Oracle® Retail Demand Forecasting

Implementation Guide

Release 15.0.1

E76450-04

August 2016

ORACLE®

Oracle Retail Demand Forecasting Implementation Guide, Release 15.0.1

Primary Author: Melissa Artley

reconfigurations, reassembly or reverse assembly, re-engineering or reverse engineering and recompilations or reverse compilations of the VAR Applications or any derivatives of the VAR Applications. You acknowledge that it shall be a breach of the agreement to utilize the relationship, and/or confidential information of the VAR Applications for purposes of competitive discovery.

The VAR Applications contain trade secrets of Oracle and Oracle's licensors and Customer shall not attempt, cause, or permit the alteration, decompilation, reverse engineering, disassembly or other reduction of the VAR Applications to a human perceivable form. Oracle reserves the right to replace, with functional equivalent software, any of the VAR Applications in future releases of the applicable program.

Contents

3 Integration

4 Installation Considerations

5 RDF Configuration Process

6 Advanced RDF Configurations

7 Batch Processing

8 Cross Promotion Effects Module (CPEM)

E Configuring the Preprocess Special Expression

F Customizing Hooks for the RDF Generate Utility and Curvebatch

G Curve Configuration Process

M Appendix: Configuring Seasonal Profiles

List of Figures

List of Tables

Send Us Your Comments

Oracle Retail Demand Forecasting Implementation Guide, Release 15.0.1.

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 Online Documentation available on the Oracle Technology Network Web site. It contains the most current Documentation Library plus all documents revised or released recently.

Send your comments to us using the electronic mail address: retail-doc_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

The Oracle Retail Demand Forecasting (RDF) Implementation Guide describes postinstallation tasks that need to be performed in order to bring RDF online and ready for production use.

Audience

This Implementation Guide is intended for the RDF application integrators and implementation staff, as well as the retailer's IT personnel. This guide is also intended for business analysts who are looking for information about processes and interfaces to validate the support for business scenarios within RDF and other systems across the enterprise.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Related Documents

For more information, see the following documents in the Oracle Retail Demand Forecasting Release 15.0.1 documentation set:

- *Oracle Retail Demand Forecasting Implementation Guide*
- *Oracle Retail Demand Forecasting Installation Guide*
- *Oracle Retail Demand Forecasting Release Notes*
- *Oracle Retail Demand Forecasting User Guide for the RPAS Classic Client*
- *Oracle Retail Demand Forecasting User Guide for the RPAS Fusion Client*
- Oracle Retail Predictive Application Server documentation

The following documentation may also be needed when implementing RDF:

- *Oracle Retail Planning Batch Script Architecture Implementation Guide*

Supplemental Documentation

The following document is available through My Oracle Support at the following URL:

https://support.oracle.com

Oracle Retail Demand Forecasting 15.0.1 Cumulative Fixed Issues (Note ID 2127030.1)

This document details the fixed issues and defects for all RDF, Curve, and Grade patch releases prior to and including the current release.

Customer Support

To contact Oracle Customer Support, access My Oracle Support at the following URL:

https://support.oracle.com

When contacting Customer Support, please provide the following:

- Product version and program/module name
- Functional and technical description of the problem (include business impact)
- Detailed step-by-step instructions to recreate
- Exact error message received
- Screen shots of each step you take

Review Patch Documentation

When you install the application for the first time, you install either a base release (for example, 15.0) or a later patch release (for example, 15.0.1). If you are installing the base release, additional patch, and bundled hot fix releases, read the documentation for all releases that have occurred since the base release before you begin installation. Documentation for patch and bundled hot fix releases can contain critical information related to the base release, as well as information about code changes since the base release.

Improved Process for Oracle Retail Documentation Corrections

To more quickly address critical corrections to Oracle Retail documentation content, Oracle Retail documentation may be republished whenever a critical correction is needed. For critical corrections, the republication of an Oracle Retail document may at times not be attached to a numbered software release; instead, the Oracle Retail document will simply be replaced on the Oracle Technology Network Web site, or, in the case of Data Models, to the applicable My Oracle Support Documentation container where they reside.

This process will prevent delays in making critical corrections available to customers. For the customer, it means that before you begin installation, you must verify that you have the most recent version of the Oracle Retail documentation set. Oracle Retail documentation is available on the Oracle Technology Network at the following URL:

http://www.oracle.com/technetwork/documentation/oracle-retail-100266.html

An updated version of the applicable Oracle Retail document is indicated by Oracle part number, as well as print date (month and year). An updated version uses the

same part number, with a higher-numbered suffix. For example, part number E123456-02 is an updated version of a document with part number E123456-01.

If a more recent version of a document is available, that version supersedes all previous versions.

Oracle Retail Documentation on the Oracle Technology Network

Oracle Retail product documentation is available on the following web site:

http://www.oracle.com/technetwork/documentation/oracle-retail-100266.html

(Data Model documents are not available through Oracle Technology Network. You can obtain them through My Oracle Support.)

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1
Introduction

Oracle Retail Demand Forecasting (RDF) is a statistical and promotional forecasting solution. It uses state-of-the-art modeling techniques to produce high quality forecasts with minimal human intervention. Forecasts produced by the RDF system enhance the retailer's supply-chain planning, allocation, and replenishment processes, enabling a profitable and customer-oriented approach to predicting and meeting product demand.

Today's progressive retail organizations know that store-level demand drives the supply chain. The ability to forecast consumer demand productively and accurately is vital to a retailer's success. The business requirements for consumer responsiveness mandate a forecasting system that more accurately forecasts at the point of sale, handles difficult demand patterns, forecasts promotions and other causal events, processes large numbers of forecasts, and minimizes the cost of human and computer resources.

Forecasting drives the business tasks of planning, replenishment, purchasing, and allocation. As forecasts become more accurate, businesses run more efficiently by buying the right inventory at the right time. This ultimately lowers inventory levels, improves safety stock requirements, improves customer service, and increases the company's profitability.

The competitive nature of business requires that retailers find ways to cut costs and improve profit margins. The accurate forecasting methodologies provided with RDF can provide tremendous benefits to businesses.

For a more detailed overview of the functionality within RDF, see the *Oracle Retail Demand Forecasting User Guide*.

Contents of this Guide

This implementation guide addresses the following topics:

Chapter 1, "Introduction": Overview of the RDF business workflow and skills needed for implementation.

Chapter 2, "Implementation Considerations": Explanation of the factors to take into consideration before performing the implementation.

Chapter 3, "Integration": Overview of integration and explanation of the RDF data flow and integration script.

Chapter 4, "Installation Considerations": Information for the setup that must be done prior to building the RDF domain and installing RDF

Chapter 5, "RDF Configuration Process": Overview of the steps needed to setup and customize RDF.

Chapter 6, "Advanced RDF Configurations": Overview of the steps needed for advanced configurations in RDF.

Chapter 7, "Batch Processing": Information on the RDF batch forecast process.

Chapter 8, "Cross Promotion Effects Module (CPEM)": Information on the module for cross promotion effects.

Chapter 9, "AutoSource": Information on the AutoSource utility.

Chapter 10, "Forecast Approval Alerts": Information on the usage and configuration of Forecast Approval Alerts.

Chapter 12, "Internationalization": Translations provided for RDF.

Appendix A, "RPAS and RDF Integration with RMS": Details RMS to RDF transformation programs, RDF to RMS extract programs, Grade (RPAS) to RMS extract programs, and Curve (RPAS) to Allocation extract programs.

Appendix B, "Configuring the Cluster Procedure": Details how retailers can use Clustering to provide insight into how various parts of their operations can be grouped together.

Appendix C, "Configuring the Clone Procedure": Describes how Cloning can generate forecasts for new items and locations by copying, or cloning history, from other items and stores.

Appendix D, "AppFunctions": Describes how the AppFunctions library supports a number of functions and special expressions for RDF.

Appendix E, "Configuring the Preprocess Special Expression": Describes the Oracle Retail Preprocess module, Preprocessing, to correct past data points that represent unusual sales values that are not representative of a general demand pattern.

Appendix F, "Customizing Hooks for the RDF Generate Utility and Curvebatch": Details the hooks used in RDF and Curve for running customized computation at certain points of the batch process.

Appendix G, "Curve Configuration Process": Overview about Curve, an RPAS solution that is used to generate ratios from historical data at user-specified intersections

Appendix H, "Grade Configuration Process": Overview about Grade, a clustering tool that provides insight into how various parts of a retailer's operations can be grouped together.

Appendix I, "CPEM Calculations": Describes the special expressions that calculate cross promotional effects for Cross Promotional Effects Module (CPEM).

Appendix J, "RDF Script Names": Lists the revised script names common to RDF Cloud Service and RDF.

Appendix K, "Configuring the Forecast150 and CausalEstimate": Describes how to configure the Forecast150 procedure that generates the forecast and estimates the promotion effects.

