > General	yes	Both	<p>Specifies how the Analytical Engine should analyze the sales_data_engine table. Use one of the following values:</p> <ul style="list-style-type: none"> <li>• yes: Use external logic to analyze this table. See "Reconfiguring the sales_data_engine Table".</li> <li>• no: Analyze the sales_data_engine table as usual.</li> </ul>	Global setting only
SdeC reateJoin	Engine > General	no	Both	<p>Specifies whether the Analytical Engine should join sales_data_engine (or its synonym) and mdp_matrix during its run. Use one of the following values:</p> <ul style="list-style-type: none"> <li>• yes: Join sales_data_engine and mdp_matrix.</li> <li>• no: Do not join these tables.</li> </ul> <p>See "Reconfiguring the sales_data_engine Table".</p>	Global setting only



Parameter	Location	Default	Engine Mode*	Details	Tuning
SdeCreateSwitch	Engine > General	internal logic	Both	<p>Specifies whether to use external logic to create the sales_data_engine table. Use one of the following values:</p> <ul style="list-style-type: none"> <li>• use internal logic (0): Create the sales_data_engine table using internal logic.</li> <li>• use external logic (1): Use external logic. If you use this option, you must rewrite the create_process_temp_table, create_object, and drop_object procedures. See "Reconfiguring the sales_data_engine Table".</li> <li>• use external logic done by engine (2). When forecasting on general levels (for example, for service parts forecasting), set to external logic engine.</li> </ul>	Global setting only
season	Engine > Time	season length	Both	<b>Read-only.</b> Season length (52 for weekly systems, 12 for monthly, 7 for daily).	Can be tuned by node
set_rb	Engine > Shell	SET transaction use rollback segment RB1	Both**	<b>Oracle 8i only; visible only to owner.</b> Set Rollback Segment command for the database. This is database dependent. See your database documentation.	Global setting only

Parameter	Location	Default	Engine Mode*	Details	Tuning
ShapeSign	Engine > Data Manipulation		Both	<p>Specifies the signs for the shape causal factors when using them in non-negative regression. Use one of the following values:</p> <ul style="list-style-type: none"> <li>0 means that after the preliminary estimation, the signs are kept as is.</li> <li>1 means that after the preliminary estimation, the shape casual factors are made positive.</li> </ul> <p>This parameter is ignored if UseNonNegRegr is set to prevent negative coefficients.</p>	Can be tuned by node
ShiftBaseCausals	Engine > Shell	0	PE only	<p><b>Parameter is visible only to owner.</b> Specifies the number of base time buckets by which the baseline causal factors should be shifted; this applies to the causal factors in the causal_factors table. Specify an integer (can be negative). The default setting (0) is recommended.</p>	Can be tuned by node

Parameter	Location	Default	Engine Mode*	Details	Tuning
ShiftDynPro moDate			PE only	<p>SQL expression that returns the number of days to add to the sales date for any given promotion; typically this is a negative number. If the resulting dates are already in the Inputs table, the Analytical Engine inserts those dates into promotion_data with is_self equal to 0.</p> <p>If this expression is null, then the default promotion dates are used.</p> <ul style="list-style-type: none"> <li>• If the expression aggregates multiple rows from promotion_data, then be sure to use an aggregate function such as DISTINCT.</li> <li>• Dates are compared to the dates in the Inputs table. If a newly generated date does not match a date in that table, then the date is deleted.</li> <li>• You can apply filters on the resulting dates, via the Promotional Causal Factor window in the Business Modeler.</li> </ul>	Global setting only

Parameter	Location	Default	Engine Mode*	Details	Tuning
ShiftPromoCausal	Engine > Shell	0	PE only	<p><b>Parameter is visible only to owner.</b> This parameter is a global setting that applies to all promotions. You may want to use ShiftDynPromoDate instead, because that gives a greater amount of control.</p> <p>This parameter specifies the number of base time buckets by which the promotional causal factors should be shifted; this applies to the causal factors in the m3_causal_factors table. Specify an integer that can be negative. For example, to make the promotional causal factors active one week after the promotions occur, specify 1 (in a weekly system).</p>	Can be tuned by node
ShiftPromoMaxValue			PE only	<p>Specifies the number of additional future time buckets to bring into history, when shifting promotions to the dates given by ShiftDynPromoDate.</p> <p>By default, the Analytical Engine considers only historical promotions and ignores any future promotions. If you shift promotion dates, that typically means you need to shift promotions that are planned for the very near future. This parameter specifies how many time buckets of the future the Analytical Engine should consider when it shifts the promotion dates.</p>	Global setting only

Parameter	Location	Default	Engine Mode*	Details	Tuning
start_date	Engine > Time	1-1-1995	Both	First sales date, the start date as it appears in the Inputs table. Can be changed according to the length of history needed for fit estimation and for the proportion mechanism. It is strongly recommended this parameter be reset to beginning actual date where history begins. See also the HistoryLength parameter.	Global setting only
start_new_run	Engine > Shell	prompt	Both	<p>Specifies whether to start a new Analytical Engine run or to perform an engine recovery. Internally, the engine records information to indicate its current processing stage. As a result, if the previous engine run did not complete, you can run recovery, and the Analytical Engine will continue from where it was interrupted.</p> <p>Use one of the following values:</p> <ul style="list-style-type: none"> <li>• yes: Always start a new run, regardless of the status of the last run.</li> <li>• no: Detect whether the previous run was complete and perform a recovery if the previous run did not complete.</li> <li>• prompt: Detect whether the previous run was complete and ask whether to perform a recovery run or a new run.</li> </ul>	Global setting only

Parameter	Location	Default	Engine Mode*	Details	Tuning
StartAverage	Engine > Data Manipulation	-12 for monthl y -52 for weekly -7 for daily	PE only	<p><b>Promotion Optimization only; parameter is visible only to owner.</b> Controls the starting date of the time span used to calculating the average baseline forecast. You specify this date relative to last_date.</p> <p>The length of this span of time is controlled by the AverageHorizon parameter.</p> <p>For information on configuring Promotion Optimization, see "Configuring Promotion Optimization for PTP".</p>	Global setting only
StdRatio	Engine > Validation	3	Both	<p>This parameter is used by the fit validation module of the Analytical Engine, and it controls the sensitivity of one of the fit validation tests. In this test, the residuals are split into two parts (earlier and later) controlled by TestPeriod. The parameter StdRatio is the maximum allowed ratio of the standard deviation of the later part to the standard deviation of the earlier part. A model is rejected if it fails the test.</p>	Can be tuned by node
T					
test_samp_len	Engine > Validation	6 for monthl y data, 26 for weekly 7 for daily	Both	<p>This parameter is used by the fit validation module of the Analytical Engine. It specifies the length of data for forecast validation.</p>	Can be tuned by node

Parameter	Location	Default	Engine Mode*	Details	Tuning
TestPeriod	Engine > Validation	6 for monthly data, 26 for weekly 7 for daily	Both	This parameter is used by the fit validation module of the Analytical Engine, and it controls the sensitivity of one of the fit validation tests. In this test, the residuals are split into two parts (earlier and later) controlled by TestPeriod. The parameter StdRatio is the maximum allowed ratio of the standard deviation of the later part to the standard deviation of the earlier part. A model is rejected if it fails the test.	Can be tuned by node
TooFew	Engine > General	2	Both	<p>This parameter is used by the preprocessing module of the Analytical Engine. It specifies the minimum number of non-zero data points that a series must have in order for the Analytical Engine to consider it model-feasible. In this test, leading zeros may or may not be considered (depending on the setting of CutTailZeros). Trailing zeros are ignored in either case.</p> <p>Must be 1 or greater.</p> <p>If the series has too few data points, the forecast failure module is run.</p>	Can be tuned by node
top_level	Engine > Shell		Both	<b>Visible only to owner; read-only.</b> Indicates the highest level of the forecast tree (the highest fictive level, HFL). This indicates the number of levels that the forecast tree contains.	Global setting only

Parameter	Location	Default	Engine Mode*	Details	Tuning
TopCoefficientLevel	Engine > Data Manipulation		PE only	<p><b>Applies only to Promotion Optimization; parameter is visible only to owner.</b></p> <p>Specifies the highest forecast tree level for which the Analytical Engine will calculate coefficients. Use any forecast tree level between BottomCoefficientLevel and the InfluenceRangeLevel, inclusive.</p> <p>For information on configuring Promotion Optimization, see "Configuring Promotion Optimization for PTP".</p>	Global setting only
TrendDampPeriod			New	<p>Used during trend detection, this parameter specifies a block of time (as a number of buckets) over which the dampening is applied. The time that contains trend is divided into blocks, as specified by this parameter. For the nth block, the Analytical Engine applies a dampening factor n times. The result is exponential dampening.</p>	Can be tuned by Node
TrendDampStep			New	<p>Used during trend detection, this parameter specifies the dampening factor, which is applied n times to the nth block of time within the trend. The result is exponential dampening. Use a value between 0 and 1, inclusive; smaller values cause dampening to happen more quickly.</p>	Can be tuned by Node



Parameter	Location	Default	Engine Mode*	Details	Tuning
TrendModelForShort			New	<p>Used during trend detection, this parameter specifies which engine model to use in order to generate the trend causal factor.</p> <ul style="list-style-type: none"> <li>• 1 means use the REGR model.</li> <li>• 2 means use the HOLT model.</li> </ul>	Can be tuned by Node
TrendOutlierRatio			New	<p>Used during trend detection, this parameter specifies how to treat outliers during model fit. It specifies a numeric weight to apply to the outliers within the long segment.</p>	Can be tuned by Node

Parameter	Location	Default	Engine Mode*	Details	Tuning
TrendPeriod	Engine > Adjustment		Both	<p>This parameter is used in two parts of the Analytical Engine.</p> <p>The adjustment module uses it as follows:</p> <ul style="list-style-type: none"> <li>• If EnableAdjustment is yes (1), then TrendPeriod specifies how far back in history the trend is measured for adjustment.</li> <li>• If zero, then no adjustment is performed. Enabling adjustment is not recommended, unless it is known that a change in trend happened recently, which is likely to be missed by the models.</li> </ul> <p>This parameter is also used by trend detection as discussed in "The Forecasting Process" on page 589. If you have disabled negative regression (via UseNonNegRegr), then it is difficult for the Analytical Engine to detect downward trends. In such cases, you should enable trend detection.</p>	Can be tuned by node
TrendPreEstimation			New	Specifies whether to perform trend detection as described in "The Forecasting Process".	Can be tuned by Node
TrendShortRatio			New	Used during trend detection, this parameter specifies how to treat outliers during model fit. It specifies a numeric weight to apply to the outliers within the short segment.	Can be tuned by Node
U					

Parameter	Location	Default	Engine Mode*	Details	Tuning
update_dead_comb			Both**	<p>Specifies whether the MANUALS_INS procedure considers dead combinations. (A combination is dead if its prediction_status setting is 99; see "Mdp_matrix".) Use one of the following values:</p> <ul style="list-style-type: none"> <li>0 means that MANUALS_INS ignores the dead combinations (this is the typical setting).</li> <li>1 means that MANUALS_INS splits aggregated data to the dead combinations and saves it to those combinations.</li> </ul>	Global setting only
UpliftThresholdMethod	Engine > Validation	1	PE only	<p>Specifies how to determine the uplift threshold:</p> <ul style="list-style-type: none"> <li>0 means use absolute values</li> <li>1 means use percent of baseline</li> </ul>	Global setting only
UpliftThresholdValue	Engine > Validation	.5	PE only	<p>Uplift and cannibalization values lower than this threshold are automatically set to null. This number must be greater than 0.</p>	Global setting only

Parameter	Location	Default	Engine Mode*	Details	Tuning
UpperUpliftBound	Engine > Validation	20	PE only	<p><b>Visible only to owner,</b> Specifies the upper allowed limit for uplifts, as a proportion of baseline. For each forecasting model, the Analytical Engine calculates the lift for each node of the forecast tree. For any given node and model, if the absolute value of the uplift is greater than this limit, then that model is not used for this node.</p> <p>The engine discards a model (for a given forecast node) in either of two cases:</p> <ul style="list-style-type: none"> <li>• If any uplift exceeds the bound given by the UpperUpliftBound parameter.</li> <li>• If the model generates too many exceptional uplifts (as specified by the LowerUpliftBound and AllowableExceptionsparameters).</li> </ul>	Can be tuned by Node

Parameter	Location	Default	Engine Mode*	Details	Tuning
UpTime	Engine > Data Manipulation	nvl(su m(nvl( UP_TI ME,1)), 1)	Both	<p>This parameter is used by the preprocessing module of the Analytical Engine. It is used to flag whether each date in sales_data should be considered a sales date or not. Use an SQL expression that returns one of the following values:</p> <ul style="list-style-type: none"> <li>• 0 (to indicate a no-sales date)</li> <li>• 1 (to indicate a date on which sales could theoretically happen)</li> </ul>	Global setting only
UpTrend	Engine > Adjustment	0.2	Both	<p>This parameter is used by the adjustment module of the Analytical Engine, if that module is enabled (via EnableAdjustment). It controls forecast adjustment for upward trend. Specifically, it represents the amount the forecast is rotated to align with recent trend in data.</p> <p>Use a value from 0 to 1, inclusive.</p> <p>Enabling adjustment is not recommended, unless it is known that a change in trend happened recently, which is likely to be missed by the models.</p>	Can be tuned by node

Parameter	Location	Default	Engine Mode*	Details	Tuning
UseBusinessFilter	Engine > Data Manipulation	no	Both	<p>Specifies whether the Analytical Engine distinguishes business and non-business days. Use one of the following values:</p> <ul style="list-style-type: none"> <li>• yes: The Analytical Engine uses only business days (as indicated by business_day_filter series in the Inputs table).</li> <li>• no: The Analytical Engine uses all days.</li> </ul>	Global setting only
UseEnvelope					

Parameter	Location	Default	Engine Mode*	Details	Tuning
UseExternalS DUpdate	Engine > Shell	no	Both	<p><b>Visible only to owner.</b> Specifies how to update the sales_data table with the current forecast. Use one of the following values:</p> <p>yes: Use an external procedure (<code>create_process_temp_table</code>).</p> <p>no: Use an internal dynamic procedure.</p> <p>The <code>create_process_temp_table</code> procedure is a template that creates the dynamic stored procedure that will be executed by the engine for the update. By default, this procedure creates the same SP that as the engine creates, but this can be overridden. The interface to this SP is as follows:</p> <ul style="list-style-type: none"> <li>• <code>is_proc_name</code> (VARCHAR2) specifies the name of the dynamic SP that the engine will execute.</li> <li>• <code>is_tmp_tbl</code> (VARCHAR2) specifies the temptable name.</li> <li>• <code>is_fore_col</code> (VARCHAR2) specifies the column name in sales_data that will be updated with the new forecast.</li> <li>• <code>is_last_date</code> (VARCHAR2) specifies a date to update mdp_matrix with.</li> </ul>	Global setting only

Parameter	Location	Default	Engine Mode*	Details	Tuning
usemodelspe rnode	Engine > Data Manipulation		Both	Specifies whether you can specify the forecasting models to use for specific nodes in the forecast tree, via the File > Analytics menu option in Promotion Effectiveness.	Global setting only
UseNonNeg Regr	Engine > Validation	yes	Both	<b>Visible only to owner.</b> Specifies whether to use non negative constraint estimation for all regression-based engine models. Use one of the following values: <ul style="list-style-type: none"> <li>yes: The coefficients are prevented from being negative.</li> <li>no</li> </ul>	Can be tuned by node
UseParamsPe rNode	Engine > Data Manipulation		Both	Specifies whether you can specify the engine parameters to use for specific nodes in the forecast tree, via the File > Analytics menu option in Promotion Effectiveness.	Global setting only
UseWeighted Regression		0	Both	Specifies whether the Analytical Engine applies a weight to each observation when fitting each model. <ul style="list-style-type: none"> <li>If this parameter is set to 1 (yes), the OBS_ERROR_STD field (in sales_data) specifies the weights for each observation.</li> <li>If this parameter is 0 (no), that field is ignored.</li> </ul>	Can be tuned by Node
W					



Parameter	Location	Default	Engine Mode*	Details	Tuning
WriteIntermediateResults	Engine > Shell	no	Both	<p><b>Applies only to the desktop products; parameter is visible only to owner.</b> Specifies whether to enable the advanced analytics function, which is available only on the desktop. Use one of the following values:</p> <ul style="list-style-type: none"> <li>yes: Retain intermediate results (coefficients for causal factors) to enable advanced analytics. The results are written to the INTERM_RESULTS table. This information includes the coefficients for each model, and the weight of each model in the forecast.</li> </ul> <p><b>Warning:</b> The Analytical Engine will run much more slowly.</p> <ul style="list-style-type: none"> <li>no</li> </ul>	Can be tuned by node

Parameter	Location	Default	Engine Mode*	Details	Tuning
WriteMissing DatesUplift	Engine > Shell	no	PE only	<p><b>Parameter is visible only to owner.</b> Specifies whether to write uplifts for dates that are missing from sales_data. Use one of the following values:</p> <ul style="list-style-type: none"> <li>yes: The Analytical Engine writes uplifts for any dates where it calculates them, even if no sales occurred. However, the uplifts will add up to the total uplift calculated by the engine.</li> <li>no: The Analytical Engine writes uplifts only for dates that have sales. This means that the uplifts will not necessarily add up to the total uplift.</li> </ul>	Global setting only
<p>* <b>PE only</b> means PE mode only; <b>Both</b> means both PE and DP modes. ** This parameter is not used directly by the engine and thus is available for use with either engine mode.</p>					

## Database Parameters for Tuning Engine Performance

The following parameters provide more advanced controls of how Demantra executes specific engine processes and allow you to tune the engine based on your implementation's unique characteristics. These parameters can be set in the INIT\_PARAMS\_% tables.

**Warning:** Using these parameters to improve engine performance depends almost entirely upon the expertise of the implementer. System issues may result if they are not used correctly. Oracle does not recommend using these parameters unless you have been trained and have previous experience tuning the analytical engine

Parameter	Details
-----------	---------

---

DBHintBranchDivision	<p>This parameter applies hints to Engine DB statements that divide MDP_MATRIX combinations among forecast engine tasks. When the hint references to table MDP_MATRIX, use #DYNTABLE1# instead. The engine will automatically replace with MDP_MATRIX. For example, to run four parallel processes when executing branch division, set this parameter to + parallel(#DYNTABLE1# 4).</p>
DBHintAggriSQL	<p>This parameter applies hints to Engine DB statements that aggregate data from the SALES_DATA table. When the hint references to table SALES_DATA, use #DYNTABLE1# instead. The engine will automatically replace with SALES_DATA. For example, to run four parallel processes when executing aggregation statements, set this parameter to + parallel(#DYNTABLE1# 4).</p>
DBHintLoadActual	<p>This parameter applies hints to the Engine DB statement that retrieves the lowest level historical data during an analytical engine run. When referencing a table use #DYNTABLE1# or #DYNTABLE2# to designate the table.</p> <p>The Engine automatically replaces dynamic strings with the relevant table name or alias. #DYNTABLE1# references SALES_DATA and #DYNTABLE2# references MDP_MATRIX. For example, to run four parallel processes on MDP_MATRIX and six parallel processes on SALES_DATA when querying lowest level information, set this parameter to + parallel(#DYNTABLE1# 6) parallel(#DYNTABLE2# 4).</p>

---

---

DBHintCreatePDE

This parameter applies hints to the Engine DB statements that create the temporary promotion data engine tables during an analytical engine run. When referencing a table, use #DYNTABLE1#, #DYNTABLE2# or #DYNTABLE3# to designate the table.

The Engine automatically replaces dynamic strings with the relevant table name or alias. Use #DYNTABLE1# to reference MDP\_MATRIX, #DYNTABLE2# to reference PROMOTION and #DYNTABLE3# to reference PROMOTION\_DATA. For example, to run four parallel processes on MDP\_MATRIX and six parallel processes on PROMOTION\_DATA when creating the PDE table, set this parameter to +  
parallel(#DYNTABLE1# 4)  
parallel(#DYNTABLE3# 6).

---

---

## Theoretical Engine Models

This chapter contains reference information for the theoretical models that the Analytical Engine uses.

This chapter covers the following topics:

- Introduction
- Flags on Causal Factors
- ARIX
- ARLOGISTIC
- ARX
- BWINT
- CMREGR
- DMULT
- ELOG
- FCROST
- HOLT
- ICMREGR
- IREGR
- LOG
- LOGISTIC
- Moving Average
- MRIDGE
- NAIVE
- REGR

## Introduction

**Note:** Oracle provides two different modes for the Analytical Engine:

- In PE mode, the engine is suitable for use with Promotion Effectiveness.
- In DP mode, the engine is suitable for use in demand planning applications.

For each model, this chapter indicates which engine modes that model can be used with.

## Flags on Causal Factors

You use the Business Modeler to apply the following flags to the causal factors; see "Configuring Global and Local Causal Factors" and "Configuring Promotional Causal Factors":

Flag*	Meaning
short	For use by the short models (BWINT, IREG, LOGREG, LOGISTIC, and REG). These models use all causal factors that they are given.
long	For use by the long models (ARLOGISTIC, CMREG, ELOG, ICMREG, and MRIDGE). These models examine all the causal factors they are given, but choose the ones that give the best results.
non-seasonal	For use by the non -seasonal models (ARIX and ARX). The only causal factors that should be flagged as non-seasonal are ones that are not a predictable function of time. For example, price varies with time, but randomly, so price should be flagged as non-seasonal.
multiplicative group 1	For use only by the DMULT model. If you are using this model, each causal factor should use one of these flags.
multiplicative group 2	See "DMULT".
*Name of flag as displayed in the Causal Factors screen or in the Promotional Causal Factors screen.	

Models not listed here use other mechanisms to choose their causal factors or do not use

causal factors at all.

## ARIX

ARIX includes integrated auto-regression terms at lag 1 and an unknown seasonal lag  $k$ , and linear causal factors.

The value of  $k$  is chosen from set of possible seasonal indexes to produce the best fit. Causal factors include the constant and events (without seasonal causal factors and without time).

### Availability

ARIX can be used with the following engine modes:

Engine Mode	Supported?
PE mode	Yes*
DP mode	Yes
*The ARIX model is never used on promotional nodes. See "Summary of the Forecasting Process".	

### Causal Factors Used by This Model

ARIX uses the non-seasonal causal factors; see "Flags on Causal Factors".

### Parameters Used by This Model

Parameter	Default	Description
Possible Season*	<b>For daily data:</b> 2, 3, 4, 5, 6, 7, 14, 30, 31, 90, 91, 92, 182, 365 <b>For weekly data:</b> 2, 4, 5, 13, 14, 26, 52 <b>For monthly data:</b> 3, 6, 12.	A vector of possible seasonal patterns of the series.  The parameter is of type vector (other parameters are defined as double in PARAM_TYPE column), with an increasing index (PARAM_INDEX) for each new PARAM_VALUE.
UseNonNegRegr	no	Specifies whether to constrain the regression coefficients to nonnegative values, within the core least squares estimation.

Parameter	Default	Description
AllowNegative	no	Specifies whether negative values of fit and forecast are allowed. If negative values are not allowed, then any non-positive fitted and forecasted values are set to zero.
*This parameter is model-specific and is not displayed in the Business Modeler; see the Parameters table.		

The ARIX parameters also apply to the ARX model.

## ARLOGISTIC

ARLOGISTIC is an extension of the LOGISTIC model and includes auto-regression and logistic regression terms.

### Availability

ARLOGISTIC can be used with the following engine modes:

Engine Mode	Supported?
PE mode	No (disable model if using this mode)
DP mode	Yes

### Causal Factors Used by This Model

ARLOGISTIC uses the long causal factors; see "Flags on Causal Factors".

### Parameters Used by This Model

ARLOGISTIC uses the same parameters as LOGISTIC; see "Parameters Used by This Model".

## ARX

This model includes auto-regression terms at lag 1 and an unknown seasonal lag k, and linear causal factors. The value of k is chosen from set of possible seasonal indexes to produce the best fit. Causal factors include the constant and events (without seasonal causal factors and without time).

### Availability

ARX can be used with the following engine modes:



Engine Mode	Supported?
PE mode	Yes*
DP mode	Yes
*The ARX model is never used on promotional nodes. See "Summary of the Forecasting Process".	

#### Causal Factors Used by This Model

ARX uses the non-seasonal causal factors; see "Flags on Causal Factors".

#### Parameters Used by This Model

ARX uses the same parameters as ARIX; see "ARIX".

## BWINT

BWINT (the Multiplicative Regression-Winters model) runs multiplicative regression on the causal factors, then exponentially smooths the resulting residuals in HOLT manner and then runs multiple regression of the smoothed residuals. BWINT models trend, seasonality and causality.

#### Availability

BWINT can be used with the following engine modes:

Engine Mode	Supported?
PE mode	No (disable model if using this mode)
DP mode	Yes

#### Causal Factors Used by This Model

BWINT uses the short causal factors; see "Flags on Causal Factors".

#### Parameters Used by This Model

Parameter	Default	Description
Alpha*	0.1	The manually set level renovation coefficient, valid only when OptimizedBwint* = 0.

Parameter	Default	Description
Gamma*	0.3	The manually set trend renovation coefficient, valid only when OptimizedBwint* = 0.
OptimizedAlphaIter*	3	The number of values on the Alpha grid for parameters optimization.
OptimizedBwint*	0	Specifies whether the parameter values (Alpha & Gamma) of the Holt procedure used here are to be optimized (1) or preset (0).
OptimizedGammaIter*	10	The number of values on the Gamma grid for parameters optimization.
Phi*	0.9	The trend damping coefficient, always set manually.
UseNonNegRegr	no	Specifies whether to constrain the regression coefficients to nonnegative values, within the core least squares estimation.
AllowNegative	no	Specifies whether negative values of fit and forecast are allowed. If negative values are not allowed, then any non-positive fitted and forecasted values are set to zero.

\*This parameter is model-specific and is not displayed in the Business Modeler; see the Parameters table.

## CMREGR

CMREGR (the Markov Chain Monte-Carlo model) fits to data an assortment of linear functions of the form:  $\text{Series} = \text{Causals} * \text{Coeff} + \text{Resid}$ .

Where:

- Causals are various subsets of causal factors, chosen by a random process from all possible combinations of factors.

The first set of causal factors consists of a collection of factors along with the lagged time series. Then, for a given length, a chain of that length is generated, and that path of that Markov chain is traveled. The states of the chain are subsets of factors, the transition probabilities for neighboring states are based on the ratio of BICs (Bayesian Information Criteria) and are zero for non- neighboring states. Neighboring states are

states that differ only by one member. At each pass, a new factor is chosen randomly. If the current model does not contain this factor, it joins, with the calculated transition probability, the group to form the next model. Thus, the greater the improvement in the model (as measured by BIC), the more probability has the model to be employed. If the current model already contains this factor, then, with the calculated transition probability, it leaves the group.

Also, a special causal factor Lag is used; this is merely the original series lagged back by one time period. When the procedure finds this causal factor useful for modeling, the meaning is that there is a significant autoregressive component in the data, which indicates the presence of random trends. If the influence of Lag is dominant over other factors, which is indicated by a large Lag coefficient, the fit will inhere the lagging effect and when plotted on the same graph as the original series, will seem to "echo" previous observations. This means that the model was unable to pick up any systematic behavior in the series, and the best it can do is to highly correlate fitted values with lagged data.

### Availability

CMREGR can be used with the following engine modes:

Engine Mode	Supported?
PE mode	Yes
DP mode	Yes

### Causal Factors Used by This Model

CMREGR uses the long causal factors; see "Flags on Causal Factors".

### Parameters Used by This Model

Parameter	Default	Description
Reset_Seed*	1	Specifies whether to reset the seed for random numbers generation at each run or simulation. If the seed is not reset, there will be different results for each run; also the simulation results will differ from batch results. 1= reset_seed; 0 = do not reset seed.  Theoretically the model assumes that the seed is not reset.
ChainLength*	500	Number of models considered for averaging.

Parameter	Default	Description
Need_Lag*		Specifies whether to use the Lag as a causal factor. Lag - the previous actual observation explains the next one.
UseNonNegRegr	no	Specifies whether to constrain the regression coefficients to nonnegative values, within the core least squares estimation.
AllowNegative	no	Specifies whether negative values of fit and forecast are allowed. If negative values are not allowed, then any non-positive fitted and forecasted values are set to zero.
UseEnvelope	no	Specifies whether Demantra will use the envelope function described in "Causal Factor Testing (Envelope Function)".
ENVELOPE_RESET_SEED*	0	Specifies whether to reset the randomization seed for the envelope function, which evaluates different sets of causal factors for different engine models.
ENVELOPE_CHAIN_LENGTH*	50	Specifies the number of variations of causal factors to try, for each model.
BestOrMix*	0	Specifies whether to use the best set of causal factors (0) or to use a mix of the causal factors (1).
*This parameter is model-specific and is not displayed in the Business Modeler; see the Parameters table.		

## DMULT

DMULT, the Multiplicative Multi-Seasonal Regression model, divides causal factors into two groups and combines them in a multiplicative linear function of the following form:

*(sum of values in causal factor group 1) \* (sum of values in causal factor group 2)*

This function can be used, for example, to combine daily and monthly seasonality.

### Availability

DMULT can be used with the following engine modes:

Engine Mode	Supported?
PE mode	Yes
DP mode	Yes

### Causal Factors Used by This Model

When you define causal factors and promotional causal factors, the Causal Factors screen and the Promotional Causal Factors screen enable you to place each factor into multiplicative group 1 or multiplicative group 2.

These options correspond to the DAILY\_VAL (multiplicative group 1) and MONTHLY\_VAL (multiplicative group 2) columns in the causal\_factors and the promotional\_causal\_factors tables.

Typically, one group contains daily causal factors such as the days of the week D1,D2,...,D7. The other group contains the remaining causal factors. Each group should include at least one causal factor, and each causal factor should be in only one group.

### Parameters Used by This Model

Parameter	Default	Description
UseNonNegRegr	no	Specifies whether to constrain the regression coefficients to nonnegative values, within the core least squares estimation.
AllowNegative	no	Specifies whether negative values of fit and forecast are allowed. If negative values are not allowed, then any non-positive fitted and forecasted values are set to zero.
MAX_ITERATIONS*	3	Specifies the maximum number of iterations used by this model. This parameter must be a whole number greater than or equal to 3. If it less than 3, the Analytical Engine uses the value 3.
SET2_COEFF_INI*	0	Specifies the initial values for the coefficients in multiplicative group 2.  The default is 0, which means that the initial values for these is zero, except for the coefficient for the constant causal factor.

Parameter	Default	Description
* This parameter is model-specific and is not displayed in the Business Modeler; see the Parameters table.		

## ELOG

ELOG (the Logarithmic CMREGR model) performs the CMREGR procedure on the log-transformed time series.

As with the CMREGR model, this model uses a special causal factor (Lag); see "CMREGR".

### Availability

ELOG can be used with the following engine modes:

Engine Mode	Supported?
PE mode	Yes
DP mode	Yes

### Causal Factors Used by This Model

ELOG uses the long causal factors; see "Flags on Causal Factors".

### Parameters Used by This Model

Parameter	Default	Description
ChainLength*	500	Length of the generated Markov Chain, that is number of models considered for averaging.
need_lag*		Specifies whether to use the Lag as a causal factor. Lag - the previous actual observation explains the next one.

Parameter	Default	Description
reset_seed*	1	<p>Specifies whether to reset the seed for random numbers generation at each run or simulation. If the seed is not reset, there will be different results for each run; also the simulation results will differ from batch results.</p> <p>1= reset_seed; 0 = do not reset seed.</p> <p>Theoretically the model assumes that the seed is not reset.</p>
LogCorrection	1	Specifies whether to use (1) or not (0) the correct form of the expectation of a lognormal variable.
UseNonNegRegr	no	Specifies whether to constrain the regression coefficients to nonnegative values, within the core least squares estimation.
AllowNegative	no	Specifies whether negative values of fit and forecast are allowed. If negative values are not allowed, then any non-positive fitted and forecasted values are set to zero.
UseEnvelope	no	Specifies whether Demantra will use the envelope function described in "Causal Factor Testing (Envelope Function)".
ENVELOPE_RESET_SEED*	0	Specifies whether to reset the randomization seed for the envelope function, which evaluates different sets of causal factors for different engine models.
ENVELOPE_CHAIN_LEN GTH*	50	Specifies the number of variations of causal factors to try, for each model.
BestOrMix*	0	Specifies whether to use the best set of causal factors (0) or to use a mix of the causal factors (1).
*This parameter is model-specific and is not displayed in the Business Modeler; see the Parameters table.		

## FCROST

FCROST (the Croston Model for Intermittent Demand) is useful for intermittent demand, which can be viewed as the demand by a distributor that supplies the product to end customers. What is visible to the demand planner is the bulk demand by the distributor, while the periodic demand of retailers is unknown. Thus, the quantities most probably reflect replenishment orders, rather than demand. Visually the data consists of peaks of random height with random intervals between the peaks.

This model is useful for data involving substantial number of zeros, and is particularly relevant for forecasting demand of slow moving parts. The model utilizes the Holt procedure for forecasting both quantities and inter-event times.

### Availability

FCROST can be used with the following engine modes:

Engine Mode	Supported?
PE mode	No (disable model if using this mode)
DP mode	Yes

### Causal Factors Used by This Model

None.

### Parameters Used by This Model

Parameter	Default	Description
AlphaQ*	0.1	Level innovation coefficient for quantities, manually set.
AlphaT *	0.1	Level innovation coefficient for inter-event times, always manually set.
GammaQ*	0.3	Trend innovation coefficient for quantities, manually set.
GammaT*	0.3	Trend innovation coefficient for inter-event times, always manually set.



Parameter	Default	Description
OptimizedAlphaIter*	3	The number of values on the Alpha grid for parameter optimization.
OptimizedFcrost*	0	For forecasting the inter-event times only. Parameter specifies whether the parameter values (AlphaQ & GammaQ) of the quantities-forecasting Holt procedure used here are to be optimized (1) or preset (0).  For forecasting the inter-event times only,
OptimizedGammaIter*	10	The number of values on the Gamma grid for parameter optimization.
Phi*	0.9	Trend damping coefficient for inter-event times, always manually set.
PhiQ*	0.9	Trend damping coefficient for quantities, always manually set.
AllowNegative	no	Specifies whether negative values of fit and forecast are allowed. If negative values are not allowed, then any non-positive fitted and forecasted values are set to zero.
*This parameter is model-specific and is not displayed in the Business Modeler; see the Parameters table.		

## HOLT

HOLT (the Double Exponential Smoothing model) provides realization for the Holt damped two-parameter exponential smoothing algorithm. The forecast is a projection of the current level estimate shifted by damped trend estimate. The level estimates are computed recursively from data as weighted averages of the current series value and the value of the previous one-step-ahead forecast. The trend (change of level) estimates are computed as weighted averages of the currently predicted level change and damped previously predicted trend. The weights and the damping coefficient are either user-supplied or can be optimized. If the optimization of parameters is chosen, they will be set so that the MAPE (Mean Square Percentage Error) is minimized.

The HOLT model is suitable for modeling time series with a slowly changing linear trend. It is usually used only to model short series (for example, 52 or fewer data points for a weekly system).

## Availability

HOLT can be used with the following engine modes:

Engine Mode	Supported?
PE mode	Yes*
DP mode	Yes
*The HOLT model is used on promotional nodes only if no other models can be used. See "Summary of the Forecasting Process".	

## Causal Factors Used by This Model

None.

## Parameters Used by This Model

Parameter	Default	Description
Alpha*	0.1	The manually set level renovation coefficient, valid only when OptimizedHolt* = 0.
Gamma*	0.3	The manually set trend renovation coefficient, valid only when OptimizedHolt* = 0.
OptimizedAlphaIter*	3	The number of values on the Alpha grid for parameters optimization.
OptimizedGammaIter*	10	The number of values on the Gamma grid (default) for parameters optimization.
OptimizedHolt*	0	Specifies whether the parameter values (Alpha & Gamma) are to be optimized (1) or preset (0).
Phi*	0.9	The trend damping coefficient, always set manually.
AllowNegative	no	Specifies whether negative values of fit and forecast are allowed. If negative values are not allowed, then any non-positive fitted and forecasted values are set to zero.

Parameter	Default	Description
*This parameter is model-specific and is not displayed in the Business Modeler; see the Parameters table.		

## ICMREGR

ICMREGR (the Intermittent CMREGR model) is an extension of both CMREGR and IREGR models.

### Availability

ICMREGR can be used with the following engine modes:

Engine Mode	Supported?
PE mode	No (disable model if using this mode)
DP mode	Yes

### Causal Factors Used by This Model

ICMREGR uses the long causal factors; see "Flags on Causal Factors".

### Parameters Used by This Model

Parameter	Default	Description
ChainLength*	500	Length of the generated Markov Chain, that is the number of models considered for averaging.
need_lag*		Specifies whether to use the Lag as a causal factor. Lag - the previous actual observation explains the next one.

Parameter	Default	Description
reset_seed*	1	<p>Specifies whether to reset the seed for random numbers generation at each run or simulation. If the seed is not reset,</p> <p>there will be different results for each run; also the simulation results will differ from batch results. 1= reset_seed; 0 = do not reset seed.</p> <p>Theoretically the model assumes that the seed is not reset.</p>
UseNonNegRegr	no	Specifies whether to constrain the regression coefficients to nonnegative values, within the core least squares estimation.
AllowNegative	no	Specifies whether negative values of fit and forecast are allowed. If negative values are not allowed, then any non-positive fitted and forecasted values are set to zero.
*This parameter is model-specific and is not displayed in the Business Modeler; see the Parameters table.		

## IREGR

IREGR (the Intermittent Regression model) is useful because the Croston model fails to consider the obvious interdependency between quantities and times between occurrences of demands in intermittent series. Moreover, due to the nature of the Holt model used by Croston, causalities and seasonality are not modeled. IREGR spreads the data into a continuous series and fits to it a regression model with unequal variances. The resulting fit and forecast may be lumped back to form spikes, after being processed by the Bayesian blending procedure.

### Availability

IREGR can be used with the following engine modes:

Engine Mode	Supported?
PE mode	No (disable model if using this mode)
DP mode	Yes

### Causal Factors Used by This Model

IREGR uses the short causal factors; see "Flags on Causal Factors".

### Parameters Used by This Model

Parameter	Default	Description
UseNonNegRegr	no	Specifies whether to constrain the regression coefficients to nonnegative values, within the core least squares estimation.
AllowNegative	no	Specifies whether negative values of fit and forecast are allowed. If negative values are not allowed, then any non-positive fitted and forecasted values are set to zero.

## LOG

LOG (the Multiple Logarithmic Regression model) performs a logarithmic regression. Using logarithms is often a good way to find linear relationships in non-linear data.

This model fits to data a linear function of the form:

$$\ln(\text{Series} + \text{ones} * \text{Shift}) = \text{Causals} * \text{Coeff} + \text{Resid}$$

Where:

- Resid is the vector of residuals.
- ones is a column vector of ones.
- Shift is a calculated value to shift the series away from non-positive values, before the logarithmic transformation.

Forecast values are obtained by back-transforming the projected regression, while considering the theoretical form of the expectation of a log-normal random variable

### Availability

LOG can be used with the following engine modes:

Engine Mode	Supported?
PE mode	Yes
DP mode	Yes

### Causal Factors Used by This Model

LOG uses the short causal factors; see "Flags on Causal Factors".

### Parameters Used by This Model

Parameter	Default	Description
LogCorrection	1	Specifies whether to use (1) or not (0) the correct form of the expectation of a lognormal variable.
UseNonNegRegr	no	Specifies whether to constrain the regression coefficients to nonnegative values, within the core least squares estimation.
AllowNegative	no	Specifies whether negative values of fit and forecast are allowed. If negative values are not allowed, then any non-positive fitted and forecasted values are set to zero.
UseEnvelope	no	Specifies whether Demantra will use the envelope function described in "Causal Factor Testing (Envelope Function)".
ENVELOPE_RESET_SEED*	0	Specifies whether to reset the randomization seed for the envelope function, which evaluates different sets of causal factors for different engine models.
ENVELOPE_CHAIN_LENGTH*	50	Specifies the number of variations of causal factors to try, for each model.
BestOrMix*	0	Specifies whether to use the best set of causal factors (0) or to use a mix of the causal factors (1).
*This parameter is model-specific and is not displayed in the Business Modeler; see the Parameters table.		

## LOGISTIC

LOGISTIC runs logistic regression on the causal factors.

### Availability

LOGISTIC can be used with the following engine modes:

Engine Mode	Supported?
PE mode	Yes
DP mode	Yes

### Causal Factors Used by This Model

LOGISTIC uses the short causal factors; see "Flags on Causal Factors".

### Parameters Used by This Model

The LOGISTIC parameters also apply to the ARLOGISTIC model.

Parameter	Default	Description
Potential*	1.5	Specifies the upper bound of market effort effect, as a multiple of maximum historical sales.
UseNonNegRegr	no	Specifies whether to constrain the regression coefficients to nonnegative values, within the core least squares estimation.
AllowNegative	no	Specifies whether negative values of fit and forecast are allowed. If negative values are not allowed, then any non-positive fitted and forecasted values are set to zero.
*This parameter is model-specific and is not displayed in the Business Modeler; see the Parameters table.		

## Moving Average

The Moving Average model considers the most recent time buckets, computes the average, and uses that for the forecast, resulting in a flat line. This forecast is generally suitable only in the near future.

This model is provided as a possible substitute for the NAIVE model, for use when all other models have failed. It does not generally interact well with other models and so is recommended only for use if no other forecast models have worked.

See "Forecast Failure", and also see "NAIVE".