Appendix L, "Configuring the Similarity Score Calculation": Describes the procedure to configure the Similarity Score Calculation.

Appendix M, "Appendix: Configuring Seasonal Profiles": Describes several options to create a seasonal profile for Curve.

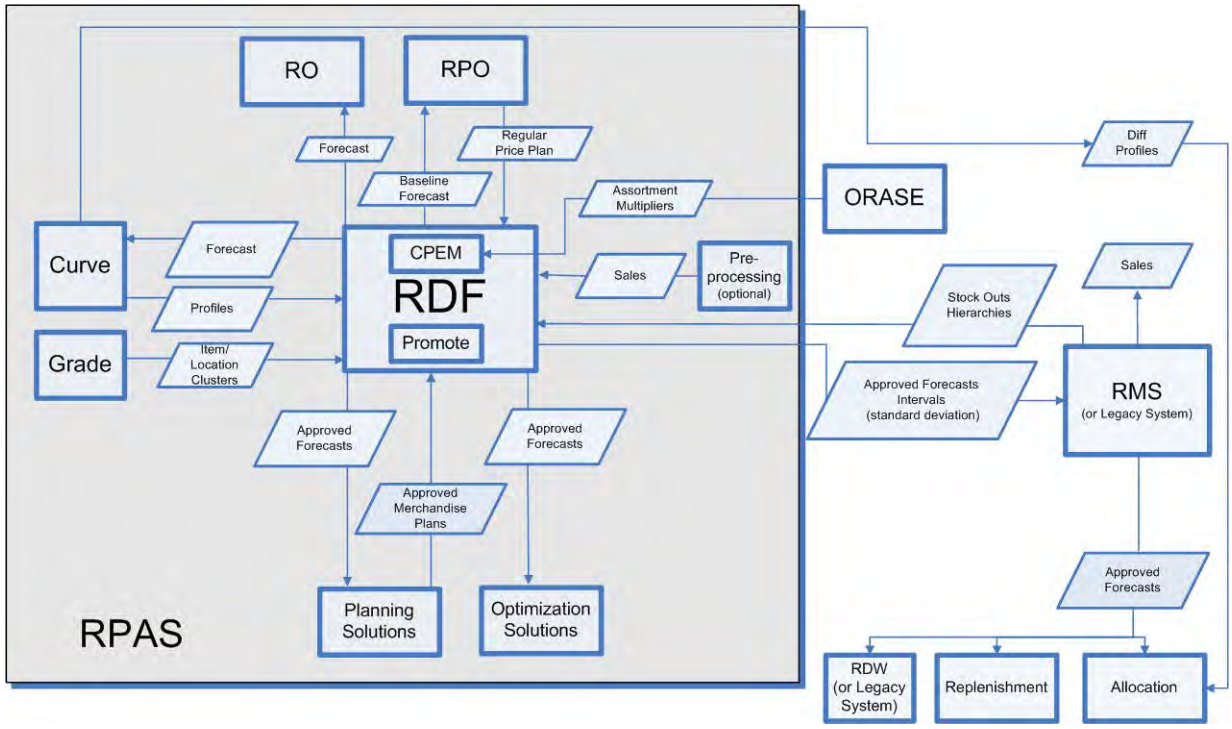
RDF and the Oracle Retail Enterprise

Oracle Retail has designed a forecasting solution separate from replenishment, allocation or planning. In order to provide a single version of the truth, it is crucial to free up the user's time and supply the tools to focus on the analysis of forecast exceptions, historical data, and different modeling techniques. This empowers the user to make better decisions, thus improving overall accuracy and confidence in the results of the forecast demand.

Within the Oracle Retail Enterprise, Oracle Retail Merchandising System (RMS) supplies RDF with Point-of-Sale (POS) and hierarchy data that is used to create a forecast. Once the forecast is approved, it is exported to RMS in order to calculate a recommended order quantity. Forecasts can also be utilized (no export process required) in any Retail Predictive Application Server (RPAS) solution to support merchandise, financial, collaborative, and price planning processes.

Figure 1–1 shows the interaction between RDF, RPAS, and other applications.

Figure 1–1 RDF and the Oracle Retail Enterprise

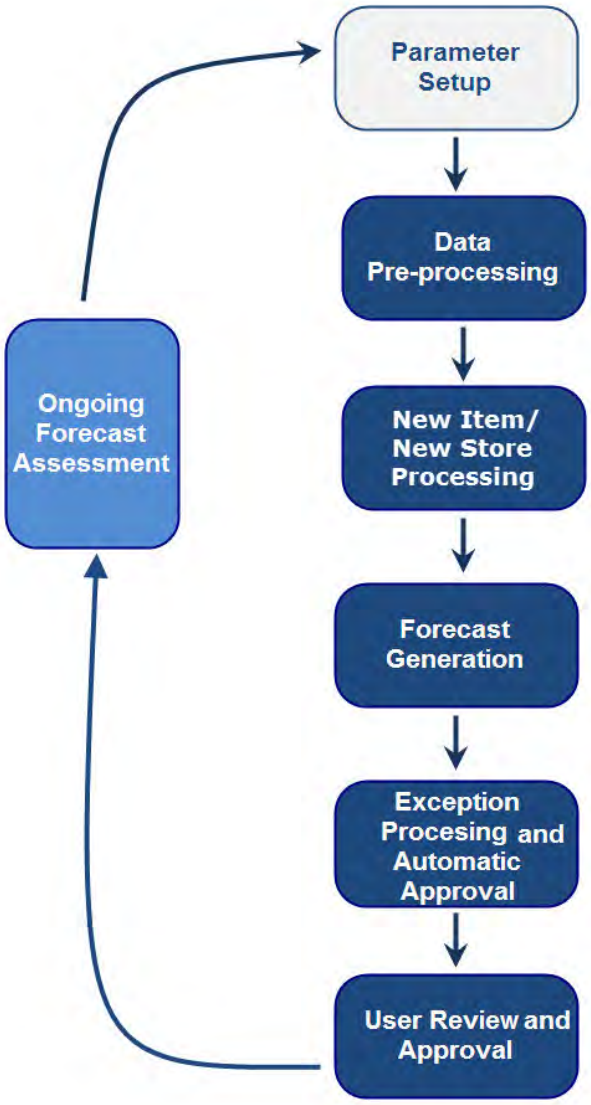


RDF Business Process Workflow

One of the challenges in retail forecasting is the data volumes. The RDF business process focuses on automation, accuracy and lends itself to easy analysis. RDF focuses on automation by automatically selecting best forecast methods and parameters as well as by automatic approval of forecasts that don't meet any exception criteria. Also, it allows users to analyze and manually approve forecasts. Forecast scorecarding allows users to monitor forecast accuracy over time and re-tune settings if necessary.

Figure 1–2 illustrates the RDF business process workflow.

Figure 1–2 Business Process Workflow



Parameter Setup

Following the initial setup, these parameters are not set on a scheduled basis, but are updated as needed.

- Preprocessing and alert parameter setup
- Sets forecast methods, parameters, and specifies source levels
- Sets history start and end dates

Data Pre-processing

For data pre-processing, RDF:

- Corrects for lost sales due to stock-outs
- Cleanses data for effects of promotions and short-term price changes (optional)

- Manual data-scrubbing (fake history and user history overrides)

New Item / New Store Processing

For new item / new store processing, RDF:

- Calculates item similarity score based on item attributes
- Generates recommendations for like item for new items
- Clones sales history

Forecast Generation

For forecast generation, RDF:

- Computes demand parameters (seasonality, level, trend)
- Optimizes exponential smoothing parameters
- Allows you to select best forecast method for item/location or use Automatic Exponential Smoothing (AutoES)

Exception Processing and Automatic Approval

For exception processing and automatic approval, RDF:

- Evaluates forecast for exceptions based on specific alert criteria
- Automatically approves non-alerted forecasts
- Allows you to review flagged exception forecasts

User Review and Approval

For user review and approval, RDF:

- Reviews and analyzes forecasts, allowing for overrides if necessary
- Approves forecasts

Ongoing Forecast Assessment

For ongoing forecast assessment, RDF:

- Assesses user overrides versus system forecasts against actuals
- Assesses forecast quality and user adoption
- Retunes parameter settings as needed

Key Features of RDF

RDF provides the following features:

- Pre-processing to correct for stock-outs and other data anomalies
- Automatic recommendation of like item for new items
- Generation of forecasts
- Optimizes forecasting methods and exponential smoothing parameters
- Selects best forecasting methods and parameters to overcome data sparsity and reliability issues

- Generation of alerts and automatic approval of forecasts
- Allows you to facilitate review of analysis and approval of forecasting

Skills Needed for Implementation

A typical RDF implementation team has technical and application/business consultants in addition to other team members.