### Availability

The Moving Average model can be used with the following engine modes:

Engine Mode	Supported?
PE mode	Yes (no lift is generated, however)
DP mode	Yes

### Causal Factors Used by This Model

None.

### Parameters Used by This Model

Parameter	Default	Description
NaiveEnable		<p>Specifies what to do at the highest forecast level, upon failure of all models.</p> <ul style="list-style-type: none"><li>no (0): Do not enable either NAIVE or Moving Average models. Do not generate a forecast.</li><li>yes (1): Enable use of the NAIVE model.</li><li>2 or higher: Enable use of the Moving Average model. In this case, the setting of NaiveEnable specifies the number of recent time buckets to use in calculating the moving average.</li></ul>

## MRIDGE

MRIDGE (the Modified Ridge Regression model) produces regression coefficients of moderate magnitude, thus assuring that lifts associated with events are of moderate size. This is equivalent to imposing a set of constraints on the coefficients in a spherical region centered at zero. In the literature, this model is of the shrinkage family.

### Availability

MRIDGE can be used with the following engine modes:



Engine Mode	Supported?
PE mode	Yes
DP mode	Yes

### Causal Factors Used by This Model

MRIDGE uses the long causal factors; see "Flags on Causal Factors".

### Parameters Used by This Model

Parameter	Default	Description
RIDGEK*	1	The larger the value of RIDGEK, the more shrinkage occurs. When RIDGEK=0, the model is equivalent to REGR.
METRIC NORM*	2	Chooses the norm for scaling the input causal factors.
UseNonNegRegr	no	Specifies whether to constrain the regression coefficients to nonnegative values, within the core least squares estimation.
AllowNegative	no	Specifies whether negative values of fit and forecast are allowed. If negative values are not allowed, then any non-positive fitted and forecasted values are set to zero.
UseEnvelope	no	Specifies whether Demantra will use the envelope function described in "Causal Factor Testing (Envelope Function)".
BestOrMix*	0	Specifies whether to use the best set of causal factors (0) or to use a mix of the causal factors (1).

\*This parameter is model-specific and is not displayed in the Business Modeler; see the Parameters table.

## NAIVE

The NAIVE model is used only at the highest forecast level, and is used only if all other

models (including HOLT) have failed. See "Forecast Failure", and also see "Moving Average".

It uses a simple averaging procedure.

### Availability

NAIVE can be used with the following engine modes:

Engine Mode	Supported?
PE mode	Yes (no lift is generated, however)
DP mode	Yes

### Causal Factors Used by This Model

None.

### Parameters Used by This Model

Parameter	Default	Description
NaiveEnable		<p>Specifies what to do at the highest forecast level, upon failure of all models.</p> <p>no (0): Do not enable either NAIVE or Moving Average models. Do not generate a forecast.</p> <p>yes (1): Enable use of the NAIVE model.</p> <p>2 or higher: Enable use of the Moving Average model. In this case, the setting of NaiveEnable specifies the number of recent time buckets to use in calculating the moving average.</p>
AllowNegative	no	<p>Specifies whether negative values of fit and forecast are allowed. If negative values are not allowed, then any non-positive fitted and forecasted values are set to zero.</p>

**Note:** When generating naive forecast at the highest forecast level, one of two methods are used. If Holt was not attempted for this node, a simplified version of the Holt model will be used and the combination will be marked with the letter T. If Holt was previously attempted a moving average based model is used instead and the node is marked with N for Naive.

## REGR

REGR (the Multiple Regression model) fits to data a linear function of the form:

$$\text{Series} = \text{Causals} * \text{Coeff} + \text{Resid}$$

Where:

- Causals is a matrix with the independent variables (causal factors) as its columns.
- Coeff is a column vector of regression coefficient.
- Resid are the (additive) residuals (errors).

Using this additive model, we are assuming that a linear relationship exists. The dependent variable is linearly related to each of the independent variables.

The regression parameters estimates are obtained by using the method of least square error.

Regression coefficients that are not statistically significant are identified by special tests and assigned the value 0.

**Note:** All regression-based models use REGR implicitly.

### Availability

REGR can be used with the following engine modes:

Engine Mode	Supported?
PE mode	Yes
DP mode	Yes

### Causal Factors Used by This Model

REGR uses the short causal factors; see "Flags on Causal Factors".

### Parameters Used by This Model

Parameter	Default	Description
UseNonNegRegr	no	Specifies whether to constrain the regression coefficients to nonnegative values, within the core least squares estimation.

Parameter	Default	Description
AllowNegative	no	Specifies whether negative values of fit and forecast are allowed. If negative values are not allowed, then any non-positive fitted and forecasted values are set to zero.
UseEnvelope	no	Specifies whether Demantra will use the envelope function described in "Causal Factor Testing (Envelope Function)".
ENVELOPE_RESET_SEED*	0	Specifies whether to reset the randomization seed for the envelope function, which evaluates different sets of causal factors for different engine models.
ENVELOPE_CHAIN_LENGTH*	50	Specifies the number of variations of causal factors to try, for each model.
BestOrMix*	0	Specifies whether to use the best set of causal factors (0) or to use a mix of the causal factors (1).
* This parameter is model-specific and is not displayed in the Business Modeler; see the Parameters table.		

# Part 7

---

## Administration



---

## Administering Demantra

This chapter briefly introduces the tasks that the system administrator for Demantra would perform. It also lists all the URLs that Demantra uses.

This chapter covers the following topics:

- Keeping the System Running
- Periodic Maintenance Tasks
- Data Cleanup Script
- Demantra URLs
- Log Files and Tables
- Illegal Characters in Demantra
- Required Third-Party Software

### Keeping the System Running

Depending on how Demantra is configured, it needs some or all of the following items to be running:

Component	When needed	Details
Database	Always	Consult Oracle database documentation.
Web server*	If solution uses any Web-based components	See documentation for the Web server.
Workflow Engine	If workflows are being used	See "Managing Workflows".

Component	When needed	Details
Database procedures listed in "Recommended Procedure Scheduling"	Always	Usually run from within workflows; see "Managing Workflows"
Possible other background processes	Varies	Contact the implementors of your Demantra system.
*For best performance, run the server without any logging.		

You should also make sure you have details on the following topics:

- The specific automated processes that the solution uses
- How often the Analytical Engine runs
- Any workflows that are in the solution
- How many components have been defined and who owns them

## Periodic Maintenance Tasks

You may periodically need to make adjustments in the following areas:

Area	Tool used	See
Maintaining users and user groups	Business Modeler	"Managing Security"
Maintaining security of menus in all Web-based products	Collaborator Workbench Administrator	"Managing Security"
Managing workflows	Workflow Manager	"Managing Workflows"
Changing ownership of worksheets and deleting unused worksheets	Business Modeler	"Managing Worksheets"

**Note:** For information on tuning the performance of the Analytical



Engine, see "Tuning the Analytical Engine".

**Note:** For information on other software configuration settings that affect performance, see the Oracle Demantra Installation Guide.

## Data Cleanup Script

The data cleanup script allows users to safely remove data that is no longer needed. It may be used to partially remove data (such as anything older than two years, or all data associated with a particular item that's not longer offered), or to perform a full cleanout of demo or test data from a pre-configured schema.

When a cleanup is performed with this script, configuration data (series, worksheets, levels, etc.) is normally preserved. Objects potentially affected by this process include data tables (mdp\_matrix, sales data, GL data tables) and level members (mostly by cascade from the data tables). Most levels should and will be purged of their members, with the exception of default level members. Some levels must be protected, as they are generic to Demantra and not specific to the customer.

**Note:** This procedure is only intended for use by experienced consultants and should never be run on a production schema.

### Before running the package:

- Make sure to do a complete database backup before cleaning out data.
- Make sure there are no active connections to the schema before executing these procedures.
- Make sure the application server is shutdown before executing these procedures.

### After running the package:

- Issue a COMMIT to make sure all transactions are committed.

### Notes about running the package:

- Alias levels may create infinite loops on the level tree; currently the process will stop deleting data once it looped more times than the number of levels in the system.
- There are some levels that are protected from being deleted, see the PROTECTED\_OBJECTS table in the schema.
- Objects that are not currently cleaned up are Worksheets, Users, Groups, Integration Profiles, Series, Series Groups, and Workflows.

### ***Cleaning Up Temporary Objects***

---

Notes	This also calls clean_schema_int.
Procedure	<code>DATA_CLEANUP.clean_schema_temps( commit_point NUMBER DEFAULT DEFAULT_COMMIT_POINT );</code>
Objects Affected	DB_EXCEPTION_LOG, INTEG_STATUS, AUDIT_PROFILES, AUDIT_TRAIL, AUDIT_PROFILE_USERS, E1_SALES, E1_BRANCH, E1_CUSTOMER, E1_ITEM, E1_ITEM_BRANCH, T_SRC_ITEM_TMPL, T_SRC_ITEM_TMPL_ERR, T_SRC_LOC_TMPL, T_SRC_LOC_TMPL_ERR, T_SRC_SALES_TMPL, T_SRC_LOC_TMPL_ERR
Description	DROP_TEMPS(0) is the standard procedure to drop temporary tables created by worksheet runs. See additional objects in clean_schema_int.
Command	<code>EXECUTE DATA_CLEANUP.CLEAN_SCHEMA_TEMP S;COMMIT;</code>

---

### ***Cleaning Up Level Data***

---

Notes	This should not delete the default member 0 in each level.
Procedure	<code>DATA_CLEANUP.clean_level_data( commit_point NUMBER DEFAULT DEFAULT_COMMIT_POINT );</code>
Objects Affected	SALES_DATA, MDP_MATRIX and the General Level data and matrix tables PROMOTION, PROMOTION_DATA and PROMOTION_MATRIX

---

Description	Recursively deletes all members for level data except protected levels in PROTECTED_OBJECTS table and the default member .
Command	EXECUTE DATA_CLEANUP.CLEAN_LEVEL_DATA;C OMMIT;

### ***Cleaning Up Integration-Related Settings***

Notes	These are mainly created by Oracle EBS collections:
Procedure	DATA_CLEANUP.clean_schema_int( commit_point NUMBER DEFAULT DEFAULT_COMMIT_POINT );
Objects Affected	DISPLAY_UNITS, REAL_VALUES (only update), DCM_PRODUCTS_UNITS, DCM_PRODUCTS_INDEX, INDEXES_FOR_UNITS, AVAIL_UNITS
Description	Deletes levels from TGROUP_RES and removes the column from INPUTS.
Command	EXECUTE DATA_CLEANUP.CLEAN_SCHEMA_INT;C OMMIT;

## **Demantra URLs**

Anyone can log into any Web-based Demantra product if they have the uniform resource locator (URL) and the appropriate access. These URLs are based upon information that you provide during installation. Make sure all your users know the URLs that they will need.

This table shows the URLs to use to access Oracle Demantra functions. When you use this table, in place of:

- `frodo`, pass the name of the Web server that runs the on which Oracle Demantra Web software

- 8080, pass the port number that you are using
- demantra, pass the name of the virtual directory that is the root of the Oracle Demantra Web software

Function	Example URL
Collaborator Workbench	<a href="http://frodo:8080/demantra/portal/loginpage.jsp">http://frodo:8080/demantra/portal/loginpage.jsp</a>
Demand Planner Web Client	<a href="http://frodo:8080/demantra/portal/partnerLogin.jsp">http://frodo:8080/demantra/portal/partnerLogin.jsp</a>
Workflow Manager	<a href="http://frodo:8080/demantra/workflow/login.jsp">http://frodo:8080/demantra/workflow/login.jsp</a>
Demantra Anywhere Version of Collaborator Workbench (Thin Client)	<a href="http://frodo:8080/demantra/portal/remoteloginpage.jsp">http://frodo:8080/demantra/portal/remoteloginpage.jsp</a>
Demantra Anywhere Version of Web Client	<a href="http://frodo:8080/demantra/portal/anywhereLogin.jsp">http://frodo:8080/demantra/portal/anywhereLogin.jsp</a>
Collaborator Workbench Administration	<a href="http://frodo:8080/demantra/portal/adminLogin.jsp">http://frodo:8080/demantra/portal/adminLogin.jsp</a>
Dynamic Open Link (DOL) access for third-party reporting tools	<a href="http://frodo:8080/demantra/portal/DOLLogin.jsp">http://frodo:8080/demantra/portal/DOLLogin.jsp</a>
Offline Access to Demantra Worksheets	<a href="http://frodo:8080/demantra/portal/launchDPWeb.jsp">http://frodo:8080/demantra/portal/launchDPWeb.jsp</a>
User Management	<a href="http://frodo:8080/demantra/portal/userManagement.jsp">http://frodo:8080/demantra/portal/userManagement.jsp</a>
Technical Administration	<a href="http://frodo:8080/demantra/admin">http://frodo:8080/demantra/admin</a>

### Direct Logins

You can log in directly to Oracle Demantra Collaborator Workbench and Demand Planner. Use this method, for example:

- For internal integrations: For example, Oracle Advanced Planning Command Center and in demonstrations that embed Oracle Demantra within Oracle Siebel
- When you want to embed Oracle Demantra within a web portal

This table shows the URLs to use to access direct login functions. When you use this table, in place of:

- `frodo`, pass the name of the Web server that runs the on which Oracle Demantra Web software
- `8080`, pass the port number that you are using
- `demantra`, pass the name of the virtual directory that is the root of the Oracle Demantra Web software

After the table, there is an explanation of the parameters to use in each

Function	Example URL
Collaborator Workbench	<code>http:// frodo:8080/demantra/portal/ directLogin.jsp?user=&lt;username&gt;&amp;pa ss=&lt;password&gt;</code>
Demand Planner	<code>http://frodo:8080/demantra/portal/ partnerLogin.jsp[?user=&lt;userID&gt;&amp;pa ss=&lt;password&gt;&amp;...]</code>

This table lists the parameters that the Collaborator Workbench direct login function uses. Both parameters are mandatory and neither has a default value.

Parameter Name	Description
<code>user</code>	Username
<code>pass</code>	Password of the user account

This table lists the parameters that the Demand Planner direct function uses. All parameters are optional and none have a default value.

Parameter Name	Description
<code>userID</code>	Username
<code>pass</code>	Password of the user account
<code>queryId</code>	ID of worksheet to open

Parameter Name	Description
combination	Additional filter for the worksheet. It contains the level ID. The separator between: <ul style="list-style-type: none"> <li>- Member ID pairs is a semicolon</li> <li>- Level and member ID is a comma.</li> </ul>

This is an example of the parameters portion of a Demand Planner URL that passes all of the parameters

```
?user=mjackson&pass=thriller&queryId=11100&combination=424,3;492,4
```

If you pass:

- No parameters: A page appears asking for the username and password
- user and pass: The function automatically logs you on
- user, pass, and queryID: The function automatically logs you on and opens the worksheet
- All parameters: The function automatically logs you on, opens the worksheet, and filters the data using the member and level IDs specified

### Running the Worksheets Window Applet as Standalone

Use this command line for running the Worksheets Window Applet as a standalone application.

```
<java_executable> -classpath <collaborator_jar_path>
-Dname=<user_name> -Dpassword=<password>
-Djava.security.policy=<policy_path>
-DcodeBase=<server_codebase> -Xmx256M
com.demantra.partner.client.main.PartnerApplet
```

For example

```
C:\j2sdk1.4.2_04\bin\javaw -classpath
"D:\Builds\Collaborator623-dev\Build\WebFiles\demantra\portal\co
llaborator.jar" -Dname=dp -Dpassword=dp
-Djava.security.policy=C:/java.policy
-DcodeBase=http://frodo:8080/demantra/portal/ -Xmx256M
com.demantra.partner.client.main.PartnerApplet
```

This table lists the parameters that the Worksheets Window Applet as Standalone function uses. All parameters are optional and none have a default value.

Parameter Name	Description
-Dname	Username
-Dpassword	Password of the user account
-DsessionId	Session ID

If you pass:

- No parameters: A page appears asking for the username and password
- -Dname and -Dpassword: The function automatically logs you on
- -DsessionId: The function binds to the existing session. You can pass the correct -Dname and -Dpassword for the session, but you do not have to.

Use executable `javaw` to suppress the black console window.

## Log Files and Tables

For your reference, Demantra writes the following log files and tables:

Table or file	Purpose
import.log	Information on the import process of the dump file.
DB_EXCEPTIONS_LOG	Errors detected when stored procedures are run; this information is displayed in the Business Modeler.
build_procedure.log	Information on the loading of the procedures into the new user.
upgrade.log	Information on the database upgrade process.

You can modify the default location of log files when running the Demantra Installer. Click "Configure Log" after launching the installer enables you to specify the log level, the log filename, and the directory to where log files are written.

The default location of these files is (Demantra\_root)\DemandPlanner\Database Objects\Oracle.

If you are using JRun as the Web server, note that all the JRun logs are in the directory JRun4\logs. All Demantra exceptions are written to default-err.log. You should delete all these log files before restarting the application server. You may want to back up the files first to another folder.

**Note:** If the server is running, you will not be able to delete some of the files because they are in use. You can instead clear the log files as follows: open each file, select all text with ctrl+A, and delete the text and save the file.

## Illegal Characters in Demantra

Within Demantra, do not use the following special characters:

Single quote (')

Double quote (")

Ampersand (&)

If you use these characters, unexpected results may occur.

## Required Third-Party Software

The *Oracle Demantra Installation Guide* lists third-party software with which Demantra works. It may be useful to review this information.



---

## Managing Security

The Demantra data and features are secured, so that not all users have access to the same data and options. This chapter describes how to maintain security.

This chapter covers the following topics:

- Creating or Modifying a Component
- Deleting a Component
- Creating or Modifying a User
- Copying a User
- Deleting a User
- Creating or Modifying a User Group
- Deleting a Group
- Providing Access to the Workflow Editor
- Logging onto the Collaborator Workbench Administrator
- Defining a Program Group
- Redefining a Program Group
- Deleting a Program Group
- Specifying Permissions for Menu Items
- Logging Out Users

### Creating or Modifying a Component

**To create or modify a component:**

1. Click Components > Create/Open Component. Or click the Create/Open Component button.

**Note:** This option may not be available, depending on the user name with which you logged onto Business Modeler.

The Create/Open Component dialog box appears.

2. Now do one of the following:

- To create a new component, click the New Component button and then click OK. Or double-click the New Component icon.

**Note:** This option is available only if you log into Business Modeler as the user with the highest permission.

- To open an existing component, double-click the icon corresponding to the component. Or click the icon and then click OK.

The Component Configuration Wizard displays its first dialog box.

3. Enter or edit general information for the user interface, as follows:

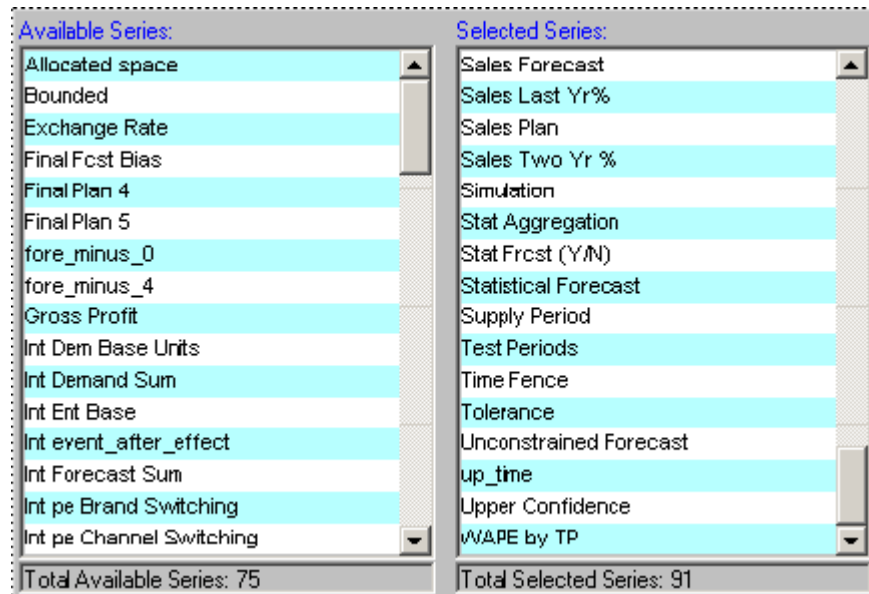
---

Component Name	Unique name for this component.
Component Description	Description
About Window Description	Optional description to include in the About page of this component.

---

4. Click Next.

The Business Modeler displays the Available Series and Selected Series lists.



5. Select the series that should be available in the component.
  1. Move all series that you want into the Selected Series list, using any of the techniques in "Working with Lists".
  2. Remove any unwanted series from the Selected Series list.
  3. When you are done specifying series, click Next.

**Note:** By default, this configuration affects all users of this component. To hide additional series for a given user, see "Creating or Modifying a User".

6. Click Next.

The Business Modeler displays the Select Component Indicators for Series window. Here you specify which series should have indicators to indicate associated promotions or notes.

**Select Indicator:** Promotion

**Available Series:**

- Outlier A
- Outlier Full
- Partner % in Fcst
- Partner 1 Acc
- Partner 2 Acc
- Partner 3 Acc
- Partner Fcst Override
- Partner Fcst
- Partner Plan 1
- Partner Plan 2
- Partner Plan 3
- Plan Dif
- POS
- Price \$
- Price change
- Price change Duration indicator

**Selected Series:**

- Final Plan

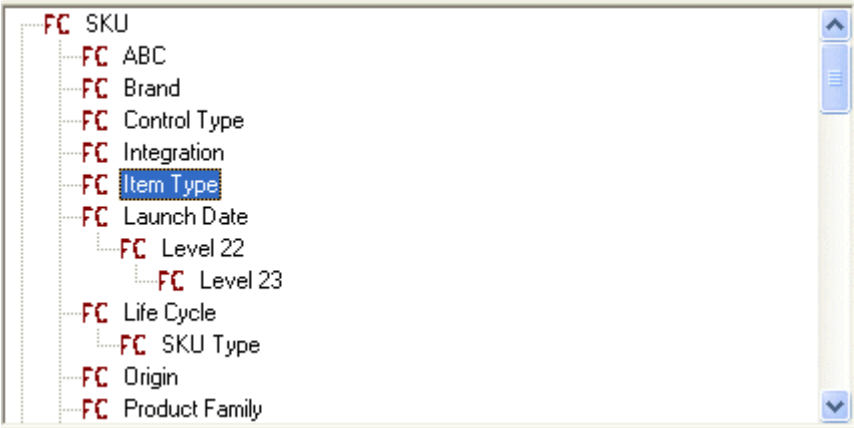
Total Available Series: 85      Total Selected Series: 1

Within a worksheet, a user can attach a promotion (in the case of Promotion Effectiveness) or a note to a given item-location combination, at a given date. If a series has been configured as using an indicator for that particular promotion or note, the series will be displayed with an indicator in all worksheet cells that correspond to that item-location combination and date.

- You can associate an indicator for any general level at the lowest level (that is, any general level that do not have child levels).
  - The default associations are different for different kinds of series. Sales series have notes indicators by default. Promotion series have both notes and promotion indicators by default.
  - This configuration affects all users of this component. No further fine tuning is possible.
7. To associate indicators with different series, do the following for each general level:
    1. In Select Indicator, select the general level, either Note or Promotion.
    2. Move all series that should use the associated indicator into the Selected Series list, using any of the techniques in "Working with Lists".
    3. Remove any unwanted series from the Selected Series list.

8. Click Next.

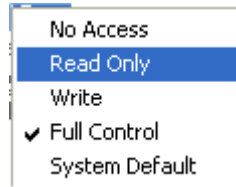
The system displays all the levels and indicates the current permission settings in this component.



The following icons indicate the permissions:

FC	Full control (including permission to delete members)
W	Read/write access
R	Read access
X	No access

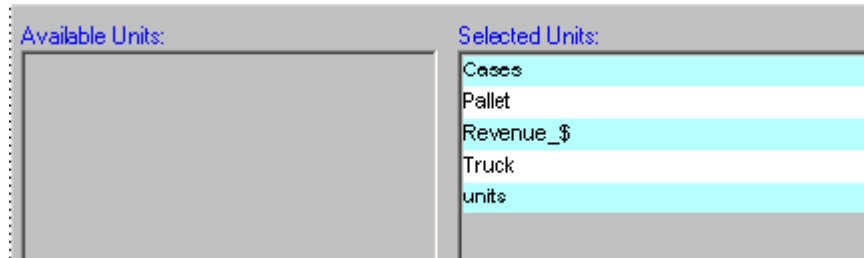
9. For each level that you want to change, right-click the level and select the appropriate permission:
- No Access (the user does not have access to this member; this option is equivalent to not including this member in the filter)
  - Read Only (the user can view this member but cannot make any changes)
  - Write (the user can view or edit this member)
  - Full Control (user can view, edit, create, and delete within this member)
  - System Default (use the default permission controlled by the DefaultLevelSecurityAccess parameter.



**Note:** By default, this configuration affects all users of this component. To fine tune permissions for a given user, see "Creating or Modifying a User".

10. Click Next.

The system displays the Available Units and Selected Units lists.



11. Select the units of measure that should be available in the component.
  1. Move all units that you want into the Selected Units list, using any of the techniques in "Working with Lists".
  2. Remove any unwanted units from the Selected Units list.

**Note:** This configuration affects all users of this component. No further fine tuning is possible.

12. Click Next.
  - The system displays the Available Indexes and Exchange Rates and Selected Indexes and Exchange Rates lists.

Available Indexes / Exchange Rates:	Selected Indexes / Exchange Rates:
	Consumer Price Index Dollar \$ Euro ?
Total Available Indexes: 0	Total Selected Indexes: 3
<b>Legend:</b> Index  Exchange Rate	

13. Select the indexes and exchange rates that should be available in the component.
  1. Move all indexes and exchange rates that you want into the Selected Indexes and Exchange Rates list, using any of the techniques in "Working with Lists".
  2. Remove any unwanted indexes and exchange rates from the Selected Indexes and Exchange Rates list.

**Note:** This configuration affects all users of this component. No further fine tuning is possible.

14. Click Next.

The next dialog box allows you to associate public worksheets with levels.

Select Level:	Available Queries:	Selected Queries:	Default Query:
City	000. Introduction 000B. Middle out Enterprise Plan 001. Store Plan 002. Creating a strategic plan 004. New Product Launch 005. Analyze historical Statistical perform 006. Clustering Report update 007. Analyze Historical Plan performance 008. Middle out Enterprise Plan 009. Store Manager Plan 010. Consensus Plan 011 Consensus Plan Cross tab analysis 012. Budget Exception Analysis 013. Lifecycle Management 014. Wall Management 015. Coupon Program		
	Total Available Queries: 39	Total Selected Queries: 0	

This association is used in two ways:

- Within the Members Browser, a user can use the right-click menu to open any of these associated worksheets directly from a member of the level (via the Open With menu option). In this case, Demantra opens the associated worksheet. The worksheet is filtered to show only data relevant to the member.
- A worksheet can include an embedded worksheet that shows details for the member that is currently selected in the worksheet. Specifically, within the worksheet designer, users can add a subtab to a worksheet. The subtab consists of any of the worksheets that are associated with a level included in the main worksheet. The embedded worksheet is filtered to show only data relevant to the member.

**Note:** This configuration affects all users of this component. No further fine tuning is possible.

15. At this point, do one of the following:

- To continue without associating any worksheets and levels, click Next.
- To associate a worksheet with a level, do the following:
  1. Click the level in the Select Level dropdown menu.
  2. Double-click the worksheet in Available Queries list, which moves it to the Selected Queries list.
  3. Move other worksheets from the Available Queries list to the Selected Queries list, as needed.
  4. Decide which worksheet in the Selected Queries list should be the default worksheet for this level. For that worksheet, click the Default check box. When the user right-clicks and selects Open, this is the worksheet that will be used.
  5. When you are done on this screen, click Next.

If you are using the PE Analytical Engine, the system displays engine profiles that could potentially be used within this component. The Business Modeler displays the Available Engine Profiles and Selected Engine Profiles lists.



Available Engine Profiles:	Selected Engine Profiles:
	Base
	Batch
	DP simulation profile
	Fast PE simulation profile
	Full PE learning simulation profile
	Simulation
Total Available Engine Profiles: 0	Total Selected Engine Profiles: 6

16. Select the engine profiles that should be available in the component. Profiles can be used only with the Promotion Effectiveness engine.
  1. Move all profiles that you want into the Selected Engine Profiles list, using any of the techniques in "Working with Lists".
  2. Remove any unwanted profiles from the Selected Engine Profiles list.
    1. When you are done specifying profiles, click Next.

**Note:** This configuration affects all users of this component.  
No further fine tuning is possible.

In the next step, you specify the user name and password of the user who owns the component. This user will be able to log into the Business Modeler and create additional users for this component.

**Define Component User.**

User Name:

User Password:

17. To specify the owner of the component:
  - In the User Name box, type the user name.

- In the User Password box, type the user password.
18. To exit and save the configuration, click OK.
  19. Modify the newly created user so that it has access to the appropriate Demantra modules. To do so, use the Security menu; see "Creating or Modifying a User".

## Deleting a Component

### To delete a component:

1. Click Components > Create/Open Component. Or click the Create/Open Component button.

**Note:** This option may not be available, depending on the user name with which you logged onto Business Modeler.

The Create/Open Component dialog box appears.

2. Click the icon corresponding to the component.
3. Click Delete.
4. Click Yes to confirm the deletion.

## Creating or Modifying a User

You can create additional users to work within the component you own.

### To create or modify a user:

1. Log on to the Business Modeler as described in "Logging onto the Business Modeler".
2. Click Security > Create/Modify User. Or click the Create/Modify User button.  
The Create/Modify User dialog box appears.
3. Next:
  - To create a new user, click the New User button, and then click OK.
  - To modify a user, click the button of that user then click OK. Or double-click the icon of the user whose details you want to modify.

The User Details dialog box appears.

**Enter User Details**

User : Jeff\_Wilson

Password: \*\*\*\*\* Integrate User: ☐

Permission Level: Supervisor

Language: English

First Name: Jeff

Last Name: Wilson

Company Name: Rory's International

Phone Number:

Fax Number:

E-Mail Address: jwilson@rorys.com

4. Specify basic user details as follows:

- Under Enter User Details, type the following information in the appropriate boxes (or select from the drop down lists):
- The user name, password, permission level, and the language in which the system will be operated. Each user name must be unique within your Demantra implementation.
- The first and last name of the user, the company name, phone and fax number, and the email address. If you set up automated email within workflows, it is important to make sure the email address is correct here.
  1. For Permission Level, see "Permission Levels".
  2. If this user also needs to work with Inventory Optimizer and Demantra Replenisher, click the Integrate User check box.

**Note:** This check box may or may not be displayed, depending on how Demantra was installed.

**Note:** Inventory Optimizer and Demantra Replenisher are licensed and documented separately from the core Demantra products.

3. Click Next.

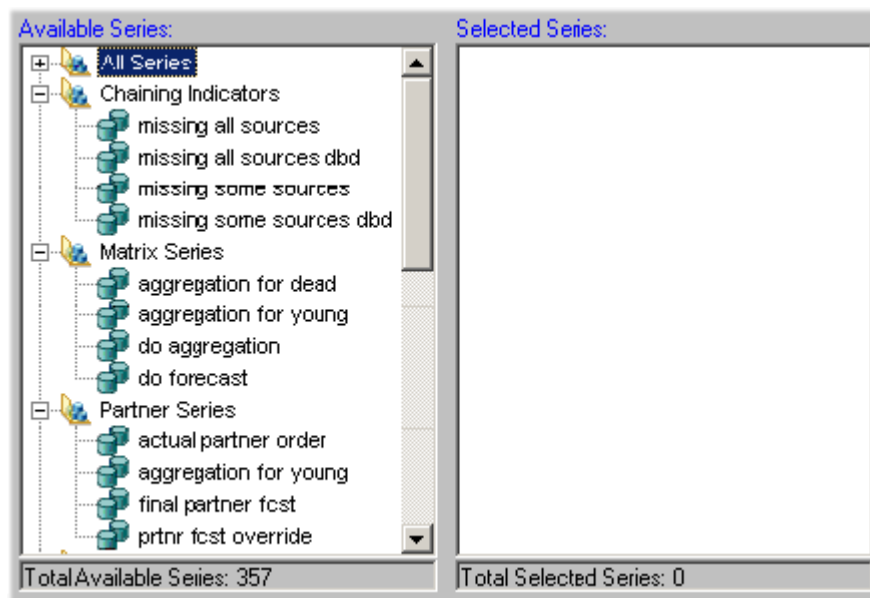
The User Modules dialog box appears. Here you specify which Demantra user

interfaces this user can access.

Name	Status	Available Named Users	Defined Concurrent Users
Demantra Administrative Tools	<input checked="" type="checkbox"/>	9983/9999	9999
Demantra Demand Planner	<input checked="" type="checkbox"/>	9982/9999	9999
Demantra Demand Planner Web	<input checked="" type="checkbox"/>	9982/9999	9999
Demantra Collaborator Workbench	<input checked="" type="checkbox"/>	9981/9999	9999
Demantra Anywhere	<input checked="" type="checkbox"/>	9990/9999	9999
Demantra Promotion Effectiveness	<input checked="" type="checkbox"/>	9981/9999	9999
Settlement Management	<input type="checkbox"/>	9999/9999	9999
Demantra Promotions Optimization	<input type="checkbox"/>	9999/9999	9999

5. Click the check box next to each module that the user needs to work with. Then click Next.

The New User - Select User Series dialog box appears. This dialog box allows you to determine what data series will be active for the new user, from the entire set of series in this component. Each list is a collapsible list of series groups and the series in them.



If a series is not active for a user, it is not available when the user creates worksheets and is not viewable in existing worksheets to which the user has access.

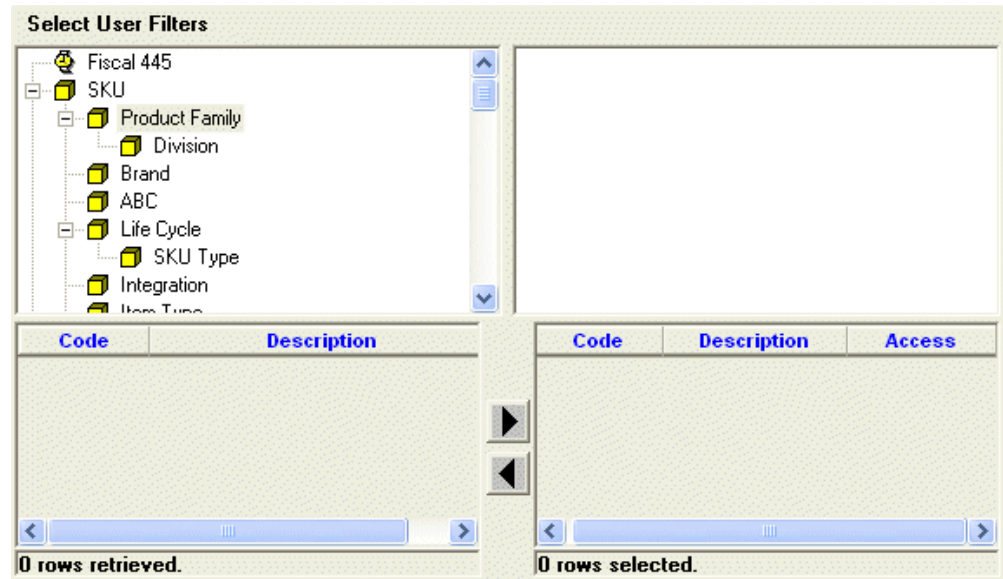
6. Specify the series that a user can see, as follows:
  1. Move all series that you want into the Selected Series list. To do so, either double-click each series or drag and drop it.

2. Remove any unwanted series from the Selected Series list.

**Note:** You can also move an entire series group from one list to the other in the same way.

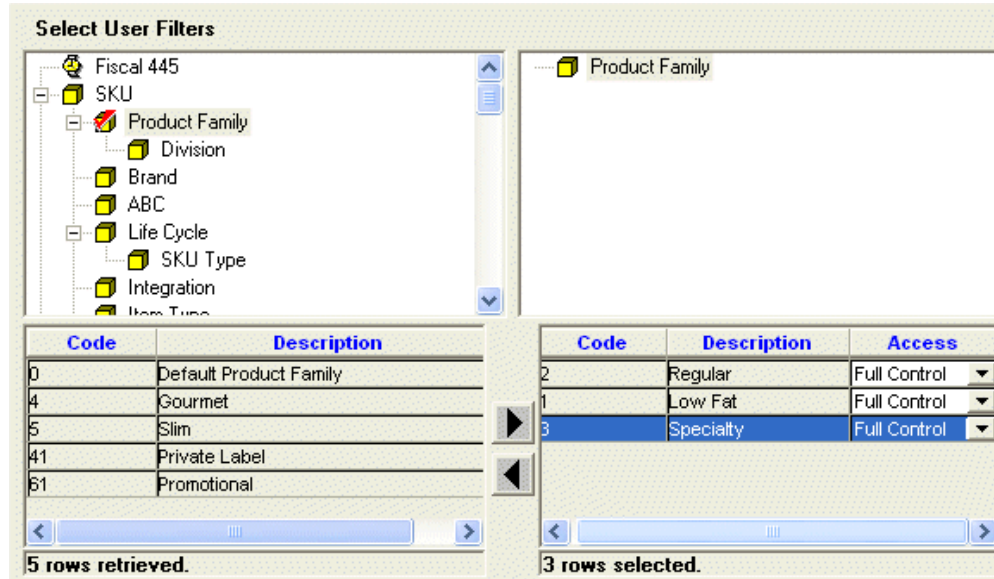
3. When you are done specifying series, click Next.

The New User - Select User Filters dialog box appears. This dialog box lets you filter the data that the user can see; specifically, you control which levels and which members of those levels the user can see.



7. Filter the data that the user can see, as follows:
  1. Click a level in the left side of the dialog box and drag it to the box on the right. Or double-click a level in the left side.
  2. Now specify which members of this level the user can see. To do so, click a member in the list, and then click the right arrow button. Or double-click the member you want to filter out.

The system moves the selected members to the box on the lower right side, as in this example:



8. Now the user can see only the selected members of this level. In the preceding example, the user can see only data that is associated with the Rainbow brand.

**Note:** The Selected Members list cannot include more than 200 members.

In the lower right, refine the security settings that control the access that the user has to each member. To do so, in the Access column, click one of the following:

1. Full Control (user can view, edit, create, and delete within this member)
  2. Read & Write (the user can view or edit this member)
  3. Read only (the user can view this member but cannot make any changes)
  4. No access (the user does not have access to this member; this option is equivalent to not including this member in the filter)
  5. System Default (use the default permission controlled by the DefaultContentSecurityAccess parameter)
9. Repeat the preceding steps for each filter you want to add. Each filter automatically limits the choices available in subsequent filters.

When you have appropriately filtered data for the user, click Next.

The New User - Select User Groups dialog box appears. This dialog box allows you to select the group or groups to which the new user will belong.

10. Specify the collaboration groups to which a user belongs, as follows:
  1. Move all groups to which the user should belong into the Selected Groups list. To do so, either double-click each group or drag and drop it.

**Note:** You can also select and move multiple groups with the standard Ctrl+click or Shift+click actions.

2. Remove any unwanted groups from the Selected Groups list.
  3. Click Next.
11. Click Finish.

See also

API to Create, Modify or Delete Users, page 13-9

Copying a User, page 61-15

Deleting a User, page 61-16

## Copying a User

If you need to create multiple similar users, it is useful to create one of those users and then copy it to create the other users.

### To copy a user:

1. Log on to the Business Modeler as described in "Logging onto the Business Modeler."
2. Click Security > Create/Modify User. Or click the Create/Modify User button.  
The Create/Modify User dialog box appears.
3. Click the button of the user you want to copy, and then click Create Copy.  
The User Details dialog box appears. Some of the information, such as user name, is blank. Other details, such as the company name, are copied from the original user.
4. Specify the user name and password for the new user.
5. Make other changes as needed.
6. Do one of the following:
  - Click Next to continue editing information for the new user. Demantra initially uses all the same values as for the original user.

- Click Finish.

Demantra also copies menu permissions of the original user; see "Specifying Permissions for Menu Items".

See also

"Creating or Modifying a User"

## Deleting a User

### **To delete a user:**

1. Log on to the Business Modeler as described in "Logging onto the Business Modeler."
2. Click Security > Create/Modify User. Or click the Create/Modify User button.  
The Create/Modify User dialog box appears.
3. Click the button of the user you want to delete, and then click Delete.  
A question box appears, inquiring if you are sure you want to delete the selected user.
4. To delete the selected user, click Yes.

See also

"Creating or Modifying a User"

## Creating or Modifying a User Group

Demantra uses user groups for several purposes:

- Group members can collaborate, within Collaborator Workbench.
- The Workflow Engine can send tasks to groups (as well as to users).
- Groups can be authorized to view and edit notes attached to worksheets.
- Groups can be authorized to use menu items.

Groups are visible in all components. Note that the users in a group can belong to different components.

### **To create or modify a group:**

1. Log on to the Business Modeler as described in "Logging onto the Business



Modeler."