The technical and application/business consultants need to have a high level understanding of other applications that RDF can integrate with, which include:

- Advanced Inventory Planning (AIP)
- Analytic Parameter Calculator for Regular Price Optimization (APC-RPO)
- Analytic Parameter Calculator for Replenishment Optimization (APC-RO)
- Replenishment Optimization (RO)
- Retail Merchandising System (RMS)
- Retail Price Optimization (RPO)

In addition, both technical and application/business consultants need to have an understanding of RPAS, its calculation engine, and multi-dimensional database concepts.

Note: Staffing models and roles and responsibilities may vary from project to project, but following is a recommendation based on best practices.

Technical Consultant Role

The technical consultant is usually responsible for the following key areas in addition to other activities:

- Interface work
- Batch scripting
- RPAS/RDF domain partitioning

Note: The technical consultant should also be well versed in Unix, Shell scripting, and batch schedulers.

Application / Business Consultant Role

The application/business consultant is responsible for:

- Designing and configuring alerts
- Configuring pre-processing rules
- Any workflow/workbook customizations needed to meet retailers business process needs

Note: The application consultant should have a strong understanding of RPAS configuration rule language, RPAS Configuration Tools, RDF plug-in, and have experience configuring solutions on RPAS.

2

Implementation Considerations

The following information needs to be considered before implementing RDF:

- Hardware Space Impacts
- Domain Partitioning
- Patch Considerations
- Batch Scheduling
- Security
- Internationalization
- Translation

Hardware Space Impacts

The following factors can affect hardware space requirements:

- Item—Number of distinct items.
- Store—Number of physical, Web, and other distinct retail outlets.
- Calendar—Number of historical and future time periods in the domain. This impacts the overall size of the environment.
- Workbooks—Amount of space used by workbooks. This is typically greater than the domain itself. The number of workbooks is related to the number of users.

Domain Partitioning

Partitioning is done to avoid competition for resources. Building a workbook and committing data are two processes that can cause contention.

How data is partitioned has an impact on the business process. The RDF domain is defined as a global domain. For performance reasons, a single subdomain is not recommended. There should be an even distribution of users across a set of local domains. For example, men's merchandise could be in a domain, women's merchandise in a domain, and children's merchandise in a domain. When a user is committing data in the men's merchandise domain, this will not affect the users in the women's or children's domains because of the use of partitioning.

Note: Domain partitioning is supported only along Product hierarchy (PROD). This is a standard RPAS hierarchy. Also source levels have to be below partion dimension, that is, if using Dept for source level forecasting, you have to partition at or below Dept.

Consider the following questions when defining the partitioning of the domain:

- How do I partition to meet my business needs?
- How do I partition my users?
- How do I create groups of users to further partition the solution?

In the GA configuration, group is a dimension label. The group dimension is a regular dimension in the product hierarchy, which the customer can rename or delete.

One of the major purposes of partitioning in RDF is to facilitate the parallelization of the batch process.

The wise selection of partition intersections can significantly reduce the batch time. Partition intersection selection should also consider business needs in such a way that contention issues are minimized. RDF has a special restriction in partitioning. Each local domain must only have one partition dimension position.

Patch Considerations

With a new release, there are two types of patches that can affect the RDF domain:

- Changes to the code in the RPAS libraries.

  The configuration is not affected by this type of patch. For these types of changes, applying the patch is a straight forward process.

- Changes to the configuration.

  These types of changes can be more complex. If a retailer has customizations in the configuration, the customizations must be redone on the new configuration before the patch is installed.

Patching Process

Before patching an RDF domain, confirm that the necessary RPAS client, server, and Configuration Tools patch updates have been successfully applied. Refer to the *Oracle Retail Predictive Application Server Installation Guide* for RPAS installation instructions.

Batch Scheduling

RDF batch is typically scheduled to run end of week with the most updated feeds of sales history and foundation data. Some tasks or batch processes can be run adhoc or as needed basis.

Following is a list of typical RDF batch tasks and scheduling considerations:

- Weekly activities:
  - Hierarchy Load
  - Data Load
  - Pre-processing

  - Forecast Generation
  - Alert Generation
  - Commit batch (committing workbooks saved to be committed later)
  - Auto Workbook build
- Adhoc/as needed
  - AutoSource

Security

To define workbook template security, the system administrator grants individual users, or user groups, access to specific workbook templates. Granting access to workbook templates provides users the ability to create, modify, save, and commit workbooks for the assigned workbook templates. Users are typically assigned to groups based on their user application (or solution) role. Users in the same group can be given access to workbook templates that belong to that group alone. Users can be assigned to more than one group and granted workbook template access without belonging to the user group that typically uses a specific workbook template. Workbook access is either denied, read-only, or full access. Read-only access allows a user to create a workbook for the template, but the user cannot edit any values or commit the workbook. The read-only workbook can be refreshed.

When users save workbooks, they assign one of three access permissions:

- World—Allow any user to open and edit the workbook.
- Group—Allow only those users in their same group to open and edit the workbooks.
- User—Allow no other users to open and edit the workbook.

Note: If you choose to customize your permissions, keep in mind the *Principle of Least Privilege* which states; only give a user enough permissions to do their job and nothing more.

Note: A user must have access to the workbook template in order to access the workbook, even if the workbook has world access rights.

Table 2–1 provides guidance on what user roles should have access to each workbook.

Table 2–1 Workbook Access Based on User Roles

Workbook	User Roles
New Item Administration	Forecast Analyst
New Item Review	Forecast Analyst
New Item Maintenance	Forecast Analyst
Business Support Like Store	Forecast Analyst
Forecaster Like Store	Forecast Analyst
Preprocess Administration	Forecast Analyst, Forecast Manager
Source Measure Maintenance	Forecast Analyst, Forecast Manager

Table 2–1 (Cont.) Workbook Access Based on User Roles

Workbook	User Roles
Floating Events Admin	Forecast Analyst, Forecast Manager
Extra Week Administration	Forecast Analyst, Forecast Manager
Short Life Cycle Maintenance	Forecast Analyst, Forecast Manager
Forecast Administration	Forecast Analyst, Forecast Manager
Forecast Maintenance	Forecast Analyst, Forecast Manager
Run Batch Forecast	Forecast Analyst, Forecast Manager
Forecast Approval	Forecast Analyst, Forecast Manager
Forecast Alert	Forecast Analyst, Forecast Manager
Forecast Delete	Forecast Analyst, Forecast Manager
Forecast Scorecard	Forecast Analyst, Forecast Manager
Interactive Forecasting	Forecast Analyst, Forecast Manager
Grouping Management	Forecast Analyst, Forecast Manager
Promotion Planner	Forecast Analyst, Forecast Manager
Promotion Management	Forecast Analyst, Forecast Manager
Promotion Maintenance	Forecast Analyst, Forecast Manager
Promo Effectiveness	Forecast Analyst, Forecast Manager
Profile Administration	Forecast Analyst, Forecast Manager
Profile Maintenance	Forecast Analyst, Forecast Manager
Run Batch Profile	Forecast Analyst, Forecast Manager
Profile Approval	Forecast Analyst, Forecast Manager
Breakpoint Administration	Forecast Analyst, Forecast Manager
Delete Cluster Run	Forecast Analyst, Forecast Manager
Cluster Review	Forecast Analyst, Forecast Manager
Generate Breakpoint Grades	Forecast Analyst, Forecast Manager
Generate Clusters	Forecast Analyst, Forecast Manager

For more information on security, see the *Oracle Retail Predictive Application Server Administration Guide for the Fusion Client*.

Internationalization

For more information on translation for RDF, see Chapter 12, "Internationalization".

Translation

Translation is the process of interpreting and adapting text from one language into another. For more information, refer to "Translation" on page 12-1

3
Integration

This chapter describes the interaction between RDF and other applications and the script used to load demand data.

RDF and RDF Cloud Service Integrations

The supported integrations differ between RDF and RDF Cloud Service. Table 3–1 shows the supported integrations.