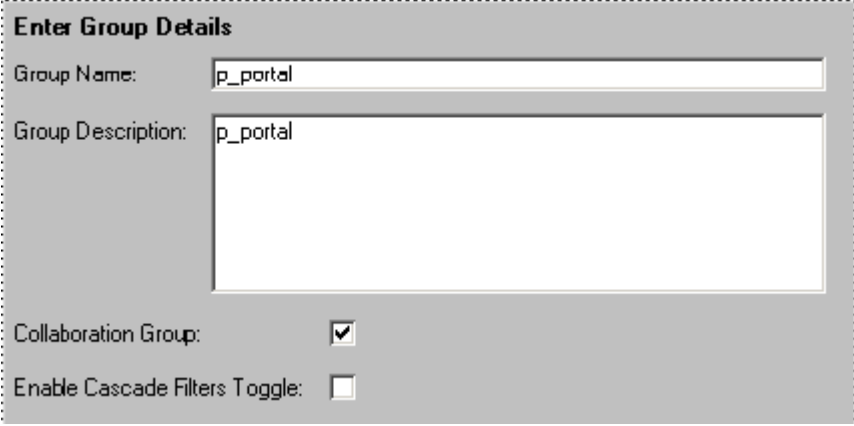
2. Click Security > Create/Modify Group. Or click the Create/Modify User Group button.

The Create/Modify Group dialog box appears.

3. Next:

- To create a new group, double-click the New Group button.
- To modify a group, click the button of that group then click OK. Or double-click the icon of the group whose details you want to modify.

The system prompts you for information about the group.

A screenshot of a dialog box titled "Enter Group Details". It has a light gray background and a dotted border. Inside, there are two text input fields: "Group Name:" and "Group Description:". Both fields contain the text "p\_portal". Below these fields are two checkboxes: "Collaboration Group:" which is checked, and "Enable Cascade Filters Toggle:" which is unchecked.

**Enter Group Details**

Group Name:

Group Description:

Collaboration Group: ☒

Enable Cascade Filters Toggle: ☐

4. Specify group details as follows:

1. Under Enter Group Details, type a name and optional description in the appropriate boxes. Each group name must be unique within your Demantra implementation.
2. If users of this group should be able to see either other in the Who's Online pane in Collaborator Workbench, make sure the Collaboration Group check box is checked. Only members of this group are able to access the Workflow Manager.

The users will also be able to send tasks to each other.

If you clear this check box, users of the group will not see one another.

3. Check or clear the Enable Cascade Filters Toggle check box.

Click this option to enable users in the group to toggle between cascade and non-cascade filter modes. If not selected, the user will have cascade filtering only.

In cascade mode, users see only members that have combinations with the previously selected members. Members that do not have combinations will not be available in the list. It is generally easier to work with filters in cascade mode.

In non-cascade mode, users see all the members of the selected level regardless of the previously selected members from other levels.

4. Click Next.

The New Group - Select Group Users dialog box appears. This dialog box allows you to select existing users who will belong to the new group.

Select Groups Users	
Available Users:	Selected Users:
Abby_Rose	dp
Bill_Feldman	dr
Max_Kan	ERP
	Guy_Catalani
	guy_yehiav
	Inv
	ip
	Jeff_Wilson
	Margie
	Marie_C
	Maya
	pe
	Replenisher
	Sharon_Cowan
Total Available Users: 3	Total Selected Users: 14

5. Specify the users in a group, as follows:
  1. Move all users that should be in this group into the Selected Users list. To do so, either double-click each user name or drag and drop it.

**Note:** You can also select and move multiple users with the standard Ctrl+click or Shift+click actions.

2. Remove any unwanted users from the Selected Users list.
3. Click Next.

6. Click Finish.

See Also

"Data Security"

"Deleting a Group"

## Deleting a Group

### To delete a group:

1. Log on to the Business Modeler as described in "Logging onto the Business Modeler."
2. Click Security > Create/Modify Group. Or click the Create/Modify Group button.  
The Create/Modify Group dialog box appears.
3. Click the button of the group that you want to delete.  
A box appears, inquiring if you are sure you want to delete the selected group.
4. Click Delete.

See also

"Data Security"

"Creating or Modifying a User Group"

## Providing Access to the Workflow Editor

In order for a given user to log into the Workflow Editor, that user must be configured a specific way.

### To provide access to the Workflow Editor:

1. Log on to the Business Modeler as described in "Logging onto the Business Modeler."
2. Create a group that includes all users who need to log into the Workflow Editor.  
See "Creating or Modifying a User Group".
3. Click Parameters > System Parameters.
4. Click the Application Server > Workflow tab.
5. Edit the workflow.group parameter. Specify the name of the group you set up; if you have multiple such groups, separate them with commas.

See also

"Managing Workflows"

## Logging onto the Collaborator Workbench Administrator

You use the Collaborator Workbench Administrator to control access to menu items.

### To log onto the Collaborator Workbench Administrator:

1. Open the administration login page:

`http://server name/virtual directory/portal/adminLogin.jsp`

For example:

`http://frodo/demantra/portal/adminLogin.jsp`

2. Enter the user name and password and click Log on.

Demantra displays the Administration page, which includes the following choices:



[Define Menus](#)  
[Define Program Groups](#)  
[Define Program Permissions](#)  
[Define Content Security](#)  
[Default Pane Layout](#)  
  
[Logout](#)  
[Login to Collaborator Workbench](#)  
[Login to Demantra Anywhere](#)

See also

"Defining a Program Group"

"Specifying Permissions for Menu Items"

"Customizing Demantra Web Pages"

## Defining a Program Group

A program group is a collection of menu items, typically related to each other in some way. You create program groups so that you can easily control access to all the menu items in the group; see "Specifying Permissions for Menu Items".

Demantra provides several predefined program groups, for convenience. These program groups contain only menu items from the right-click menus.

Program group	Menu items in this group, by default
Add	New <i>member</i> right-click menu option for every level in the system.
Edit	Edit <i>member</i> Unmapped Conditional Text: HelpOnly  right-click menu option for every level in the system.
Delete	Delete <i>member</i> right-click menu option for every level in the system.
View	View <i>member</i> right-click menu option for every level in the system.
Copy	Copy, Paste, and Paste from Clipboard right-click menu options for every applicable level in the system. (Note that this option is available only for promotional-type levels.)
Open	Open and Open With right-click menu options for every level in the system.

### To define a program group:

1. Log into the Collaborator Workbench Administrator. See "Logging onto the Collaborator Workbench Administrator".  
The Administration page appears.
2. Click Define Program Groups.  
The system displays a page that lists the existing program groups.

Program Group Name	Action	
<a href="#">Add</a>		
<a href="#">Edit</a>		
<a href="#">Delete</a>		
<a href="#">View</a>		
<a href="#">Copy</a>		
<a href="#">Open</a>		



3. Click the Add Program Group button.

Demantra displays a page where you can define a new program group:

Name:

Description:

Program Type Filter:  Level Filter:

	Program Object	Selected
		<input type="button" value="Select All"/>
	Object Menu	
	ABC	<input type="checkbox"/>
	New ABC	<input checked="" type="checkbox"/>
	Edit ABC	<input type="checkbox"/>
	Delete ABC	<input type="checkbox"/>
	View ABC	<input type="checkbox"/>
	Open	<input type="checkbox"/>
	Open With	<input type="checkbox"/>
	Add Note	<input type="checkbox"/>
	Account	<input type="checkbox"/>

4. For Name and Description, specify a name and optional description for this program group.
5. Optionally select an item from the Program Type Filter selection list, to reduce the number of menus and menu items shown on this screen.
  - To display only options on the right-click menus, click Object Menu.
  - To display only options on the menu bars, click Menu.
6. Optionally select a level from the Level Filter selection list, to reduce the number of menus and menu items shown on this screen. (This filtering is available only if you are viewing right-click menus.)
7. In the table, expand the menus as needed.
8. In the Selected column, select the check box for each menu item to include within this program group.
9. Click OK.

You are now ready to define permissions for this program group; see "Specifying Permissions for Menu Items".

See also

"Deleting a Program Group"

## Redefining a Program Group

### To redefine a program group:

1. Log into the Collaborator Workbench Administrator. See "Logging onto the Collaborator Workbench Administrator".  
The Administration page appears.
2. Click Define Program Groups.  
The system displays a page that lists the existing program groups.
3. In the row corresponding to the group you want to redefine, click the Edit Program Group button.  
Demantra displays a page where you can edit this program group.
4. Optionally edit the Name and Description.
5. Optionally select an item from the Program Type Filter selection list, to reduce the number of menus and menu items shown on this screen.
  - To display only options on the right-click menus, click Object Menu.
  - To display only options on the menu bars, click Menu.
6. Optionally select a level from the Level Filter selection list, to reduce the number of menus and menu items shown on this screen. (This filtering is available only if you are viewing right-click menus.)
7. In the table, expand the menus as needed.
8. In the Selected column, select the check box for each menu item to include within this program group.
9. Click OK.

See also

"Deleting a Program Group"



## Deleting a Program Group

### To delete a program group:

1. Log into the Collaborator Workbench Administrator. See "Logging onto the Collaborator Workbench Administrator."

The Administration page appears.

2. Click Define Program Groups.

The system displays a page that lists the existing program groups.

3. In the row corresponding to the group you want to delete, click the Delete Program Group button. No confirmation message is displayed; the group is deleted immediately.

See also

"Defining a Program Group"

## Specifying Permissions for Menu Items

### To specify permissions for menu items:

1. Log into the Collaborator Workbench Administrator. See "Logging onto the Collaborator Workbench Administrator".

The Administration page appears.

2. Click Define Program Permissions.

The system displays a page where you specify the category upon which to apply the menu availability.

The screenshot shows a 'Security Scope' configuration window. It contains four radio buttons for selecting the scope: 'Current Component' (which is selected), 'User Permission', 'Group', and 'User'. To the right of the 'User Permission', 'Group', and 'User' radio buttons are dropdown menus. The 'User Permission' dropdown shows 'System Manager', the 'Group' dropdown shows 'p\_portal', and the 'User' dropdown shows 'dp'. At the bottom of the window, there is a 'Module Name' dropdown menu showing 'Settlement Management'.

3. To define the scope, check one of the following radio buttons and select an item from the associated drop down list:

---

Current Component	Use this option to enable or disable menu items for all users of the component that you own.
User Permission	Use this option to enable or disable menu items for a specific permission level. See "Permission Levels".
Group	Use this option to enable or disable menu items for a specific user.
User	Use this option to enable or disable menu items for a specific user group.
Module Name	Use this option to specify if the changes you make should apply to all modules or to specific modules.

---

4. Click Next.

Demantra displays an expandable hierarchy that shows all the menu items you chose, like the following example:

Program Type Filter: All Level Filter: All

Program Object	Hidden	Disabled	Inherited Permission
<b>Settlement Management</b>	<b>Select All</b>	<b>Select All</b>	<b>Select All</b>
<b>File</b>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
<b>Worksheet</b>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
<b>Edit</b>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
<b>View</b>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
<b>Options</b>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
<b>Data</b>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
<b>Help</b>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
<b>Program Groups</b>			
<b>Object Menu</b>			

Initially, the Inherited Permission check boxes are all checked, which means that the permissions that will be used are inherited from higher in the security hierarchies. Likewise, the Hidden and Disabled check boxes display the current inherited settings.

5. Optionally select an item from the Program Type Filter selection list, to reduce the number of menus and menu items shown on this screen.
  - To display only options on the right-click menus, click Object Menu.
  - To display only options on the menu bars, click Menu.
6. Optionally select a level from the Level Filter selection list, to reduce the number of menus and menu items shown on this screen. (This filtering is available only if you are viewing right-click menus.)
7. In the table, expand the menus as needed.
8. For each item in this table, specify permissions as follows:

Desired outcome	Hidden	Disabled	Inherited Permission
Menu option is explicitly hidden	Checked	Irrelevant	Unchecked
Menu option is explicitly displayed but disabled	Unchecked	Checked	Unchecked

Desired outcome	Hidden	Disabled	Inherited Permission
Menu option is explicitly displayed and enabled	Unchecked	Unchecked	Unchecked
Use implicit permissions for this menu item	Unchecked	Unchecked	Checked
<b>Note:</b> To understand how multiple permissions are combined, see "How Demantra Combines Multiple Permissions".			

9. Click Finish. The settings are saved.

See also

"Configuring Menus in Collaborator Workbench"

## Logging Out Users

Demantra provides a tool that you can use to log out users whose sessions have hung due to network or other problems. This applies only to the users of the Web-based products

**Note:** A user with permission level below System Manager can log into this tool and end his or her own session. Other users will not be visible.

### To log a user out of Demantra:

1. Browse to the following case-sensitive URL:

`http://server name/virtual directory/portal/userManagement.jsp`

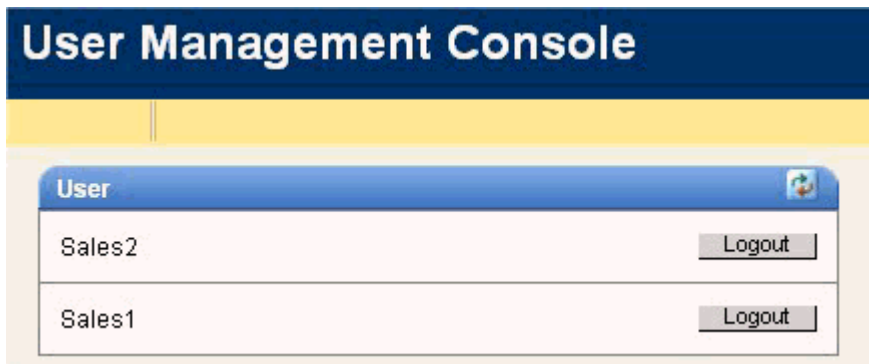
For example:

`http://frodo/demantra/portal/userManagement.jsp`

A login page appears.

2. Type your username and password and then click Log on.

Demantra displays the following screen:



3. Click Logout in the row corresponding to the user you want to log out.



---

## Managing Workflows

This chapter describes how to use the Workflow Manager to start, stop, and view workflow instances and to manage schema groups.

This chapter covers the following topics:







- Viewing Workflow Status
- Starting Workflow Instances
- Scheduling Workflow Instances
- Stopping Workflow Instances
- Creating or Editing a Schema Group
- Deleting a Schema Group
- Viewing the Workflow Process Log
- Recovery and the Workflow Engine
- Web Services for Demantra Workflows

### Viewing Workflow Status

You can view all the status of all public workflow schemas and all private workflow schemas that you created. This means that you can see how many instances of those schemas are running, as well as the status of each instance.

#### **To view overall status of the workflows:**

The Workflow Manager displays the overall status information for the workflows, for the currently selected schema group (All in this case).

View according to Schema Groups							<a href="#">New</a> <a href="#">Modify</a> <a href="#">Delete</a>			
All										
Schema ID	Schema name	Owner	Creation Date	Last Modified	Instances	Status	Action			
1	<a href="#">Partner Plan collaboration</a>	dp	Feb 28 20:10:02 2002	Aug 21 16:56:46 2003	0		<a href="#">Edit</a>	<a href="#">Start</a>	<a href="#">Schedule</a>	<a href="#">Delete</a>
2	<a href="#">Space_consumption_Alert</a>	guy_yehiav	Apr 14 21:21:45 2003	May 07 18:05:45 2003	0		<a href="#">Edit</a>	<a href="#">Start</a>	<a href="#">Schedule</a>	<a href="#">Delete</a>
22	<a href="#">Stockout_Alert</a>	guy_yehiav	Apr 14 22:40:47 2003	Sep 22 16:31:00 2003	0		<a href="#">Edit</a>	<a href="#">Start</a>	<a href="#">Schedule</a>	<a href="#">Delete</a>
23	<a href="#">CPFR_Step_1</a>	guy_yehiav	Apr 29 17:46:04 2003	Apr 29 17:46:34 2003	0		<a href="#">Edit</a>	<a href="#">Start</a>	<a href="#">Schedule</a>	<a href="#">Delete</a>
43	<a href="#">Stockout_Alert_Per_Store</a>	guy_yehiav	May 08 13:05:53 2003	May 08 13:05:53 2003	0		<a href="#">Edit</a>	<a href="#">Start</a>	<a href="#">Schedule</a>	<a href="#">Delete</a>
63	<a href="#">Export Dynamic Data</a>	Inv	May 16 09:59:59 2003	May 16 10:00:14 2003	0		<a href="#">Edit</a>	<a href="#">Start</a>	<a href="#">Schedule</a>	<a href="#">Delete</a>
							<a href="#">Process Log</a> <a href="#">New Schema</a> <a href="#">Refresh</a>			

Each row corresponds to a workflow schema. The Instances column indicates how many instances of this workflow schema are currently running, if any. The Status column uses the following color codes:

Green	The workflow schema is live and you may execute it, creating a workflow instance.
Red	The workflow schema is archived and cannot be executed.
Yellow	There is a data error or other fault within this schema.

### To specify which workflow schemas to display:

1. To see all workflow schemas, select All in the Schema Groups drop-down list. Then click View.
2. To see a subset of the schemas, select a group in the Schema Groups drop-down list. Then click View.

### To refresh the display:

1. Click Refresh.

### To view the currently running instances of a schema:

1. If the workflow schema is not visible, use the drop-down menu and select the name



of a schema group to which it belongs. Or select All.

2. Click the Instances link in the row corresponding to that workflow.

The Workflow Manager lists all the instances of that schema that are currently running.

Process ID	Started at	Initiator	Current step	Action
7	Mon Dec 08 14:53:16 EST 2003	dp	ExportMasterData	<a href="#">Terminate</a>

[Back](#) [Refresh](#)

3. When you are done, click Back.

See also

"Logging into the Workflow Manager"

"Viewing the Workflow Process Log"

## Starting Workflow Instances

You can start an instance of any public workflow schema or any private workflow schema that you created.

### To start a workflow instance:

1. If the workflow schema is not visible, use the drop-down menu and select the name of a schema group to which it belongs. Or select All.
2. Click Start next to the schema that you want to start.

The Workflow Engine starts an instance of the workflow and increments the number of instances in the Instances column by one.

**Note:** Although you can generally run as many instances of a workflow as you want at the same time, be careful not to activate conflicting processes and tasks.

See also

"Logging into the Workflow Manager"

"Stopping Workflow Instances"

"Creating Workflows"

## Scheduling Workflow Instances

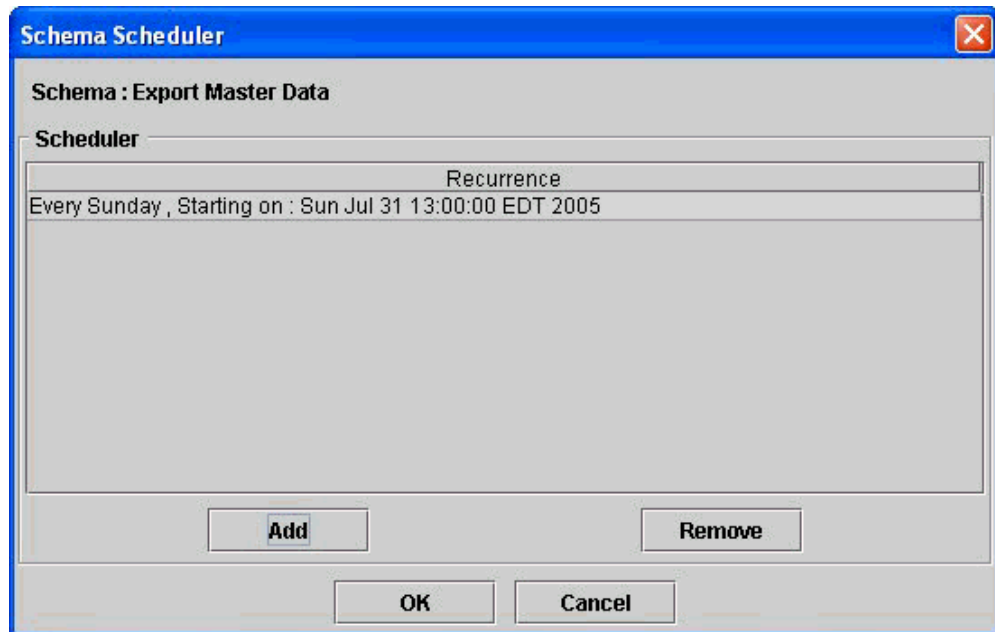
If you are the owner of a workflow, you can schedule an instance to start at a specific

time or times. If you are not the owner, you cannot schedule it, although you can start it manually, as described in "Starting Workflow Instances".

**To schedule a workflow instance:**

1. If the workflow schema is not visible, use the drop-down menu and select the name of a schema group to which it belongs. Or select All.
2. Click Schedule in the row corresponding to that workflow.

The system displays the Schema Scheduler page, which lists all the times when Workflow Engine will start an instance of this schema.



The screenshot shows a dialog box titled "Schema Scheduler" with a blue header bar and a red close button. The main content area is titled "Schema : Export Master Data" and contains a section labeled "Scheduler". Inside the Scheduler section, there is a table with one row. The table has two columns: "Recurrence" and "Starting on". The "Recurrence" column contains the text "Every Sunday , Starting on : Sun Jul 31 13:00:00 EDT 2005". Below the table, there are two buttons: "Add" and "Remove". At the bottom of the dialog box, there are two buttons: "OK" and "Cancel".

Recurrence
Every Sunday , Starting on : Sun Jul 31 13:00:00 EDT 2005

Buttons: Add, Remove, OK, Cancel

3. Click Add.

The system displays the following page.

**Recurrence**

Schema : Export Master Data

Recurrence

Schedule Schema :      At :      Start On :

Daily      05:59 PM      07/28/2005

Schedule Schema Daily

Every 1 day(s)

OK      Cancel

4. In the Schedule Schema drop-down list, select the option that specifies how often to start an instance of this workflow:
  - Daily
  - Weekly
  - Monthly
  - Once
  - At Startup (This option launches the workflow whenever the Web server is started.)
  - Periodic (in this option, you can start a workflow at periodic intervals (measured in seconds, minutes, hours, days, weeks, months, or years. Note that you cannot choose the starting time.)

Depending on the choice you make here, the system displays additional scheduling options in the bottom part of the page.
5. In the rest of the page, finish specifying the schedule.

6. Click OK.

#### To unschedule a workflow instance:

1. On the Workflow Management page, click Schedule in the row corresponding to that workflow.

The system displays the Schema Scheduler page. This page displays one row for each scheduling entry for this workflow.

2. Click the row corresponding to the scheduling entry you want to remove.
3. Click Remove.

See also

"Starting Workflow Instances"

"Stopping Workflow Instances"

## Stopping Workflow Instances

You can stop any workflow instance that you started. You cannot stop a workflow instance started by another user.

#### To stop a workflow instance:

1. If the workflow schema is not visible, use the drop-down menu and select the name of a schema group to which it belongs. Or select All.
2. Click the number in the Instances column that corresponds to that workflow.

The system lists all the instances of that schema.

Process ID	Started at	Initiator	Current step	Action
7	Mon Dec 08 14:53:16 EST 2003	dp	ExportMasterData	<a href="#">Terminate</a>

[Back](#) [Refresh](#)

**Note:** Instances that show in red are instances of Fail-To-Execute Step steps. For more information, see "Fail-To-Execute Step".

3. Click Terminate next to the instance that you want to stop.
4. Click OK.

The instance is stopped and is removed from the list of instances.

**Note:** Terminate stops only the workflow instance itself. It does not cancel any work that the instance may have initiated (such as tasks that were sent or requests placed in the Simulation Engine or Business Logic Engine queues). These items must be cancelled manually.

See also

"Creating Workflows" "Starting Workflow Instances"

"Scheduling Workflow Instances"

## Creating or Editing a Schema Group

Schema groups affect only the display within the Workflow Manager. A schema can belong to multiple groups. Also, a schema group can be public (viewable by all users who log into Workflow Manager) or private (viewable only by you).

You can edit any schema group that you created; this has no effect on the workflow schemas themselves. You cannot edit schema groups created by other users.

### To create or edit a schema group:

1. On the Workflow Management page, do one of the following:
  - To create a new schema group, click New at the top of this page.
  - To edit a schema group, select the group from the drop-down list at the top of this page. Then click Modify.

The Workflow Manager displays the following page:

**Name** \* DF/DR schemas

**Description**

**Available Schemas**

- Partner Plan collaboration
- Space\_consumption\_Alert
- Stockout\_Alert
- CPFR\_Step\_1
- Stockout\_Alert\_Per\_Store
- Export Dynamic Data
- Export Master Data
- Create Division
- CreateCombinations
- Create User

**Selected Schemas**

- Import Replenishment Proposal
- Import Replenishment Requirements
- Import Stock Data from DF
- Run Immediate Batch in DF/DR
- Run Scheduled Batch In DF/DR
- Upload data to DF/DR

**Permission** Public

\* - Mandatory fields

Cancel X OK ✓

2. For Name, type a unique name for this schema group.
3. For Description, type an optional description.
4. Specify the workflow schemas to include in this group. To do so, move schemas from the left list to the right list.
5. For Permission, click Public or Private, depending on whether you want other users to be able to see this schema group.
6. Click .

See also

"Creating Workflows"

## Deleting a Schema Group

You can delete any schema group that you created; this has no effect on the workflow schemas themselves. You cannot delete schema groups created by other users.

### To create or edit a schema group:

1. On the Workflow Management page, select the group from the drop-down list at

the top of this page.

2. Click Delete.
3. Click OK to confirm the deletion.

## Viewing the Workflow Process Log

The workflow process log displays information on all the workflow instances that have run or that are running.

### To view the process log:

1. On the bottom of the Workflow Management page, click Process Log.

The Process Log page appears.

View Processes:		All	View			
PID	Schema Id	Initiator	Start Time	End Time	Last Step	Status
1	72	dp	2003-12-08 14:51:41.0	2003-12-08 14:51:42.0	RunBatch	Completed
2	72	dp	2003-12-08 14:51:53.0	2003-12-08 14:51:53.0	RunBatch	Completed
3	72	dp	2003-12-08 14:52:30.0	2003-12-08 14:52:30.0	RunBatch	Completed
4	72	dp	2003-12-08 14:52:33.0	2003-12-08 14:52:33.0	RunBatch	Completed
5	72	dp	2003-12-08 14:52:36.0	2003-12-08 14:52:36.0	RunBatch	Completed
6	65	dp	2003-12-08 14:53:19.0	2003-12-08 14:53:19.0	CreateDivision	Completed
8	67	dp	2003-12-08 14:54:51.0	2003-12-08 14:54:53.0	Compos	Completed
9	66	dp	2003-12-08 14:55:07.0	2003-12-08 14:55:07.0	DataAlignment	Completed

Back Refresh

### To filter process log entries:

1. Select the required filter from the View Processes drop-down menu.
2. Click View.

The filtered processes are shown.

See also:

"Viewing Workflow Status"

## Recovery and the Workflow Engine

Each time it starts up, the Workflow Engine checks to see if there are any workflow instances that are running, that is, an instance whose status is not completed or terminated. For each instance that is currently running, the engine performs a recovery procedure.

An instance is considered to be in mid-step and therefore running, even when it is between steps. The current step is the last step that was running when the crash happened.

For information on standard workflow messages, see "Tips and Troubleshooting".

## Web Services for Demantra Workflows

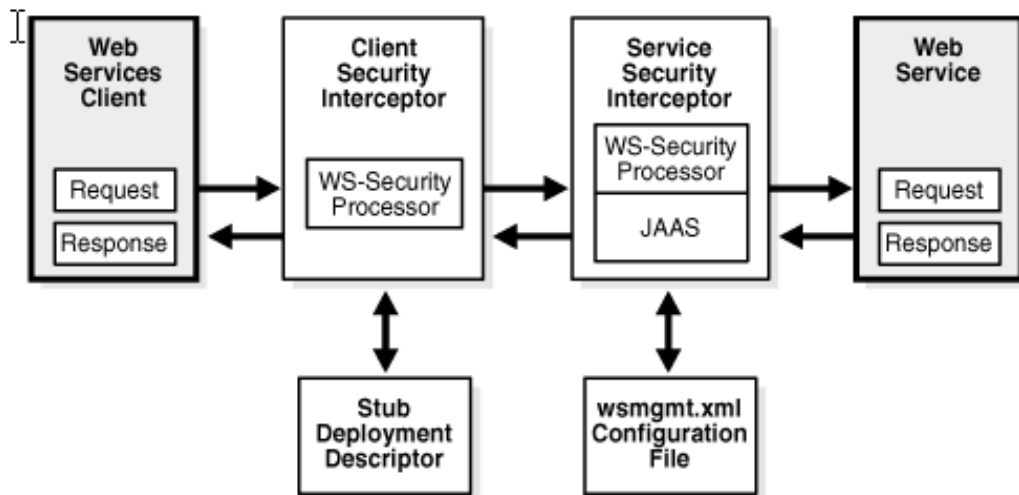
A Demantra workflow can be called as a web service. There are four methods available (see Web Service Methods below for detailed explanations of each):

- GetWFProcessStatus
- RunWFProcess
- RunWFProcessWithContext
- TerminateWFProcess

**Note:** The workflow web service runs only on the Oracle Application Server and WebLogic.

## Web Service Architecture

Demantra's Web Service is developed to align with the Oracle Application Server WS architecture. The following diagram illustrates the overall architecture of WS-Security architecture for the Oracle Web service stack.



## Web Services Library Overview

The following tables identify the methods and parameter types for calling Demantra



web services.

---

Name:	Demantra Workflow Web Service name="MSC_WS_DEMANTRA_WORKFLOW"
Description:	Provides web services interface for the activation/termination and monitoring of Workflow processes.
WSDL:	http://localhost/demantra/MSC_WS_DEMANTRA_WORKFLOWSoapHttpPort
Last Modified:	January 2008

---

### Class Descriptions

---

Dictionary	Description:	This class contains an implementation of a Dictionary, which hold an array of Entries. Each Entry contains Key, Value, and Type. This Dictionary will be used for the WF Process, and will be a part of a Web Service implementation.
	Type:	Complex Type
	Required:	No
	Default:	No
Entry	Description:	This class represents an Entry object that contains a Key that points to a specific Value. This object will be used in the WF Process as part of the Web Service implementation, and will hold Attributes, Levels, Members, Population, Dates, etc. Always assume case sensitivity.
	Type:	Complex Type

---

key	Required:	No
	Default:	No
	Description:	The key is a member attribute of the Entry it is a unique identifier (hash).
	Type:	string
	Required:	Yes
numericValue	Default:	No
	Description:	A numeric value variable that may be contained in the Entry; will usually hold level attribute values.
	Type:	Double
	Required:	No
	Occurrence:	Minimum 0 Maximum 1
populationValue	Description:	A population descriptor that may be contained in the Entry.
	Type:	Population
	Required:	No
	Occurrence:	Minimum 0 Maximum 1
	Description:	A date value that may be contained in the Entry; will usually hold level attribute values.
dateValue	Type:	dateTime
	Required:	No

stringValue	Occurrence:	Minimum 0 Maximum 1
	Description:	A string value that may be contained in the Entry; will usually hold level attribute values.
	Type:	String
	Required:	No
booleanValue	Occurrence:	Minimum 0 Maximum 1
	Description:	Boolean value used for the internal parameter in our application which needed inside the process (e.g. - a flag that indicates whether we need to create new combinations, in new member step)
	Type:	Boolean
	Required:	No
Population	Occurrence:	Minimum 0 Maximum 1
	Description:	This class holds a Population object that will be used for the Web Service implementation. The Population contains Dates (range from-until) and Filters (level-members).
	Type:	Complex Type
	Required:	No
fromDate	Default:	No
	Description:	This element is the start date of the population date range.

---

untilDate	Type:	dateTime
	Required:	Yes
	Default:	No
	Description:	This element is the end date of the population date range.
levelIds	Type:	dateTime
	Required:	Yes
	Default:	No
	Description:	This element represents the array of level ids that define the dimensions of the population. Each element in this array corresponds to LevelMembers held in memberIds.
memberIds	Type:	int
	Required:	Yes
	Occurrence:	1 to unbound
	Description:	In this complex object we finally hold the members of the population related to selected population dimensionality.
	Type:	LevelMembers
	Required:	Yes
	Occurrence:	1 to unbound

---

LevelMembers	Description:	This class holds an array of Member Ids that will be used for the Web Service implementation - these Members related to a Level and are part of a Population Object.
	Type:	Complex Type
	Required:	No
	Default:	No
members	Description:	Array of Member Ids that will be used for the Web Service implementation - these Members related to a Level and are part of a Population Object.
	Type:	int
	Required:	Yes
	Occurrence:	1 to unbound

## Method Details

### RunWFProcess

Description:	Run a Workflow Schema. This will be the same like Starting a WF Process via the Workflow Page. WF Schema Name is Case Sensitive.
Access:	remote
Return Type:	long
Output:	None

---

Implemented In:	com.demantra.common.webServices.workflow w.WorkfloServices
-----------------	---------------------------------------------------------------

Arguments:

---

***schemaName***

---

Description:	The name of the Workflow schema that is requested for remote execution.
--------------	-------------------------------------------------------------------------

Type:	String
-------	--------

Required:	Yes
-----------	-----

Default:	No (Case sensitive)
----------	---------------------

---

---

***runSynch***

---

Description:	Specifies whether the workflow schema is to be executed is synchronized or asynchronous mode.
--------------	-----------------------------------------------------------------------------------------------

Type:	Boolean
-------	---------

Required:	Yes
-----------	-----

Default:	No
----------	----

---

---

**RunWFProcessWithContext**

---

Description:	Retrieve a query object of the posts to blogs specified. Maximum of 50 records returned. WF Schema Name is Case Sensitive.
--------------	----------------------------------------------------------------------------------------------------------------------------

---

---

Access:	remote
Return Type:	long
Output:	false
Implemented In:	com.demantra.common.webServices.workflow. w.WorkfloServices

---

---

Arguments:

---

***schemaName***

---

Description:	The name of the Workflow schema that is requested for remote execution.
Type:	String
Required:	Yes
Default:	No (Case Sensitive)

---

---

***dictionary***

---

Description:	This dictionary contains execution context parameters for the Workflow. Entry/Parameters are case sensitive.
Type:	Dictionary
Required:	Yes
Default:	No

---

---

***runSynch***

---

Description:	Specifies whether the workflow schema is to be executed is synchronized or asynchronous mode.
--------------	-----------------------------------------------------------------------------------------------

---



Type:	boolean
Required:	Yes
Default:	No

---

### GetWFProcessStatus

---

Description:	Returns the current Status and Step Name of the WF Process. e.g. - 1. Process Active [ImportSKU]. 2. Process Terminated [RunWS]. 3. Process Completed [ExportSKU].
Access:	remote
Return Type:	String
Output:	false
Implemented In:	com.demantra.common.webServices.workflow.WorflowServices
Arguments:	

#### ***processId***

---

Description:	A parameter that identifies the workflow process that requires monitoring.
Type:	long
Required:	Yes
Default:	No

---



---

### TerminateWFProcess

Description:	Terminate the current (identified by process ID) Active WF Process..								
Access:	remote								
Return Type:	No								
Output:	false								
Implemented In:	com.demantra.common.webServices.workflow. w.WorkfloServices								
Arguments:	<div> <div><b><i>processId</i></b></div> <table> <tr> <td>Description:</td><td>A parameter that identifies the workflow process that requires monitoring.</td></tr> <tr> <td>Type:</td><td>long</td></tr> <tr> <td>Required:</td><td>Yes</td></tr> <tr> <td>Default:</td><td>No</td></tr> </table> </div>	Description:	A parameter that identifies the workflow process that requires monitoring.	Type:	long	Required:	Yes	Default:	No
Description:	A parameter that identifies the workflow process that requires monitoring.								
Type:	long								
Required:	Yes								
Default:	No								

## Integration with JAAS

OracleAS Web Services provides an implementation of Java Authentication and Authorization Service (JAAS) for J2EE applications that is fully integrated with J2EE declarative security. This allows Demantra to take advantage of the JAAS constructs such as principal-based security and pluggable login modules. OracleAS Web Services Security provides out-of-the-box JASS authentication login modules that allow J2EE applications running on OracleAS Web Services to leverage the central security services of Oracle Identity Management.

The JAAS Provider ensures secure access to and execution of Java applications, and integration of Java-based applications with Oracle Application Server Single Sign-On.

Demantra has implemented a custom login module that is deployed in the OracleAS.

## Configuring the Security Provider of the Application Server

Demantra has implemented a custom login module that is deployed in the OracleAS. The following procedure configures the application server to use this login module:

1. Connect to Oracle Enterprise Manager.
2. Deploy Demantra
3. Define Security Provider as follows:
  - Select – 'Administration' -> 'Security Providers' (by Go to Task) -> 'demantra' application -> edit (demantra) -> 'Change Security Provider'.
  - In the Drop Down of 'Security Provider Type', select 'Custom Security Provider'.
  - In the 'JAAS Login Module Class' field, set 'com.demantra.common.authentication.DemantraLoginModule'.
  - Click OK.

**Note:** This step can be done only through a deployment of the Demantra application
4. Return to the Home Page and select 'Web Service'.
  - If no web services are found, open the link of the WF Web Service: [http://\(ROOT\)/demantra/MSO\\_WS\\_DEMANTRA\\_WORKFLOWSoapHttpPort](http://(ROOT)/demantra/MSO_WS_DEMANTRA_WORKFLOWSoapHttpPort)
  - Then refresh the Enterprise Manager page.
  - Click on the link: MSO\_WS\_DEMANTRA\_WORKFLOWSoapHttpPort.
  - Select 'Administration' -> 'Enable/Disable Features'
  - Move 'Security' to the 'Enabled Features' box and then click OK.
  - Make sure that the 'Security' Feature is marked as Enabled.
5. On the same page, select the 'Edit Configuration' icon of the 'Security' Feature, Press the 'Inbound Policies' button, and in the 'Authentication' tab, mark the checkbox 'Use Username/Password Authentication' and select 'Password Type' = 'Plain Text'. Click OK.
6. Test the Workflow Web Service by providing User Name & Password in the WS-Security section.

## Details of the Demantra custom login module

Location: **package** com.demantra.common.authentication;

**DemantraLoginModule.java**

Methods

- `public void initialize(Subject subject, CallbackHandler callbackHandler, Map sharedState, Map options)`

Initialize this LoginModule.

Parameters:

**subject** the Subject to be authenticated.

**callbackHandler** a CallbackHandler for communicating with the end user (prompting for usernames and passwords, for example).

**sharedState** shared LoginModule state.

**options** options specified in the login Configuration for this particular LoginModule.

- `public boolean login() throws LoginException`

Authenticate the user by prompting for a username and password.

**Returns:** true if the authentication succeeded, or false if this LoginModule should be ignored. **Throws:**

FailedLoginException if the authentication fails.

LoginException if this LoginModule is unable to perform the authentication.

- `public boolean commit() throws LoginException`

This method is called if the LoginContext's overall authentication succeeded (the relevant REQUIRED, REQUISITE, SUFFICIENT and OPTIONAL LoginModules succeeded).

If this LoginModule's own authentication attempt succeeded (checked by retrieving the private state saved by the login method), then this method associates a Principal with the Subject located in the LoginModule. If this LoginModule's own authentication attempted failed, then this method removes any state that was originally saved.

**Returns:** true if this LoginModule's own login and commit attempts succeeded, or false otherwise.

**Throws:** LoginException if the commit fails.

- `public boolean abort() throws LoginException`

This method is called if the LoginContext's overall authentication failed. (the relevant REQUIRED, REQUISITE, SUFFICIENT and OPTIONAL LoginModules did not succeed).

If this LoginModule's own authentication attempt succeeded (checked by retrieving the private state saved by the login and commit methods), then this method cleans up any state that was originally saved.

**Returns:** false if this LoginModule's own login and/or commit attempts failed, and true otherwise.

**Throws:** LoginException if the abort fails.

- public boolean logout() throws LoginException

Logout the user.

This method removes the Principal that was added by the commit method.

**Returns:** true in all cases since this LoginModule should not be ignored.

**Throws:** LoginException if the logout fails.



---

## Managing Worksheets

Worksheets are created within the user interfaces, but you can manage them from the Business Modeler.

This chapter covers the following topics:

- Viewing the Worksheets
- Changing Worksheet Ownership
- Changing Worksheet Access
- Deleting a Worksheet

### Viewing the Worksheets

There may be a large number of worksheets within your system. You can use the Worksheet Manager to view the worksheets, change their ownership, and delete worksheets. The Worksheet Manager also keeps track of changes made to the worksheets.

#### To view the worksheets:

1. Log onto the Business Modeler as described in "Logging onto the Business Modeler."
2. Click Tools > Worksheet Management.

The Worksheet Manager is displayed. In this window, a table displays a row for each worksheet with the following information:

---

Owner	Demantra user who has permission to modify this worksheet. By default, this is the user who created the worksheet.
-------	--------------------------------------------------------------------------------------------------------------------

---

Permission Type	Specifies whether this worksheet is private (seen only by its owner) or public (visible to all users).
Open	Indicates whether this worksheet is currently being used.
Last Access	Indicates when this worksheet was last opened.
Accessed By	Indicates the user who last opened this worksheet.

## Changing Worksheet Ownership

Only the owner of a worksheet can edit it.

### To change who owns a worksheet:

1. Log onto the Business Modeler as described in "Logging onto the Business Modeler."
2. Click Tools > Worksheet Management.  
The Worksheet Manager is displayed.
3. In the row corresponding to the worksheet, click the Owner field.
4. Select a new owner and click OK.

## Changing Worksheet Access

If a worksheet is private, it can be seen only by its owner. If it is public, it is visible to all users.

### To change who can access a worksheet:

1. Log onto the Business Modeler as described in "Logging onto the Business Modeler."
2. Click Tools > Worksheet Management.  
The Worksheet Manager is displayed.



3. In the row corresponding to the worksheet, click the Permission Type field.
4. Click Private or Public and click OK.

## Deleting a Worksheet

### **To delete a worksheet:**

1. Log onto the Business Modeler as described in "Logging onto the Business Modeler" .
2. Click Tools > Worksheet Management.  
The Worksheet Manager is displayed.
3. Click the row corresponding to the worksheet.
4. Click Delete.  
Demantra asks for confirmation.
5. Click OK.



---

## Other Administration

Demantra provides a Web-based interface to perform other, less common administrative tasks, described here.