Table 3–1 RDF and RDF Cloud Service Supported Integrations

Integration	RDF	RDF Cloud Service
RMS to RDF	Yes	Yes
RDF to RMS	Yes	Yes
APC-RPO to RDF	Yes	No
RPO to RDF	Yes	No
RDF to RPO	Yes	No
RDF to APC-RO	Yes	Yes
RDF to AIP	Yes	Yes

Oracle Retail Application Integration

RDF is integrated with the following Oracle Retail applications listed by group:

- Integrated Inventory Planning Suite Data Flow
  - From RDF to AIP
  - From RDF to APC-RO
  - From APC-RO to RDF
- RDF Supporting RMS Replenishment and Allocation
  - From RMS to RDF
  - From RDF to RMS
- RDF Data Flow with RPO
- APC-RPO, RPO, RDF Integration
  - From APC-RPO to RPO
  - From APC-RPO to RDF
  - From RPO to RDF

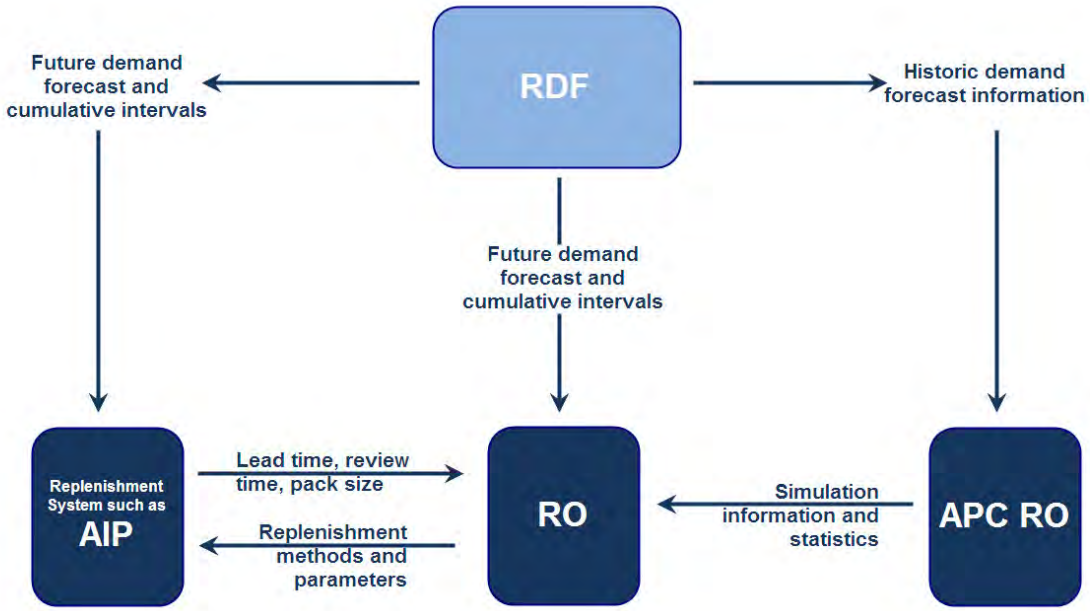
- From RDF to RPO

Integrated Inventory Planning Suite Data Flow

Figure 3–1 shows the integration of the Integrated Inventory Planning Suite applications and the flow of data among those applications. Note that Figure 3–1 shows a replenishment system. This can be AIP or any other replenishment system. The demand forecasting application can be RDF or any other forecasting system. RDF forecasts are used as input to RO for simulation-determined replenishment parameters. RDF forecasts and associated statistics are used by AIP to plan time-phased replenishment

This solution supports data sharing among these applications. Note that the data sharing functionality is not dependent on the presence of all these applications. The defined data sharing between any of the applications works for the entire suite as well as for a subset of the applications.

Figure 3–1 Integrated Inventory Planning Suite Data Flow



From RDF to AIP

The data flow from RDF to AIP includes weekly and daily forecasts and cumulative intervals.

For detailed information about the RDF to AIP interface, see the following:

- Chapter 7, "Batch Processing"
- *Oracle Retail Advanced Inventory Planning Implementation Guide*

From RDF to APC-RO

RDF's forecasts are important inputs for APC-RO. Forecasts are provided for initial load into APC-RO. A rolling 52 forecasts are generated and exported to APC-RO, each

of the forecasts starts one week after another. The main purpose of the integration is to generate RDF forecast in RDF GA domain and export the forecasts from a RDF domain and covert the forecasts into the format required by APC-RO.

APC-RO provides to RDF a list of item/stores and RDF only exports the forecasts for those item/stores to APC-RO.

For detailed information about the RDF to APC-RO interface, see the following:

- Chapter 7, "Batch Processing"
- *Oracle Retail Analytic Parameter Calculator for Replenishment Optimization Implementation Guide*

From APC-RO to RDF

APC-RO provides to RDF a list of item/stores and RDF only exports the forecasts for those item/stores to APC-RO.

For detailed information about the APC-RO to RDF interface, see the following:

- Chapter 7, "Batch Processing"
- *Oracle Retail Analytic Parameter Calculator for Replenishment Optimization Implementation Guide*

RDF Supporting RMS Replenishment and Allocation

RDF integrates with Retail Merchandising System (RMS) to receive foundation data. In addition, it also sends weekly and daily forecasts to RMS (replenishment and allocation). These descriptions explain the data flows between RMS and RDF.

For detailed information about the RMS and RDF interface, see the following:

Appendix A, "RPAS and RDF Integration with RMS"

Oracle Retail Merchandising System Operations Guide, Volume 1

From RMS to RDF

The data flow from RMS to RDF includes:

- Product hierarchy
- Location hierarchy
- Calendar hierarchy

From RDF to RMS

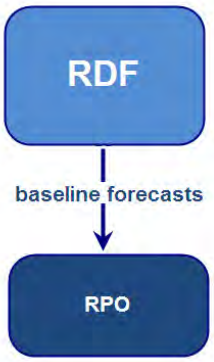
The data flow from RDF to RMS includes:

- Weekly and daily forecasts and cumulative intervals

RDF Data Flow with RPO

RDF sends baseline forecasts to RPO.

Figure 3–2 RDF Data Flow with RPO



APC-RPO, RPO, RDF Integration

This section describes the integration and data flow between APC-RPO, RPO, and RDF.

Figure 3–3 APC-RPO, RPO, RDF Integration

From APC-RPO to RPO

The data flow from APC-RPO to RPO sends:

- Item and cross item elasticities of items. RPO uses these elasticities to optimize prices.
- Maximum and minimum historical prices of items. RPO uses this data to optimize prices.
- Anchor prices of items. Anchor prices are the baseline prices that APC-RPO uses to calculate price elasticity. RPO uses the anchor prices to calculate price drift metrics.
- Maximum price increase and decrease percentages, self-item elasticity standard errors, and self-item elasticities t-statistics. RPO uses the maximum price increase and decrease percentages to setup up the default minimum and maximum price percentage change. Meanwhile, the RPO user can refer to the self-item elasticity standard error and t-statistics to adjust the price constraint.

From APC-RPO to RDF

The data flow from APC-RPO to RDF sends:

- Regular price self elasticities to RDF. The self elasticities, together with the price plan, allow RDF to calculate the item elasticity lift.
- Regular price cross-item elasticities to RDF. There are two types of cross-item elasticities: halo and cannibalization. These cross elasticities, together with the price plan, allow RDF to calculate the cross-item lift for both halo and cannibalization effects related to the corresponding elasticities.
- Anchor prices to RDF for reference purposes.
- Historical prices. RDF uses these to calculate the regular price lifts.

From RPO to RDF

The data flow from RPO to RDF sends the price plan that allows RDF to calculate the three components of the regular price lift: regular price self effect, regular price cannibalization effect, and regular price halo effect.

From RDF to RPO

The data flow from RDF to RPO sends forecasts to RPO. These forecasts represent the base demand at the item/store level. RPO aggregates the forecasts to the item/price zone level and uses that data to optimize prices.

4

Installation Considerations

This chapter describes the setup that must be done before building the RDF domain and the batch script that must be run to build the domain.

Note: For information on building the Curve and Grade domains, refer to the *Oracle Retail Predictive Application Server Installation Guide*.

Installation Dependencies

RPAS and RDF must be installed before setting up and configuring RDF:

- For information on installing RPAS, see the *Oracle Retail Predictive Application Server Installation Guide*.
- For information on installing RDF, see the *Oracle Retail Demand Forecasting Installation Guide*.

RPAS Installation

The Java-based RPAS installation programs that are included with the installation package are used to install the server-side RPAS components on UNIX operating systems.

The RPAS installer performs the following functions:

- Installs the RPAS server
- Installs the Configuration Tools on the server
  - On Windows, an InstallShield package is used to install the Configuration Tools.
- Defines the DomainDaemon port

RPAS Client Installation

The RPAS server installation package also includes the following RPAS clients:

- RPAS Classic Client: A Windows-based client interface for end users and system administrators of an RPAS domain.
- RPAS Fusion Client: A Web-based client developed using Oracle Application Development Framework (ADF).

Note: RDF Cloud Service only supports the RPAS Fusion Client.

Each RPAS client installation package includes a separate installer to help you install the client. For more information on installing the RPAS clients, refer to the *Oracle Retail Predictive Application Server Installation Guide*.

RDF Installation

In addition to the RPAS installer, the installation package also includes the Java-based RPAS installation program for the RDF application.

The RDF installer automates the following tasks:

- Installs the RDF mock install configuration
- Installs RDF plug-ins for the Configuration Tools
- Installs Language Translation files
- Optionally creates a sample RDF domain

RDF Taskflow for the RPAS Fusion Client

The RDF installation software enables you to install the taskflow and online help files for the RPAS Fusion Client. In order to install the taskflow files, the RPAS Fusion Client must already be installed. For more information on installing the RPAS Fusion Client, see the *Oracle Retail Predictive Application Server Installation Guide*.

During the RPAS Fusion Client installation, the installer automatically sets up the RPAS domain connection configurations in the `ProfileList.xml` file. If you choose to set up the domain connection after the installation or set up an additional domain, you must manually set up the connection. For more information, see the *Oracle Retail Predictive Application Server Administration Guide for the Fusion Client*.

Environment Variables

In addition to the regular RPAS environment variables, including RPAS_HOME, you need to set up the following environment variables and export them.

Table 4–1 Environment Variables

Environment Variable	Description	Use
RIDE_HOME	The location of the Configuration Tools.	Mandatory for Installation and Patching
PATH	The standard operating system path. Running the script `$RPAS_HOME/rpaslogin.ksh` is the preferred method to update and export the PATH variable.	Mandatory
RDF_BATCH_TIMEOUT	This numeric value, set in seconds, controls the amount of time that the Batch Forecast workbook will wait for another Batch Forecast workbook process to finish. This allows several users in different local domains to start a Batch Forecast workbook simultaneously.	Optional

Files Needed to Build the RDF Domain

Before the domain is built, the following types of files need to be set up:

- Standard RPAS Hierarchy Files
- Data Files

- Historical Data
- Loading and Extracting Data

Standard RPAS Hierarchy Files

The following hierarchy files are needed:

- Calendar (CLND) Hierarchy File
- Product (PROD) Hierarchy File)
- Location (LOC) Hierarchy File
- Time Series Grouping (Attribute) Hierarchy File

> **Note:** As with all standard RPAS hierarchies, these hierarchies are configurable. For information about configuring these hierarchies, see Chapter 5, "RDF Configuration Process."

Calendar (CLND) Hierarchy File

File name: clnd.csv.dat
File format: comma-separated values file

Table 4–2 describes the fields in the file:

Table 4–2 Calendar Hierarchy Fields

Name	Description	Parent
DAY	Day ID (YYYYMMDD format)	None
DAY label	Day label	not applicable
WEEK	Week ID	DAY
WEEK label	Week label	not applicable
MNTH	Month ID	WEEK
MNTH label	Month label	not applicable
QRTR	Quarter ID	MNTH
QRTR label	Quarter label	not applicable
SSN	Fiscal Half ID	QRTR
SSN label	Fiscal Half label	not applicable
YEAR	Year ID	SSN
YEAR label	Year label	not applicable
DOW	Day of Week ID	DAY
DOW label	Day of Week label	not applicable
DOS	Day of Season ID	DAY
DOS label	Day of Season label	not applicable
WOY	Week of Year ID	WEEK
WOY label	Week of Year label	not applicable
WOS	Week of Season ID	WEEK

Table 4–2 (Cont.) Calendar Hierarchy Fields

Name	Description	Parent
WOS label	Week of Season label	not applicable

Example 4–1 CLND Format

```
20050130,01/30/2005,w01_2005,01/30/2005,JAN_2005,January 2005,Q1_
2005,Quarter 1 2005,H1_2005,2005 First Half, A2005, Year 2005, SAT,
Saturday,DOS01,DOS 01,WY01, Week 01, WS01,WOS 01
```

There is also one user dimension supported for the calendar hierarchy. This dimension is not loaded. It is maintained in the Hierarchy Maintenance workbook. For more information, see the *Oracle Retail Predictive Application Server Administration Guide for the Fusion Client*.

Name	Description	Parent
WG1	Week grouping ID	WEEK

Note: After upgrading a domain or loading a new calendar hierarchy into the RDF domain, ensure that the current date measures value is within the new calendar hierarchy.

Product (PROD) Hierarchy File

File name: prod.csv.dat
File format: comma-separated values file

Table 4–3 describes the fields in the file with the GA hierarchy definition. The hierarchy may be configured per implementation, in which case the implementer will need to provide the file format. Note that the product hierarchy must have Item (ITEM), Subclass (SCLS), and Group (PGRP) dimensions.

Table 4–3 Product Hierarchy Fields

Name	Description	Parent
ITEM	Item ID	None
ITEM label	Item label	not applicable
ITPT	Parent ID	ITEM
ITPT label	Parent label	not applicable
ITGP	Grand Parent ID	ITPT
ITGP label	Grand Parent label	not applicable
SCLS	Subclass ID	ITGP
SCLS label	Subclass label	not applicable
CLSS	Class ID	SCLS
CLSS label	Class label	not applicable
DEPT	Department ID	
DEPT label	Department label	not applicable
PGRP	Group	

Table 4–3 (Cont.) Product Hierarchy Fields

Name	Description	Parent
PGRP label	Group label	not applicable
DVSN	Division ID	PGRP
DVSN label	Division label	not applicable
SPLR	Supplier ID	ITEM
SPLR label	Supplier label	not applicable
The following fields are unavailable in RDF Cloud Service		
PTD1	Parent Diff 1ID	ITEM
PTD1 label	Parent Diff 1 label	not applicable
GPD1	Grand Parent Diff 1 ID	PTD1
GPD1 label	Grand Parent Diff 1 label	not applicable
SCD1	Subclass Diff 1 ID	GPD1
SCD1 label	Subclass Diff 1 label	not applicable
CLD1	Class Diff 1 ID	SCD1
CLD1 label	Class Diff 1 label	not applicable
DPD1	Dept Diff 1 ID	CLD1
DPD1 label	Dept Diff 1 label	not applicable
DIF1 1	Diff 1 ID	DPD1
DIF1 1 label	Diff 1 label	not applicable

Example 4–2 PROD Format (RDF Cloud Service)

```
10000010,10000010Leather Loafer - Black 6 B,10000010,10000010Leather
Loafer - Black 6 B,10000009,10000009Leather
Loafer,122,122Loafer,1312,1312Casual*,1310,1310Footwear
Women's*,1300,Group 1,1,All Product,1000,Supplier 1
```

Example 4–3 PROD Format (RDF)

```
10000010,10000010Leather Loafer - Black 6 B, 10000010, 10000010Leather
Loafer - Black 6 B, 10000009, 10000009Leather Loafer, 122, 122Loafer,
1312, 1312Casual, 1310, 1310Footwear Women's, 1300, Group 1, 1, All
Product, 1000, Supplier 1, 10000010_sml, 10000010Leather Loafer - Black 6
B Small, 10000009_sml, 10000009Leather Loafer Small, 122_sml, 122Loafer
Small, 1312_sml,,  1312Casual* Small, 1310_sml,,  1310Footwear Women's*
Small, _sml, Small
```

There are also two user dimensions supported for the product hierarchy. These dimensions are not loaded. They are maintained in the Hierarchy Maintenance workbook.. For more information, see the *Oracle Retail Predictive Application Server Administration Guide for the Fusion Client*.

Name	Description	Parent
ITG1	Item Grouping 1	IDITEM
ITG2	Item Grouping 2	IDITEM

Location (LOC) Hierarchy File

File name: loc.csv.dat
File format: comma-separated values file

Table 4–4 describes the fields in the file with the GA hierarchy definition. The hierarchy may be configured per implementation, in which case the implementer will need to provide the file format. Note that the location hierarchy must have Store (STR) and Region (RGN) dimensions.

Table 4–4 Location Hierarchy Fields

Name	Description	Parent
STR	Store ID	none
STR label	Store label	not applicable
DSTR	District ID	STR
DSTR label	District label	not applicable
RGN	Region ID	DSTR
RGN label	Region label	not applicable
AREA	Area ID	RGN
AREA label	Area label	not applicable
CHN	Chain ID	AREA
CHN label	Chain label	not applicable
CMPN	Company ID	CHN
CMPN label	Company label	not applicable
SFMT	Store Format ID	STR
SFMT label	Store Format label	not applicable
STCL	Store Class	STR
STCL label	Store Class Label	not applicable

Example 4–4 LOC Format

```
1000, New York City, 1000, US, 1000, North America, 1000, The Americas,
1000, Bricks & Mortar, 100, JCB Trading Company, 4, 4, A, A
```

There are also two user dimensions supported for the product hierarchy. These dimensions are not loaded. They are maintained in the Hierarchy Maintenance workbook.. For more information, see the *Oracle Retail Predictive Application Server Administration Guide for the Fusion Client*.

Name	Description	Parent
STG1	Store Grouping 1 ID	STR
STG2	Store Grouping 2 ID	STR

Product Attributes (ATTR) Hierarchy File

File name: attr.csv.dat
File format: comma-separated values file

Table 4–5 describes the fields in the file with the GA hierarchy definition.

Table 4–5 Time Series Grouping Hierarchy Fields

Name	Description	Parent
ATN	Attribute name ID	none
ATN label	Attribute name label	not applicable

Example 4–5 ATTR Format

```
10000,Activity
```

There are no user dimensions supported for the product attributes hierarchy.

Time Series Grouping (Attribute) Hierarchy File

File name: grph.csv.dat
File format: comma-separated values file

Table 4–6 describes the fields in the file.