This chapter covers the following topics:

- Other Web-based Administration Tools
- Logging Messages of the Application Server
- Managing Level Caching
- Viewing and Modifying Cache Properties

### Other Web-based Administration Tools

Browse to the following case-sensitive URL:

`http://server name/virtual directory/admin`

For example:

`http://frodo/demantra/admin`

The following page appears:

# Admin Tools

1. [DB Connection Pool Status](#)
2. [Thread Pool Status](#)
3. [JVM Memory Status](#)
4. [Cache Manager](#)
5. [Logger Manager](#)

Ignore options 1, 2, and 3, which are currently non functional.

## Logging Messages of the Application Server

By default, the Application Server writes logs into the directory `Demantra_root/Collaborator/virtual_directory/portal/logs`. These logs record activity of the server and clients.

To change the behavior of this logging, edit the file `Demantra_root/Collaborator/virtual_directory/portal/conf/logconf.lcf`. In this file, you can specify items such as the following:

- Name and location of the log file
- Maximum size of the log file
- Number of log files to keep

For details, see the comments in `Demantra_root/Collaborator/virtual_directory/portal/conf/logconf.lcf`.

**Important:** If the default language uses a **non-ASCII character set** (such as Korean, Japanese, Chinese, Russian) then the text editor for viewing server log files must support the UTF-8 character set. Otherwise the text may not display correctly.

## Managing Level Caching

Because caching is a trade-off between memory and speed, a new caching mechanism

allows you to specify the needed caching policy on a level-by-level basis, depending on the size and usage patterns of your levels.

### Specifying Level Caching Policies

To specify how to cache a given level, edit the `group_tables_cache` table, as follows:

Field	Purpose
GROUP_TABLE_ID	Specifies the ID of the level, as given in the <code>group_tables</code> table.
CACHE_CAPACITY	<p>If <code>CACHE_TYPE</code> is <code>LRU</code>, <code>CACHE_CAPACITY</code> specifies the maximum number of members of this level that will be cached at any time.</p> <p>This field is ignored when <code>CACHE_TYPE</code> is set to <code>SIMPLE</code>.</p>
CACHE_IS_LAZY	<p>Setting to control member auto load at server startup time or on demand.</p> <p>Default value is "0"</p> <ul style="list-style-type: none"><li>"0" = at startup, cache is loaded on startup</li><li>"1" = on demand, cache is not loaded on startup</li></ul> <p><code>CACHE_TYPE</code> and <code>CACHE_CAPACITY</code> work with <code>CACHE_IS_LAZY</code> to specify whether the cache for this level is completely loaded upon server startup:</p> <ul style="list-style-type: none"><li><code>CACHE_TYPE</code> = <code>SIMPLE</code> will hold all members</li><li><code>CACHE_TYPE</code> = <code>LRU</code> will load up to <code>CACHE_CAPACITY</code> members</li></ul>

Field	Purpose
CACHE_TYPE	<p>Specifies the caching policy for this level. Use one of the following values:</p> <ul style="list-style-type: none"> <li>• SIMPLE - The cache is unlimited for this level. Simple will hold all members in memory as long as the server is running.</li> <li>• LRU - LRU will hold members in memory up to the limit set in CACHE_CAPACITY. Whenever the server loads a member beyond this limit of CACHE_CAPACITY, the least recently used member is flushed from memory.)</li> </ul> <p>The default value is SIMPLE.</p>
CHE_LOAD_EXTRA_FROM	Used to filter startup initialization. See CACHE_LOAD_EXTRA_WHERE.
CACHE_LOAD_EXTRA_WHERE	Used to filter startup initialization. This field and CACHE_LOAD_EXTRA_FROM allow you to define SQL criteria for loading the cache. For example, if the implementation works mostly on current year promotions, then you can add the From part "promotion_dates" and to the Where part "promotion_dates.from_date > '1/1/2006'". In this way, the cache will be initially loaded with promotions from 2006. This does not mean that the cache will be in any way limited to 2006 after its initial load.
ENABLE_STATISTICS	* Specifies whether the server should collect statistics on use of this cache. "0" = false; "1" = true
ENABLE_DEBUG	* Specifies whether the server should record information for possible use in debugging issues related to this cache. "0" = false; "1" = true
* Can be changed later through a Web-based interface.	

By default, Demantra uses the following caching policy for each level:

- CACHE\_TYPE is SIMPLE (meaning that CACHE\_CAPACITY is ignored)
- CACHE\_IS\_LAZY is "0" = false
- ENABLE\_STATISTICS is "0" = false

- `ENABLE_DEBUG` is "0" = false

## Recommended Settings

Any level with more than 10,000 members should be cached. When levels contain 10,000 or more members, system memory may be consumed rapidly and performance significantly affected, especially when retrieving worksheets and data. Therefore, it is recommended that implementations with 10,000 or more members do the following:

- Set `CACHE_IS_LAZY` to '0'.
- Set `CACHE_TYPE` to 'LRU'.
- Set `CACHE_CAPACITY` to the desired number of members to cache per level (i.e. the count of rows returned by `CACHE_LOAD_EXTRA_WHERE` + delta, where delta is the number of expected future members that will be loaded).
- If it is not possible to predefine the range of level members using `CACHE_LOAD_EXTRA_WHERE`, then set `CACHE_CAPACITY` to a value less than 10,000.
- Define `CACHE_LOAD_EXTRA_WHERE` and `CACHE_LOAD_EXTRA_FROM`. This will enable level members to be cached in memory when the application server starts up, rather than retrieving them from the database.

## Viewing and Modifying Cache Properties

1. Browse to the following case-sensitive URL:

`http://server name/virtual directory/admin`

For example:

`http://frodo/demantra/admin`

2. Click Cache Manager.

A page like the following appears.

---

## Cache Manager

Debug: [true](#)

#	Level Name	Objects Type	Cache Type
1	<a href="#">District</a>	members	SIMPLE
2	<a href="#">Promotion Group</a>	members	SIMPLE
3	<a href="#">Brand</a>	members	SIMPLE
4	<a href="#">Category</a>	members	SIMPLE
5	<a href="#">Plans</a>	members	SIMPLE
6	<a href="#">Sales Area</a>	members	SIMPLE
7	<a href="#">Aggregation User Status</a>	members	SIMPLE
8	<a href="#">Optimization Goal</a>	members	SIMPLE
9	<a href="#">Division</a>	members	SIMPLE
10	<a href="#">Scenarios</a>	members	SIMPLE

This page lists each level in your system, along with the CACHE\_TYPE setting for that level.

3. Click a level.

A page like the following appears.



---

# Cache Promotion.members

Level Id: 232

Lazy: false

Size: 500

Statistics: [false](#)

Debug: [false](#)

Capacity:

Memory Size:   Bytes.

## Cache Manager

This screen shows the following details:

---

Level ID	Internal Demantra identifier for the level.
Lazy	Indicates whether the cache for this level is completely loaded upon server startup.  If true, the cache is not loaded on startup.
Size	Indicates the maximum number of members in the cache.
Statistics	Specifies whether the server should collect statistics on use of this cache. Click the true/false link to change this setting.
Debug	Specifies whether the server should record information for possible use in debugging issues related to this cache. Click the true/false link to change this setting.

---

---

Capacity	<p>Specifies the maximum number of members of this level that the cache can include.</p> <p>You can enter a new value and click Update to save the new value.</p> <p>This option is shown only if the CACHE_TYPE is LRU.</p>
Memory Size	<p>Click the Calculate button to see how much memory this cache is currently using.</p>

---

---

## Upgrading Using Oracle Demantra Business Application Language

This chapter describes the Business Application Language tool (BAL) that helps you to compare old and new schemas, and make choices about how to handle differences. The BAL Explorer is launched during the installation process when you select to manually upgrade to the latest application version.

This chapter covers the following topics:

- Overview
- The Upgrade Process
- BAL Explorer Screen Elements
- Using the BAL Explorer

### Overview

When upgrading to new versions of Demantra, your data needs to be changed to the new data schema.

Demantra provides you with a tool called the Business Application Language (BAL) that helps you to compare old and new schemas, and make choices about how to handle differences. It also provides you with the capability to assemble key elements from other schemas and merge them into a target schema (for testing purposes, for example).

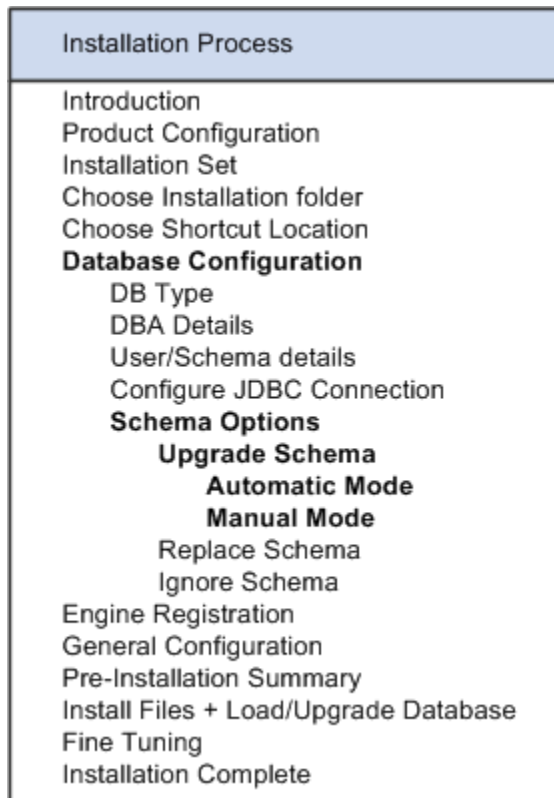
The BAL Explorer is launched during the installation process when you select to manually upgrade to the latest application version. Currently, it is only available when upgrading from Demantra version 7.1.1 to newer versions.

**Important:** The BAL is only available to customers who have implemented the standard pre-configured applications available from Demantra 7.1.1 onwards. Customers who are using the standard Oracle

application configuration with supported modifications like adding their own new levels/series, can use the "Application and Platform Upgrade" option in the installer. Customers who are not using the standard Oracle application configuration should , choose the platform upgrade option which preserves custom configurations and enables new platform functionality. It does not introduce new application objects. If you are uncertain which option to choose, please contact support for guidance.

## The Upgrade Process

For those using Demantra 7.1.1 or later, you can make choices about how you want to migrate your data to the latest application version. The upgrade schema choices are presented when you install Demantra and configure the database. This diagram shows the placement of the upgrade schema options during the installation process:



Two upgrade schema options are available: automatic mode and manual mode. In the automatic mode, the installer upgrades your schema based on a set of default actions:

- New objects are inserted

- Overlapping objects are reconciled by aligning some of the objects to the new application definition and creating duplicate objects where relevant.

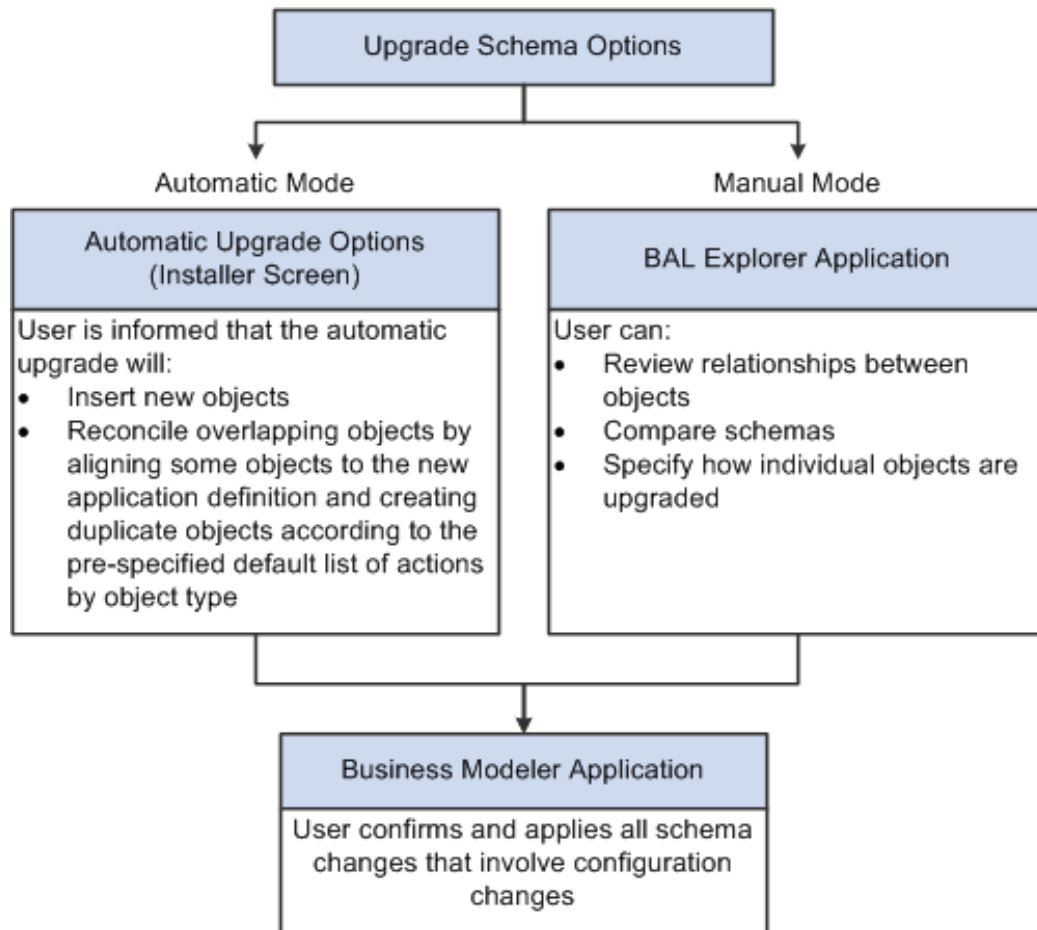
The automatic mode runs the BAL Explorer in the background without any personal interaction.

If you want to make specific choices about how objects are upgraded to the new schema, then you should choose manual mode. This mode launches the BAL Explorer and enables you to:

- Review the relationship between objects
- Compare schemas
- Specify how individual objects are upgraded when conflicts occur between the schemas

The Business Modeler is invoked at the end of each upgrade mode, enabling you to apply any configuration changes identified in the new schema.

The following diagram shows the process involved with the two upgrade schema options:



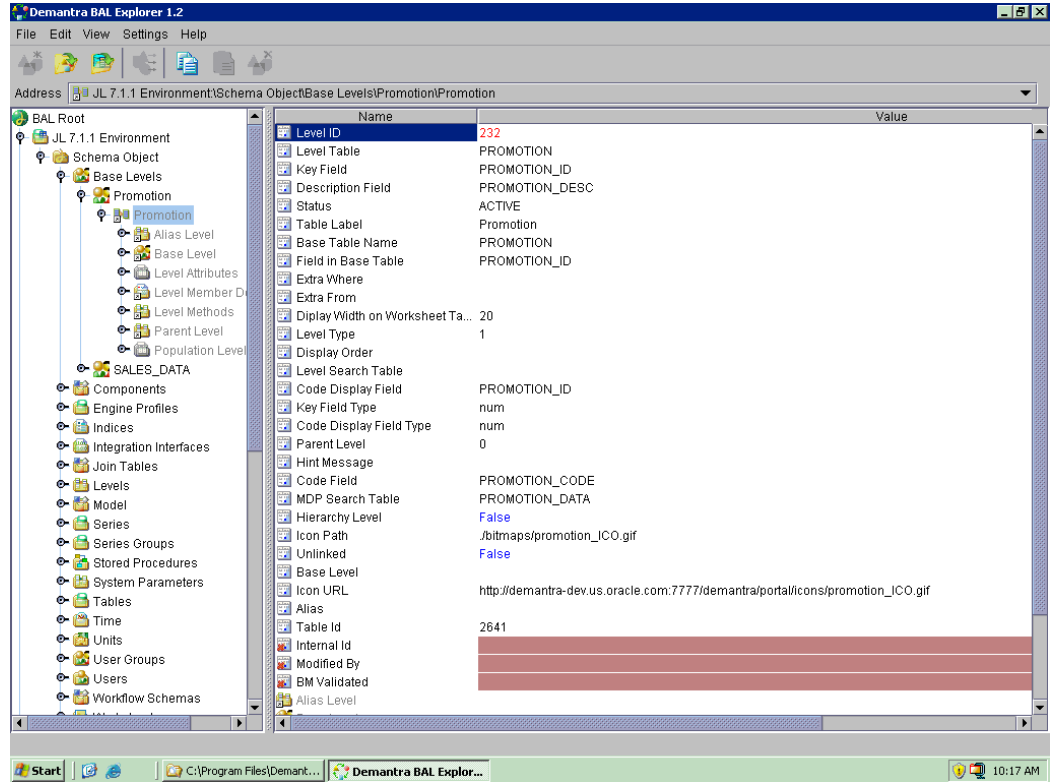
When the upgrade is complete, the Business Modeler displays the Upgrade Complete screen, which summarizes the objects that were upgraded.

**Important:** Oracle strongly recommends that you backup your database before attempting to upgrade from an earlier schema.

## BAL Explorer Screen Elements

The BAL Explorer main screen, shown below, is comprised of two main components:

- Schema Browser pane (on the left)
- Object Details pane (on the right)



## The Schema Browser Pane

The Schema Browser displays all the schemas currently defined in the BAL Explorer. These schemas can be expanded to display the objects contained within the schema. Depending on the type of object you select, the attributes are displayed with icons representing their data type. For example, time attributes are displayed with a clock icon.

The Schema Browser also provides information about how objects are related.

- Black represents a container relationship: object A contains object B (shown in black). Object B has meaning only within object A. Object B is shown in black within object A. For example, a list for a series has meaning only within that series.
- Gray represents a referring relationship: object A refers to object B. Object B has meaning independent of object A. Object B is shown in gray within object A. For example, a worksheet refers to the Demand series.

## Object Details Pane

The Object Details pane provides information about the object currently selected in the Schema browser, including:

- Database fields that correspond to the configuration settings in the Demantra platform tools (Business Modeler and Workflow Editor).
- Folders of objects that are contained within or referred to by the selected object.

The Object Details pane also displays any mismatches between the selected schema (the database) and the latest Demantra schema (the configuration files). Mismatches between the database and the configuration files appear as follows:

- Green: Field is in the database but not found in the latest BAL configuration files.
- Red: Field is in the BAL configuration files but not in the database. Missing fields are also indicated by the red x on the field icon.

## Legend

For a better understanding of the many icons used in the BAL Explorer, a legend is available.

To view the legend:

1. From the View menu, select Legend. The Legend window appears on the far right.
2. Close the Legend window when you are not using it.

## Status Bar

To view the progress of an upgrade, you can activate the status bar.

To turn on the status bar, from the View menu, select Status Bar.

## Using the BAL Explorer

This section includes the following:

- Understanding Objects, Schemas, Folders, Files and Repositories
- Schema Procedures
- Folder Procedures
- Upgrading Procedures
- Troubleshooting

## Understanding Objects, Schemas, Folders, Files and Repositories

A schema object is a logical data storage structure. Schema objects do not have a



one-to-one correspondence to physical files on the disk that stores their information. Oracle stores a schema object logically within a tablespace of the database. The data of each object is physically contained in one or more of the datafiles of the tablespace. For example, each Demantra object generally corresponds to a row in a particular table of the Demantra database, and further details might be located in other tables.

A schema is a collection of schema objects that also defines how they relate to one another. No relationship exists between schemas and tablespaces: a tablespace can contain objects from different schemas, and the objects for a schema can be contained in different tablespaces.

Configuration files describe the target Demantra schema. BAL compares these configuration files to the source database. Presently, Demantra recommends using standard configuration files for upgrading purposes. Customized configuration files might be available in future releases.

The BAL Explorer organizes objects into folders for convenience and readability. By default, the BAL Explorer groups the objects by type and displays each type in separate folders, as follows:

- Base levels
- Components
- Engine profiles
- Indices
- Integration interfaces
- Join tables
- Levels
- Model
- Series
- Series groups
- Stored procedures
- System parameters
- Tables
- Time
- Unit

- User groups
- Users
- Workflow schemas
- Worksheets

The default folders used to display different types of objects have no inherent meaning aside from organizing the object information for viewing purposes.

When upgrading to a new version of the Demantra software in manual mode, you can create one or more schema aliases that you can compare with the latest Demantra schema. You can represent an existing schema as is, or you can create new schemas that are customized with objects from a full schema for testing purposes. You can create additional folders within schemas for testing purposes and copy objects from other schemas into them. Successfully upgrading the contents of a personal folder will add these objects to the parent schema.

**Note:** When you upgrade from the installer in manual mode, a schema is created automatically.

When the BAL Explorer first connects to a schema, it creates a BAL repository, saving the information it needs in its own internal tables. This repository can reside either within the same database user or a different database user so that nothing new is introduced into the original repository.

## Schema Procedures

**Note:** If you are upgrading your schema from the installer, the schema definition is created automatically. Go to Upgrading Procedures, page 65-15 for detailed instructions.

### Creating a Schema Connection

To view the contents of a Demantra database or compare databases, you must create a visual representation of the schema. This alias schema appears in the Schema Browser pane.

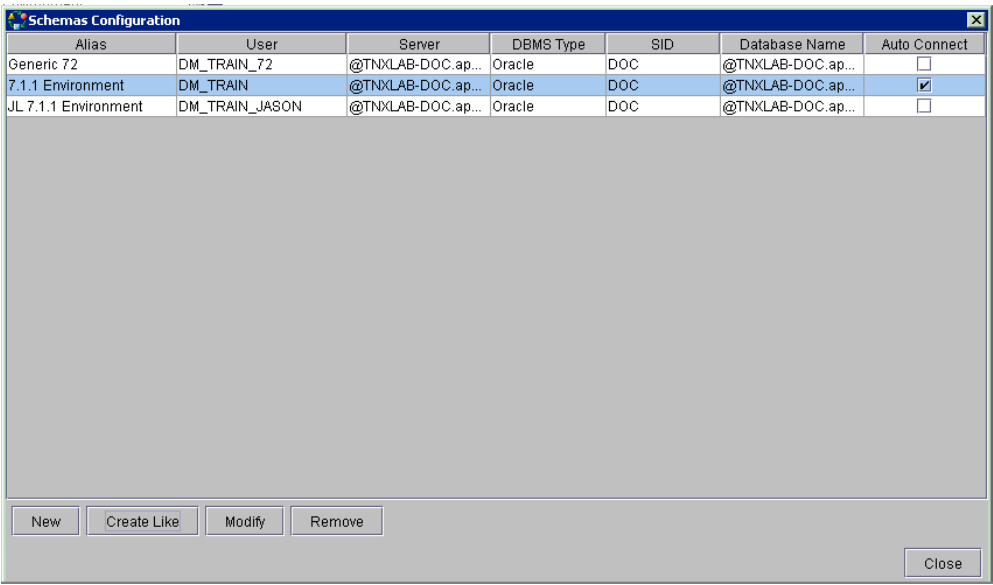
**Caution:** Do not use any unusual characters when defining the name such as apostrophes, brackets, and so on. This usage might cause your comparison report to not save.

**Important:** When you create a schema, you can create the repository in

the current database user or a different user. If you choose the latter, make sure that the second database user exists before creating the schema.

To create a schema connection:

1. Click Open Schema. The Schema Configuration dialog box appears.



2. Click New. The New Connection dialog box appears.

**New Connection**

**Schema**

Database Type: Oracle

Server Name:

User Name:

Password:

SID:

Port: <1521>

Database Name:

Alias:

Test Connection

☐ Define Separate Repository

Cancel Apply

3. Enter the following data in the following fields:

Field	Description
Database Type	Select your database type.
Server Name	The name of the server where your database resides.
User Name	The user name of the database.
Password	The password of the database.
SID	If you are using an Oracle database, enter the SID.
Port <1521>	The port your database is using. The default is 1521.
Alias	The schema alias to appear in the Schema Browser.

4. Select Define Separate Repository if you do not want the repository created in the current database tables. The Repository information area appears in the New Connection dialog box:

**Modify Connection**

Schema	Repository
Database Type: Oracle	Database Type: Oracle
Server Name: @TNXLAB-DOC.aps.jdedwards.com	Server Name: @TNXLAB-DOC.aps.jdedwards.com
User Name: DM_TRAIN	User Name: PTP
Password: d	Password: p
SID: DOC	SID: DOC
Port: <1521> 1521	Port: <1521> 1521
Database Name:	Database Name:
Alias: 7.1.1 Environment	
<input type="button" value="Test Connection"/>	<input type="button" value="Test Connection"/>

☒ Define Separate Repository

**Note:** A second database user must already exist for the repository to be successfully defined.

5. Click Test Connection to see if your schema is properly configured.
6. Click Apply. The new schema alias appears in the Schemas Configuration dialog box.
7. Click AutoConnect if you always want the schema to be active when you are working in the BAL Explorer.
8. Click Close. The new schema appears in the Schema Browser Pane.

## Editing a Schema Connection

To edit a schema connection:

1. Click Open Schema.
2. Select the schema you want to modify from the list of schema aliases.
3. Click Modify.
4. Make any changes necessary to your schema.
5. Click Apply.

6. Click AutoConnect if you always want the schema to be active when you are working in the BAL Explorer.
7. Click Close.

### **Creating a Schema Connection Based on Another Schema Connection**

To create a schema connection based on another schema connection:

1. Click Open Schema.
2. Select the schema connection you want to base your new schema connection on.
3. Click Create Like. The New Connection dialog box appears with the configuration details of the original schema.
4. In the Alias field, change the schema alias name.
5. Make any additional changes to the new schema.
6. Click Apply. The new schema appears in the list of schemas.
7. Click Close. Your new schema appears in the Schemas Browser Pane.

### **Deleting a Schema Connection**

To delete a schema:

1. Click Open Schema.
2. Select the schema connection you want to delete.
3. Click Remove.
4. Click Yes when prompted for confirmation. The schema alias is removed from the list of schemas in the schemas configuration dialog box.
5. Click Close. The schema connection no longer appears in the Schemas Browser pane.

### **Connecting to the Schema Repository**

If you did not select the autoconnect option for your schema, you can manually connect to your schema when necessary.

To connect to a schema:

1. In the Schema Browser, right-click the schema to which you want to connect.

2. Select Connect. Your schema becomes active.

### Disconnecting from a Schema Repository

To disconnect from a schema:

1. In the Schema Browser, right-click the schema that you want to disconnect.
2. Select Disconnect. Your schema becomes inactive and appears gray in the Schema Browser.

### Rebuilding a Schema Repository

You rebuild a schema repository if you have made configuration changes to the database since the schema was created in the BAL Explorer (for example, using Business Modeler). Rebuilding the schema repository incorporates all of the latest database contents, including any objects that were already upgraded into the schema. However, any personal folders associated with the schema that have not been upgraded are discarded when the schema is rebuilt.

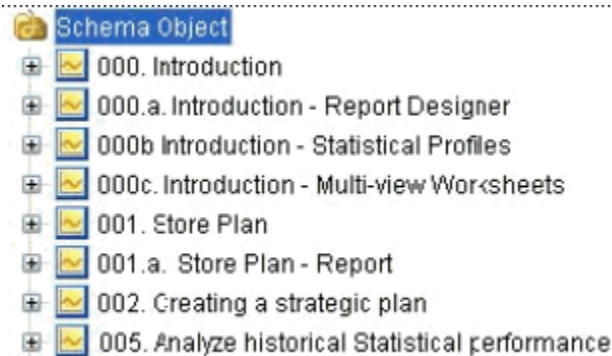
To rebuild a schema repository:

1. In the Schema Browser, right-click the schema that you want to rebuild.
2. Select Rebuild. Your schema repository is rebuilt.

**Note:** Rebuilding your schema repository also has the effect of connecting your schema

### Changing Object Order and Display

By default, the BAL Explorer sorts the display of objects by their types within a schema or folder. You can change this default and display the objects alphabetically instead. The alphabetical display may appear as follows:



To change object order and display:

1. Right-click the schema or folder that you want to display differently. A check next to the View by Type option indicates that the option is currently selected.
2. Select View by Type to change the appearance.

## Folder Procedures

Folders are used for testing purposes and to move objects from one schema to another. If you want to upgrade specific objects from schema A to schema B, you create a folder in schema B. All objects the folder contains will become valid objects in the target schema if the upgrade process completes successfully. The folder itself does not have meaning to Demantra.

### Creating Folders

To create a folder:

1. Right-click the schema in which you want the folder to be located.
2. Select New Folder. The folder icon appears in the Schema Browser.
3. Type the name of the folder and press Enter.

### Renaming Folders

To rename a folder:

1. Right-click the folder that you want to rename.
2. Select Rename.
3. Type the new folder name and press Enter. The name has been changed.

### Deleting Folders

To delete a folder:

1. Right-click the folder you want to delete.
2. Select Delete.
3. Click OK when Prompted to confirm the deletion. The folder and its contents are deleted.

### Copying Objects from a Schema to a Folder

Within the BAL Explorer, you can copy objects from more than one schema to a folder.



You can use the Shift or Control keys to copy more than one object at a time. The Shift key enables you to copy adjacent objects. The Control key enables you to copy multiple objects that are not adjacent.

After the objects are pasted into the target folder, they appear in blue italics to show that they were not upgraded to the target folder.

To copy an object from a schema to a folder:

1. Right-click the object that you want to copy to a folder.
2. Select Copy.
3. Right-click the folder to which you want to copy the object.
4. Select Paste. The object appears in the target folder.

### Deleting an Object from a Folder

Objects can be deleted from a folder, but not from a schema.

To delete an object from a folder:

1. Right-click the object that you want to delete from a folder.
2. Select Delete.
3. Click OK when prompted to confirm the deletion. The object is deleted from the folder, but it remains in the original schema.

## Upgrading Procedures

**Important:** Before upgrading to a new version, it is strongly recommended that you take a backup of your existing database.

### Setting the Global Upgrade Preferences

When you upgrade an existing schema to a new schema definition, conflicts can occur between the two schemas. The Upgrade Settings option provides guidance to the BAL Explorer when performing an upgrade. New objects are always inserted. Overlapping objects can be duplicated, aligned to the new object, or ignored.

There is a set of default options available for new and overlapping objects. The default options by object type are listed in a table below. When you choose the "Automatic Upgrade" option when installing, you are not able to interact with the BAL Explorer to change the default options. When you choose the "Manual Upgrade" option when installing, an upgrade package is created by BAL with the same default options but you can choose to change or override the default options in the BAL Explorer.

By default:

- New objects are inserted.
- Overlapping objects are either duplicated, aligned or ignored based on the object type itself.
- Customizations to standard integration interfaces are not preserved during the automatic upgrade process.

### Integration Interfaces and Data Profiles

Type of Upgrade	Result
Automatic	BAL Explorer aligns modified standard integration interfaces and associated data profiles to the new version,
Manual	Same as the Automatic Upgrade except you can change the default upgrade action.

After you upgrade:

- Review your pre-upgrade schema backup for any standard integration interfaces that you may have customized. These interfaces would have been aligned to the new version in the upgraded schema.
- Add back those customizations to the new version of the integration interface in the upgraded schema.

### Worksheets

Type of Upgrade	Result
Automatic	BAL Explorer retains customized standard worksheets in your existing schema and duplicates them in the new schema. See 'Naming convention for duplicate objects' for more details.
Manual	Same as the Automatic Upgrade except you can change the default upgrade action.

**Note:** Only those standard worksheets that have changes between the

two versions will be duplicated. If a worksheet has not changed between your version and the new upgrade version, it is not duplicated.

After you upgrade, Oracle recommends that you evaluate each worksheet and:

- Manually merge your pre-upgrade customizations from the pre-upgrade version to the new version and use the new version of the worksheet.
- Optionally, delete the old unused worksheet so as to prevent the application from having too many old and unused worksheets.

**Note:** Workflow schemas also follow the same upgrade behavior as worksheets.

### Naming Convention for Duplicate Objects

For objects that are duplicated during the upgrade, certain naming conventions are followed. During an upgrade, if an object has to be duplicated, the following naming conventions are used:

- The existing object in your schema will be renamed according to the below convention:

Objectname\_version number being upgraded to\_build number  
being upgraded to

#### Example

If you are upgrading from version 7.1.1 to 7.3.0.1 and the upgrade process has to duplicate a worksheet called 'Demand Analysis Item & Org', then the existing 7.1.1 worksheet is renamed as 'Demand Analysis Item & Org\_730\_125'. Here version number = 7.3.0, build number = 125 (a fictional build number). The build and version numbers corresponding to the new release you are upgrading to can be found in the table version\_details after the upgrade is complete. (Every released version corresponds to a specific version number and build number. For example, version 7.2.0.2 corresponds to version number 7.2.0 and build number = 226, version 7.3.0 corresponds to version number 7.3.0 and build number = 68.)

- The new object introduced in your schema by the upgrade process is named according to the same name in the Demantra standard schema, with no suffixes.

In the above example, the new version of the worksheet will be introduced as 'Demand Analysis Item & Org', with no suffixes

**Important Note:** There is a change in the naming behavior of duplicate objects from earlier versions (7.2.0.2 and 7.3) to the behavior in version 7.3.0.1:

- From version 7.2.0.2 until version 7.3, new/duplicate objects were inserted with a suffix of the new version and the existing objects were left intact. For example,

when upgrading from version 7.1.1 to version 7.3, the new 7.3 version of a worksheet called 'Demand Analysis Item & Org' would have been inserted as 'Demand Analysis Item & Org\_730', while the existing worksheet would be left intact as 'Demand Analysis Item & Org'.

- From version 7.3.0.1 onwards, the new/duplicate worksheets are inserted without a suffix with the same name in the Demantra standard schema, while the existing object will be suffixed with the version and build number as explained above.

Therefore, if your schema has gone through multiple application upgrades prior to version 7.3.0.1, you may see objects with a '\_720' or a '\_730' suffix, which means that these objects were introduced during an upgrade to 7.2.0.2 or 7.3.0 respectively and you may see objects with a '\_730\_125' suffix, which means that these objects were suffixed and 'archived' during an upgrade to version 7.3.0.1 ( In this example, version number = 7.3.0 and build number = 125).

#### **Table of Default Upgrade Actions for Overlapping Objects**

This table describes the upgrade reconciliation actions for overlapping objects when you choose the Default upgrade option:

<b>Object</b>	<b>Sub-Objects</b>	<b>Reconciliation Action</b>	<b>Comments</b>
Series	-	Align	Existing series definition is aligned to the new series definition.

Object	Sub-Objects	Reconciliation Action	Comments
Worksheet	-	Duplicate	<p>If there is a change in the worksheet definition between the existing version and the new version, then the new version of worksheets are created as duplicate objects. Existing worksheets are retained in their original version.</p> <p>After the upgrade, it is recommended that you manually merge your pre-upgrade customizations from the pre-upgrade version to the new version and use the new version of the worksheet thereafter.</p>
Level	-	Ignored; sub-objects evaluated.	The level definition is retained. Sub-objects of level are handled as indicated below.
Level	Method	Align with Merge; method input.output arguments evaluated.	Existing methods are aligned to new version.
Level	Method Input Arguments	Align	Input arguments are aligned to the new version.
Level	Method Output Arguments	Align	Output arguments are aligned to the new version.

<b>Object</b>	<b>Sub-Objects</b>	<b>Reconciliation Action</b>	<b>Comments</b>
Level	Level Attributes	Align with merge	Level attributes are aligned and merged so that post upgrade, both customer level attributes and level attributes in the new version are included.
Level	Unit Levels	Align with merge	The customer unit levels are aligned and merged with the new version unit levels. After the upgrade, the original customer unit levels are included with any new unit levels.
Level	Population Attributes	Align	Population attributes are aligned to the population attribute definition in the new version.
Level	Level Member Defaults	Ignore	Customer level member defaults are retained.

Object	Sub-Objects	Reconciliation Action	Comments
Integration Interface	-	Align	<p>If there is a change in the integration interface, data profile, or level profile between the existing and new application versions, the existing version of the integration interface is aligned (overwritten) with the new version. When an integration interface is aligned, any data profiles within it are also aligned to the new version.</p> <p>Therefore, customizations made to the standard integration interfaces are not preserved during the upgrade process.</p> <p>After the upgrade,:</p> <ul style="list-style-type: none"> <li>- Review your pre-upgrade schema backup for the integration interfaces that have been aligned</li> <li>-Add back any customizations you have made to the new version of the integration interface.</li> </ul>

Object	Sub-Objects	Reconciliation Action	Comments
Workflow Schema	-	Duplicate	<p>If there is a change in the workflow schema between the existing and new application version, then the new version of the workflow schema is created as a duplicate object.</p> <p>After upgrade, it is recommended that you manually merge your pre-upgrade customizations from the pre-upgrade version to the new version and use the new version of the workflow.</p>
Users	-	Align	-
Model	-	Align	Changes to the existing entries in the standard data model are aligned with the definition in the new version.



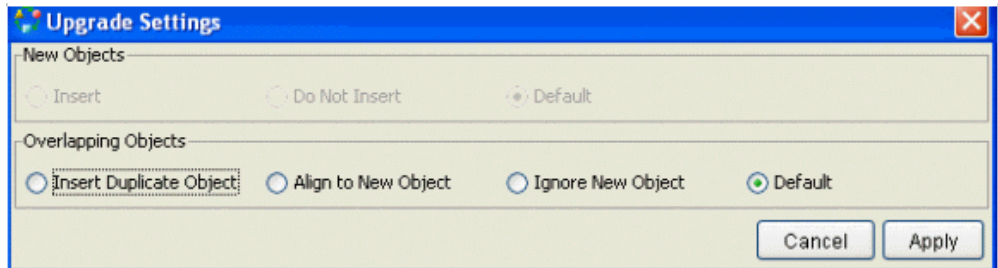
Object	Sub-Objects	Reconciliation Action	Comments
Components (Component Wizard of the Business Modeler)	All sub-objects <ul style="list-style-type: none"> <li>• Component Series</li> <li>• Component Levels</li> <li>• Component Units</li> <li>• Component Indexes</li> <li>• Worksheet for Levels</li> <li>• Engine Profiles</li> </ul>	Align with Merge	<p>All sub-objects under Components (for example, component levels, series, units, and so on) are merged with the new version so that the original and new settings are available.</p> <p>For example, if a customer has associated a COGS unit to a component and in the newer version, there is a new unit XYZ associated with this component, then both COGS and XYZ units are available after the upgrade.</p> <p><b>Note:</b> The existing component level permissions are overwritten by the new level permissions. It is important to keep track of the existing component level permissions before the upgrade. After the upgrade, these component level permissions can be reconfigured in the Business Modeler to match the pre-upgrade settings.</p>

Object	Sub-Objects	Reconciliation Action	Comments
Group Users	-	Align with Merge	Any users a customer has added to a user group are merged with any users associated with this group in the new version.
Indexes	-	Align	-
<b>Note:</b> These are not database indexes, but rather indexes created in Business Modeler through the Configure > Configure Indexes menu option.			
Series Group	-	Align with Merge	Custom series added to existing series group will be retained along with any standard series in that group.
Any other objects not mentioned in this table	-	Align	Objects not specifically mentioned in this table are aligned to the new version.

Based on your settings, the Object Resolution dialog box displays default upgrade suggestions for each object during the upgrade process. Choices can be changed on an object-by-object basis using the Set Object Resolution option.

**To set the global upgrade preferences:**

1. From the Settings menu, select Upgrade Settings. The Upgrade Settings dialog box appears.



**Note:** New objects are always inserted.

2. For Overlapping Objects, select:

- **Insert Duplicate Object:** Objects exist in both the original and new schema definitions. The original object is suffixed as "objectname\_version number\_build number", while the new object is created as a duplicate object with the same name as in the Demantra standard schema. See Naming Convention for Duplicate Objects, page 65-17 for more information.

**Note:** Only overlapping objects that have changed between the current and new versions are duplicated. For example, if a worksheet in 7.1.1 has not changed compared to the 7.3 worksheet, then it is not duplicated when upgrading to 7.3.

- **Align to New Object:** Original objects are merged with the objects in the new version schema definition. In cases where the table indicates "Align with Merge", the original objects are merged with the new objects; both exist after the upgrade.
- **Ignore New Object:** Existing object is retained in its original form; new version of the object is ignored.
- **Default:** Objects and sub-objects are upgraded based on the action type specified in the Table of Default Upgrade Actions for Overlapping Objects, page 65-18 above.

3. Click Apply. All upgrade suggestions are based on these upgrade setting preferences from this point forward.

### Upgrading the Objects in a Folder

To add objects from one schema to another, you create a folder in the target schema. Once you have added objects to the folder, you can upgrade these objects to the target schema database. During the upgrade process, the BAL Explorer checks to see whether

all the necessary related objects, such as levels and series, exist in the schema. If not, you are prompted to substitute other objects or cancel the operation. Some objects may require configuration changes. These changes can be implemented through the Business Modeler.

**Note:** If you want to add more objects after you have upgraded a folder, add the objects to a new folder and, when ready, upgrade that folder.

To upgrade objects in a folder:

1. Right-click the folder that you want to upgrade.
2. Select Upgrade.

## Upgrading Schemas

Upgrading a schema begins with comparing two schemas to detect differences. The BAL Explorer compares the schemas object by object, and it returns a detailed report that itemizes the differences. This report can be saved in html format for future reference. If you are satisfied with the comparison, you can prepare the upgrade package. Once the upgrade package has been prepared, you can start the upgrade process.

**Note:** When upgrading from the installer, the BAL Explorer automatically performs the comparison of the two schemas and displays the report. The upgrade package is created automatically.