Table 4–6 Time Series Grouping Hierarchy Fields

Name	Description	Parent
GRPD	Group Dimension	none
GRPD label	Group Dimension label	not applicable

Example 4–6 Attribute Format

```
group1,low ros low demand summer
group2,low ros low demand fall
```

Data Files

Table 4–7 describes the measures as defined in the GA configuration. The levels and intersections may be configured per implementation, in which case the implementer will need to provide updated information.

Table 4–7 Data Files

Measure Name	Measure Label	Base Intersection	Data Type	On-premise or Cloud	Required or Optional	Loaded By
Category: Sales						
dpos	Daily Sales	day / item /str	Real	On-premise	Required	rdf_load_measures.ksh
pos	Weekly Sales	week / item /str [1]	Real	Both	Required	rdf_load_measures.ksh
Category: Preprocessing						
ldpreoosind	Loaded Out of Stock Indicator	week / item /str	Boolean	Both	Optional	rdf_load_measures.ksh
preppiind	Promotion Indicator	week / item /str	Boolean	Both	Optional	rdf_load_measures.ksh
preseaprof	Seasonal Profile	woy / scls / rgn	Real	Both	Optional	rdf_load_measures.ksh
Category: New Item / New Store						
nitattval	Attribute Value	item / atn[1]	String	Both	Only required for Attribute Based New Item	rdf_load_measures.ksh
nitattvaltyp	Attribute Value Type	atn	Integer	Both	Only required for Attribute Based New Item	rdf_load_measures.ksh
nitdattwgt	Attribute Weight	clss / atn[3]	Real	Both	Only for required Attribute Based New Item	rdf_load_measures.ksh
nitdatttau	Numeric Attribute Tau	clss / atn[3]	Real	Both	Only required for Attribute Based New Item	rdf_load_measures.ksh
nitfcststovr	Forecast Start Date Override	item / str[2]	Date	Both	Optional	rdf_load_measures.ksh
nishiststovr	History Start	item / str[2]	Date	Both	Optional	rdf_load_measures.ksh

Table 4–7 (Cont.) Data Files

Measure Name	Measure Label	Base Intersection	Data Type	On-premise or Cloud	Required or Optional	Loaded By
nititm2itr	Item to Item RHS map	item/iter	Boolean	Both	Only required for Attribute Based New Item	rdf_load_measures.ksh
Category: Grouping						
grpasnmt	Group Assignment	item / str	String	Both	Only required if intersection of forecast level includes time series group dimension	rdf_load_measures.ksh
Category: Promotions						
pvarxlpprc	Promotion Variable Price Discount	week / item /str	Real	On-premise	Optional	rdf_load_measures.ksh
pvarxlcirc	Promotion Variable Circular	week /item / str	Boolean	On-premise	Optional	rdf_load_measures.ksh
pvarxlisd	Promotion Variable In-Store Display	week /item / str	Boolean	On-premise	Optional	rdf_load_measures.ksh
pvarxlcovr	Promotion Variable Cover	week /item / str	Boolean	On-premise	Optional	rdf_load_measures.ksh
pvarxlmuby	Promotion Variable Multibuy	week /item / str	Boolean	On-premise	Optional	rdf_load_measures.ksh
pvarxltvad	Promotion Variable TV Ad	week /item / str	Boolean	On-premise	Optional	rdf_load_measures.ksh
pvarxlbts	Promotion Variable Back to School	week / item	Boolean	On-premise	Optional	rdf_load_measures.ksh
pvarxldml	Promotion Variable Direct Mail	day / item / str	Boolean	On-premise	Optional	rdf_load_measures.ksh
pvarxleas	Promotion Variable Easter	day / item	Boolean	On-premise	Optional	rdf_load_measures.ksh
pvarxlnyd	Promotion Variable New Years Day	day / item	Boolean	On-premise	Optional	rdf_load_measures.ksh
pvarxlsgn	Promotion Variable Signage	day / item / str	Boolean	On-premise	Optional	rdf_load_measures.ksh

Table 4–7 (Cont.) Data Files

Measure Name	Measure Label	Base Intersection	Data Type	On-premise or Cloud	Required or Optional	Loaded By
pvarxlxmas	Promotion Variable Christmas	day / clss	Boolean	On-premise	Optional	rdf_load_measures.ksh
pvarxlxmss	Promotion Variable Christmas Season	week / item	Boolean	On-premise	Optional	rdf_load_measures.ksh
promoenable	Enable Promotions	prmo / prml	Boolean	On-premise		rdf_load_measures.ksh
pvarxlbp1	Promotion Variable Boolean Promotion 1	week / item /str^4	Boolean	Cloud	Optional	rdf_load_measures.ksh
pvarxlbp2	Promotion Variable Boolean Promotion 2	week / item /str^4	Boolean	Cloud	Optional	rdf_load_measures.ksh
pvarxlbp3	Promotion Variable Boolean Promotion 3	week / item /str^4	Boolean	Cloud	Optional	rdf_load_measures.ksh
pvarxlbp4	Promotion Variable Boolean Promotion 4	week / item /str^4	Boolean	Cloud	Optional	rdf_load_measures.ksh
pvarxlbp5	Promotion Variable Boolean Promotion 5	week / item^4	Boolean	Cloud	Optional	rdf_load_measures.ksh
pvarxlbp6	Promotion Variable Holiday 1	week / item /str^4	Boolean	Cloud	Optional	rdf_load_measures.ksh
pvarxlbp7	Promotion Variable Holiday 2	week / item /str^4	Boolean	Cloud	Optional	rdf_load_measures.ksh
pvarxlbp8	Promotion Variable Holiday 3	week / item /str^4	Boolean	Cloud	Optional	rdf_load_measures.ksh
pvarxlbp9	Promotion Variable Holiday 4	week / item /str^4	Boolean	Cloud	Optional	rdf_load_measures.ksh
pvarxlbp10	Promotion Variable Holiday 5	week / item /str^4	Boolean	Cloud	Optional	rdf_load_measures.ksh
pvarxlexpp	Promotion Variable Price Discount	week / item /str^4	Real	Cloud	Optional	rdf_load_measures.ksh
Category: Causal Set Up						

Table 4–7 (Cont.) Data Files

Measure Name	Measure Label	Base Intersection	Data Type	On-premise or Cloud	Required or Optional	Loaded By
calcintxlxb	Causal Calculation Intersection	pgrp / flvl	String	On-premise	Optional	rdf_load_measures.ksh
causalextbxlxb	Causal Baseline	pgrp / flvl	String	Both	Only required for causal	rdf_load_measures.ksh
dowprof[6]	Day of Week Profile	dow	Real	Both	Optional	rdf_load_measures.ksh, rdf_e_apcro.ksh
Category: Bayesian Plan						
bayesianplan01	Bayesian Plan 01 - Final Level	week / item /str[5]	Real	Cloud	Only required for levels with Bayesian forecast method	rdf_load_bayesian_ plan.ksh
bayesianplan02	Bayesian Plan 02 - Source Level	week / scls/ str[5]	Real	Cloud	Only required for levels with Bayesian forecast method	rdf_load_bayesian_ plan.ksh
bayesianplan03	Bayesian Plan 03 - Source Level	week / pgrp/ grpd[5]	Real	Cloud	Only required for levels with Bayesian forecast method	rdf_load_bayesian_ plan.ksh
bayesianplan04	Bayesian Plan 04 - Source Level	week / item / rgn[5]	Real	Cloud	Only required for levels with Bayesian forecast method	rdf_load_bayesian_ plan.ksh
bayesianplan05	Bayesian Plan 05 - Source Level	week / dept / str[5]	Real	Cloud	Only required for levels with Bayesian forecast method	rdf_load_bayesian_ plan.ksh
bayesianplan06	Bayesian Plan 06 - Source Level	week / item/ chn[5]	Real	Cloud	Only required for levels with Bayesian forecast method	rdf_load_bayesian_ plan.ksh
Category: APC-RO Export						
apcroexptmask	Export to APC-RO Mask	item /str	Boolean	Both	Optional	rdf_e_apcro.ksh
dowprof[6]	Day of Week Profile	dow	Real	Both	Optional	rdf_load_measures.ksh, rdf_e_apcro.ksh
Category: Cross Promotion						