If you have chosen the upgrade setting that allows you to change the resolution action, you will be able to choose how to handle the differences between schemas on an object-by-object basis. BAL then processes the schema upgrade. Finally, the Business Modeler launches and you activate the configuration changes to complete the upgrade. The following diagram illustrates the upgrade process:

Upgrade Process
Compare Schemas (automatic from installer)
Prepare upgrade package (automatic from installer)
Start upgrade of schema
Choose how to upgrade each conflicting object
Complete upgrade of schema
Activate configuration changes in Business Modeler

The schema repository, which contains all upgrades to that particular schema, is not deleted after the upgrade unless you choose to remove it completely. It is updated with future upgrades.

To upgrade a schema:

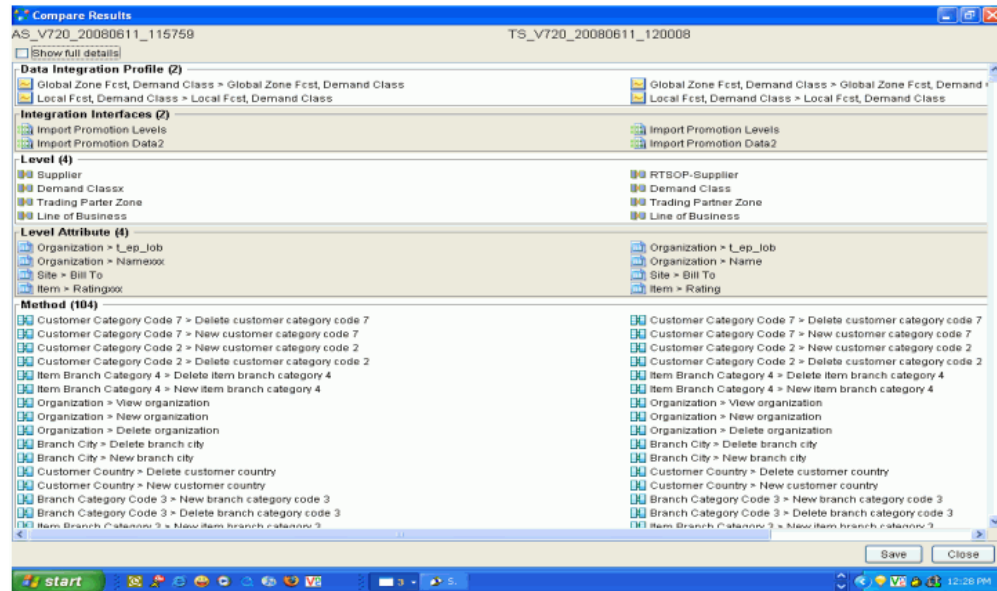
**Note:** Go directly to Compare Results Step, page 65-27 if you are upgrading manually from the installer. The other steps are performed automatically.

1. Click the source schema (the original schema).
2. Ctrl-click the target schema (the new schema).
3. For manual upgrades, right-click one of the selected schemas and select Compare. The Compare Results screen appears, listing the objects that differ between the two schemas. The system displays the source schema on the left (AS\_V730\_TIMESTAMP) and the target schema on the right (TS\_V730\_TIMESTAMP).

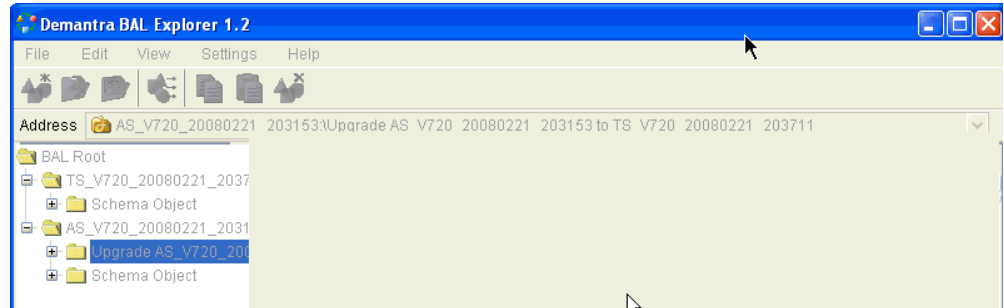
The installer takes you directly to the Compare Results screen.

**Note:** Once you have selected the manual upgrade, the installer begins to upgrade the schema and prepare the BAL repositories. The Manual Upgrade screen appears while the BAL Explorer is comparing the source and target schemas. Both the Next and the Previous buttons are greyed out during this process. The comparison may take a long time, and during this time, the BAL Explorer may appear to be gray for the most part. To see whether the comparison is still running, note the bottom left-hand corner in

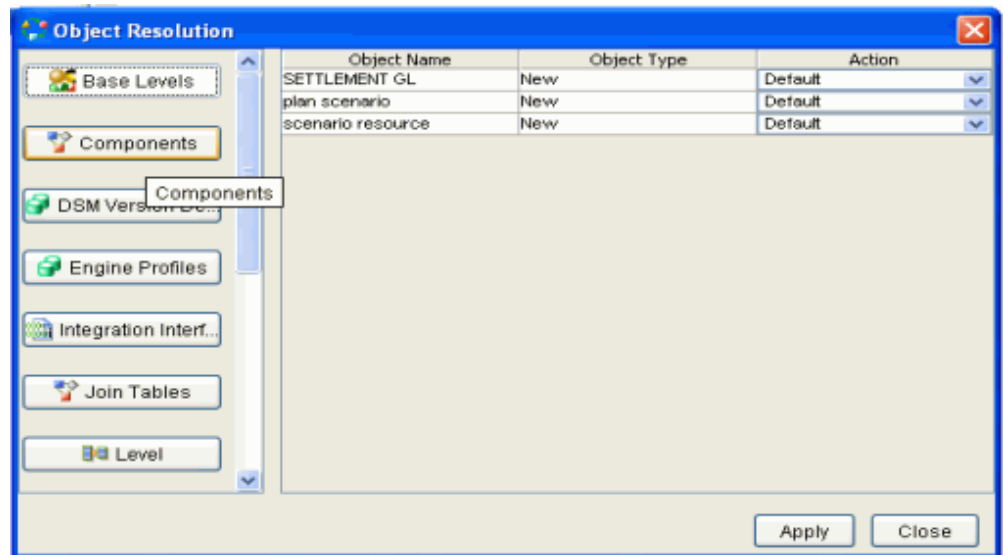
the BAL Explorer. A status bar displays the comparison currently being done.



4. Click Show full details to show the specific differences between the schemas.
5. Click Save to save the report in html format for future reference. The html reports are saved in the <Demantra install folder>\Demand Planner\BAL\logs\AS\_V730\_Timestamp to TS\_V730\_Timestamp folder. You can print the report from the saved document. Review the html comparison reports in detail to determine the optimum upgrade actions to use on your schema.
6. If you are not upgrading from the installer, click Prepare Upgrade for the schema that you want to upgrade. A progress bar appears so you can monitor the production of the upgrade package. When complete, the upgrade package appears in the Schema Browser before the schema object folder. In the following example, the AS\_V730\_20080221\_2031 schema was prepared for upgrading. The upgrade folder appears before the schema object folder:



7. Click Close to close the Compare Results dialog box.
8. From the Settings menu, select Set Object Resolution. This selection enables you to modify the upgrade suggestions for each object individually.
9. Right-click the upgrade package and select Upgrade. The Object Resolution dialog box appears. The type of object conflict appears with the action suggested based on your global upgrade settings.



10. Choose how you want to handle each new or overlapping object.

For each new object, your options are:

- Insert: Add the new object to the target database.
- Default: Add the new object to the target database.

For each overlapping object, your options are:

- Align to New: Align the source object with the target schema object.

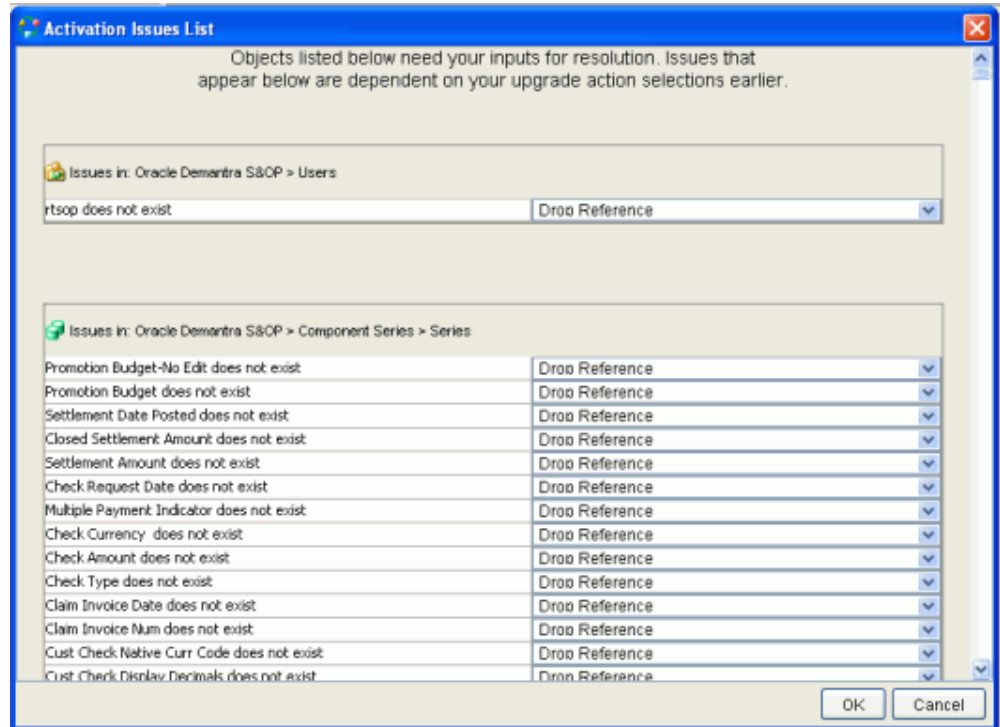
- **Insert Duplicate:** Create the new version of the object as a duplicate in the target schema.

**Note:** This option may not be available depending on the type of object.

- **Ignore:** Retain the source object in its original form and ignore the new version of the object in the target schema.
- **Default:** Objects and their sub-objects are upgraded according to the Table of Default Upgrade Actions for Overlapping Objects, page 65-18 .

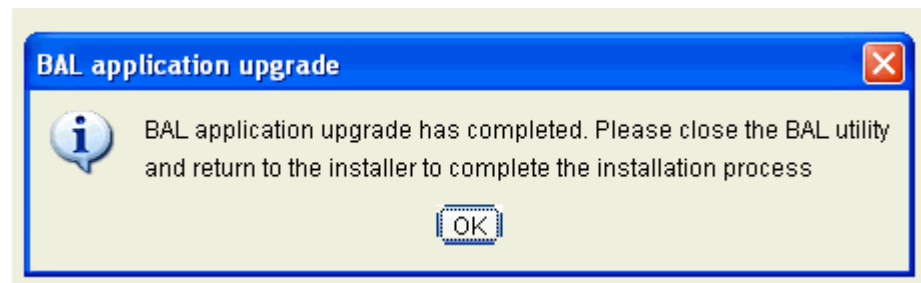
11. Click **Apply** when done. The source schema is upgraded to the target schema format based on your upgrade choices. A progress bar appears.
12. If there are any issues that need your intervention to resolve, the activation issues list is displayed. These issues can occur if you accidentally delete any objects that are referred to by other objects in the upgrade package.





For each issue displayed, you can accept the default Drop Reference which removes the link between the new objects and the existing object.

13. When you have specified how to deal with all the issues presented, click OK.
14. A prompt appears when the BAL application upgrade is complete. The following prompt is displayed for those upgrading using the installer. A different prompt appears if you are running the BAL without the installer.



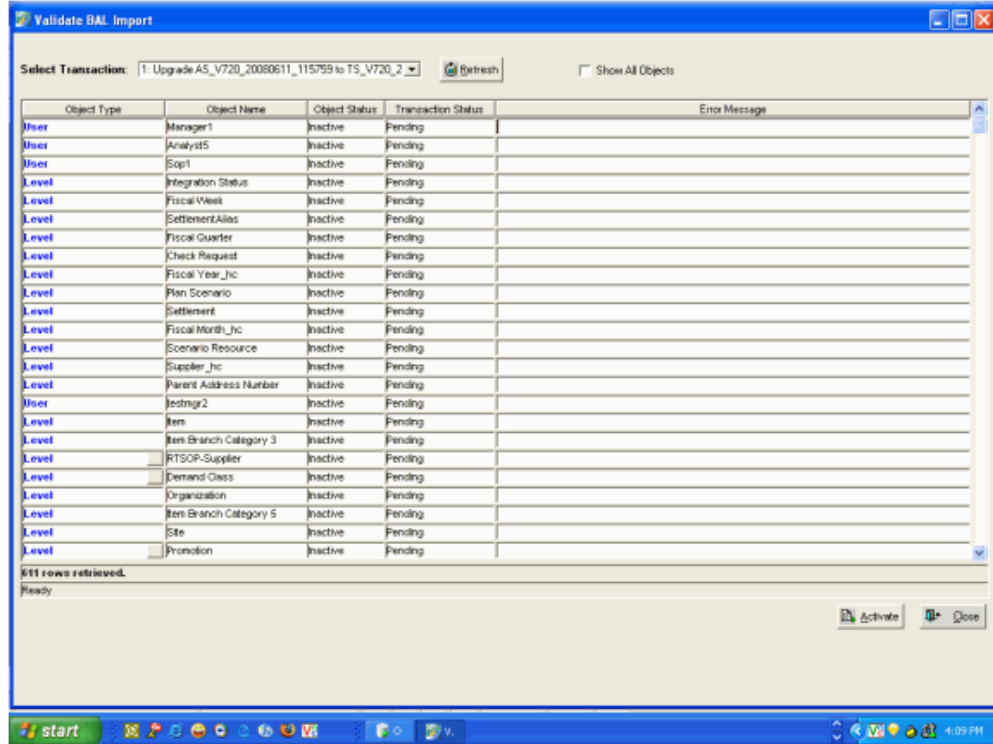
**Important:** Do not close the BAL utility before receiving the prompt or the upgrade may fail.

Click OK to continue.

15. Close the BAL Explorer. If you are upgrading through the installer, the Business

Modeler application opens automatically (this may take a few minutes). Otherwise, start the Business Modeler to continue the upgrade process.

16. The Validate BAL Import screen appears. If are not updating from the installer, from the Configuration menu, select Validate BAL Import, and select the update package to continue.



17. Click the Activate button. The configuration changes are made through the Business Modeler to the database.
18. Close the Business Modeler. A prompt appears that indicates that warnings were encountered while trying to activate some objects.
19. Click Resume (recommended) to return to the installer. Alternatively, click View to log file or Exit to quit the installation.
20. Click Done to complete the installation. For more details about the upgrade, review the bal\_bm.log.

## Troubleshooting

### Starting The BAL Explorer Application

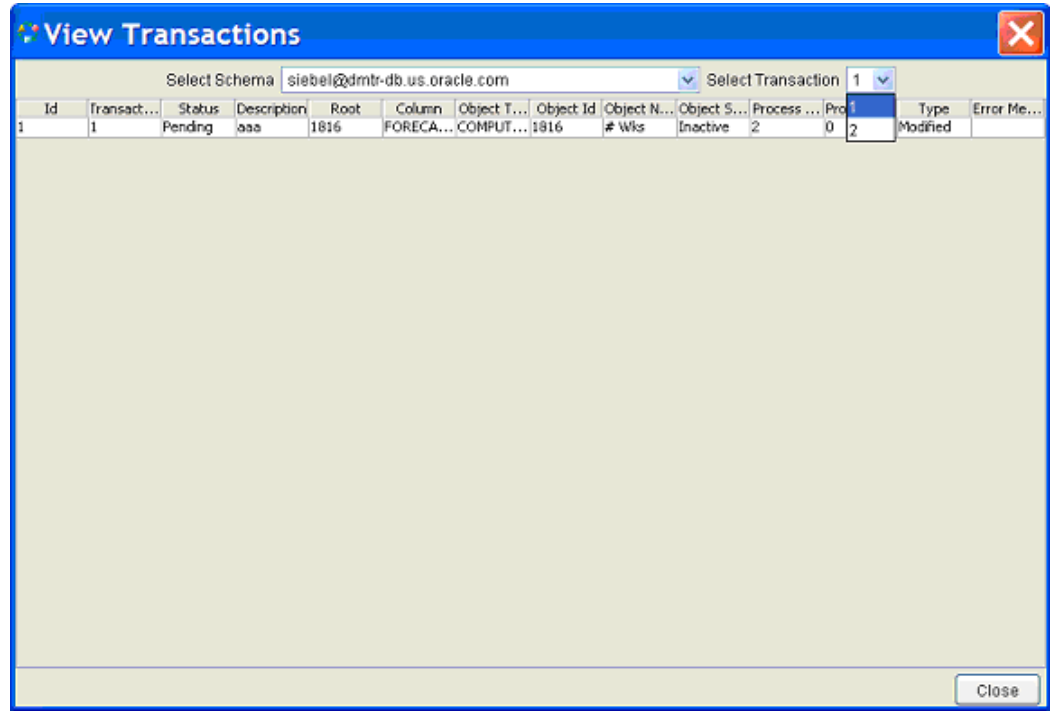
For the following troubleshooting options, you must start the BAL Explorer manually.

From the < Demantra install directory>\Demand Planner\BAL, double-click on bal.bat.

## Viewing Transactions

The View Transactions window displays information about all the schema upgrades that have been made within BAL, including the status, description, root, column, object table, type, and error message. You can filter the results by schema or transaction.

To view transactions, open the BAL Explorer application, select Transactions. The View Transactions window appears.



## Including the Upgrade SQL in the Log File

You can view the PL/SQL run by the BAL Explorer during the upgrade once the upgrade is complete. The SQL appears in the bal\_log.txt file located in the <root directory>\Demand Planner\BAL\Logs directory. This option provides you with many more details about the upgrade than would normally be captured.

To include the upgrade SQL in the bal\_log.txt file:

1. From the View menu, select View Upgrade SQL.
2. Perform a folder or schema upgrade.
3. When complete, open the <root directory>\Demand Planner\BAL\Logs\bal\_log.txt file to view the upgrade details.

## Viewing Log Files

If your upgrade fails, you may find the reasons for the failure in the log file. The `bal_log.txt` file and the `log.properties` files are located as follows:

- `<root directory>\Demand Planner\BAL\Logs\bal_log.txt` This file shows details about the processing of an upgrade, including errors. If you selected the View Upgrade SQL option, then the SQL run by the BAL Explorer during the upgrade appears when the upgrade completes.
- `<root directory>\Demand Planner \BAL\log.properties`

You can customize the amount of detail shown in the `bal_log.txt` file by altering the settings in this file.

---

## Tips and Troubleshooting

For reference, the first section describes the first-time login procedure for Demantra applications, followed by several sections of tips. After that, this chapter lists possible errors that users may encounter and describes how to resolve them. The errors are listed alphabetically by message text or general description.

See Also

Oracle Demantra Release Notes Oracle Demantra Installation Guide

This chapter covers the following topics:

- Initial Logon to Demantra
- About Demantra Configuration Settings
- Key Settings Controlled by the Installer
- Redirecting Demantra to a Different Database
- Java Tips
- Tomcat Tips
- Error Messages or Trouble Descriptions

### Initial Logon to Demantra

The first time you log onto Demantra Web applications, Demantra downloads and installs software. For reference, this section repeats and expands on the details from the user guides.

#### **Initial Logon to Collaborator Workbench:**

This operation is necessary only once for each computer.

1. Launch a supported web browser.
2. Enter the URL for Collaborator Workbench:

`http://server name/virtual directory/portal/loginpage.jsp`

3. In the Log On dialog box, enter your user name and password.
4. Click Login.
5. If you are prompted to install JRE, do so. Choose the Typical installation and accept all the default values, or follow your own site practices.

After you install this software, Collaborator Workbench comes up, displaying your personal page.

Demantra displays a dialog box that asks if you want to trust the signed application distributed by Oracle.

**Note:** This dialog box is sometimes displayed as soon as Collaborator Workbench comes up. In other cases, you do not see it until you click a worksheet name.

6. Click Yes, (or Always) or Start, depending on which dialog box is displayed.

#### **Initial Logon to the Web Client:**

This operation is necessary only once for each computer.

1. Launch a supported web browser.
2. Enter the web address for the Web client:

`http://server name/virtual directory/portal/partnerLogin.jsp`

3. Type your name and password and click Login.

Demantra prompts you to install JRE.

4. When you are prompted to install JRE, do so. Choose the Typical installation and accept all the default values, or follow your own site practices.

Demantra next displays a dialog box that asks if you want to trust the signed application distributed by Oracle.

5. Click Yes (or Always) or Start, depending on which dialog box is displayed.

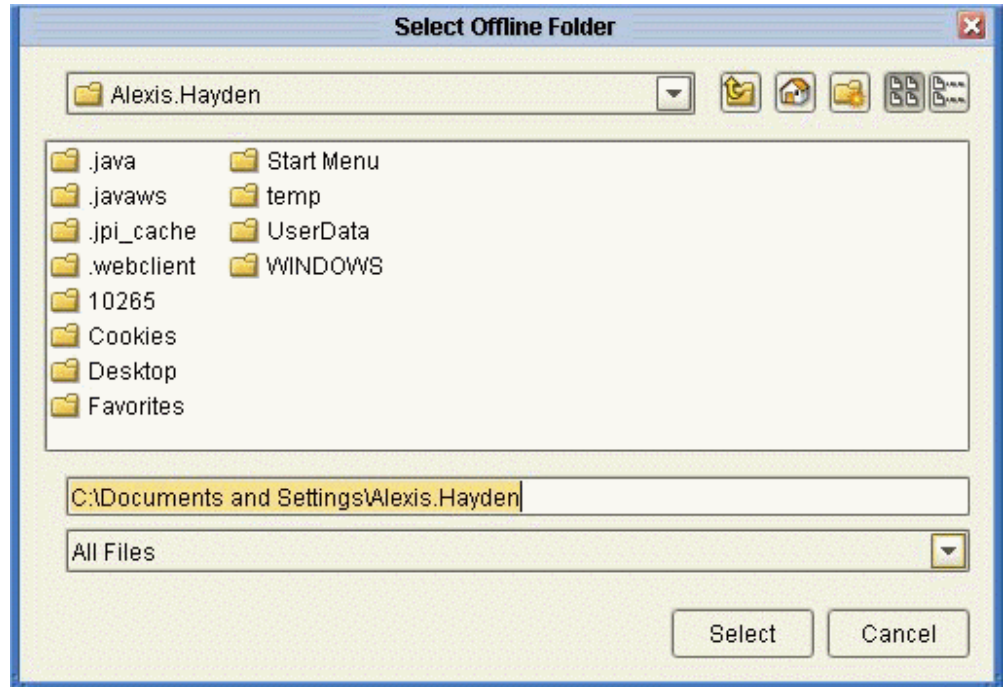
#### **Initial Setup and Logon of Offline Environment:**

This operation is necessary only once for user on a given computer.

**Note:** When you perform these steps, you must have network access to the Demantra server, which must be running.

1. Within a normal Demantra worksheet, click File > Export for Offline Use.

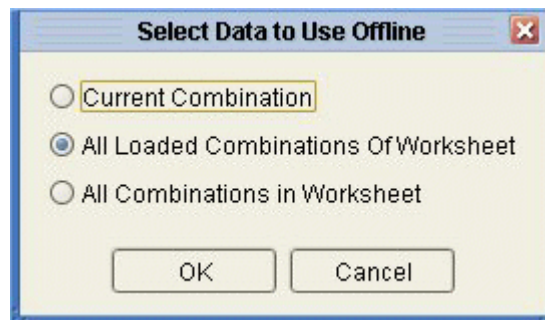
Demantra displays the following screen:



Here you specify the directory where Demantra will store your offline information. By default, this is c:/Documents and Settings/username, but you can choose a different location. The offline directory itself is always named Demantra.

2. Specify the offline directory location and click OK.

Demantra then displays the following screen:



Here you specify which data to take offline. You might want to work with only a single combination, for example.

3. Choose one of the options and click OK.

4. Log off Demantra.
5. Enter the web address supplied by your system administrator for offline access.

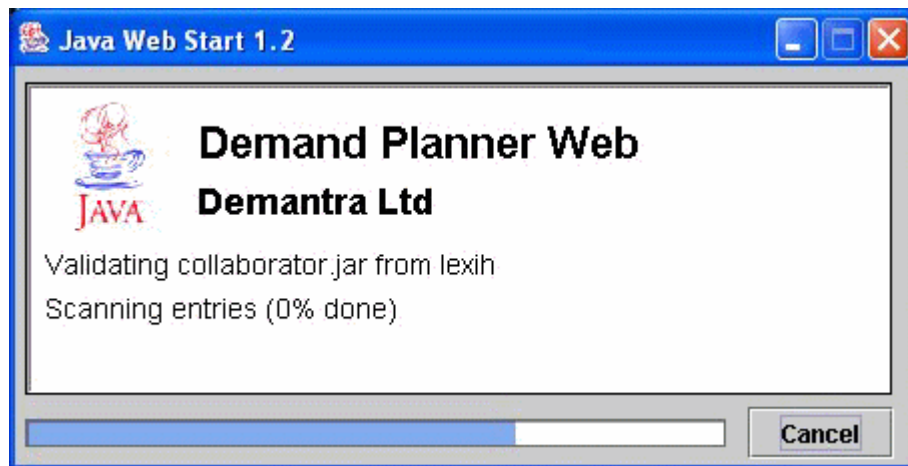
This URL probably has the following format:

`http://server name/virtual directory/portal/launchDPWeb.jsp`

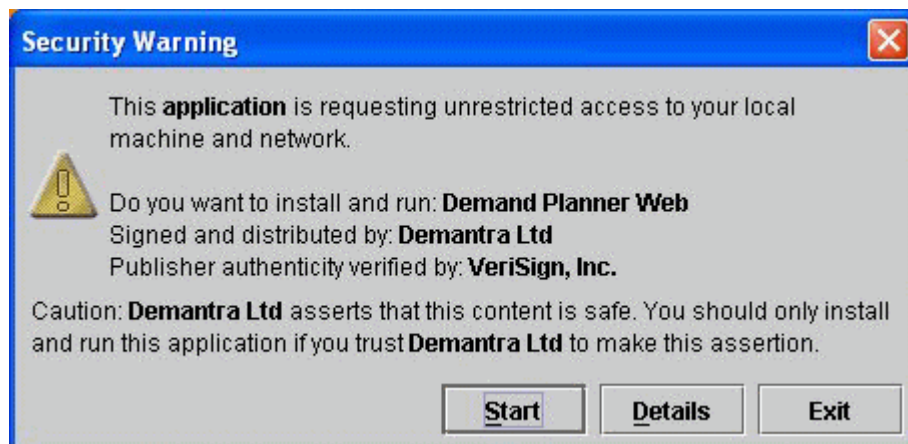
For example:

`http://frodo/demantra/portal/launchDPWeb.jsp`

Depending on what is already installed on this computer, Demantra then briefly displays a screen titled Java Web Start, which shows its progress in scanning and downloading the required JAR file.



After that, Demantra displays a dialog box that asks if you want to trust the signed application distributed by Oracle:



**Note:** If Demantra does not display this screen, that means that you

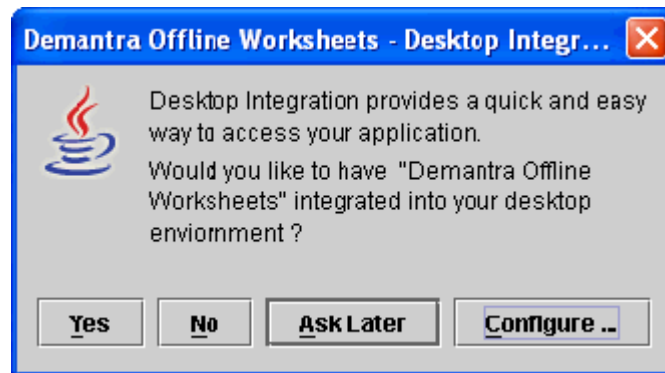


have already downloaded the applet to this machine. You do not need to download it again.

**Note:** Depending your system configuration, the screen might be slightly different, with the following options instead: Yes, Always, and No.

6. Click Start or Yes (or Always), depending on which dialog box is displayed.

Next, Demantra displays the following prompt:



7. Click Yes.

Demantra creates a shortcut labeled Demantra Offline Worksheets, which you use to access your offline worksheets in the future. This shortcut is on your desktop and on your Start menu.

This user is now configured to use worksheets offline on this machine.

## About Demantra Configuration Settings

The core Demantra configuration details are stored in multiple locations:

- The desktop executables (Business Modeler, Demand Planner, Analytical Engine, and so on) get the configuration information from the following file:

*Demantra\_root*\Demand Planner\Security Management\ds.ini

Parts of this are encrypted and must be edited with a utility provided by Demantra (encryption.exe); see "Redirecting Demantra to a Different Database".

- The Web-based products get configuration information from the APS\_PARAMS table and the server.xml file, which is located at \Oracle Demantra\Collaborator\Tomcat\conf\ on the machine where the server is

installed.

**Note:** Almost all the parameters of this file can be edited from within the Business Modeler, and it is better to use the Business Modeler to make changes so that the audit trail can record them. The Business Modeler also provides drop-down menus of possible options, where applicable.

**Note:** To access these parameters within Business Modeler, click Parameters > System Parameters.

- Other settings are stored in the Demantra database, in the form of parameters. These can be edited through the Business Modeler, as well.
- The Web-based products also use configuration information in the XML files.

## Key Settings Controlled by the Installer

This section summarizes the key settings that the installer controls and indicates where those settings are stored.

Installer Screen	Installer Option	In the APS_PARAMS table	In ds.ini
DBA Information	DBA username (to access database as DBA and load data)	Not saved here	Not saved in this file.
	Password		
	TNS Name		Tnsname
Configure Oracle Database User	Database type	DBType	DBType
	User (to store Demantra data)	DBUser	LogID**
	Password	DBPassword	LogPassword**

Installer Screen	Installer Option	In the APS_PARAMS table	In ds.ini
Configure JDBC Connection	Server name (host machine or IP address on which database resides)	ServerName	ServerName
	Port	DBPort	DBPort
	Oracle SID (Oracle only)*	DBName	Database
Specify Web Address	Root address and virtual directory	AppServerURL server.generalurl	Not saved in this file.
**Encrypted in the ds.ini file.			

## APSMoDe Parameter

The APSMoDe parameter (stored only in the ds.ini file) controls whether to use the Stand-Alone Integration Tool (aps.bat). This tool consists of a subset of the APS, packaged as an executable file.

The installer automatically sets this parameter. This parameter has the following effect:

- 0: do not use Stand-Alone Integration Tool. When you use encryption.exe to edit ds.ini, only General tab is displayed.
- 1: use the Stand-Alone Integration Tool (Demantra\_root/Demand Planner/Integration/aps.bat). Also, when you use encryption.exe to edit ds.ini, the ASP Stand Alone tab is displayed, in addition to the General tab.

For information on using aps.bat, see "Executing an Integration Interface".

## Other Parameters

The installer also sets parameters for the following purposes:

- The tablespaces Demantra should use
- The configuration of the administrator email account

For these parameters, see "Database" and "Email".

## JAVA\_HOME System Environment Variable

Tomcat requires JDK, which means that the JAVA\_HOME system environment variable must be set (not a user environment variable). The installer automatically installs JDK if appropriate and sets this environment variable. JAVA\_HOME should be set equal to the directory that contains the bin directory of JDK.

## Other Configuration Files

The installer also makes edits to the following files. If you make a change to a port or protocol or other, you must be sure to make the change in the following files:

- Demantra\_root/Collaborator/virtual\_directory/WEB-INF/web.xml
- If you are using Tomcat: Demantra\_root/Collaborator/Tomcat/conf/server.xml (refers to the Demantra host and port, as well as the path to the Demantra virtual directory).

**Note:** When you start Tomcat, Tomcat creates or updates the file Demantra\_root/Collaborator/Tomcat/conf/Catalina/localhost/virtual\_directory.xml, as needed.

- If you are using WebSphere:
  - WAS\_HOME/installedApps/host\_name/demantra.war/demantra.war/WEB-INF/web.xml
  - WAS\_HOME/config/cells/host\_name/applications/demantra.war/deployments/demantra/demantra.war/WEB-INF/web.xml

Back up any file before making edits, and then carefully search and replace as needed.

## Redirecting Demantra to a Different Database

Oracle does not support Microsoft SQL Server in this release. Please monitor My Oracle Support for versions supporting SQL Server. Items marked with \*\* are not valid unless support for Microsoft SQL Server is available.

In Demantra 7.3.0, the database connection (and data source configuration) is controlled by the Java Naming Directory Interface (JNDI).

To point Demantra to a different database without rerunning the installer, complete the following steps:

1. Make a backup copy of server.xml. This file is located in ... \Oracle Demantra \Collaborator \Tomcat \conf \.

2. Open server.xml for editing.

3. Locate the following sections:

```
Context path="/demantra"
docBase="E:\Program Files\Oracle Demantra
73b37\Collaborator\demantra"
crossContext="false"
debug="0"
reloadable="false"

Resource
name="jdbc/DemantraDS"
auth="Container"
type="javax.sql.DataSource"
driverClassName="oracle.jdbc.driver.OracleDriver"
url="jdbc:oracle:thin:@dempm2db.us.oracle.com:1521:ORCL"

username="demo73b37"
password="manager"
```

4. Modify the "username" and "password" sections to point to the new schema and to use the new password.
5. To point to a different DB, modify the DB host and SID within the URL, which in this example are "dempm2db.us.oracle.com" and "ORCL", respectively.
6. Restart the web server. (If you are not using Apache Jakarta Tomcat, refer to your application server's version-specific documentation to learn how to modify the database hostname, username, password, and SID (system identifier) specified by the JNDI.)

## Java Tips

This section contains background information about how Demantra uses Java. The Demantra Web client (Demand Planner Web, Promotion Effectiveness, or Settlement Management) uses JRE. Each machine that runs the Web client should have JRE, which Demantra automatically downloads when necessary.

**Note:** JDK is needed only if you are using Tomcat. The JDK is needed on the machine that runs Tomcat, not on the client machines. For information on Tomcat, see the Oracle Demantra Installation Guide.

## Java Versions and Older Demantra Installations

JRE versions are generally backwards compatible. If you are using an older version of the Web client, you can use the same JRE as the current Demantra. This means that,

from a single machine, you can log into different Demantra installations, even if they use different versions of Java.

If the client machine either does not have a version of JRE installed or the latest installed version is older than what Demantra currently supports, the user will be prompted to download and install the latest supported version. If the machine has at least one supported version installed, Demantra will use the latest supported version.

## Tips for a Clean Java Installation

It is possible, but tricky, to keep multiple versions of Java running on a single machine. Oracle recommends that you carefully remove all Java versions other than the current version used by Demantra; to remove them, use the Add or Remove Programs control panel.

It is also useful to check your PATH system environment variable. Java is added to this, and you should make sure it includes only the Java that you intend to use. Note that Oracle provides Java as well; you do not need to uninstall these, but you should probably remove those versions from the PATH system environment variable.

Finally, you should make sure that the supported web browser is configured to use the correct Java version:

1. Click Tools > Internet Options.
2. Click the Advanced tab.
3. Within the Java item, make sure that the correct version of Java is selected for use with applets, as specified in the Oracle Demantra Installation Guide.

## Setting client.JREMaxMemory with configureEnv.jsp

The configureEnv.jsp tool sets JRE maximum memory on each client's installed java plugin {located at: user home directory}/Application Data/Sun/Java). It affects all applications that run with that plugin. To run this utility, use the following url (filling in the appropriate server and port number):

`http://application_server_host:port/application_root/portal/configureEnv.jsp` For example, if running the client on port 8080 of the local machine, you would use:  
`http://localhost:8080/demantra/portal/configureEnv.jsp`.

## Tomcat Tips

For demos or in production you can use Tomcat as the Web server, and the installer supports this option for convenience. Oracle has tested with Apache Jakarta Tomcat 6.x.

## Installing with Tomcat

This section briefly notes the differences between a demo installation and the usual installation.

1. Apache Jakarta Tomcat 6.x requires JDK 1.5 (This can be obtained free at [www.sun.com](http://www.sun.com).) You do not have to pre-install this, but you should make sure you do not have an earlier version of JRE on the machine. If so, uninstall that.
2. Run the installer like usual, except choose Demo for Web Server type.
3. If prompted, specify the desired value for the JAVA\_HOME system environment variable. The installer prompts you for this if more than one Java is installed on the machine.

## Changing the Default Tomcat Port

The Tomcat default port is 8080. The installer does not change the default configuration for the port. This must be done manually in the file `Demantra_root/Collaborator/Tomcat/conf/server.xml`.

**Note:** If you do use the 8080 port, note that the Oracle XDB database user tries to use that port. See "Port Collision with Oracle XDB User".

## Starting the Server if Using Tomcat

If you chose the Demo Web Server type, the installer adds Start menu options to start and stop Tomcat.

1. In Windows, click Start and click Programs.
2. Click Demantra > Demantra Spectrum release > Start Web Server.

## Clearing the Tomcat Cache

To clear the Tomcat cache, delete the directory **Demantra\_root** `/Collaborator/Tomcat/work/standalone/localhost`.

You may need to do this if you receive the "Object Error" message; see "Object Error".

## Renaming the Installation Root Directory

It is safest to reinstall Demantra rather than to rename the root directory where it is installed. However, if you are using Tomcat, you can rename the Demantra root directory and redirect Tomcat. To redirect Tomcat, edit the file `Demantra_root`

/Collaborator/Tomcat/conf/server.xml. In this file, edit the parameter docBase. This parameter should specify the full path to the Demantra virtual directory.

## Writing the Tomcat Log to a File

By default, the Tomcat log is written to the console. To reconfigure Tomcat to write its log to a file, edit the file *Demantra\_root/Collaborator/Tomcat/conf/server.xml*.

Find the Logger section and edit it as follows:

```
<Logger name="tc_log"
 path="logs/tomcat.log"
 verbosityLevel = "INFORMATION" /> f
```

## Error Messages or Trouble Descriptions

### APS.bat Runs, Then Disappears - ClassNotFoundException

#### Scenario

When running the standalone integration from a custom batch file other than a direct call to APS.bat, incorrect specification of the path can lead to failures. In other words, the command window opens and then immediately disappears. Nothing apparently happens.

#### Explanation

Example of an invalid call, creating a new batch file with the following syntax will fail:

```
"C:\Demantra Spectrum\Demand Planner\Integration\aps.bat" IMPORT_DATA
TEST TEST
```

Since the APS.bat file relies on relative paths to connect to the Java class path and call the aps.jar main class, if the call comes from a different batch file or command line window that is located outside of the APS.bat file folder, these calls may fail with a 'ClassNotFoundException' error.

#### Resolution

To resolve the issue, several solutions ensure the process will use the correct directory:

1. Make sure that the calling batch file switches to the APS.bat folder before calling the APS.bat file with the parameters. For example:

```
CD "C:\Demantra\Demand Planner\Integration" APS.bat IMPORT_DATA TEST
TEST"
```

2. Use the START command to call the APS.bat file and pass in the APS.bat folder location with the parameters. For example:

```
Start /B /D "C:\Demantra\Demand Planner\Integration\" aps.bat
IMPORT_DATA TEST TEST
```



**Note:** Note: This second option is most useful when the calling batch file includes more commands and avoids the need to change the current directory back to its original location.

3. The third option leverages the workflow to run the .bat file. The workflow can run with a relative path (leveraging the token) to the application server root parameter.

**Note:** For information about relative path requirements, see *Parameters Used as Arguments for a Workflow*, page 31-4.

## Cannot Connect to Database

### Scenario

A user tries to log into a Demantra desktop product but receives the following message:

Cannot connect to database

This message is also displayed in the DOS window on the server.

### Explanation

The database is not running.

### Resolution

Start the database.

## Cannot Connect to Server

### Scenario

A user receives the following message when trying to run a worksheet in one of the Web products:

Cannot connect to server

The worksheet is not displayed.

### Explanation

There is a Java problem on this user's machine.

### Resolution

1. Stop the Web server.
2. On the user's machine, open the Java Plug-in control panel, clear the Java cache, and remove the certificate.
3. Restart the Web server.

4. When the user next opens a worksheet, he or she should accept a new certificate as usual.

## **Could Not Allocate Space**

### **Scenario**

When you started the Web server, you received a message complaining that Demantra could not allocate space.

### **Explanation**

The tablespaces assigned to Demantra are not large enough.

### **Resolution**

Contact the database administrator and increase all the Demantra tablespaces.

## **Data Is Exported as Text, Not Currency**

### **Scenario**

In the Web products, the user exports to Excel, and some of the values are formatted as text (General) rather than as currency.

### **Explanation**

When receiving data from an external source, Microsoft Excel uses the Regional Options in the Windows Control Panel to determine whether a given cell should be formatted as Currency or General (as is or text).

### **Resolution**

Later versions of Excel provide an option for converting problematic cells that it recognizes. If Excel does not provide any such option, do the following:

1. Open the Windows Control Panel.
2. Double-click Regional and Language Options.
3. On the Regional Options tab, make sure that the Currency setting uses the same currency symbol as Demantra.
4. Export again from Demantra.

## **DOL Link to Excel Uses Wrong Link**

### **Scenario**

The user tries to link Demantra data into Excel via DOL, but the wrong link is provided.

Specifically, in Excel, the user clicks Data > Get External Data > New Web Query, and then browse to the file DOLLogin.jsp. The user then logs in and selects the query. When

the user returns to Excel, the link shown is the wrong one.

#### **Explanation**

This problem is caused by a defect in older versions of Excel.

#### **Resolution**

Copy and paste the link from the web page directly.