Table 4–7 (Cont.) Data Files

Measure Name	Measure Label	Base Intersection	Data Type	On-premise or Cloud	Required or Optional	Loaded By
appcanneff	Approved Promo Cannibalization Spreading Profile	item / rgn / iter	Real	On-premise	Only required for cannibalization cross promotions	rdf_load_measures.ksh
apphaloeff	Approved Promo Halo Spreading Profile	clss / rgn / clsr	Real	On-premise	Only required for halo cross promotions	rdf_load_measures.ksh
halochgratio	Halo Transfer Ratio	rgn / clsr	Real	On-premise	Only required for halo cross promotions	rdf_load_measures.ksh
halomaxratio	Halo Max Ratio	clss / rgn	Real	On-premise	Only required for halo cross promotions	rdf_load_measures.ksh
cannchgratio	Cannibalization Transfer Ratio	item / rgn	Real	On-premise	Only required for cannibalization cross promotions	rdf_load_measures.ksh
cannmaxratio	Cannibalization Max Ratio	item / rgn	Real	On-premise	Only required for cannibalization cross promotions	rdf_load_measures.ksh
Category: Regular Price						
promoind07xb	Promotion Indicator 07 - itm/str/week-Weekly Causal Final	week / item / str	Boolean	On-premise	Only required for regular price effects	rdf_load_measures.ksh
rdfgamma	Price Elasticity	item /str / iter	Real	On-premise	Only required for regular price effects	rdf_load_measures.ksh
rdfprice	Regular Price	week / item / str	Real	On-premise	Only required for regular price effects	rdf_load_measures.ksh
rdfpromoind	Promotion Indicators	week / item / str	Boolean	On-premise	Only required for regular price effects	rdf_load_measures.ksh
regprice07xb	Regular Price 07 - itm/str/week-Weekly Causal Final	week / item / str	Real	On-premise	Only required for regular price effects	rdf_load_measures.ksh
Category: Short Life Cycle Causal						
slcitemind	Short Lifecycle Item Flag	item	Boolean	On-premise	Optional	rdf_load_measures.ksh

Table 4–7 (Cont.) Data Files

Measure Name	Measure Label	Base Intersection	Data Type	On-premise or Cloud	Required or Optional	Loaded By
salesprice	Sales Price	week / item / str	Real	On-premise	Optional	rdf_load_measures.ksh
Category: Floating Events						
preevnindt01 through preevnindt20	Float Event <level> Indicator	week / item / str[7]	Boolean	On-premise	Optional	rdf_load_measures.ksh

Table Key

Following is the key for Table 4–7:

[1] On RDF Cloud Services, the intersection will be the same as the Intersection specified for level 01 in the RDF configuration tools plug-in.

[2] The product and location dimensions of the intersection will match the product dimension and location dimensions of the measure specified in the Clone Input field in the New Item configuration tools plug-in.

[3] The product dimension of the intersection will match the product dimension specified in the Attribute Weight Level field in the New Item configuration tools plug-in.

[4] The intersection will be the same as the Intersection specified for the corresponding promotion in the Promote configuration tools plug-in.

[5] The intersection will be the same as the Intersection specified for the corresponding level in the RDF configuration tools plug-in.

[6] The dowprof measure is used for both Causal Set Up and APC-RO export.

[7] The intersection will be the same as the Intersection specified for level 01 in the RDF configuration tools plug-in.

Historical Data

It is required that you have at least two years of historical sales to produce an optimal forecast.

Loading and Extracting Data

Data is loaded into RDF using the standard RPAS approach. For information on formatting the load data files and the utilities that enable administrators to load data into RPAS, see the *Oracle Retail Predictive Application Server Administration Guide for the Fusion Client*. Information is provided on all the batch scripts that are used for loading, processing, and extracting data.

5

RDF Configuration Process

RDF is a statistical forecasting solution that uses state-of-the-art modeling techniques to produce high quality forecasts with minimal human intervention. Forecasts produced by RDF enhance the retailer's supply-chain planning, allocation, and replenishment processes, which enables a profitable and customer-oriented approach to predicting and meeting product demand.

RDF supports pre-processing, new item/store processing, profile creation and forecast generation. To obtain good forecast results, the above features need to be configured to work together. RDF is highly configurable and extremely flexible. To streamline RDF implementation and shorten implementation time, several plug-ins are provided to work together with RPAS Configuration Tools. These plug-ins let users input configuration options through the GUI or xml files and automatically generate configuration solutions based on the RDF GA master template and user inputs. Plug-ins commonly used in RDF are Forecast Common, Prepare Demand, New Item, Curve, RDF and Promote. Another plug-in, Grade, is also included in the RDF package to generate clustering solutions. The plug-ins auto-generate the hierarchies, measures, rules, and workbook templates that are required by RDF to support the forecasting configuration entered in through the plug-in interface:

Table 5–1 Autogenerated Items from Plug-ins

Autogenerated Item	Description
Hierarchies	The DATA hierarchy will be updated with the required base internal hierarchy and dimensions.
Measures	All measures necessary to support the base solution will be created.
Rules	All Rule Sets, Rule Groups, and Rules to support the base solution will be created.
Workbook Templates	All pre-defined workbook templates to support the base solution will be created.

You may continue to make changes to the plug-in configurations, and the autogeneration process may be repeated as often as needed prior to the installation. Prior to domain build/patching, the plug-ins should also been invoked to generate the RDF internal hierarchy file and measure data files.

Note: RDF only supports global domain configuration. Simple domain configuration is not supported.

Steps to Modify the RDF or RDF Cloud Service Configuration

Implementers are expected to start with the RDF/RDFCS GA configuration and create their own RDF configuration using the steps in the following sections. If an implementer wants to start their configuration from scratch, the new configuration must have all the hierarchies in the RDF GA configuration and the hierarchies must be in the same order. The easiest way to start fresh is to copy the hierarchy.xml file from the RDF GA configuration into the blank configuration and then make modifications.

Perform these steps in the order described to configure and create an RDF or RDF Cloud Service domain:

1. Register the RDF Libraries
2. Set up the Hierarchy

   Note: This step must be skipped for RDF Cloud Service. In RDF Cloud Service, the hierarchy setup cannot be modified.

3. Set up the Partition Dimension and RDF Configuration Type
4. Set up the Common, Grouping and PrepDemandCommon Solutions

   Note: This step must be skipped for RDF Cloud Service. RDF Cloud Service does not allow customization of these modules.

5. Set up the Prep Demand Solution
6. Set up the NewItem Solution

   Note: This step can be skipped if you are not to configuring NewItem in your RDF Configuration.

7. Configure the Curve Solution

   Note: This step must be skipped if you are configuring for RDF Cloud Service. Curve is not included in the RDF Cloud Service package.

8. Configure the RDF Solution
9. Configure the Promote Solution
10. Set up the Task Flow
11. Build an RDF Domain Using the RDF Configuration

Register the RDF Libraries

Prior to configuring the RDF, register the RDF libraries to support proper validation of the RDF-specific rules. Open the Function Library Manager and add AppFunctions, LostSaleFunctions, RdfFunctions, and ClusterEngine.

For more information on how to add the function libraries, refer to the *Oracle Retail Predictive Application Server Configuration Tools User Guide*.

Set up the Hierarchy

Note: This step must be skipped for RDF Cloud Service. In RDF Cloud Service, the hierarchy setup cannot be modified.

The following step only applies to RDF on premise:

Caution: From the GA configuration, **do not**:

- Change any hierarchy name
- Change any hierarchy order
- Delete any hierarchy

For all RDF and RPAS internal hierarchies, do not change any dimension names.

Table 5–2 lists the required hierarchies for RDF.

Table 5–2 RDF Required Hierarchies

Required Hierarchy	Common Requirement
Calendar	The calendar hierarchy must have Day, Week, Day of Week (DOW) dimensions.
Relative Week	RDF internal hierarchy. Always use the GA data file. Do not edit.
Run Round	RDF internal hierarchy used by preprocessing. Always use the GA data file generated by Prepare Demand plug-in.
Product	The product hierarchy must have item and pgrp (Group) dimension. The pgrp (Group) dimension is the partition dimension of the global domain. It must only have one position in each local domain. The global domain partition must be performed on product hierarchy.
Location	The location hierarchy must have store (STR) dimensions
Admu	RPAS internal hierarchy. Do not edit.
Data	RPAS internal hierarchy. Do not edit.
Promotions	RDF internal hierarchy used by Promote. Always use the GA data file generated by Promote plug-in.
Causal Levels	RDF internal hierarchy used by Promote. Always use the GA data file generated by Promote plug-in.
Product RHS (pror)	This is a duplication of the product hierarchy. The dimensions and their roll ups have to be a mirror of the product hierarchy. The hierarchy name cannot be changed.
Lngs	RPAS internal hierarchy. Do not edit.
Time Series Grouping	The hierarchy used in grouping level forecast.
Product Attributes	The attribute hierarchy is needed for new item. The like item recommendation is generated based on product attribute.
Location RHS	This is a duplication of the location hierarchy. The dimensions and their roll ups have to be a mirror of the location hierarchy. The hierarchy name cannot be changed.

Table 5–2 (Cont.) RDF Required Hierarchies

Required Hierarchy	Common Requirement
Path	RDF internal hierarchy used by preprocessing. Always use the GA data file generated by Prepare Demand plug-in.