## **Error in Process Execution**

See "Workflow Process Has Failed".

## **Error Loading Tasks, Error on Page**

#### **Scenario**

In the My Tasks area of Collaborator Workbench, a user sees one of the following messages:

```
error on page error
 loading tasks
```

#### **Explanation**

On this machine, the Regional Settings are not US English.

#### **Resolution**

Change the Regional Settings to US English., as described in "Date Dialog Boxes Behave Strangely".

## **Error When Opening a Worksheet from the Members Browser**

#### **Scenario**

The user tries to use the Open or Open With menu options to open a worksheet from the Members Browser (or a content pane that uses the Members Browser). An error occurs.

#### **Explanation**

There may be an underlying problem with the Java plug-in.

#### **Resolution**

1. Make sure that only the correct version of JRE is installed on your machine (see "Initial Logon to Demantra").
2. If you are using Tomcat, clear the Tomcat cache on the server by deleting the directory **Demantra\_root/Collaborator/Tomcat/work/standalone/localhost**.
3. Restart the Web server and try again to open the worksheet.

4. If this does not fix the problem, then uninstall JRE from the user's machine. To do so, use Add/Remove Programs and restart the machine if prompted to do so.
5. Then complete the steps in "Initial Logon to Demantra".

## Error While Running Worksheet Window: Object Error

See "Object Error".

## File Download When Using launchDPWeb.jsp or partnerLogin.jsp

### Scenario

When the user accesses either of these URLs (<http://server name/virtual directory/portal/partnerLogin.jsp> or <http://server name/virtual directory/portal/launchDPWeb.jsp>), the following dialog box is displayed:

```
File download
Do you want to save or open this file
Name: partnerLogin.jsp [or launchDPWeb.jsp]
```

### Explanation

This message is displayed if Web Start is not installed on this computer. Normally, the latest version of Java Web Start is installed automatically along with the JRE when you first run Collaborator Workbench or any of the Web client interfaces.

### Resolution

1. Log into Collaborator Workbench and check to see if it has a link labeled Click here to install Java Web Start.
  1. If so, click that link.
  2. This installs Java Web Start 1.4.2\_10.
  3. Demantra next prompts you to install JRE.
2. On the other hand, if JRE is already installed on this computer, uninstall it. To do so, use Add/Remove Programs and restart the machine if prompted to do so.
3. Log onto Collaborator Workbench.
4. Follow the steps in "Initial Logon to Collaborator Workbench".

## Internal Error: Please Check Database and Network Connections

See "Workflow Internal Error".

## Invalid Argument to Function

### Scenario

When a user tries to run a specific worksheet in the desktop (Demand Planner), the following message is displayed:

Invalid argument to function

### Explanation

There is an error in the client expression of at least one series used in this worksheet.

### Resolution

1. Make a list of all the series in the affected worksheet.
2. Within the Business Modeler, check the client expressions of those series. You can click the Verify Expressions button in the toolbar to verify all server and client expressions.
3. Correct the client expression, save the changes, and reload the changes to Demand Planner.

## Java Out of Memory

### Scenario

The user receives a message from Java saying that it is out of memory.

### Possible Resolutions

- Increase the amount of memory allowed for Java. To do so, go to the Java Plug-in control panel. Click the Cache tab and increase the cache size setting.
- Reduce the amount of memory that Java requires. To do so, use fewer worksheets simultaneously and reduce the amount of data in any worksheet. The main way to reduce the size of a worksheet is to filter it, but you can also remove unneeded series.

## License File Has Expired

### Scenario

After logging into a Demantra Web product, a user receives the following message:

Your Security License File has expired

The user is not able to proceed further.

### Possible Explanations

This message can occur for several reasons:

- The Demantra license is expired.
- The user tried to log onto Workflow Manager but is not a member of the workflow group.
- The Demantra database is not running.
- Demantra is configured incorrectly. Specifically, the DBName parameter has not been set correctly.

#### **Resolution**

1. If the user tried to log onto Workflow Manager, make sure that the user is a member of the workflow group, as specified by the workflow.group parameter; see "Providing Access to the Workflow Editor".
2. Make sure that the database is running.
3. Make sure that the DBName parameter has been set correctly. Typically, with an Oracle installation, the problem is that you have set this parameter to the host name of the database machine, rather than the Oracle SID. See the Oracle Demantra Installation Guide.
4. Contact Oracle and get a new license.

## **No Shortcut to Access Offline Worksheets**

#### **Scenario**

A user is trying to log into an offline worksheet but does not have a shortcut set up on the machine to do this.

#### **Explanation**

The user either deleted the shortcuts or failed to create them when prompted.

#### **Resolution**

First make sure that the following directory exists on this user's machine:

C:\Documents and Settings\*username*\.javaws\cache\indirect\

This directory should include a file with a name such as indirect46519.ind. The filename will include a different number.

- If this file does not exist, probably the user has not performed the setup steps to configure offline access. See "Initial Setup and Logon of Offline Environment".
- If this file does exist, then create a shortcut as follows:

Detail	Value to Use
Title	Demantra Offline Worksheets
Target type	Application
Target location	Java Web Start
Target	"C:\Program Files\Java Web Start\javaws.exe" "@C:\Documents and Settings\username\javaws\cache\indirect\ind_file_as_above"
Start in	leave blank
Run	Normal window

## No Suitable Driver (Integration)

### Scenario

A user tries to perform import or export via the Stand-Alone Integration Tool (Demantra\_root/Demand Planner/Integration/aps.bat), but gets the following message in the DOS shell:

```
java.lang.ClassNotFoundException:
...
Error in Connection to Database, No suitable driver.
Retrying to Connect...
```

### Explanation

This error occurs if you have the incorrect setting of the ApsMode parameter.

### Resolution

See the Oracle Demantra Installation Guide.

## Object Error

### Scenario

A user receives the following message when trying to launch a worksheet in one of the Web products:

```
error while running Worksheet Window: object error
```

### Possible Explanations

This message can occur for several reasons:

- There may be an incorrect setting in an XML file on the Web server.
- There may be a problem on this user's machine.
- It may be necessary to clean the Tomcat cache on the Web server.

### Resolution

1. First make sure that the Web server is pointing to the correct location.
  1. On the server, open the following file:  
*Demantra\_root/Collaborator/virtual\_directory/WEB-INF/web.xml* For example:  
*Demantra\_root/Collaborator/demantra/WEB-INF/web.xml*
  2. In this file, check the value of the parameter `electric.http.url`. This parameter should have the following format and value:  
`http://server name/virtual_directory/glue`  
 For example: **http://frodo/demantra/glue**
  3. Edit the file if necessary, save the changes, and then restart the Web server.
2. Then try to resolve a Java problem on the user's machine:
  1. Make sure that the browser option "Use Java2 v1.4.1\_03 for applet" is unchecked. To access this option in the browser, click Tools > Internet Options...and click the Advanced tab.
  2. Stop the Web server, clear the user's Java cache and certificate, and restart the Web server, as described in "Cannot Connect to Server".
  3. If this does not fix the problem, then uninstall JRE. To do so, use Add/Remove Programs and restart the machine if prompted to do so.
  4. Then see "Initial Logon to Demantra".
3. Clear the Tomcat cache on the server. To do so, delete the directory **Demantra\_root** /Collaborator/Tomcat/work/standalone/localhost.

## Page Cannot Be Displayed

### Scenario

A user accesses one of the Demantra URLs and receives the following message:

The page cannot be displayed

### Explanation

The Web server is not running.



### Resolution

Start the Web server.

## Please Handle Recovery

### Scenario

A user receives an email message that includes the following text:

Please handle recovery for the following process

### Explanation

This error message is the default message that the Workflow Engine sends when asking a user how to recover from a failed workflow instance. (The message itself is controlled by the mail.strings.recovery parameter.)

### Resolution

Identify the workflow instance that failed. The Workflow Manager shows all workflows and all workflow instances.

Recovery depends on the workflow and on your environment.

## Port Collision with Oracle XDB User

### Scenario

When the user tries to log on, he or she receives the following message:



### Explanation

In Oracle 9i, the XDB database user tries to use port 8080. If you use port 8080 for Demantra, then users will encounter the message described above.

### Resolution

Reconfigure the Oracle XDB user to use a different port, as follows:

1. Open the Oracle Enterprise Manager and log in as a DBA.
2. On the left side of the screen, expand the XML Database item and click the Configuration item.
3. On the right side of the screen, edit the http-port field and change the value from 8080 to another four-digit number.
4. Save the change.
5. To verify the fix, try to access <http://localhost:8080>. You should get a blank page.

## Possible Jar Problem

### Scenario and Explanation

If Java is not using the correct jar files on a user's machine, different errors can occur, specifically:

- The Demantra Web pages seem wrong.
- Error messages occur.
- You suspect that the correct jar files were not downloaded.

### Resolution

1. Check the dates of the jar files that Demantra is using:
  1. Start the Web server.
  2. Note the messages that the APS writes into the DOS window. Look for the line that is similar to the following example:  
  
Starting Service Demantra Application Server/7.0.0 (27)
  3. The number in parentheses indicates the jar version that the APS is using.
2. Check the dates of the jar files that Java is actually using, as follows:
  1. Open the Java plug-in control panel.
  2. Click the Cache tab.
  3. Click View.

## Process Is Terminated

### Scenario

A user receives an email message with the following subject line:

The following process is terminated

#### **Explanation**

This error message is the default message that the Workflow Engine sends when a workflow instance is terminated. (The message itself is controlled by the mail.strings.processterminated parameter.)

#### **Resolution**

Ask your Demantra administrator if he or she terminated the workflow instance.

## **Tasks Timed Out**

#### **Scenario**

A user receives an email message that includes the following subject line:

Task(s) timed out in workflow

Depending on the workflow, the text of the message includes one of the following lines:

Treatment period for this task(s) was finished and one or more of the group members haven't respond. The process moved to alternative treatment.

Treatment period for this task(s) was finished and the process moved to alternative treatment

#### **Explanation**

These error message are the default message that the Workflow Engine sends when a User Step or a Group Step times out. (The messages themselves are controlled by the mail.strings.taskstimedoutsubject, mail.strings.timeout.user, and mail.strings.timeout.group parameters.)

The workflow definition defines the timeout periods and the timeout behavior.

#### **Resolution**

Identify which workflow step was timed out, and consult your Oracle implementors for details on that workflow. Depending on how the workflow was defined, the alternative treatment may be sufficient for your needs.

## **Treatment Period Was Finished**

See "Tasks Timed Out".

## **Unable to Launch Desktop from Collaborator**

#### **Scenario**

A user tries to launch the desktop from Collaborator Workbench, but encounters an error.

**Explanation**

There may be a problem with the TNS configuration.

**Resolution**

Make sure the TNS name matches the server name. To do so:

1. Make sure you have no spaces in your path to dp.exe OR put dp.exe in your path (able to launch it from a command prompt). The menu should be set up correctly as follows:

Program Target: dp.exe

Type: desktop initiation

2. Set up a TNS on your machine whose name is the same as the database server name. For example, if your database server is WYSIWYG, create a TNS named WYSIWYG. Make sure the corresponding TNS exists on the database server.
3. Edit the ServerName parameter in the Business Modeler. This parameter is on the Application Server > App Server tab.

## Unable to Log into Collaborator After Expired Session

**Scenario**

A user has logged into Collaborator Workbench and the session has expired. Now the user cannot log in again.

**Explanation**

The browser has not been updated correctly with the user status.

**Resolution**

Close the browser window and open a new browser. In the new browser, the user will be able to log on again.

## User Does Not Receive Email

**Scenario**

A user fails to receive an email message, either from an automated workflow or from another user of Collaborator Workbench.

**Explanation**

Demantra uses the email addresses configured in the Business Modeler.

**Resolution**

In the Business Modeler, make sure that the email address is configured correctly for this user. See "Creating or Modifying a User".

Also check with IT department to make sure that the Demantra administrative user is configured with the appropriate permissions on the mail server.

## User Is Already Active

### Scenario

A user tries to log into one of the Web products and receives the following message:

User is already active

### Explanation

Any given user can open only one session at a time. This applies whether or not the user is trying to log on from the same computer.

### Resolution

Either wait for the user session to time out or manually end the user session. Contact Oracle Support Services for details.

## Workflow Internal Error

### Scenario

A user receives an email message with the following subject line:

Workflow internal error

The text of the message itself includes the following:

Internal error: please check database and network connections.

### Explanation

This error message is the default message that the Workflow Engine sends when an internal error occurs in the workflow module. (The message itself is controlled by the mail.strings.internalerror.subject and mail.strings.internalerror.message parameters.)

### Resolution

First try to determine if the error was caused by a database communication failure, lost network connection, or unavailability of the Web server. Correct the situation and re-execute the workflow instance.

If you cannot determine the cause of the error, gather as much information as possible and contact Oracle Support Services.

## Workflow Process Has Failed

### Scenario

A user receives an email message with the following subject line:

Workflow process has failed

The text of the message itself includes the following:

Error in process execution

### Explanation

This error message is the default message that the Workflow Engine sends to the initiator of a workflow when it fails to execute any step in that workflow. (See "Fail-To-Execute Step". (The message itself is controlled by the mail.strings.taskfailuresubject and mail.strings.processfailuresubject parameters.)

In such cases, the Workflow Engine also sends a selection task to the My Tasks module for the same user. This task provides options for continuing.

### Resolution

1. Identify the workflow that failed and try to identify the cause of the failure.

A failure can happen for a variety of reasons, for example, an invalid worksheet or user, a database communication error, the Web server being down, or failure of an invoked external application. Check for such error conditions.

2. The user who initiated the workflow should log onto Collaborator Workbench, go to My Tasks, and specify how to proceed.
  - If you have corrected the underlying problem and you want to rerun the step that failed, click Retry.
  - If you have corrected the underlying problem and have performed the failed step manually, click Continue.
  - If you want to cancel execution of this workflow instance, click Abort.

Then click the Save & Refresh link at the bottom of the task list.

3. If you cannot determine the cause, gather as much related information as possible, and contact Oracle Support Services.

## Worksheet Is Empty

### Scenario

A user opens a worksheet, but the worksheet is empty.

### Explanation

The worksheet contains no data. There are multiple possible reasons:

- The user may not have access to the specific data. For example, a worksheet shows data for a specific account, but the user is not authorized for that account.
- The user may not be not permitted to see data at the aggregation levels in the worksheet.

- The user may not have access to the series shown in the worksheet.
- There may be no data within the span of time that the worksheet uses.
- There may be an exception-type filter applied to the worksheet, but no data meets the exception condition.

#### **Resolution**

1. Try increasing the span of time of the worksheet.
2. Check the user's permissions.
3. Check the worksheet's filter and exception filter. Remember that if you launch a worksheet via the Open With menu option, the worksheet is filtered by the member from which you started. This additional filter is not visible in the worksheet designer.

## **Worksheet Runs Slowly**

#### **Scenario**

Your system has a large number of series, and worksheets take a long time to run.

#### **Explanation**

The tables that store the series might benefit from being rebuilt.

#### **Resolution**

Run the REBUILD\_TABLES procedure, which rebuilds the sales\_data and mdp\_matrix tables by default. You can pass a table name as an argument to the procedure.

## **Zero Length Column Error**

#### **Scenario**

When working with a cached worksheet, the following error is encountered:

```
Zero length column
```

#### **Explanation**

For technical reasons, a worksheet cache cannot be created if any server expressions in that worksheet return null or zero-length values.

#### **Resolution**

Check the server expressions for all series and modify them if any return such values. Use the expression to\_number(null,0) to express null values that can be cached.

## ORA-4043 Error When Running MSDDEMLD

Loader Worker fails with the following: Creating dummy log file ... Parent Program Name: MSDDEMLD This IS part of a Plan run. Username:SQL\*Loader-941: Error during describe of table DMTRA\_TEMPLATE.T\_SRC\_SALES\_TMPL ORA-04043: object DMTRA\_TEMPLATE.T\_SRC\_SALES\_TMPL does not exist SQL\*Loader: Release 10.1.0.5.0 - Production on Wed Dec 12 16:03:29 2007 Copyright (c) 1982, 2005, Oracle. All rights reserved. Program exited with status 1

The control file (\$MSC\_TOP/patch/115/import/T\_SRC\_SALES\_TMPL.ctl) used to load Shipment and Booking Data from flat files to the Demantra schema table T\_SRC\_SALES\_TMPL has the schema name 'DMTRA\_TEMPLATE' hardcoded in it

If the demantra schema name is other than 'DMTRA\_TEMPLATE' and the customer wants to use the Self Service program for loading Shipment and Booking History through flat files, then this control file should be manually updated to use the correct Demantra schema name.



---

## Key Tables

### T\_SRC Tables

#### T\_SRC\_ITEM\_TMPL

This staging table is used by the ep\_load\_main procedure. Each record corresponds to a unique item entity based on all lowest item levels in the model. For information on the hierarchy, see "Item Levels".

Name	Type	Purpose
EQ_UNIT	NUMBER(20,10)	Conversion ratio between units and equivalent units
T_EP_I_ATT_1	VARCHAR2(100)	Item Description
T_EP_I_ATT_2	VARCHAR2(100)	Franchise, not used in PTP model, may have N/A
T_EP_I_ATT_3	VARCHAR2(100)	Genre, not used in PTP model, may have N/A
T_EP_I_ATT_4	VARCHAR2(100)	Platform, not used in PTP model, may have N/A
T_EP_I_ATT_5	VARCHAR2(100)	Rating, not used in PTP model, may have N/A
T_EP_I_ATT_6	VARCHAR2(100)	Release Month, not used in PTP model, may have N/A
T_EP_I_ATT_7	VARCHAR2(100)	UPC Code, not used in PTP model, may have N/A

<b>Name</b>	<b>Type</b>	<b>Purpose</b>
T_EP_I_ATT_8	VARCHAR2(100)	Media Type, not used in PTP model, may have N/A
T_EP_I_ATT_9	VARCHAR2(100)	Product size, not used in PTP model, may have N/A
T_EP_I_ATT_10	VARCHAR2(100)	Product Weight, not used in PTP model, may have N/A
T_EP_P1	VARCHAR2(100)	Parent model information
T_EP_P2	VARCHAR2(100)	Product Subgroup level information
T_EP_P2A	VARCHAR2(100)	Brand level information
T_EP_P2A1	VARCHAR2(100)	Segment level information
T_EP_P2A2	VARCHAR2(100)	Category level information
T_EP_P2B	VARCHAR2(100)	Promotion Group level information
T_EP_P3	VARCHAR2(100)	Product Group level information
T_EP_P4	VARCHAR2(100)	Product Line level information
DM_ITEM_CODE	VARCHAR2(240)	Item Level Code
DM_ITEM_DESC	VARCHAR2(240)	Item Level Description
E1_PLANNING_UOM	VARCHAR2(100)	Unit of measure of E1 demand plan
E1_SHIPPING_UOM	VARCHAR2(100)	Unit of measure E1 shipped
E1_SHIPPING_UOM_MULTIPLE	NUMBER(20,10)	Conversion factor to E1 Shipping UOM
E1_WEIGHT_UOM	VARCHAR2(100)	E1 weight of 1 UOM

<b>Name</b>	<b>Type</b>	<b>Purpose</b>
E1_WEIGHT_UOM_MULTIPLE	NUMBER(20,10)	Conversion factor to E1 weight UOM
E1_VOLUME_UOM	VARCHAR2(100)	E1 weight of 1 UOM
E1_VOLUME_UOM_MULTIPLE	NUMBER(20,10)	Conversion factor to E1 volume UOM
E1_PRIMARY_UOM	VARCHAR2(100)	E1 primary UOM
E1_PRIMARY_PLANNING_FACTOR	NUMBER(20,10)	Conversion of E1 primary UOM to planning units
E1_ITEM_CATEGORY_1	VARCHAR2(100)	Item Category from E1 to Demantra
E1_ITEM_CATEGORY_2	VARCHAR2(100)	Item Category from E1 to Demantra
E1_ITEM_CATEGORY_3	VARCHAR2(100)	Item Category from E1 to Demantra
E1_ITEM_CATEGORY_4	VARCHAR2(100)	Item Category from E1 to Demantra
E1_ITEM_CATEGORY_5	VARCHAR2(100)	Item Category from E1 to Demantra
E1_ITEM_CATEGORY_6	VARCHAR2(100)	Item Category from E1 to Demantra
E1_ITEM_CATEGORY_7	VARCHAR2(100)	Item Category from E1 to Demantra
E1_ITEM_CATEGORY_8	VARCHAR2(100)	Placeholder for E1 Category code. Currently not loaded
E1_ITEM_CATEGORY_9	VARCHAR2(100)	Placeholder for E1 Category code. Currently not loaded

<b>Name</b>	<b>Type</b>	<b>Purpose</b>
E1_ITEM_CATEGOR Y_10	VARCHAR2(100)	Placeholder for E1 Category code. Currently not loaded
E1_ITEM_CATEGOR Y_11	VARCHAR2(100)	Placeholder for E1 Category code. Currently not loaded
E1_ITEM_CATEGOR Y_12	VARCHAR2(100)	Placeholder for E1 Category code. Currently not loaded
E1_ITEM_CATEGOR Y_13	VARCHAR2(100)	Placeholder for E1 Category code. Currently not loaded
E1_ITEM_CATEGOR Y_14	VARCHAR2(100)	Placeholder for E1 Category code. Currently not loaded
E1_ITEM_CATEGOR Y_15	VARCHAR2(100)	Placeholder for E1 Category code. Currently not loaded
E1_ITEM_CATEGOR Y_16	VARCHAR2(100)	Placeholder for E1 Category code. Currently not loaded
E1_ITEM_CATEGOR Y_17	VARCHAR2(100)	Placeholder for E1 Category code. Currently not loaded
E1_ITEM_CATEGOR Y_18	VARCHAR2(100)	Placeholder for E1 Category code. Currently not loaded
E1_ITEM_CATEGOR Y_19	VARCHAR2(100)	Placeholder for E1 Category code. Currently not loaded
E1_ITEM_CATEGOR Y_20	VARCHAR2(100)	Placeholder for E1 Category code. Currently not loaded
E1_ITEM_CATEGOR Y_21	VARCHAR2(100)	Placeholder for E1 Category code. Currently not loaded
E1_ITEM_CATEGOR Y_22	VARCHAR2(100)	Placeholder for E1 Category code. Currently not loaded
E1_ITEM_CATEGOR Y_23	VARCHAR2(100)	Placeholder for E1 Category code. Currently not loaded

<b>Name</b>	<b>Type</b>	<b>Purpose</b>
EBS_ACCOUNT_D ESC	VARCHAR2(240)	EBS Account description
EBS_PRODUCT_CA TEGORY_CODE	VARCHAR2(240)	Product Category level code
EBS_PRODUCT_CA TEGORY_DESC	VARCHAR2(240)	Product Category level description
EBS_PRODUCT_FA MILY_CODE	VARCHAR2(240)	Product Family level code
EBS_PRODUCT_FA MILY_DESC	VARCHAR2(240)	Product Family level description
EBS_DEMAND_CL ASS_CODE	VARCHAR2(240)	Demand Class code
EBS_DEMAND_CL ASS_DESC	VARCHAR2(240)	Demand Class Description
EBS_ITEM_DEST_K EY	NUMBER(10)	Item Destination Key- Numeric
EBS_DEMAND_CL ASS_DEST_KEY	NUMBER(10)	Demand Class Destination Key- Numeric

#### **T\_SRC\_LOC\_TMPL**

This staging table is used by the ep\_load\_main procedure. Each record corresponds to a unique location entity based on all lowest item levels in the model. For information on the hierarchy, see "Location Levels".

<b>Name</b>	<b>Type</b>	<b>Purpose</b>
LR1_ATTR1	VARCHAR2(100)	Generic Retailer Attribute 1
LR1_ATTR2	VARCHAR2(100)	Generic Retailer Attribute 2
T_EP_L_ATT_1	VARCHAR2(100)	Store Size, Not used in PTP, default to N/A

<b>Name</b>	<b>Type</b>	<b>Purpose</b>
T_EP_L_ATT_2	VARCHAR2(100)	Store Classification, Not used in PTP, default to N/A
T_EP_L_ATT_3	VARCHAR2(100)	Store Type, Not used in PTP, default to N/A
T_EP_L_ATT_4	VARCHAR2(100)	Customer Segment, Not used in PTP, default to N/A
T_EP_L_ATT_5	VARCHAR2(100)	Store Manager, Not used in PTP, default to N/A
T_EP_L_ATT_6	VARCHAR2(100)	Store Language, Not used in PTP, default to N/A
T_EP_L_ATT_7	VARCHAR2(100)	Store Revenue Class, Not used in PTP, default to N/A
T_EP_L_ATT_8	VARCHAR2(100)	Site Zip Code, Not used in PTP, default to N/A
T_EP_L_ATT_9	VARCHAR2(100)	Store Demographic, Not used in PTP, default to N/A
T_EP_L_ATT_10	VARCHAR2(100)	Site City - Not used in PTP, default to N/A
T_EP_LR1	VARCHAR2(100)	Supplier Code for RTS
T_EP_LR2	VARCHAR2(100)	Bill-To code
T_EP_LR2A	VARCHAR2(100)	Retailer information
T_EP_LS1	VARCHAR2(100)	Territory Information
T_EP_LS2	VARCHAR2(100)	District Information
T_EP_LS3	VARCHAR2(100)	Region Information
T_EP_LS4	VARCHAR2(100)	Sales Area Information
T_EP_LS5	VARCHAR2(100)	Division Information
T_EP_LS6	VARCHAR2(100)	Company Information

<b>Name</b>	<b>Type</b>	<b>Purpose</b>
DM_SITE_CODE	VARCHAR2(240)	Site level code
DM_SITE_DESC	VARCHAR2(240)	Site level description
DM_ORG_CODE	VARCHAR2(240)	Organization level code
DM_ORG_DESC	VARCHAR2(240)	Organization level description
E1_BRANCH_CITY	VARCHAR2(100)	E1 Branch information City
E1_BRANCH_STAT E	VARCHAR2(100)	E1 Branch information State
E1_BRANCH_COU NTR	VARCHAR2(100)	E1 Branch information County
E1_BRANCH_ZIP_ CODE	VARCHAR2(100)	E1 Branch information Zip Code
E1_BRANCH_CATE GORY_1	VARCHAR2(100)	E1 Branch Category Code
E1_BRANCH_CATE GORY_2	VARCHAR2(100)	E1 Branch Category Code
E1_BRANCH_CATE GORY_3	VARCHAR2(100)	E1 Branch Category Code
E1_BRANCH_CATE GORY_4	VARCHAR2(100)	E1 Branch Category Code
E1_BRANCH_CATE GORY_5	VARCHAR2(100)	E1 Branch Category Code
E1_BRANCH_CATE GORY_6	VARCHAR2(100)	Placeholder for E1 Branch Category code. Not loaded
E1_BRANCH_CATE GORY_7	VARCHAR2(100)	Placeholder for E1 Branch Category code. Not loaded

<b>Name</b>	<b>Type</b>	<b>Purpose</b>
E1_BRANCH_CATE GORY_8	VARCHAR2(100)	Placeholder for E1 Branch Category code. Not loaded
E1_BRANCH_CATE GORY_9	VARCHAR2(100)	Placeholder for E1 Branch Category code. Not loaded
E1_BRANCH_CATE GORY_10	VARCHAR2(100)	Placeholder for E1 Branch Category code. Not loaded
E1_BRANCH_CATE GORY_11	VARCHAR2(100)	Placeholder for E1 Branch Category code. Not loaded
E1_BRANCH_CATE GORY_12	VARCHAR2(100)	Placeholder for E1 Branch Category code. Not loaded
E1_BRANCH_CATE GORY_13	VARCHAR2(100)	Placeholder for E1 Branch Category code. Not loaded
E1_BRANCH_CATE GORY_14	VARCHAR2(100)	Placeholder for E1 Branch Category code. Not loaded
E1_BRANCH_CATE GORY_15	VARCHAR2(100)	Placeholder for E1 Branch Category code. Not loaded
E1_BRANCH_CATE GORY_16	VARCHAR2(100)	Placeholder for E1 Branch Category code. Not loaded
E1_BRANCH_CATE GORY_17	VARCHAR2(100)	Placeholder for E1 Branch Category code. Not loaded
E1_BRANCH_CATE GORY_18	VARCHAR2(100)	Placeholder for E1 Branch Category code. Not loaded
E1_BRANCH_CATE GORY_19	VARCHAR2(100)	Placeholder for E1 Branch Category code. Not loaded
E1_BRANCH_CATE GORY_20	VARCHAR2(100)	Placeholder for E1 Branch Category code. Not loaded
E1_BRANCH_CATE GORY_21	VARCHAR2(100)	Placeholder for E1 Branch Category code. Not loaded



<b>Name</b>	<b>Type</b>	<b>Purpose</b>
E1_BRANCH_CATE GORY_22	VARCHAR2(100)	Placeholder for E1 Branch Category code. Not loaded
E1_BRANCH_CATE GORY_23	VARCHAR2(100)	Placeholder for E1 Branch Category code. Not loaded
E1_BRANCH_CATE GORY_24	VARCHAR2(100)	Placeholder for E1 Branch Category code. Not loaded
E1_BRANCH_CATE GORY_25	VARCHAR2(100)	Placeholder for E1 Branch Category code. Not loaded
E1_BRANCH_CATE GORY_26	VARCHAR2(100)	Placeholder for E1 Branch Category code. Not loaded
E1_BRANCH_CATE GORY_27	VARCHAR2(100)	Placeholder for E1 Branch Category code. Not loaded
E1_BRANCH_CATE GORY_28	VARCHAR2(100)	Placeholder for E1 Branch Category code. Not loaded
E1_BRANCH_CATE GORY_29	VARCHAR2(100)	Placeholder for E1 Branch Category code. Not loaded
E1_BRANCH_CATE GORY_30	VARCHAR2(100)	Placeholder for E1 Branch Category code. Not loaded
E1_CUSTOMER_CI TY	VARCHAR2(100)	E1 Customer information city
E1_CUSTOMER_ST ATE	VARCHAR2(100)	E1 Customer information state
E1_CUSTOMER_CO UNTRY	VARCHAR2(100)	E1 Customer information county
E1_CUSTOMER_BU SINESS_UNIT	VARCHAR2(100)	E1 Customer information business unit
E1_CUSTOMER_M AILING_NAME	VARCHAR2(100)	E1 Customer information mailing name

<b>Name</b>	<b>Type</b>	<b>Purpose</b>
E1_CUSTOMER_ZIP_CODE	VARCHAR2(100)	E1 Customer information zip code
E1_CUSTOMER_CATEGORY_1	VARCHAR2(100)	E1 Customer Category Code
E1_CUSTOMER_CATEGORY_2	VARCHAR2(100)	E1 Customer Category Code
E1_CUSTOMER_CATEGORY_3	VARCHAR2(100)	E1 Customer Category Code
E1_CUSTOMER_CATEGORY_4	VARCHAR2(100)	E1 Customer Category Code
E1_CUSTOMER_CATEGORY_5	VARCHAR2(100)	E1 Customer Category Code
E1_CUSTOMER_CATEGORY_6	VARCHAR2(100)	E1 Customer Category Code
E1_CUSTOMER_CATEGORY_7	VARCHAR2(100)	E1 Customer Category Code
E1_CUSTOMER_CATEGORY_8	VARCHAR2(100)	Placeholder for E1 Customer Category code. Not loaded
E1_CUSTOMER_CATEGORY_9	VARCHAR2(100)	Placeholder for E1 Customer Category code. Not loaded
E1_CUSTOMER_CATEGORY_10	VARCHAR2(100)	Placeholder for E1 Customer Category code. Not loaded
E1_CUSTOMER_CATEGORY_11	VARCHAR2(100)	Placeholder for E1 Customer Category code. Not loaded
E1_CUSTOMER_CATEGORY_12	VARCHAR2(100)	Placeholder for E1 Customer Category code. Not loaded
E1_CUSTOMER_CATEGORY_13	VARCHAR2(100)	Placeholder for E1 Customer Category code. Not loaded

<b>Name</b>	<b>Type</b>	<b>Purpose</b>
E1_CUSTOMER_CATEGORY_14	VARCHAR2(100)	Placeholder for E1 Customer Category code. Not loaded
E1_CUSTOMER_CATEGORY_15	VARCHAR2(100)	Placeholder for E1 Customer Category code. Not loaded
E1_CUSTOMER_CATEGORY_16	VARCHAR2(100)	Placeholder for E1 Customer Category code. Not loaded
E1_CUSTOMER_CATEGORY_17	VARCHAR2(100)	Placeholder for E1 Customer Category code. Not loaded
E1_CUSTOMER_CATEGORY_18	VARCHAR2(100)	Placeholder for E1 Customer Category code. Not loaded
E1_CUSTOMER_CATEGORY_19	VARCHAR2(100)	Placeholder for E1 Customer Category code. Not loaded
E1_CUSTOMER_CATEGORY_20	VARCHAR2(100)	Placeholder for E1 Customer Category code. Not loaded
E1_CUSTOMER_CATEGORY_21	VARCHAR2(100)	Placeholder for E1 Customer Category code. Not loaded
E1_CUSTOMER_CATEGORY_22	VARCHAR2(100)	Placeholder for E1 Customer Category code. Not loaded
E1_CUSTOMER_CATEGORY_23	VARCHAR2(100)	Placeholder for E1 Customer Category code. Not loaded
E1_CUSTOMER_CATEGORY_24	VARCHAR2(100)	Placeholder for E1 Customer Category code. Not loaded
E1_CUSTOMER_CATEGORY_25	VARCHAR2(100)	Placeholder for E1 Customer Category code. Not loaded
E1_CUSTOMER_CATEGORY_26	VARCHAR2(100)	Placeholder for E1 Customer Category code. Not loaded
E1_CUSTOMER_CATEGORY_27	VARCHAR2(100)	Placeholder for E1 Customer Category code. Not loaded

<b>Name</b>	<b>Type</b>	<b>Purpose</b>
E1_CUSTOMER_CATEGORY_28	VARCHAR2(100)	Placeholder for E1 Customer Category code. Not loaded
E1_CUSTOMER_CATEGORY_29	VARCHAR2(100)	Placeholder for E1 Customer Category code. Not loaded
E1_CUSTOMER_CATEGORY_30	VARCHAR2(100)	Placeholder for E1 Customer Category code. Not loaded
EBS_TP_ZONE_CODE	VARCHAR2(240)	EBS Trading Partner Zone code
EBS_TP_ZONE_DESCRIPTION	VARCHAR2(240)	EBS Trading Partner Zone description
EBS_ZONE_CODE	VARCHAR2(240)	EBS Zone code
EBS_ZONE_DESCRIPTION	VARCHAR2(240)	EBS Zone description
EBS_ACCOUNT_CODE	VARCHAR2(240)	EBS Account code
EBS_CUSTOMER_CODE	VARCHAR2(240)	EBS Customer code
EBS_CUSTOMER_DESCRIPTION	VARCHAR2(240)	EBS Customer Description
EBS_CUSTOMER_CLASS_CODE	VARCHAR2(240)	EBS Customer Class code
EBS_CUSTOMER_CLASS_DESCRIPTION	VARCHAR2(240)	EBS Customer Class Description
EBS_OPERATION_UNIT_CODE	VARCHAR2(240)	Unit code for EBS operation
EBS_OPERATION_UNIT_DESCRIPTION	VARCHAR2(240)	Unit description for EBS operation
EBS_BUSINESS_GROUP_CODE	VARCHAR2(240)	EBS Business Group code

<b>Name</b>	<b>Type</b>	<b>Purpose</b>
EBS_BUSINESS_GRP_DESC	VARCHAR2(240)	EBS Business Group description
EBS_LEGAL_ENTITY_CODE	VARCHAR2(240)	EBS Legal Entity code
EBS_LEGAL_ENTITY_DESC	VARCHAR2(240)	EBS Legal Entity description
EBS_SALES_CHANNEL_CODE	VARCHAR2(240)	EBS Sales Channel code
EBS_SALES_CHANNEL_DESC	VARCHAR2(240)	EBS Sales Channel description
EBS_SITE_DEST_KEY	NUMBER(10)	EBS Site Destination Key - Numeric
EBS_ORG_DEST_KEY	NUMBER(10)	EBS Organization Destination Key - Numeric
EBS_SALES_CHANNEL_DEST_KEY	NUMBER(10)	EBS Sales Channel Destination Key - Numeric
EBS_SUPPLIER_CODE	VARCHAR2(240)	EBS Supplier code
EBS_SUPPLIER_DESC	VARCHAR2(240)	EBS Supplier description
E1_PARENT_ADDRESS_NUM	VARCHAR2(100)	E1 Parent address information - number
T_EP_LR2_DESC	VARCHAR2(200)	Bill-To description
E1_PARENT_ADDRESS_DESC	VARCHAR2(200)	E1 Parent address information - description

#### **T\_SRC\_SALES\_TMPL**

This staging table is used by the ep\_load\_main procedure. Each record corresponds to sales data for a given item and location combination, based on all lowest item levels in the model. For information on the hierarchy, see "Item Levels" and "Location Levels".

<b>Name</b>	<b>Type</b>	<b>Purpose</b>
T_EP_P1	VARCHAR2(100)	Parent Model
T_EP_LR1	VARCHAR2(100)	Ship To
T_EP_LS1	VARCHAR2(100)	Territory
ACTUAL_QTY	NUMBER(20,10)	Populates the Actuals Ttl series
BASE_EVT_DOL_R TL	NUMBER(20,10)	Populates the Base Evt \$ Rtl sd series
INCR_EVT_DOL_R TL	NUMBER(20,10)	Populates the Incr Evt \$ Rtl sd series.
ITEM_PRICE	NUMBER(20,10)	Populates the Avg Rtl sd series.
SALES_DATE	DATE	Date of the sale.
SDATA10	NUMBER(20,10)	Populates the % ACV DISP series.
SDATA11	NUMBER(20,10)	Populates the % ACV FEAT series.
SDATA12	NUMBER(20,10)	Populates the % ACV FEAT series.
SDATA13	NUMBER(20,10)	Populates the % ACV TPR series.
SDATA14	NUMBER(20,10)	Populates the % ACV FREQSHOPPER series.
SDATA4	NUMBER(20,10)	Populates the Shipments series
SDATA5	NUMBER(20,10)	Populates the Actuals Base series.
SDATA6	NUMBER(20,10)	Populates the Actuals Incr series.
SDATA7	NUMBER(20,10)	Populates the COGS series.
SDATA8	NUMBER(20,10)	Populates the List Price series.
SDATA9	NUMBER(20,10)	Populates the % ACV ANY PROMO series.

<b>Name</b>	<b>Type</b>	<b>Purpose</b>
SHELF_PRICE_SD	NUMBER(20,10)	Populates the Shelf Price sd series.
T_EP_M1	VARCHAR2(100)	Not used.
T_EP_M2	VARCHAR2(100)	Not used.
SDATA15	NUMBER(20,10)	Not used.
DM_ITEM_CODE	VARCHAR2(240)	Item Code
DM_ORG_CODE	VARCHAR2(240)	Organization Code
DM_SITE_CODE	VARCHAR2(240)	Site Code
E1_ITEM_BRANCH _CATEGORY_1	VARCHAR2(100)	E1 Item Branch Category Code
E1_ITEM_BRANCH _CATEGORY_2	VARCHAR2(100)	E1 Item Branch Category Code
E1_ITEM_BRANCH _CATEGORY_3	VARCHAR2(100)	E1 Item Branch Category Code
E1_ITEM_BRANCH _CATEGORY_4	VARCHAR2(100)	E1 Item Branch Category Code
E1_ITEM_BRANCH _CATEGORY_5	VARCHAR2(100)	E1 Item Branch Category Code
E1_ITEM_BRANCH _CATEGORY_6	VARCHAR2(100)	Placeholder for E1 Item Branch Category code. Not loaded
E1_ITEM_BRANCH _CATEGORY_7	VARCHAR2(100)	Placeholder for E1 Item Branch Category code. Not loaded
E1_ITEM_BRANCH _CATEGORY_8	VARCHAR2(100)	Placeholder for E1 Item Branch Category code. Not loaded
E1_ITEM_BRANCH _CATEGORY_9	VARCHAR2(100)	Placeholder for E1 Item Branch Category code. Not loaded

<b>Name</b>	<b>Type</b>	<b>Purpose</b>
E1_ITEM_BRANCH_CATEGORY_10	VARCHAR2(100)	Placeholder for E1 Item Branch Category code. Not loaded
E1_ITEM_BRANCH_CATEGORY_11	VARCHAR2(100)	Placeholder for E1 Item Branch Category code. Not loaded
E1_ITEM_BRANCH_CATEGORY_12	VARCHAR2(100)	Placeholder for E1 Item Branch Category code. Not loaded
E1_ITEM_BRANCH_CATEGORY_13	VARCHAR2(100)	Placeholder for E1 Item Branch Category code. Not loaded
E1_ITEM_BRANCH_CATEGORY_14	VARCHAR2(100)	Placeholder for E1 Item Branch Category code. Not loaded
E1_ITEM_BRANCH_CATEGORY_15	VARCHAR2(100)	Placeholder for E1 Item Branch Category code. Not loaded
E1_ITEM_BRANCH_CATEGORY_16	VARCHAR2(100)	Placeholder for E1 Item Branch Category code. Not loaded
E1_ITEM_BRANCH_CATEGORY_17	VARCHAR2(100)	Placeholder for E1 Item Branch Category code. Not loaded
E1_ITEM_BRANCH_CATEGORY_18	VARCHAR2(100)	Placeholder for E1 Item Branch Category code. Not loaded
E1_ITEM_BRANCH_CATEGORY_19	VARCHAR2(100)	Placeholder for E1 Item Branch Category code. Not loaded
E1_ITEM_BRANCH_CATEGORY_20	VARCHAR2(100)	Placeholder for E1 Item Branch Category code. Not loaded
E1_ITEM_BRANCH_CATEGORY_21	VARCHAR2(100)	Placeholder for E1 Item Branch Category code. Not loaded
E1_ITEM_BRANCH_CATEGORY_22	VARCHAR2(100)	Placeholder for E1 Item Branch Category code. Not loaded
E1_ITEM_BRANCH_CATEGORY_23	VARCHAR2(100)	Placeholder for E1 Item Branch Category code. Not loaded



Name	Type	Purpose
EBS_DEMAND_CL ASS_CODE	VARCHAR2(240)	EBS Demand Class level code
EBS_SALES_CHAN NEL_CODE	VARCHAR2(240)	EBS Sales Channel level code
EBS_BOOK_HIST_B OOK_QTY_BD	NUMBER(20,10)	Historical Booking supplied quantity by booking date
EBS_BOOK_HIST_R EQ_QTY_BD	NUMBER(20,10)	Historical Booking requested quantity by booking date
EBS_BOOK_HIST_B OOK_QTY_RD	NUMBER(20,10)	Historical Booking supplied quantity by requested date
EBS_BOOK_HIST_R EQ_QTY_RD	NUMBER(20,10)	Historical Booking requested quantity by booking date
EBS_SHIP_HIST_SH IP_QTY_SD	NUMBER(20,10)	Historical Shipments sent shipment by ship date
EBS_SHIP_HIST_SH IP_QTY_RD	NUMBER(20,10)	Historical Shipments sent shipment by requested date
EBS_SHIP_HIST_RE Q_QTY_RD	NUMBER(20,10)	Historical Shipments requested shipment by requested date
EBS_ITEM_SR_PK	NUMBER	EBS item Code
EBS_ORG_SR_PK	NUMBER	EBS Organization Code
EBS_SITE_SR_PK	NUMBER	EBS Site Code
EBS_DEMAND_CL ASS_SR_PK	VARCHAR2(100)	EBS Demand Class Code
EBS_SALES_CHAN NEL_SR_PK	VARCHAR2(100)	EBS Sales Channel Code

**Important:** If using text files, they must contain the same fields as the staging tables above contain and in the same order.



---

# Index

## A

---

- Abort recovery option, 39-4
- Abs function, 38-3
- accumulatedOrUpdate parameter, 34-2, 43-11
- active\_forecasts\_versions parameter, 34-2, 43-2, 50-21, 50-21
- active combination, 36-9, 50-19
- Activity Browser
  - configuring, 40-3
  - introduction, 8-5, 50-6
- activity shape modeling, 53-10
- ACV data, 40-9, 40-26
- add\_zero\_combos\_to\_mdp parameter, 43-14, 58-2
- AdditionalConnectionsExpression parameter, 43-6
- addressing within influence groups, 57-3
- adjustment of forecast, 57-15
- administration
  - data cleanup script, 60-3
- Administrative Utilities, 1-5
- advanced analytics
  - enabling on desktop, 55-7
- Aggregated Base Level option, 27-19
- aggregated data, importing, 13-17, 13-18
- aggregation
  - across promotions, 50-11
  - backwards versus forwards, 25-4
  - by base level, 27-19
  - by item, 6-23, 43-12
  - by promotion or by sales, 27-19
  - by unlinked level, 26-9
  - during forecasting, 36-4, 50-20, 51-4, 57-15
  - how controlled, 3-3
  - introduction, 2-3, 2-6
  - kinds of levels, 5-6
  - SQL functions, 37-4
- aggri\_98, aggri\_99 data fields, 36-4
- alert phase of workflow step, 9-14
- align\_sales\_data\_levels\_in\_loading parameter, 34-3, 35-25, 43-11, 55-9, 55-12, 56-12
- AllowableExceptions parameter, 55-5, 58-2
- AllowNegative parameter, 57-11, 58-3
- Alpha parameter, 59-5, 59-14
- AlphaQ parameter, 59-12
- AlphaT parameter, 59-12
- ampersand, in Demantra Spectrum, 49-10, 60-10
- Analytical Engine
  - and promotion status, 40-23
  - and simulation within a workflow, 39-30
  - batch or simulation, 2-11, 49-3
    - choosing mode, 56-9
    - comparison, 57-17
  - components, 57-18
  - components and processing flow, 57-18
  - configuration settings, 66-5
  - configuring, 51-1
  - distributed mode
    - details, 57-20
    - introduction, 49-2
  - DP and PE modes, 1-5, 49-2, 55-16, 58-1, 59-2
  - forecasting and coefficients, 2-10, 50-1
  - forecasting models and engine flow, 2-10, 50-8
  - forecast tree, 2-10, 50-8

- forecast versions, 6-7
- log options and other settings, 56-4
- memory issues, 56-13
- overview, 4-3, 49-1
- PE forecast mode, 55-8, 58-4
- registering, 41-2, 56-2
- running
  - from Administrator, 56-8
  - from DOS or workflow, 56-11
  - from Start menu, 56-11
  - general notes, 56-2
  - in recovery mode, 56-17
  - workflow, 9-13
- running before using Promotion
- Optimization, 41-1
- server, 57-18
- stopping, 56-10, 56-17
- troubleshooting, 56-12, 56-12, 56-12
- tuning, 55-6
- use of promotion status, 40-5
- ANALYZE\_SCHEMA procedure, 35-3
- AnalyzeMdp parameter, 58-3
- API\_ADD\_ORA\_DEM\_LEVEL\_FILTERS, 13-9
- API\_CONFIG\_SETTLEMENT procedure, 35-3
- API\_CREATE\_ORA\_DEM\_USER, 13-9
- API\_MODIFY\_INTEG\_SERIES\_ATTR, 13-7
- API\_MODIFY\_INTEG\_SERIES\_FDATE, 13-8
- API\_MODIFY\_INTEG\_SERIES\_RDATE, 13-8
- API\_NOTIFY\_APS\_INTEGRATION, 13-7
- applet
  - installing, 66-2, 66-2, 66-4
  - version, 66-20
- application\_root parameter, 34-3
- applicationDateFormat parameter, 34-3, 43-8
- applicationDateTimeFormat parameter, 34-3, 43-8
- Application Server, 1-4
- Application Server queue, 43-2
- APPPROC\_REBUILD\_CONTEXT\_CACHES procedure, 35-3
- ApprovalProcessScope parameter, 34-4
- AppServerLocation parameter, 34-4
- AppServerSupport parameter, 34-4
- AppServerURL parameter, 34-4, 66-7
- APS
  - configuration file, 66-5
  - introduction, 1-5
- APS.bat
  - and license, 66-7
  - arguments, 29-34
- APSMODE parameter, 66-7, 66-19
- APS queue, 43-2
- ARIX model, 59-3
- ARLOGISTIC model, 59-4
- ARX model, 59-4
- ASCP integration with S&OP, 16-17
- Ask recovery option, 39-4
- associating
  - indicators with series, 6-29, 61-4
  - levels and units of measure, 28-8
  - levels and worksheets (for Open With), 61-7
  - series and series groups, 27-37
  - series and users, 27-32
  - units and levels, 28-8
  - users and groups, 61-18
- associating levels and worksheets (for Open With), 40-19
- attribute
  - adding to level, 26-9
  - and method execution, 9-4
  - automatically creating as level, 26-11
  - creating drop-down list, 26-11
  - creating lookup level as attribute, 26-2, 26-7
  - creating with the Data Model Wizard, 25-15
  - displaying when method is completed, 32-8
  - exporting value, 29-22, 29-26
  - filtering drop-down list, 26-14
  - introduction, 5-9
  - making editable by users, 32-7
  - passing to method, 32-7
  - promotional
    - and aggregation, 50-11
    - as a causal factor, 50-6
    - converting to causal factor, 54-4
    - retrieving for causal factor, 54-10
  - prompting user for value, 9-4
  - quantitative versus qualitative, 54-4
  - specifying as input to workflow, 9-18, 31-5
  - specifying default value, 26-11
  - using within method, 9-17, 32-11
- attribute, population
  - adding to a level, 26-18
  - loading, 29-25
  - passing to method, 9-16, 32-10

- presentation styles in methods, 32-8
- specifying presentation style, 32-6
- table where stored, 29-27
- using to filter condition step, 39-7
- using to filter exception step, 39-18
- audit\_history\_length parameter, 34-5, 35-5, 43-12
- AuditMailAddress parameter, 34-5, 43-10
- audit trail
  - and parameter changes, 23-7, 66-6
  - clearing old data, 35-5
  - specifying length, 43-12
- autogenerated ID, 25-12
- automating data transfers, 29-34
- AutoRunMode parameter, 34-5, 43-22
- AverageHorizon parameter, 58-4
- Avg(), 37-1
- axes of a worksheet
  - introduction, 8-6

## B

---

- Bal Explorer
  - creating schema connection, 65-8
  - creating schema connection based on another schema connection, 65-12
- BAL Explorer, 65-1
  - changing object order and display, 65-13
  - connecting to schema repository, 65-12
  - copying objects from schema to folder, 65-14
  - creating folders, 65-14
  - deleting folders, 65-14
  - deleting object from folder, 65-15
  - deleting schema connection, 65-12
  - disconnecting from schema repository, 65-13
  - editing schema connection, 65-11
  - including upgrade SQL in log file, 65-33
  - rebuilding a schema repository, 65-13
  - renaming folders, 65-14
  - setting global upgrade preferences, 65-15
  - starting manually, 65-32
  - upgrading objects in folder, 65-25
  - upgrading schemas, 65-26
  - viewing log files, 65-34
  - viewing transactions, 65-33
- batch file
  - for loading sales, item, or location data, 25-8
  - running from within workflow, 39-19

- to export data, 29-12
- to import data, 29-6
- to import supplementary data, 30-4
- batch run
  - and Distributed Engine, 57-20
  - and engine components, 57-18
  - and forecast versions, 50-21
  - and intermediate results, 56-17
  - choosing mode, 56-9
  - compared to simulation run, 2-11, 49-3, 57-17
  - starting from a workflow, 9-13
  - starting from DOS, 56-11
  - starting from Start menu, 56-11
- BatchRunMode parameter, 55-8, 58-4
- Bayesian blending, 57-14
- Bayesian Model Manager, 55-15
- bce\_log\_parm parameter, 34-5
- Bera-Jarque test (Bjtest), 57-12
- BestOrMix parameter, 57-13, 59-8, 59-11, 59-18, 59-21, 59-24
- BIIO\_IMP\_PROMO\_DATA staging table, 40-11
- BIIO\_Population staging table, 40-15
- BIIO\_Promotion staging table, 40-12
- Bjtest (Bera-Jarque test), 57-12
- BLE Step
  - and other Demantra processing steps, 9-8
  - reference, 39-3
- BLEThreadPoolSize parameter, 34-5, 43-20
- BLETimeOut parameter, 34-5
- BLEtime-out parameter, 43-20
- BottomCoefficientLevel parameter, 58-5
- branch\_data\_items table
  - introduction, 6-23, 43-12
  - specifying whether to use, 34-39
  - using, 27-35
- branch\_data table, 6-28, 27-18, 27-23, 27-27
- BranchID Multiple field, 56-6, 56-6
- brokers, 42-14
- buildNumber parameter, 34-6
- BulkLoaderBlockSize parameter, 55-7, 58-5
- BulkLoaderEnableRecovery parameter, 55-7, 58-5
- Business Application Language, 65-1
- Business Logic Engine (BLE)
  - and Analytical Engine, 56-2
  - and client expressions, 3-4, 13-17, 29-10
  - and workflows, 9-20
  - introduction, 4-4

- threading mechanism, 43-20
- workflow step, 39-3
- Business Modeler
  - associating units with levels, 28-8
  - choosing engine models, 55-14
  - configuration settings, 66-5
  - configuring causal factors, 53-6
  - configuring components, 61-1
  - configuring exchange rates, 28-1
  - configuring forecast tree, 52-1
  - configuring import for supplementary data, 30-1
  - configuring indexes, 28-1
  - configuring integration interfaces, 29-2
  - configuring levels, 5-1
  - configuring methods, 32-1
  - configuring parameters, 23-7, 55-1
  - configuring promotional causal factors, 54-8
  - configuring series, 27-2
  - configuring time units, 28-10
  - configuring units, 28-5
  - configuring update-lock expressions, 28-11
  - creating data for global causals, 53-3
  - Data Model Wizard, 25-3
  - introduction, 4-1
  - refreshing, 23-6
  - starting, 23-3
  - viewing series dependencies, 27-37
  - when to use, 4-6, 4-7, 4-8, 23-2
- Buydown series, 40-12
- BWINT model, 59-5

## C

---

- cache
  - engine
    - during initial scan, 57-3
    - during learning phase, 57-5
  - Java
    - clearing, 66-13
    - specifying size, 66-17
  - level members
    - modifying cache properties, 64-5
    - specifying policies, 64-3
  - series aggregated by items, 6-23
  - series data aggregated by item, 27-35
  - Tomcat, clearing, 66-11, 66-15

- worksheet
  - refreshing in the background, 9-8
- worksheets
  - enabling globally, 34-16, 43-22
- CACHE\_CAPACITY, 64-3
- CACHE\_IS\_LAZY, 64-3
- CACHE\_LOAD\_EXTRA\_FROM, 64-4
- CACHE\_LOAD\_EXTRA\_WHERE, 64-4
- CACHE\_TYPE, 64-4
- Cache Manager, 64-5
- CachePath parameter, 58-6
- caching
  - worksheet, 8-11
- CalcOptimizationInput parameter, 58-6
- cannibalism parameter, 58-7
- CannibalizationIgnore parameter, 55-4, 58-7
- cannot connect to database, 66-13
- cannot connect to server, 66-13
- cascade mode for filter, enabling, 61-18
- Case function, 38-4
- casual user, 10-5
- causal factors
  - and demand and forecast, 2-10, 50-1
  - and Min Len field, 55-15
  - and promotion attributes, 5-9, 54-4
  - and shape modeling, 53-10
  - base factors, 50-3
  - configuring, 53-6
  - creating data for global causals, 53-3
  - creating data for local causals, 53-5
  - data and configuration details, 50-3
  - data location, 53-1
  - data requirements, 51-2
  - deleting, 53-13
  - flags for engine models, 53-7, 59-2
  - for promotions, 54-9
  - guidelines, 51-7
  - influence of promotions on related products, 50-14
  - number of, 57-7
  - overview, 2-10, 50-2
  - promotion attributes used as, 50-6
  - self, own, and other, 50-14
  - specifying in the Data Model Wizard, 25-16
  - typical settings, 53-3
  - when to configure, 4-3, 4-7, 23-2
- chaining

- and delta\_def, 58-10
- Chaining Management
  - launching from within Collaborator, 44-7
- CHAINING procedure, 35-5
- ChainLength parameter, 59-7, 59-10, 59-15
- characters to avoid, 23-3, 49-10, 60-10
- charsetEncoding parameter, 34-7
- check\_validity\_installation parameter, 34-7
- Check Finish After option, 39-4
- Check Finish Every option, 39-4
- check request level, 5-8, 5-16, 42-16
- check requests, 42-14
- child, 5-5
- claim
  - introduction, 42-2
  - origin of, 42-3
- CLEAN\_LOG\_TABLES procedure, 34-5, 35-5
- client.enableOpenNoteWithDoubleClick parameter, 34-7
- client.JREMaxMemory parameter, 34-8
- client.lookAndFeel parameter, 34-8
- client.MaxCombsLoadChunkSize parameter, 34-8
- client.worksheet.privateAccessType parameter, 34-8
- ClientDDCacheCapacity parameter, 34-8
- client expression
  - allowed operators, 38-2
  - and export, 13-17, 29-10
  - dependencies on other series, 27-38
  - functions used in, 38-xxx
  - introduction, 3-3
  - introduction to editor, 27-21
  - literal negative values, 27-28
  - syntax, 6-8, 27-24
  - used as an update-lock expression, 28-11
  - used for series summary, 27-9
  - used with server expression, 3-5, 6-11
  - verifying all, 27-28
  - weighted average, 27-10
  - when evaluated, 3-8
- CMREGR model, 59-6
- code field
  - and extra FROM and extra WHERE, 26-5
  - and integration, 29-6
  - for dropdown-type series, 27-13
  - specifying, 25-12, 26-4
  - specifying in Data Model Wizard, 25-11
  - viewing for members, 26-22
- coefficients
  - for shapes, 53-10, 54-14
  - introduction, 50-1
  - mathematical requirements for calculating, 51-7
  - preventing negative values, 55-4
  - writing to database, 58-65
- Collaboration Services, 1-5
- collaborator.searchURL parameter, 34-9, 43-3
- collaborator.supportURL parameter, 34-8, 43-3
- Collaborator Workbench
  - and workflow, 9-7
  - browser window title, 43-17
  - cannot launch desktop from, 66-23
  - configuring, 44-1, 61-20
  - configuring content security, 44-10
  - configuring links, 44-12
  - configuring login page, 44-12
  - configuring menus, 44-2
  - customizing graphics, 40-21, 44-10
  - error in My Tasks, 66-15
  - how configurable, 1-7
  - parameters, 43-2
  - parameters related to, 43-2
  - specifying menu permissions, 61-25
  - title of home page, 43-17
  - workflow and My Tasks, 39-2
- Collaborator Workbench Administrator
  - configuring default layout of Collaborator, 44-7
  - configuring menu permissions in Collaborator, 61-21, 61-24, 61-25, 61-25
  - configuring menus in Collaborator, 44-2
  - configuring pane security in Collaborator, 44-10
  - logging on, 44-1, 61-20
  - overview, 4-4
- ColorCodingLevel parameter, 34-9, 40-3
- color expression, 6-18, 27-29
  - dependencies on other series, 27-38
- colors in worksheet table, 27-29
- combination
  - and delta, 43-13, 58-10
  - and matrix series, 6-3
  - delta field, 11-4, 43-13, 58-10

- displaying data for, 8-5
- importing new, 29-7
- inserting new forecast records, 29-7
- introduction, 2-5
- maximum number to display in graph, 43-5
- P1, P2, and so on, 11-4, 11-5, 11-6
- selecting which to display, 8-5
- combination level
  - introduction, 5-6
- combination series
  - where stored (if stored), 27-3
- company.name parameter, 34-9, 43-10, 43-17, 43-21
- COMPETITION\_ITEM parameter, 52-9, 57-3, 58-8
- COMPETITION\_LOCATION parameter, 52-9, 57-3, 58-8
- component
  - creating or modifying, 61-1
  - deleting, 61-10
  - engine profiles, specifying, 61-8
  - indexes and exchange rates
    - specifying, 61-6
    - viewing, 28-4
  - introduction, 10-1, 10-1
  - levels, specifying, 61-5
  - owner, 61-9
  - restarting server after changing, 23-9
  - series
    - adding, 61-2
    - viewing, 27-32
  - series indicators, specifying, 61-3
  - units
    - specifying, 61-6
    - viewing, 28-7
  - users, specifying, 61-10
  - worksheet-level associations, 61-7
- Component Configuration Wizard, 61-2
- COMPUTE\_STD\_ERR procedure, 35-6
- condition
  - detecting, 38-4, 38-12
  - flagging with color, 6-18, 27-29, 27-29
- conditional functions
  - Case, 38-4
  - If, 38-12
- Condition Step
  - and other logic steps, 9-8
  - reference, 39-5
- configure to order, 51-8
- configure to order levels, 5-8
- connection handles
  - deleting link, 31-4
  - introduction, 9-14
  - linking, 31-3
- constraint optimization, 47-1
- Constraint Profit Optimizer
  - creating profile, 47-2
  - deleting profile, 47-8
  - introduction, 47-1
  - running profile, 47-8
- constructor methods, 9-3, 32-3
- Container Step
  - and other logic steps, 9-8
  - reference, 39-7
- content
  - defaults for content wizard, 43-23
  - parameters related to, 43-4
  - setting security, 44-10
- Continue recovery option, 39-4
- CopyPasteIntersect parameter, 34-10
- Count(), 37-1
- cp
  - publish forecast, 14-53
- CPI (Consumer Price Index), 28-4
- CPO Engine, 47-1
- create\_object procedure, 55-14
- CREATE\_OPT\_STRUCT procedure, 35-6
- CREATE\_PE\_STRUCT procedure, 35-6
- create\_process\_temp\_table procedure, 55-14, 58-63, 58-63
- Create Member Step, 9-7, 39-9
- crosstab layout, 8-6
- Croston Model for Intermittent Demand, 59-12
- CTL file, 25-8, 30-5
- CTO worksheets
  - associating multiple price lists, 14-40
- CurrentRow function, 38-5
- custom methods, 9-3, 32-3
- Custom Step
  - and other external functions, 9-13
  - reference, 39-10
- CutTailZeros parameter, 57-4, 57-9, 58-9



## D

---

- DAILY model, 59-8
- DampPeriod parameter, 58-9
- DampStep parameter, 57-15, 58-9
- dashboard
  - customizing graphics, 40-21
- data
  - how calculated, 3-3
  - how stored, 3-1
  - importing or exporting automatically, 29-34
  - loading, 25-24
  - types of data allowed in series, 27-18
- database
  - cannot connect to, 66-13
  - creating table, 24-1
  - maintaining, 55-6
  - maintaining tables, 24-2
  - maintenance tasks, 35-2
  - recompiling functions and procedures, 24-3
  - using a different database without reinstalling, 66-8
- DatabaseEncoding parameter, 34-11
- database server, 1-5
- data-dependent time unit, 28-10
- data loading
  - data needs, 40-9
  - overview of tools, 40-10
- data model
  - choosing sources, 25-5
  - deleting, 25-24
  - importing or exporting, 25-24, 25-25
  - loading data into, 25-24
  - opening, 25-24
  - saving with new name, 25-24
  - selecting number and format of sources, 25-4
  - specifying format of source files, 25-6
- Data Model Wizard, 25-3
- data profile, 13-6, 29-4, 29-10
- data transformations, 57-11
- date/time format
  - for new dates, 43-11
  - parameters, 43-8
- date format
  - imported data, 25-7
- date functions
  - Day, 38-6
  - Month, 38-13
  - Today, 38-18
  - Year, 38-19
- date series, 27-18
- Day function, 38-6
- db\_exception\_log table, 35-5
- DBConnectionTimeout parameter, 34-11, 43-6
- DBDateFormat parameter, 34-11, 43-6, 43-8
- DBHintAggriSQL, 58-67
- DBHintBranchDivision, 58-67
- DBHintCreatePDE, 58-68
- DBHintLoadActual, 58-67
- DBIdleTimeOut parameter, 34-11, 43-6
- dbms\_connect parameter, 41-3
- dbms\_type parameter, 41-3
- DBName parameter, 34-12, 66-7, 66-18
- DBPassword parameter, 34-12, 66-6
- DBPort parameter, 34-12, 66-7
- DBType parameter, 34-12, 66-6
- DBUser parameter, 34-12, 66-6
- dead combination, 36-9, 50-19
- debugging engine run, 56-15
- DebugMode parameter, 34-14
- Debug parameter, 34-13
- Decode(), 37-2
- deduction
  - introduction, 42-2
- deductions
  - origin, 42-3
- def\_delta parameter, 11-4, 58-10
- default
  - attribute values for manually created member, 5-9, 26-11
  - combinations in worksheet, 29-17
  - delays in workflow processing, 9-14
  - Demantra graphics (Web products), 40-21, 44-10
  - general level indicators for series, 61-4
  - index or exchange rate, 28-2
  - links in Collaborator Workbench, 44-12
  - methods, 1-7, 9-2
  - pane configuration in Collaborator Workbench, 44-7
  - parent members for manually created member, 5-9, 26-15
  - right-click options, 9-2

- series in new worksheet, 27-6
- unit of measure, 13-3
- worksheet associated with a level, 40-20, 61-8
- DefaultContentSecurityAccess parameter, 34-14
- DefaultLevelSecurityAccess parameter, 34-14
- DELETE\_FORE\_COL procedure, 35-6
- DELETE\_INVALID\_GL\_POPULATION procedure, 35-6
- delete access, security for, 10-4
- DeleteIsSelfCondition parameter, 57-2, 58-11
- DeleteIsSelfRows parameter, 57-2, 58-11
- Delete Member Step, 9-8, 39-13
- delta, delta\_d, delta\_w data fields, 36-4
- delta value for item-location combination, 11-4, 43-13, 58-10
- demand
  - and causal factors, 2-10, 50-2
  - modeling as a shape, 50-4
  - profile, 50-4
  - useful series, 2-6
- Demand Planner Web
  - introduction to user interface, 8-5
- Demantra-Siebel integration extracts, 20-66
  - Baseline Volume Forecast, 20-66
  - Promotional Lift Coefficients, 20-68
  - Promotions, 20-71
- Demantra Spectrum
  - solution components, 1-2
- Demantra to EnterpriseOne Integration
  - Demand Management, 17-2
- Demantra to EnterpriseOne Integration Integration Points
  - Demand Management, 17-2
- description field, 26-4, 26-22
- DeskTopNotification parameter, 34-14
- destructor methods, 9-3, 32-3
- detect\_cp parameter, 57-10, 58-12
- detect\_outlier parameter, 57-9, 58-12
- DetectModelOutliers parameter, 57-14, 58-13
- DeviationFactor parameter, 57-12, 58-13
- dir.onlineHelp parameter, 34-14, 43-3
- direct customers, 42-14
- display style for population attributes
  - examples, 32-8
  - specifying, 32-6
- Distributed Engine
  - and sales\_data\_engine, 51-6
  - details of behavior, 57-20
  - introduction, 1-5, 49-2
- DMULT model, 59-8
- do\_aggri data field, 36-4
- do\_fore data field, 36-5
- DOL
  - URL, 60-6
  - using, 27-36
  - wrong link, 66-14
- DOL Services, 1-5
- double-click for notes, disabling, 43-22
- Double Exponential Smoothing Model, 59-13
- DownTrend parameter, 57-15, 58-14
- dp.exe, 44-7
- drop\_object procedure, 55-14
- dropdown security, 26-12
- ds.ini, 66-5
- DSM
  - configuring, 42-5
  - configuring matching, 42-11
  - configuring promotions, 42-8
  - configuring write-offs, 42-12
  - customer relationships, 42-14
  - data flow, 42-4
  - loading data, 42-13
  - overview, 42-1
  - parameters, 42-11
  - setting up database structures, 42-6
- DSMAllShipDateDifference parameter, 34-15, 35-33, 42-11
- DSMOICheckProduct parameter, 34-15
- DSMOIPercentDifference parameter, 34-15, 35-34, 42-12
- DSMOIShipDateDifference parameter, 34-15, 35-34, 42-12
- DSMP EOIAmountSeries parameter, 34-15, 42-11
- DSMPEShipDateSeries parameter, 34-15, 42-11
- DSMPromotionBudgetSeries parameter, 34-15, 42-11
- DSMWriteOffThreshold parameter, 34-16, 42-12
- DSR Integration
  - business process, 22-1
  - importing dsr data into Demantra, 22-3
  - objects for dsr integration, 22-2
  - overview of procedures, 22-2
  - running multiple forecasts, 22-3
  - solution overview, 22-1

- dying\_time data field, 36-5
- dying\_time parameter, 36-5, 36-9, 50-17, 58-14
- DYNAMIC\_SYNC procedure, 27-35, 34-39
- Dynamic Open Link
  - URL, 60-6
  - wrong link, 66-14

## E

---

- EBS integration with S&OP, 16-1
- edit-lock expression
  - and other features that control editability, 6-6
  - creating, 27-29
  - dependencies on other series, 27-38
  - introduction, 6-6
- Edit Member Step, 9-7, 39-13
- edit methods, 9-3, 32-3
- ELOG model, 57-11, 59-10
- email
  - creating link for webmaster, 44-13
  - Demantra administrator
    - reconfiguring, 43-9
  - in automated workflow, 9-7, 39-14, 39-21, 39-36
  - not received, possible cause, 66-24
  - specifying for users, 61-11
- Email Step
  - and other interactive steps, 9-7
  - reference, 39-14
- embedded worksheets
  - backend configuration, 61-8
  - introduction, 8-8
- empty rows
  - saving setting, 8-11
- empty time bucket, referencing, 6-9
- ENABLE\_DEBUG, 64-4
- ENABLE\_STATISTICS, 64-4
- EnableAdjustment parameter, 57-8, 57-15, 58-15
- EnableFitValidation parameter, 57-8, 57-12, 58-15
- EnableForecastValidation parameter, 57-8, 58-16
- EnableIncrementalLoading parameter, 34-16
- EnableModifiedVariance parameter, 58-16
- EnableOpenNoteWithDoubleClick parameter, 43-22
- EnableSimGLFilter parameter, 58-17
- EnableWorksheetCaching parameter, 34-16
- encryption.exe, 66-5

- Engine Administrator, 56-1
  - configuring settings, 56-4
  - running engine from, 56-8
  - starting, 56-3
- engine log
  - filtering, 56-6, 56-7
  - settings, 56-6
  - viewing, 56-15
- Engine Manager
  - log settings, 56-6
  - overview, 57-18
  - role, 57-18
  - starting, 56-9
  - stopping, 56-10
- EngineManager.exe, 44-7, 56-9
- engine output threshold
  - description, 43-15
- engine profile
  - creating or renaming, 55-2, 55-2
  - including in a component, 61-9
  - introduction, 2-11, 49-4, 55-1
  - specifying in DOS, 56-11
- ENVELOPE\_CHAIN\_LENGTH parameter, 57-13, 59-8, 59-11, 59-18, 59-24
- ENVELOPE\_RESET\_SEED parameter, 57-13, 59-8, 59-11, 59-18, 59-24
- envelope function, 57-13
- envelope test, 57-14
- EP\_LOAD\_ITEMS procedure, 35-10
- EP\_LOAD\_LOCATIONS Procedure, 35-13
- EP\_LOAD\_MAIN procedure, 25-2, 35-7
- EP\_LOAD\_MDP\_LEVEL Procedure, 35-7
- EP\_LOAD\_SALES Procedure, 35-16
- Error in process execution, 66-25
- error log
  - data loading, 25-6
  - for procedures, 24-3
  - integration, 29-9
- error on page or error loading tasks, 66-15
- error tables for integration, 29-28
- error while running Worksheet Window, 66-19
- estimation, 57-7
- event
  - default settings, 40-18
- event causal factor, 53-7
- Excel
  - data exported to as text, 66-14

- wrong link to Demantra Spectrum, 66-14
- exceptions
  - and other filters, 2-7
  - and promotion series, 34-39
  - flagging series for use in, 27-18
  - in data profile, 29-19
  - introduction, 2-8
  - using in a workflow, 39-16
  - using in workflow, 9-7
- Exception Step
  - and other interactive steps, 9-7
  - and other logic steps, 9-8
  - reference, 39-16
- exchange rate
  - and monetary unit, 28-6
  - configuring, 28-1
  - editing values, 28-4
  - including in a component, 61-7
  - in integration, 29-17
  - used in component, 28-4
- Executable Step
  - and other external functions, 9-13
  - reference, 39-19
- EXECUTE\_PROFILES procedure, 35-23, 35-24, 45-4, 57-2
- executeStep function, 39-12
- execution.shell parameter, 34-16, 43-21
- Exp function, 38-6
- expiration time, 43-3
- expired license, 66-17
- expired session, 66-24
- export
  - exporting a data model, 25-25
  - setting up, 29-2
  - within automated workflow, 39-34
- EXPOSE\_PROMOTIONS procedure, 35-23, 40-5
- extending demantra, 1-2
- Extra From
  - for a drop-down attribute, 26-14
  - for a level, 26-5
  - for a series drop-down, 27-15
  - for a server expression, 27-27
- extra SQL, 6-28, 6-28, 27-27
- Extra Where
  - for a drop-down attribute, 26-14
  - for a level, 26-5
  - for a series dropdown, 27-15

- for a server expression, 27-27

## F

---

- Fail-To-Execute
  - in Container Step, 39-8
  - in Exception Step, 39-18
- FCROST model, 59-12
- FillMethod parameter, 57-9, 58-20
- FillParameter parameter, 57-9, 58-20
- filter
  - and Open With menu, 40-20, 61-8
  - and other performance tips, 8-14
  - appearance to user, 2-7
  - at user level, 10-1, 10-3, 61-13
  - business days versus non-business days, 58-62
  - cascade versus non-cascade mode, per user, 61-17
  - controlling order of levels in, 26-7
  - creating filtered levels, 26-4
  - data displayed in, 26-8
  - for promotional causal factors
    - introduction, 54-5
    - specifying, 54-10
    - use during engine flow, 57-2
  - in a worksheet, 2-7
  - in data profile for import or export, 29-18
  - passing to method, 9-15, 32-9
  - using autogenerated ID as code field, 25-12
  - using in definition of drop-down list for attribute, 26-14
  - using in definition of drop-down list for series, 27-15
  - using in definition of series, 6-28, 27-27
- FIRSTDAYINWEEK parameter, 34-17, 43-2
- fit
  - and residuals, 57-11
  - calculation, 57-7
  - validation, 57-8, 57-16
- forecast
  - calculation, 57-13
  - failure procedure, 57-15
  - models, 59-2
  - split forecast by series, 51-5
  - validation of, 57-13
- ForecastFirstRow function, 38-7
- ForecastGenerationHorizon parameter, 55-7, 58-

ForecastMeanRelativeDistance parameter, 57-14, 58-21

#### forecast models

- ARIX, 59-3
- ARLOGISTIC, 59-4
- ARX, 59-4
- BWINT, 59-5
- CMREGR, 59-6
- DMULT, 59-8
- ELOG, 59-10
- FCROST, 59-12
- HOLT, 59-13
- ICMREGR, 59-15
- IREGR, 59-16
- LOGISTIC, 59-18
- LOGREGR, 59-17
- MRIDGE, 59-20
- NAIVE, 59-21
- REGR, 59-23

#### forecast tree

- and advanced analytics, 55-16
- and the engine, 2-11
- configuring, 4-3, 52-1
- example, 50-11
- guidelines, 51-6
- maximum forecast level, 52-10, 58-29
- minimum forecast level, 52-10
- promotion levels, 52-4
- splitting by Engine Manager, 56-6, 57-18
- structure, 51-3
- tuning engine by node, 55-16
- used by Analytical Engine, 2-10, 50-8

#### forecast tree configuration

- pooled time series, 52-5

#### forecast versions, 43-2, 50-21

format.dates.longFormat parameter, 34-17, 43-8

format.dates.shortFormat parameter, 34-17, 43-8

Fpos function, 38-7

Fsum function, 38-8

## G

Gamma parameter, 59-6, 59-14

GammaQ parameter, 59-12

GammaT parameter, 59-12

Gantt chart

introduction, 8-5

general.copyright parameter, 34-17

general.homepage.title parameter, 34-17, 43-17

general.title.text parameter, 34-18, 43-17

general.userList.tasks parameter, 34-18

general.userList.whoisonline parameter, 34-18, 43-3, 43-4

general.userList.worksheets parameter, 34-18

#### general level

- adding a population attribute, 26-18

- and series indicators, 10-2

- configuring Activity Browser, 40-3

- exporting, 29-19

- flagging as analytical, 26-8

- importing, 29-23

- introduction, 5-6

- specifying presentation style for population, 32-8

- specifying series indicators, 6-29, 61-4

#### General Levels

- Proport, 11-11

GetRow function, 38-9

glob\_prop, 11-3

glob\_prop data field, 36-5

#### global factor

- and other causal factors, 50-2

- creating, 53-3

#### gl synchronization

- overview, 46-1

#### graph

- controlling available series, 27-8

- hiding or displaying, 8-5

- introduction, 8-5

- parameters for controlling, 43-4

Graph.MaxLabelWidth parameter, 34-18, 43-4

graphics, customizing, 40-21, 44-11

GrossRemove parameter, 57-10, 58-22

#### group

- access to Workflow Editor/Manager, 61-19

- adding users to, 61-14

- creating, 61-16

- deleting, 61-19

- introduction, 10-1

- overview, 10-1

- selecting users for, 61-18

- users in, 61-18

group\_tables table, 52-8

group expressions, 27-33

Group Step

and other interactive steps, 9-7

reference, 39-21

GUI Services, 1-5

## H

---

hierarchy

adding level, 25-14

controlling order of levels in display, 26-7

data model, 5-5

forecast, configuring, 52-1

introduction, 5-5

HighestSquaring parameter, 58-22

hint message

level, 26-8

series, 6-8, 27-7, 38-11

hist\_glob\_prop, 11-3

hist\_glob\_prop data field, 36-5

hist\_glob\_prop parameter, 36-5, 58-22

historical data, 2-3

HistoryLength parameter, 58-23

HOLT model, 57-15, 59-13

https, 1-6

hung session

ending, 61-28

## I

---

ICMREGR model, 59-15

If function, 38-12

IGL, 50-13

illegal characters, 23-3, 49-10, 60-10

images, customizing, 40-21, 44-11

import

aggregated data, 13-17

data other than series or levels, 13-16

importing a data model, 25-25

series data, at any level, 29-5

setting up, 29-2

within automated workflow, 39-34

ImportBlockSize parameter, 34-18, 43-11

incremental loading, enabling, 43-22

index (financial)

and monetary unit, 28-6

and worksheet, 7-1

configuring, 28-1

editing values, 28-4

including in a component, 61-7

in integration, 29-17

introduction, 2-2

restarting server after changing, 23-9

used in component, 28-4

index (seasonal), 59-3, 59-4

index (table)

rebuilding, 35-34

where stored, 43-7, 55-11

index adviser, 24-6

indexspace parameter, 34-19, 43-7, 55-11

indicators for notes, promotions, or general levels, 61-3

indirect customers, 42-14

InfluenceGroupLevel parameter, 58-23

influence groups

absolute and relative addressing, 57-3

aggregating attributes within, 54-8

defining, 52-4

introduction, 50-13

level in forecast tree, 51-4

shape modeling for, 54-12

InfluenceRangeLevel parameter, 58-23

influence ranges

defining, 52-4

introduction, 50-13

level in forecast tree, 51-4

initial\_param parameter, 34-19, 43-7, 55-10

input data

allowed formats, 13-5

introduction, 2-3

joining, 25-8

Inputs table

adding data via the Business Modeler, 53-3

used for global causals, 50-4, 53-1

used for time aggregations, 5-7, 26-4, 26-20

used for time units, 28-10

INSERT\_UNITS procedure, 34-37, 35-23, 43-12, 45-4, 55-8

InsertDateTimeFormat parameter, 34-19, 43-8, 43-11

Insertmissingvalues parameter, 34-19, 43-11

Integrate User option, 61-11

integration

checking progress of, 29-35

data needs, 40-9

- introduction, 1-6
- overview of options, 13-6
- overview of tools, 40-10
- reference for interfaces, 40-17
- Integration
  - EBS to S&OP, 16-1
  - EnterpriseOne to DM, 17-1
  - EnterpriseOne to DSM, 19-1
  - EnterpriseOne to PTP, 18-1
- Integration Business Process
  - Demand Management to EnterpriseOne, 17-3
- integration interface
  - defining, 29-2
  - deleting, 29-27
  - executing from DOS, 13-7, 29-34
  - executing from workflow, 9-7, 39-34
  - introduction, 13-6
  - restarting server after modifying, 23-9
- Integration Tool (standalone), 13-7, 29-34
- INTERM\_RESULTS table, 56-17, 58-65
- IntermitCriterion parameter, 57-9, 57-16, 58-24
- intermittency detection and processing, 57-9
- Intermittent CMREGR Model, 59-15
- Intermittent flow
  - details, 57-16
  - introduction, 57-7
- Intermittent Regression Model, 59-16
- Internal error
  - check database and network connections, 66-25
- IntUpdate parameter, 57-16, 58-24
- IREGR model, 59-16
- IRL, 50-13, 51-4
- is\_fictive data field, 36-6
- Is\_Modified function, 38-12
- is\_self data field, 36-12, 57-2
- item\_id data field, 36-1, 36-3, 36-10
- item\_node field, 36-10, 56-16
- item-location combination
  - displaying data for, 8-5
  - introduction, 2-5
- items table, 26-4
- item-type levels
  - introduction, 5-6

## J

---

- J2EE application server, 1-4
- Java
  - including in workflow, 39-12
  - using as method, 32-1
- Java applet
  - version, 66-20
- Java out of memory, 66-17
- join, defining, 25-8
- JREMaxMemory parameter, 43-22
- Jump test, 57-14

## K

---

- KillEngine.bat, 56-17

## L

---

- Lag, 59-7
- languages, 43-26
- last\_date\_backup parameter, 58-25
- last\_date parameter, 58-25
- layout, Collaborator Workbench
  - configuring for all users, 44-7
- layout, worksheet
  - and summary rows, 38-11
  - examples, 8-6
  - saving changes, 8-9
  - tips, 8-14
- leading zeros, 57-9
- lead parameter, 57-8, 57-13, 58-25
- legend
  - of content pane, maximum number of entries, 43-4
  - showing by default, 34-31
  - showing by default in desktop, 34-31
- Legend.MaxLegendItemWidth parameter, 34-19, 43-4
- level
  - adding attributes, 26-9
  - adding right-click option, 5-15, 9-1
  - associating with worksheets (for Open With), 40-19, 61-7
  - attributes, 5-9
  - caching members, 64-2
  - configuring, 26-6
  - default member, 5-9
  - default member, specifying, 26-17
  - deleting, 26-23

- disabling, 26-22
- exporting members and attributes, 29-20
- filtering, 26-4
- importing or exporting, 29-24
- including in component, 61-5
- introduction, 5-4
- joining to another table, 26-4
- predefined default member, renaming or hiding, 26-18
- predefined default members, 5-9
- restarting server after changing, 23-9
- specifying security at component level, 61-5
- specifying units of measure, 28-8
- table where stored, 52-8
- where and when created, 5-16
- level\_id field, 36-10, 56-16
- level profile, 13-6
- level renovation coefficient, 59-5
- level series, 6-4
- license
  - expired, 66-17
- License.ExpirationMessage parameter, 34-19
- link between steps
  - creating, 31-3
  - deleting, 31-4
- LinkedDataAutoCopy parameter, 34-20
- links in Collaborator Workbench
  - adding to menus, 44-4
  - configuring, 43-3, 44-12
- live combination, 36-9, 50-19
- Load\_data.bat, 25-24
- LoadDataStop parameter, 34-20, 43-11
- Loading Veh \$ series, 40-12
- lob procedures, 14-57
- loc\_node field, 36-10, 56-16
- local causal factor, 50-3
- location\_id data field, 36-1, 36-4, 36-11
- location data, input file, A-5
- location table, 26-4
- location-type level
  - introduction, 5-6
- LockTimeout parameter, 34-20, 43-6
- log.history parameter, 34-20, 43-12, 43-21
- Logarithmic CMREGR model, 59-10
- logconf.lcf, 64-2
- LogCorrection parameter, 57-11, 58-26, 59-11, 59-18

- log file
  - for loading data, 25-6
  - integration, 29-9
- log for procedure errors, 24-3
- logging off
  - if session hangs, 61-28
- logging on, 60-5, 66-1
- login page, configuring, 44-12
- LOGISTIC model, 59-18
- LogLevel parameter, 58-27
- LOG model, 57-11
- LOGREGR model, 59-17
- lookup level
  - creating as attribute, 26-2, 26-7
- Lower(), 37-2
- LowerUpliftBound parameter, 55-5, 58-28
- lowest promotional level (LPL)
  - setting, 52-4
- lowest promotion level (LPL), 51-4
- LRU caching, 64-4
- Ltrim(), 37-2

## M

---

- mail\_recipient parameter, 34-23
- mail.outgoing.server parameter, 34-20
- mail.server parameter, 34-21, 43-9
- mail.strings.from.system parameter, 34-21, 43-9, 43-10
- mail.strings.internalerror.message parameter, 34-21, 43-10, 66-25
- mail.strings.internalerror.subject parameter, 34-21, 43-10, 66-25
- mail.strings.processfailuresubject parameter, 34-21, 43-10, 66-26
- mail.strings.processterminated parameter, 34-21, 43-10
- mail.strings.recovery parameter, 34-22, 43-10, 66-21
- mail.strings.taskfailuresubject parameter, 34-22, 43-10, 66-26
- mail.strings.taskstimatedoutsubject parameter, 34-22, 43-10, 66-23
- mail.strings.timeout.group parameter, 34-22, 43-10, 66-23
- mail.strings.timeout.user parameter, 34-22, 43-11, 66-23



- mailAddress parameter, 34-23, 43-9
- mail parameter, 34-20, 43-9
- mail-password parameter, 34-22
- mailProtocol parameter, 34-23
- mail-username parameter, 34-23
- MA model, 57-16, 59-19
- ManualRefreshAsDefault parameter, 34-23
- MANUALS\_INS\_INTEGRATION, 13-9
- MANUALS\_INS\_INTEGRATION procedure, 13-9, 29-35, 34-37, 35-25, 43-12
- MANUALS\_INS\_RECOVERY procedure, 35-25
- MANUALS\_INS procedure, 35-25
- Markov Chain Monte-Carlo model, 59-6
- MatrixCombs parameter, 34-24
- matrix series
  - created for causal factor, 53-10
  - introduction, 6-3
- mature\_age parameter, 36-9, 50-17, 58-28
- max\_accept\_num parameter, 58-29
- max\_fore\_level parameter, 52-10, 58-29
- max\_fore\_sales\_date parameter, 34-24
- max\_initial\_members parameter, 34-24
- MAX\_ITERATIONS parameter, 59-9
- max\_records\_for\_commit parameter, 34-24, 43-7
- max\_sales\_date parameter, 34-24
- Max(), 37-2
- MaxAvailableFilterMembers, 43-24
- MaxBackgroundLoad parameter, 34-24
- MaxCombsLoadChunkSize parameter, 43-22
- MaxDBConnections parameter, 34-24, 43-6
- MaxEngMemory parameter, 56-13, 58-30
- Max Len field, 55-15
- MaxSaleVal parameter, 34-25
- MaxUpdateThreads parameter, 34-25, 43-18
- MDP\_ADD procedure, 34-29, 35-25, 43-11, 56-12
- mdp\_matrix table
  - and combination levels, 26-4
  - and combination series, 27-3
  - and engine models, 55-16
  - fields in, 36-3
  - key settings used by engine, 50-16
- Mean\_check parameter, 57-12
- MeanRelativeDistance parameter, 57-12, 58-30
- Mean test, 57-14
- member
  - adding right-click option, 5-15, 9-1
  - introduction, 5-9
- Member Management
  - and delta\_def, 58-10
  - launching from within Collaborator, 44-7
  - number of members shown at once, 34-24
- Members Browser
  - comparison to selection lists, 8-5
- member security, 61-14
- menu
  - Collaborator Workbench
    - configuring, 44-2
    - configuring menu permissions, 61-25
  - right-click
    - configuring, 2-8, 5-15, 9-1
    - configuring Open with menu, 61-8
- menu, right-click
  - configuring Open with menu, 40-20
  - removing options, 40-21
- method
  - action when complete, 32-5
  - creating, 32-1
  - default methods, 9-2
  - deleting, 32-12
  - input dialog box, 9-4, 9-5
  - output dialog box, 9-5
  - overview, 2-8, 5-15, 9-1
  - passing arguments to, 9-14, 9-17, 31-4, 32-9
  - restarting server after changing, 23-9
  - thread pool, 43-19
  - when to define, 4-8
- min\_fore\_level parameter, 52-10, 58-30
- min\_fore\_level parameter, 55-6
- min\_fore\_sales\_date parameter, 34-25
- min\_sales\_date parameter, 34-25
- Min(), 37-2
- MinDBConnections parameter, 34-25, 43-6
- Min Len field, 55-15
- MinLengthForDetect parameter, 57-9, 58-31
- missing\_all\_sources data field, 36-6
- missing\_some\_sources data field, 36-6
- missing\_values, 58-39
- missing values, 57-9
- mix\_max\_members parameter, 34-25
- MODEL\_PATH parameter, 41-3
- models
  - and causal factors, 53-2
  - and engine flow, 2-10, 50-8
  - enabling globally, 55-14

- general introduction, 49-3
- long, 53-2
- non seasonal, 53-2
- reference information, 59-2
- seeing how used for nodes, 56-12
- short, 53-2
- specifying for specific combinations, 55-16
- Mod function, 38-13
- modulus, 38-13
- Month function, 38-13
- Moving Average model, 57-16, 59-19
- MRIDGE model, 59-20
- Multiple Logarithmic Regression Model, 59-17
- Multiple Regression Model, 59-23
- Multiplicative Multi-Seasonal Regression model, 59-8
- Multiplicative Regression-Winters model, 59-5
- My Tasks
  - and workflow, 39-2
  - configuring user access, 44-10
  - default availability and location, 44-7
  - errors on, 66-15
  - introduction, 1-7
  - origin of tasks in, 9-6
  - refreshing, 66-26
- My Worksheets
  - configuring user access, 44-10
  - default availability and location, 44-7

## N

---

- NaiveEnable parameter, 57-8, 57-15, 58-31, 59-20, 59-22
- NAIVE model, 57-15, 59-21
  - when used, 52-10, 58-29
- navBarController.addNewContentLink.Text parameter, 34-25, 43-3
- navBarController.interpriseContent.text parameter, 34-25
- need\_lag parameter, 59-8, 59-10, 59-15
- need\_spread parameter, 57-16, 58-32
- negative coefficients, preventing, 55-4
- negative values within client expression, 27-28
- negative values within server expression, 27-26
- new\_member data field, 36-7
- next\_param parameter, 34-25, 43-7, 55-10
- nodal tuning, 55-16

- node\_forecast\_details parameter, 55-7, 56-17, 58-32
- NODE\_FORECAST table, 56-17
- non-cascade mode for filter, enabling, 61-18
- NonNegRegrMaxTolMult parameter, 58-33
- NormalizationFactor parameter, 58-33
- NormalizeResults parameter, 36-12, 55-5, 58-34
- normalizing historical engine results, 55-5
- No suitable driver, 66-19
- note
  - access at group level, 61-16
  - configuring indicators in worksheet table, 61-3
  - controlling double-click access, 43-22
- notifications, missed, 34-4
- NpiUseWsFilter parameter, 34-26
- numeric functions
  - Abs, 38-3
  - Exp, 38-6
  - Mod, 38-13
  - Pi, 38-14
  - Rand, 38-14
  - Round, 38-15
  - Sqrt, 38-16
  - Truncate, 38-18
- NumShapes parameter, 53-13, 55-4, 58-35
- Nvl(), 37-2

## O

---

- object error message, 66-19
- OBS\_ERROR\_STD data field, 36-2
- off-invoice settlement
  - introduction, 42-2
  - origin, 42-3
- offline
  - cannot access offline worksheets, 66-18
- OpenWithContext parameter, 34-26, 43-23
- Open With menu
  - associating levels and worksheets, 40-19, 61-7
  - customizing, 43-23
  - errors, 66-15
- opti\_level\_item parameter, 41-3
- opti\_level\_location parameter, 41-4
- OptimizedAlphaIter parameter, 59-6, 59-14
- OptimizedBwint parameter, 59-6
- OptimizedGammaIter parameter, 59-6, 59-14
- OptimizedHolt parameter, 59-14

## Oracle

- monitoring sessions, 24-6
- terminating sessions, 24-6
- XDB user and port collision, 66-21
- oracle\_optimization\_mode parameter, 43-8, 58-35
- other promotional causal factor, 57-4
- outlier
  - detecting during engine preprocessing, 57-9
  - detecting during fit validation, 57-12
  - gross, 57-10
  - points needed to detect, 58-31
- outlier data field, 36-3, 36-8
- Outlier parameter, 57-12
- OutliersPercent parameter, 57-10, 58-35
- OutlierStdError parameter, 57-10, 58-35
- own promotional causal factor, 57-4

## P

---

- P1, P2, ..., P12 data fields, 36-8
- P1 value for item-location combination, 11-4, 11-5, 11-6
- page cannot be displayed, 66-20
- parameters, 58-xxxii
  - and engine profiles, 55-1
  - APS queue, 43-2
  - Collaborator Workbench, 43-2
  - configuring, 23-7
  - database, 43-5
  - date/time formats, 43-8
  - email, 43-9
  - for database procedures, 39-33
  - for default promotion start and span, 40-18
  - for Demantra Spectrum executables, 44-7
  - for engine, 55-1
  - for optimization step, 41-2
  - for programs run from Collaborator Workbench, 44-5
  - graphs, 43-4
  - in Integration Queue, 29-35
  - in methods, 9-17, 9-17, 31-5, 32-11
  - integration and import, 43-11
  - logs and old data, 43-12
  - passing to Java, 39-12
  - Promotions Effectiveness, 40-3
  - Settlement Management, 42-11
  - solution basics, 43-2
  - solution branding, 43-17
  - specifying for Java class, 39-11
  - threading, 43-17
  - workflow, 43-21
  - worksheets, 43-21
- parent, 5-5
- partial chaining indicators, 36-6
- PartitionColumnItem parameter, 55-10, 58-36
- PartitionColumnLoc parameter, 55-10, 58-36
- parts\_factor\_source parameter, 34-27
- password
  - permission needed for changing, 10-9
  - policy, 10-10
- Paste Member Step, 39-24
- PATH environment variable, 41-2
- Pause option, 39-4
- Pay Type series, 40-12
- pct\_increase\_for\_analyze parameter, 34-27, 43-8
- PD1, PD2, ... PD7 data fields, 36-3, 36-8
- PercentOfZeros parameter, 57-15, 58-36
- performance
  - suggestions on fixing, 66-27
  - tips for worksheet design, 8-14
  - tuning high-volume systems, 55-7
- permission level
  - customizing, 10-5
  - introduction, 10-1
  - specifying for user, 61-11, 61-11
  - specifying menu items for, 61-26
- phase\_in\_out parameter, 34-27
- Phi parameter, 59-6, 59-13, 59-14
- PhiQ parameter, 59-13
- Pi function, 38-14
- pivot table, 8-6
- Planning Applications menu
  - customizing, 44-6
- Platform Server, 1-4
- Please handle recovery, 66-21
- POP\_ALL\_MATCH\_PROPOSAL procedure, 35-33, 42-12
- POP\_OI\_MATCH\_PROPOSAL procedure, 35-34, 42-12
- population attribute
  - adding to level, 26-18
  - loading, 29-25
  - passing to method, 9-16, 32-10
  - presentation style in methods, 32-8

- specifying presentation style, 32-6
- table where stored, 29-27
- using to filter condition step, 39-7
- using to filter exception step, 39-18
- portalVersion parameter, 34-27
- ports, changing, 66-8
- Possible\_Season parameter, 59-3
- post\_effect data field, 36-8
- Potential parameter, 59-19
- power user, 10-5
- pre\_effect data field, 36-8
- PRE\_LOGON procedure, 35-34
- predefined default member
  - introduction, 5-10
  - renaming or hiding, 26-18
- prediction\_status data field, 36-9
- prediction status
  - field where stored, 36-9
  - for fictive combinations, 50-19
  - for real combinations, 50-19
  - how used to clear old forecast data, 55-8
  - purpose, 50-18
  - reference, 36-9
  - setting indirectly with do\_fore, 50-18
  - used by constraint profit optimization, 47-4
  - when computed, 11-10
- prepare\_data procedure, 25-24
- preprocessing, 57-8
- presentation style for population attributes, 32-8
  - specifying, 32-6
- price, 53-6
- procedures, 35-2
  - error log, 24-3
  - recompiling, 24-3
  - running from within a workflow, 9-8
  - to load data into data model, 25-24
  - when to wrap, 4-9
  - wrapping, 24-5
- process log, 62-9
- profile
  - for constraint optimization
    - creating, 47-2
    - deleting, 47-8
    - introduction, 47-1
    - running, 47-8
  - for demand, 50-4
  - for engine parameters, 4-3
  - adding to component, 61-8
  - creating or renaming, 55-2
  - introduction, 49-4
  - specifying when running engine, 56-8, 56-11
  - for integration
    - creating, 29-2
    - executing from workflow, 9-7
    - introduction, 13-6
  - for loading supplementary data, 13-16
  - for rolling data
    - executing, 35-23
    - introduction, 45-1
- program group
  - defining, 61-21, 61-24
  - deleting, 61-25
- PROMO\_AGGR\_LEVEL parameter, 55-16, 58-38
- PromoDefaultSpan, 43-23
- PromoDefaultSpan parameter, 34-27, 40-3, 40-18
- PromoDefaultStart, 43-24
- PromoDefaultStart parameter, 34-27, 40-3, 40-18
- PromoStartDate parameter, 55-4, 57-4
- promotion
  - color coding, 40-3
  - configuring indicators in worksheet table, 61-3
  - configuring the Activity Browser, 40-3
  - copy/paste operations, 6-23
  - default date settings, 40-3
  - default settings, 40-18
  - specifying lowest level, 52-4
- promotion\_data\_engine table, 57-2
- promotion\_data table
  - and promotion series, 27-3
  - fields in, 36-10
- promotion\_id field, 36-11
- promotion\_status\_lookup table, 40-5
- promotional causal factors
  - adjusting promotion dates, 54-13
  - aggregating to the LPL, 54-5
  - aggregating within the IGL, 54-8
  - and engine flow, 57-2
  - and other causal factors, 50-3
  - configuring, 54-8
  - filtering, 54-5
  - key options, 54-4
  - merging, 54-7
  - self, own, and other, 57-4

- transposing, 54-6
- promotion level, 5-16
- Promotion Optimization
  - configuring, 41-2
  - methods, 4-6
  - overview of setup, 41-1
  - setting up without the installer, 41-2
- Promotions Effectiveness
  - adding population attributes to general level, 26-18
  - configuring Activity Browser, 40-3
  - introduction to user interface, 8-5
  - level series, 6-3, 27-10
  - parameters, 40-3
  - setting lowest promotional level (LPL), 52-4
- promotion series
  - introduction, 6-3
  - specifying how to preserve during copy/paste, 6-23, 27-20, 27-20
  - where stored (if stored), 27-3
- promotion shape modeling, 54-14, 54-15
- PromotionStartDate parameter, 58-38
- promotion table, 40-5
- proof of performance
  - introduction, 42-3
- prop\_changes data field, 36-10
- proport
  - and integration, 29-7
  - below min\_fore\_level, 52-10, 58-30
  - calling from mdp\_add procedure, 43-11
  - during import, 29-7
  - overview, 3-5, 11-1
  - parallelization, 43-14
  - specifying reference series, 27-19
  - summary of parameters, 43-13
  - used by engine, 2-10, 50-8
  - when using engine profiles, 11-11
- Proport
  - General Levels, 11-11
- proport\_missing, 11-3
- proport\_missing parameter, 11-6, 58-39
- proport\_spread parameter, 11-7, 58-40
- proport\_threshold parameter, 11-7, 58-42
- proport\_parallelization, 43-14
- ProportParallelJobs parameter, 43-15, 58-42
- ProportRunsInCycle parameter, 43-15, 58-42
- ProportTableLabel parameter, 43-15, 58-43

- PTP Integration
  - promotion method configuration, 18-7
- PW1, PW2, ..., PW6 data fields, 36-3, 36-10

## Q

---

- QAD series, 53-11
- Quantile parameter, 57-12, 57-14, 58-43
- quantity\_form, 11-3
- quantity\_form parameter, 58-43
- quantity alignment duration series, 53-11
- query, 2-1, 8-1
- Query.MaxCombinations parameter, 34-28, 43-5
- Query.MaxSeries parameter, 34-28, 43-5
- Query.TopBottom.MaxCombinations parameter, 43-5
- QueryMechanismTimeout parameter, 34-28, 43-2
- QueryRunMaximumThreadUser parameter, 34-28
- quote marks, in Demantra Spectrum, 49-10, 60-10

## R

---

- Rand function, 38-14
- random numbers, 38-14
- Rapid Planning integration with S&OP, 16-19
- REBUILD\_INDEXES procedure, 35-3, 35-34
- Rebuild\_Sales\_Table parameter, 34-28, 35-35, 43-6
- REBUILD\_SCHEMA procedure, 35-34
- REBUILD\_TABLES procedure, 34-28, 35-3, 35-35, 43-6, 66-27
- ReceiveCollaborationMessages parameter, 34-29
- recompiling database, 24-3
- recovering from engine failure, 56-17
- recovery, 62-9
- recovery options, workflow, 39-4
- refitting model, 57-12
- refreshing, 66-26
- Refresh Population Step, 39-25
  - and other Demantra processing steps, 9-8
- RefreshTimer parameter, 34-29
- regime\_change field, 36-3
- regime change
  - detecting, 57-10
  - flags for detecting, 57-10
  - points needed to detect, 58-31

- sensitivity of detection, 58-44, 58-44
- RegimeThreshold parameter, 57-10, 58-44
- registering engine, 56-2
- REGR model, 59-23
- reloading data automatically, 43-22
- RemoveResidOutlier parameter, 57-12
- REPLACE\_APOSTROPHE\_LEVELS procedure, 35-35
- reset\_seed parameter, 59-7, 59-11, 59-16
- ResetForeVals parameter, 55-8, 58-44
- resetmat parameter, 58-45
- residuals calculation, 57-7
- RestartOnFailureInterval parameter, 34-29
- retailer
  - changing default profile, 40-18
  - list price, 40-9
  - shipments, 40-10
- Retry recovery option, 39-4
- right-click menu
  - configuring, 2-8, 5-15, 9-1
  - configuring Open With menu, 40-20, 61-8
- rolling\_profiles table, 35-23
- rolling average demand, 11-3
- rolling data
  - and forecast versions, 50-21
  - configuring series for use in, 27-5
  - creating profiles, 45-2
  - executing profiles, 35-23, 35-24, 45-3, 57-2
  - overview, 45-1
  - rolling data profile groups, 45-4
  - using to compare forecast versions, 50-22
- Round(), 37-2
- Round function, 38-15
- RowCount function, 38-16
- row functions
  - CurrentRow, 38-5
  - ForecastFirstRow, 38-7
  - GetRow, 38-9
  - RowCount, 38-16
- Rtrim(), 37-3
- Run\_full\_matrix\_proport parameter, 11-9, 34-29, 36-7, 36-10
- Run Engine window, 56-8
- RunInsertUnits parameter, 35-23, 45-4, 55-8, 57-1, 58-46
- RUNMODE parameter, 58-47
  - and checking engine version, 55-16

- and troubleshooting, 56-2
- RunProportInMdp\_add parameter, 34-29, 35-25, 43-11
- Run TextPro, 56-6

## S

---

- S&OP-Hyperion Planning integration, 21-1
- Safe\_Division(), 37-3
- safety stock
  - inventory level, based on service level, 38-20
  - procedure for computing, 35-6
- Sale Price series, 40-11
- sales\_data\_engine\_index\_space parameter, 34-30, 43-7, 55-11
- sales\_data\_engine\_space parameter, 34-30, 43-7, 55-11
- sales\_data\_engine table/view
  - and engine server, 57-19
  - reconfiguring for performance, 55-11
  - tablespace for, 43-6
  - where created, 51-6
- sales\_data table
  - adjusting for direct use by engine, 55-9
  - and branch\_data synonym, 27-18
  - and local causals, 50-4
  - and price causal factor, 50-3
  - and sales series, 27-3
  - fields in, 36-1
  - inserting records, 29-7
  - rebuilding, 43-6
- sales data, 2-3
- sales data, input file, A-1, A-13
- sales series
  - introduction, 6-3
  - where stored (if stored), 27-3
- Scheduler
  - procedures, 35-2
  - recommended scheduling, 35-2
- scheduling
  - database procedures
    - recommendations, 35-2
  - workflow instances, 62-3
- schema (database)
  - analyzing, 35-3
  - rebuilding, 35-34
- schema (workflow)

- creating, 31-1
  - deleting, 31-12
- schema group (workflow)
  - creating, 62-7
  - deleting, 62-8
- SCI Data
  - importing from EBS to S&OP, 16-22
- sdata5, sdata6, and so on, 40-26
- SdeAnalyzeSwitch parameter, 55-13, 58-48
- SdeCreateJoin parameter, 55-13, 58-48
- SdeCreateSwitch parameter, 55-12, 58-49
- season parameter, 58-49
- security
  - dropdown, 26-12
- security changes, missed, 34-4
- SecurityFromSystem parameter, 34-30
- security model, 10-1
- Selection Step
  - and other interactive steps, 9-7
  - and other logic steps, 9-8
  - reference, 39-27
- selection task
  - adding to workflow, 39-27
- self promotional causal factor, 57-4
- series
  - available to user, 61-12
  - color expression, 6-18, 27-29
  - configuration as drop-down, 27-12
  - configuring data in, 27-16
  - configuring server expression, 27-25
  - configuring table display, 27-8
  - creating, 27-2
  - data in, 3-3
  - dependencies among, 27-38
  - edit-lock expression, 6-6, 27-28
  - exporting data at any level, 29-10
  - including in a component, 61-3
  - indicators for notes or promotions, 61-3
  - making available for specific user, 61-12
  - restarting server after changing, 23-9
  - specifying general information, 27-5
  - where and when created, 6-30
- server, cannot connect to, 66-13
- server.generalurl parameter, 34-30, 43-21, 66-7
- Server.SessionExpiration parameter, 34-30, 43-3
- server expression
  - allowed operators, 37-4
  - and unit token, 7-3
  - as source for rolling data, 45-3
  - configuring, 27-25
  - Demantra tokens in, 37-5
  - filtering source data, 6-28, 27-27
  - functions used in, 37-1
  - introduction, 3-3
  - introduction to editor, 27-21
  - literal negative values, 27-26
  - referring to forecast version, 6-7
  - referring to other tables, 6-28, 27-27
  - syntax, 6-6, 27-23
  - used with client expression, 3-5, 6-11
  - verifying all, 27-28
  - when evaluated, 3-8
- ServerName parameter, 34-30, 66-7, 66-23
- Service Parts Forecasting, 15-1
- service parts forecasting levels, 5-9
- Service Parts Planning integration with DM, 15-1
- session
  - expiration time, 43-3
  - limit per user, 66-25
  - lock timeout period, 43-6
  - rolling forecasts, 45-3
- session, Oracle
  - monitoring, 24-6
  - terminating, 24-6
- set\_rb parameter, 43-8, 58-49
- SET2\_COEFF\_INI parameter, 59-9
- setParameters function, 39-12
- settings.xml, 56-5
- settlement level, 5-8, 5-16, 42-18
- SettlementLocationExtension parameter, 34-30, 42-7
- Settlement Management
  - configuring, 42-5
  - configuring matching, 42-11
  - configuring promotions, 42-8
  - configuring write-offs, 42-12
  - customer relationships, 42-14
  - data flow, 42-4
  - loading data, 42-13
  - overview, 42-1
  - parameters, 42-11
  - setting up database structures, 42-6
- SettlementProductExtension parameter, 34-31, 42-7

- settlements
  - default settings, 42-11
  - introduction, 42-1
  - trade versus non-trade, 42-2
- setup
  - configure to order, 51-8
- shape modeling
  - activity shape modeling
    - enabling, 53-12
    - introduction, 50-4
    - samples, 53-12
    - shape alignment, 53-10
  - comparison of engine variants, 50-6, 53-10
  - data requirements, 51-7
  - general introduction, 50-4, 53-10
  - impact on count of causal factors, 51-7
  - promotion shape modeling
    - enabling, 54-15
    - introduction, 50-8, 54-14
  - related parameters, 55-4
- ShapeSign parameter, 55-4, 58-50
- ShiftBaseCausals parameter, 58-50
- ShiftDynPromoDate parameter, 54-13, 55-5, 58-51
- ShiftPromoCausals parameter, 54-13, 55-5, 57-4, 58-52
- show\_legend parameter, 34-31
- Siebel integration, 20-1
  - main batch file configuration, 20-28
  - running, 20-31
- Siebel integration batch files, 20-34
- Siebel integration extracts, 20-34
  - Account Hierarchy Planning, 20-34
  - Plan Master List, 20-38
  - Pricing and Cost, 20-40
  - Product Planning Hierarchy, 20-44
  - Promotional Tactics, 20-63
  - Promotions, 20-49
  - Promotion Status, 20-48
  - Sales Activity, 20-65
- Siebel integration setup, 20-9
- SimMaxSize parameter, 34-31
- SIMPLE caching, 64-4
- simple time unit, 28-10
- simulation
  - adding to workflow, 9-8
  - and Distributed Engine, 57-20
  - and promotion-level filtering, 58-17
  - and workflow engine, 39-30
  - choosing mode, 56-9
  - compared to batch run, 2-11, 49-3, 57-17
  - controlling maximum size, 43-16
  - fast simulation, 55-11
  - related parameters, 43-16
  - starting engine from DOS, 56-11
  - starting engine from Start menu, 56-11
  - tablespace for, 43-6
- Simulation Engine, 49-3
- simulationindexspace parameter, 34-31, 43-7, 55-11
- simulation notification, missed, 34-4
- simulationspace parameter, 34-31, 43-7, 55-11
- Simulation Step
  - and other Demantra processing steps, 9-8
  - reference, 39-30
- SimWarnSize parameter, 34-31
- Slotting \$ series, 40-11
- SNO integration with S&OP, 16-14
- solution components, 1-2
- sorting
  - saving sort order, 8-11
- sortMemberbycodeinFilter, 34-32
- source tables, 13-5
- special characters (to avoid), 23-3, 49-10, 60-10
- SPF\_Default\_Profile, 34-32
- SPF\_Enable\_Worksheet\_Calculations, 34-32, 43-24
- SPF\_History\_Periods, 34-32
- splitting
  - below min\_fore\_level, 52-10, 58-30
  - during forecasting, 2-10, 50-8
  - during import, 29-7
  - overview, 3-5, 11-1
  - specifying reference series, 27-19
- SQL procedures, compiling, 24-3
- Sqrt function, 38-16
- square root, 38-16
- staging table
  - for item data, A-1
  - for location data, A-5
  - for promotion data, 40-11
  - for promotion populations, 40-15
  - for promotions, 40-12
  - for sales data, A-13
- staging tables



- and error tables, 29-28
- for data and level integration, 29-28
- for data import
  - specifying name, 29-8
  - specifying order of levels, 29-17
  - structure, 29-28
- for Data Model wizard
  - introduction, 25-4
  - joining, 25-8
  - specifying how to use fields, 25-10
  - specifying structure, 25-6
- for level import
  - creating, 29-27
  - structure of main table, 29-30
  - structure of population table, 29-32
- Standalone Integration Tool
  - using, 29-34
- Stand-Alone Integration Tool
  - and license, 66-7
  - using, 13-7
- standard error process, 48-1
- start\_date parameter, 58-53
- start\_new\_run parameter, 55-6, 56-17, 58-53
- START\_OR\_END\_WEEK parameter, 34-32
- StartAverage parameter, 58-54
- starting
  - Analytical Engine, 56-11
  - Business Modeler, 23-3
  - workflow instance, 62-3
- StartUpdateQueue parameter, 34-33, 43-2
- status of promotion
  - configuring, 40-4
- Std\_check parameter, 57-12
- StdRatio parameter, 57-12, 58-54, 58-54, 58-55
- step
  - adding, 31-3
  - common properties, 9-14
  - connecting to another step, 31-3
  - connection handles, 9-14
  - deleting, 31-4
  - editing, 31-4
  - overview of kinds, 9-6
  - reference for types, 39-2
- Stored Procedure Step
  - and other Demantra processing steps, 9-8
  - reference, 39-33
- string series, 27-18

- SubStr(), 37-3
- sub tabs of worksheet
  - overview, 8-5
- Sum(), 37-3
- summaries, 6-20
- summary row
  - and levels on x-axis, 38-10
  - computing average in, 38-16
  - computing maximum in, 38-17
  - computing minimum in, 38-17
  - computing row count in, 38-17
  - computing sum in, 38-17
  - computing weighted average in, 38-18
  - configuring for series, 27-9
  - dependencies on other series, 27-39
  - determining relative level, 38-10
  - introduction and examples, 6-18
- synchro field, 27-18
- synchronization
  - between views, 8-10
- syndicated data
  - in TPMO, 40-9
- sysadmin\_email\_address parameter, 34-33
- sysadmin\_password parameter, 34-33
- sysadmin\_username parameter, 34-34
- Sysdate(), 37-3
- System Manager permission
  - and delete access, 10-4
- SystemStatus parameter, 34-34

## T

---

- T\_SRC\_ITEM\_TMPL staging table, A-1
- T\_SRC\_LOC\_TMPL staging table, A-5
- T\_SRC\_SALES\_TMPL staging table, A-13
- tables
  - creating, 24-1
  - editing, 24-2
  - maintaining, 24-6
- tablespace parameter, 34-34, 43-7, 55-10
- tablespaces
  - checking storage, 55-10
  - used by Demantra Spectrum, 43-7, 55-10
- task
  - and workflow, 9-2
  - introduction, 1-7, 9-7
  - selection tasks, 39-27

- specifying, 39-2
- timeout, 9-13
- task.default.option parameter, 34-34
- task.message.length parameter, 34-34
- task.message.taskadded parameter, 34-34
- tasklist.showTimeoutTasks parameter, 34-34
- tasks timed out in workflow, 66-23
- templates for data models, 25-24
- temporary tables, cleaning up, 24-6
- test\_samp\_len parameter, 57-14, 58-54
- TestPeriod parameter, 57-12, 58-54, 58-55
- threading parameters, 43-17
- threadpool.attributesUpdate.per\_comb parameter, 34-34
- threadpool.attributesUpdate.size parameter, 34-35
- threadpool.attributesUpdate.timeout parameter, 34-35
- threadpool.copy\_paste.per\_process parameter, 34-35, 43-19
- threadpool.copy\_paste.size parameter, 34-35, 43-19
- threadpool.copy\_paste.timeout parameter, 34-35
- threadpool.copy\_paste.time-out parameter, 43-19
- threadpool.default.size parameter, 34-35
- threadpool.default.timeout parameter, 34-35
- threadpool.level\_method.block parameter, 34-35, 43-20
- threadpool.level\_method.size parameter, 34-36, 43-19
- threadpool.level\_method.timeout parameter, 34-36
- threadpool.level\_method.time-out parameter, 43-19
- threadpool.query\_run.size parameter, 34-36, 43-20
- threadpool.query\_run.timeout parameter, 34-36
- threadpool.query\_run.time-out parameter, 43-20
- threadpool.update.size parameter, 34-36, 43-18
- threadpool.update.timeout parameter, 34-36
- threadpool.update.time-out parameter, 43-19
- time aggregation
  - and time units, 5-7, 26-20
  - creating, 26-20
  - introduction, 5-6
- time aggregations
  - creating, 5-7
- time bucket
  - introduction, 2-3
  - referencing if empty, 6-9
- timeout
  - in BLE Step, 39-5
  - in Container Step, 39-9
  - in workflow, 9-13
- Timeout>Alert Time option, 39-5
- Timeout>Timer option, 39-5
- time resolution
  - base
    - if changed, 25-23
    - setting, 34-37
    - specifying, 25-3
  - used during import, 2-3
- time unit, 5-6, 5-7
  - and time aggregations, 26-20
  - and time axis, 5-8, 26-20
  - conceptual example, 7-4
  - creating, 28-10
  - deleting, 28-11
  - rules in Demantra Spectrum, 7-3
  - simple versus data-dependent, 7-3, 28-10
- timeunit parameter, 34-
- timeunit parametertimeunitTimeresolution parameter, 34-37
- Tnsname parameter, 66-6
- To\_char(), 37-3
- To\_date(), 37-3
- To\_number(), 37-4
- Today function, 38-18
- Tomcat
  - clearing cache, 66-11
  - non-default port, 66-11
  - port, 66-11
  - starting, 66-11
  - writing log to a file, 66-12
- TooFew parameter, 57-9, 58-55
- Tools and Applications menu
  - customizing, 44-6
- top\_level parameter, 58-55
- TopCoefficientLevel parameter, 58-56
- traffic lighting, 27-29
- Transfer Step
  - and other Demantra processing steps, 9-7
  - reference, 39-34
- treatment period was finished, 66-23

- trend
  - adjustment, 57-8, 57-15, 58-15
  - alignment with, 58-14, 58-61
  - damping coefficient, 59-6, 59-14
  - damping coefficients, 59-13
  - innovation coefficients, 59-12
  - measuring for adjustment, 58-58
  - renovation coefficient, 59-6, 59-14
- trend damp period, 34-37
- TrendDampPeriod parameter, 57-11
- TrendDampStep parameter, 57-11
- TrendModelForShort parameter, 57-11
- TrendOutlierRatio parameter, 57-11
- TrendPeriod parameter, 57-11, 57-15, 58-58
- TrendPreEstimation parameter, 57-10
- TrendShortRatio parameter, 57-11
- troubleshooting
  - engine problems, 56-12
  - if engine does not run, 56-12, 56-12
- Truncate function, 38-18
- tuning (performance), 55-7

## U

---

- units of measure
  - associating with aggregation levels, 28-8
  - configuring, 28-5
  - including in a component, 61-6
  - introduction, 2-2
  - in worksheets, 7-1
  - restarting server after changing, 23-9
  - where and when created, 7-8
- update\_dead\_comb, 34-37
- update\_units\_by\_items parameter, 34-37, 43-12
- UpdateAudit parameter, 34-37, 43-12
- Update Data Step, 39-35
- update-lock expression
  - configuring, 28-11
- UpdateQueueTimeout parameter, 34-38, 43-2
- Update Rate, 56-10
- UpdateThreadTimeout parameter, 34-38
- UpdateThreadtime-out parameter, 43-18
- UPGRADE\_TO\_SHAPE\_MODELLING
  - procedure, 35-35, 53-12
- Upgrading Schemas, 65-1
- upk
  - configuring, 43-30

- UpliftThresholdMethod parameter, 55-6, 58-59
- UpliftThresholdValue parameter, 55-6
- Upper(), 37-4
- UpperUpliftBound parameter, 55-5, 58-60
- UpTime parameter, 58-61
- UpTrend parameter, 57-15, 58-61
- URLs, 60-5, 66-1
  - changing, 66-8
- Use autogenerated ID as code for filter, 25-12
- UseBusinessFilter parameter, 58-62
- UseDateRangeMatrix, 34-38
- UseEnvelope parameter, 57-13, 58-62, 59-8, 59-11, 59-18, 59-21, 59-24
- UseExternalSDUpdate parameter, 58-63
- useGLEExceptions parameter, 34-39
- UseItemsAggri parameter, 27-35, 34-39, 43-12
- usemodelspernode parameter, 55-16, 58-64
- UseNonNegRegr parameter, 55-4, 58-64
- UseParamsPerNode parameter, 55-16, 58-64
- user
  - access to levels, 61-5
  - access to series, 61-2, 61-3, 61-12
  - access to worksheets, 63-2
  - adding to groups, 61-14, 61-18
  - copying, 61-15
  - creating, 61-10
  - filtering data, 61-13
  - filter mode (cascade vs non-cascade), 61-17
  - forcing logout, 61-28
  - overview, 10-1, 10-1
  - restarting server after changing, 23-9
  - specifying modules, 61-11
- user does not receive email, 66-24
- user is already active, 66-25
- UserListContentProvider.commontitle
  - parameter, 34-40, 43-4
- User Step
  - and other interactive steps, 9-7
  - reference, 39-36
- UserTitleContentProvider.TimeToSleep
  - parameter, 34-40, 43-4
- UseWeightedRegression parameter, 57-11

## V

---

- Valid\_fit parameter, 57-12
- verifying all expressions, 27-28

- vertical formula
  - definition and details, 6-8, 27-24
  - precedence in recalculation, 6-10

- view
  - and worksheet, 8-6
  - x-axis and y-axis, 8-6

- virtual directory
  - parameter where saved, 66-7

- VisibleUserName, 34-40

## W

---

- Wait Until Step
  - and other logic steps, 9-9
  - reference, 39-38

- Web Platform Server, 1-4

- Web server
  - configuration file, 66-5
  - introduction, 1-4
  - launching workflows when starting, 62-5
  - threading, 43-17
  - using Tomcat, 66-10

- Whoxd5 s Online
  - controlling user access, 44-10
  - customizing, 43-4
  - default availability and location, 44-7

- width of series, configuring, 27-10

- workflow
  - available steps, 9-2
  - email addresses used in, 61-11
  - introduction, 1-6
  - length of history, 43-12
  - parameters, 43-21
  - responding to external events, 9-6
  - schema
    - creating, 31-1
    - examples, 9-18
    - running an integration interface, 13-6, 13-17, 29-34
    - using as method, 32-4
    - when called by method, 9-5, 9-6
  - schema group
    - creating, 62-7
    - deleting, 62-8
  - schema instances
    - launching at Web server start, 62-5
    - scheduling, 62-3

- starting, 62-3
  - stopping, 62-6

- where and when created, 9-20

- workflow.group parameter, 34-40, 43-21, 61-19

- Workflow Editor/Manager
  - login access, 43-21, 61-19
  - scheduling workflow instances, 62-3
  - starting workflow instances, 62-3
  - stopping workflow instances, 62-6
  - using, 31-1
  - viewing process log, 62-9
  - viewing workflow status, 62-1

- Workflow Engine, 9-2

- Workflow internal error, 66-25

- Workflow process has failed, 66-25

- workflows
  - web services, 62-10

- Workflow Services, 1-4

- worksheet
  - changing access, 63-2
  - changing owner, 63-2
  - configuration
    - adding right-click options to, 2-8, 5-15, 9-1
    - adding to Open With menu, 40-19, 61-7
    - indicators for notes and promotions, 61-3
    - overview, 1-7
    - use of units and indexes, 7-1
    - using colors to flag conditions, 27-29

- data in, 2-6, 3-3

- deleting, 63-3

- design
    - filtering, 2-7
    - tips, 8-14

- elements of, 8-5

- evaluating client expressions in batch mode, 9-8

- how aggregated and organized, 2-5

- introduction, 2-1

- loading data automatically, 43-22

- object error message when opening, 66-19

- offline, cannot access, 66-18

- ownership and security, 8-9

- reloading data automatically, 43-22

- saving definition, 8-9

- table, 8-5

- viewing all worksheets, 63-1

- views in, 8-6
- when to define, 4-8, 23-3
- x-axis and y-axis, 8-6
- worksheet.private\_access\_type parameter, 43-23
- WorksheetBeanContentProvider.
- memberCombinationsLimit parameter, 34-41
- Worksheet Cache Step
  - and other Demantra processing steps, 9-8
  - reference, 39-41
- worksheet caching, 8-11
- WorksheetDefaultDateChoiceMethod parameter, 34-41, 43-23
- WorksheetDefaultSpan parameter, 34-41, 43-23
- Worksheet Manager, 63-1
- worksheet step
  - reference, 39-44
- wrap30.bat, 24-5
- WriteIntermediateResults parameter, 55-7, 56-17, 58-65
- WriteMissingDatesUplift parameter, 55-6, 58-66

## **X**

---

- x-axis
  - introduction, 8-6
- XDB database user, 66-21

## **Y**

---

- y-axis
  - introduction, 8-6
- Year function, 38-19
- young combination, 36-9, 50-19

## **Z**

---

- Z\_Val function, 38-20
- zero forecast
  - and prediction status, 36-9
  - requesting, 36-5
- zero length column error, 66-27
- zeros, stripping from data, 58-9