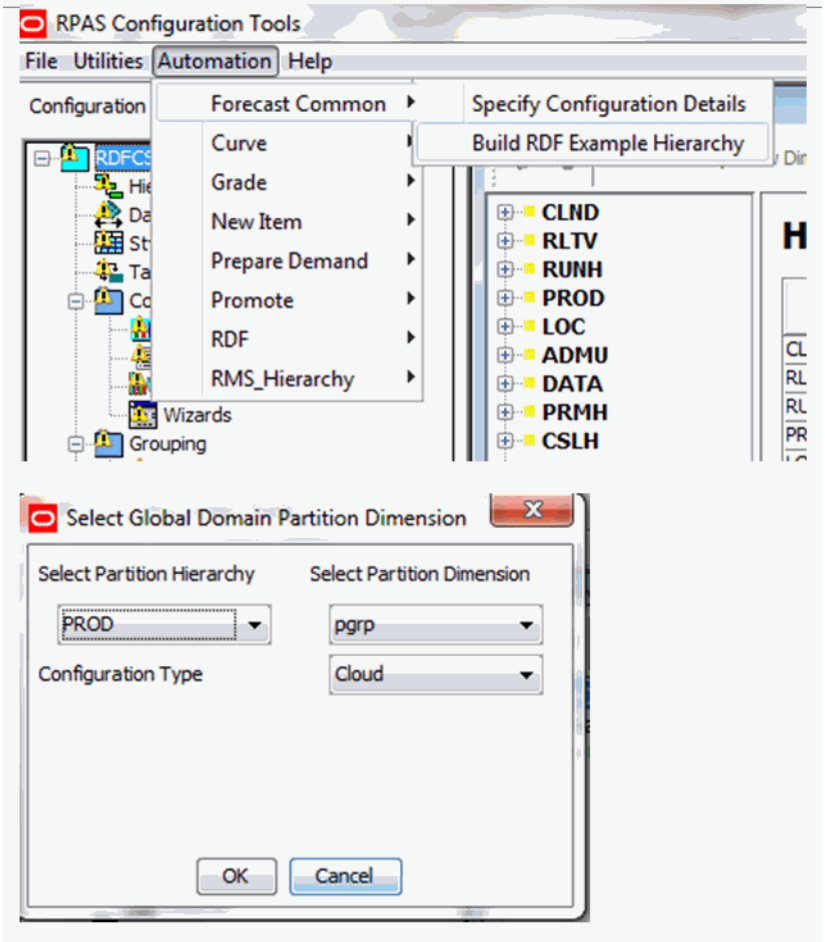
Set up the Partition Dimension and RDF Configuration Type

For this step, it is required to specify the partition dimension and configuration type through the Forecast Common plug-in:

1. From the Configuration Tools toolbar, select the Automation menu and then from the Forecast Common option, select Specify Configuration Details The partition hierarchy must be Product and the partition dimension must satisfy the requirement of one position per domain. The configuration type can be either Cloud Service or on-premise.
2. Click **OK** after setting up all inputs.

> **Note:** RDF does not support simple domain configuration. RDF must be configured for the global domain.

Figure 5–1 Configuration Tools: Forecast Common



Set up the Common, Grouping and PrepDemandCommon Solutions

Note: This step must be skipped for RDF Cloud Service. RDF Cloud Service does not allow customization of these modules.

Open a RDF GA configuration to see the common, grouping and prepdemand common modules.

The purpose of these three modules is to set up important input/output measures for the whole RDF project. The content created in these modules will not be modified by the plug-ins. The measures created in these modules are external measures for the plug-ins, and they will serve as inputs to plug-ins. It is strongly suggested that any customization to the RDF product should be done in these three modules, or you can create your own module to house the customized measures, rules, and workbook templates.

In RDF GA, the common solution is used to set up sales history inputs/outputs to:

- PrepDemand
- NewItem
- Curve
- RDF Solutions

The PrepDemandCommon solution is used to set up the optional inputs to PrepDemand solution, such as outlier indicator, out-of-stock indicator and seasonal profile.

The Grouping solution is used to set up rules to generate timeseries grouping membership if there is a forecast source level intersection containing the timeseries grouping dimension.

An implementer can modify grouping rules and criteria in this solution. The GA measures and rules in the previously listed configurations are examples for the implementer. The implementer can modify them based on customer's needs.

The plug-ins reautomation should not change the contents of these modules except when named intersections generated by the plug-ins are used.

Set up the Prep Demand Solution

The purpose of the Prep Demand module, also referred to as Preprocessing, is to correct past data points that represent unusual sales values that are not representative of a general demand pattern. Such corrections may be necessary when an item is out-of-stock and cannot be sold, which usually results in low sales. Preprocessing will adjust for stockout for both the current week and the following week because it assumes that the out-of-stock indicators represent end-of-week-stockout. Data Correction may also be necessary in a period when demand is unusually high. The Preprocessing module allows you to automatically make adjustments to the raw POS (Point of Sales) data so that subsequent demand forecasts do not replicate undesired patterns that are caused by lost sales or unusually high demand. Preprocessing can also be used to remove promotion spikes when a promotion indicator is available. Inclusion of a promotion spike can seriously skew the baseline forecasting. It is ideal to remove seasonal effects from sales so that the seasonal pattern does not interfere with the causal estimation, especially when promotion and significant seasonal patterns overlap.

Based on the usage, sales history should be preprocessed in different ways. In RDF/ RDF Cloud Service GA configuration, two preprocessing paths are configured. Path 01 is used to preprocess sales for baseline forecasting. Path 02 is used to preprocess sales for causal forecasting. For baseline forecasting, the sales history goes through four stages:

- Out-of-stock correction
- Outlier correction
- Promotional spike removal
- Smoothing

For causal forecasting, the sales history goes through three stages: out-of-stock correction, outlier correction and seasonal pattern removal. Each stage is called RUN in preprocessing.

Based on the customer's needs, an implementer can decide how many paths to configure, what kinds of runs to include in each path, and what the input and output measures are for each path. Once the information is fed to the PrepDemand plug-in, the PrepDemand configuration will be auto-generated with all necessary measures, rules and workbook templates. The rdf_preprocessing.ctl file should also be updated to reflect the preprocessing path changes. For more information about the rdf_ preprocessing.ctl file changes, refer to Chapter 7, "Batch Processing."

PrepDemand Configuration Process

The following sections describe the PrepDemand configuration process.

PreDemand Pre-configuration Data Requirement

There are several parameters within the PrepDemand configuration that may reference other measures that are configured external to the solution. Prior to configuring an PrepDemand solution, it is required that these measures already exist within the project:

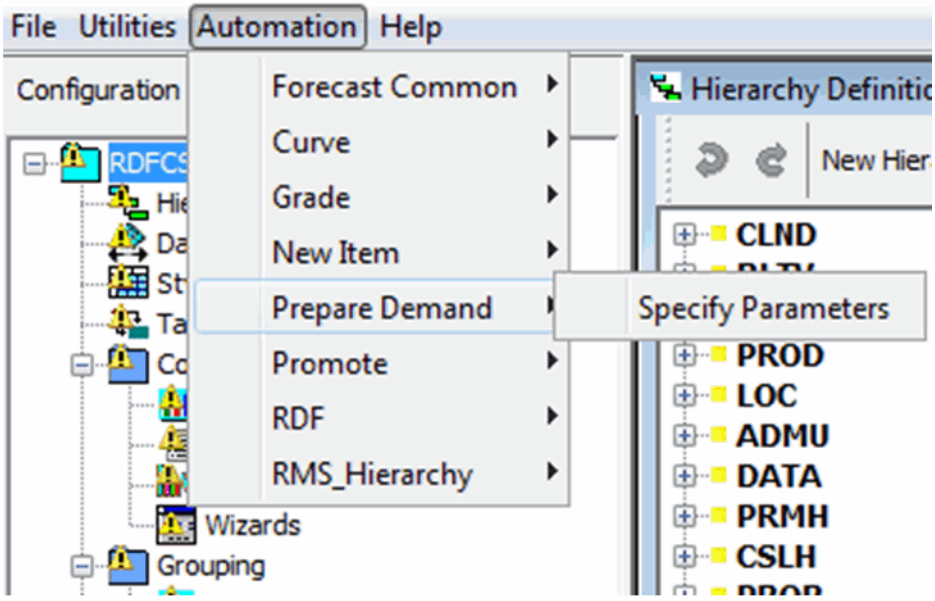
Measure	Description
Data Source	The prep demand plug-in provides a list of existing numeric measures based on the user-specified intersection for a preprocessing path. An implementer selects the measure that stores the input data for preprocessing. This measure should be configured in the Common module.
Output	The prep demand plug-in provides a list of existing numeric measures based on the user-specified intersection for a preprocessing path. An implementer selects the measure that stores the output data from the preprocess. This measure is also normally an input to NewItem/RDF plug-in. This measure should be configured in the Common module.
First Aux Measures	For each preprocessing path, the PrepDemand plug-in allows a maximum of six runs, which means six preprocessing stages (one preprocessing method per stage). Each preprocessing method may require supporting measures as inputs. Two supporting measures are allowed for each run. These supporting measures are specified in the fields of First and Second Aux measures. Examples of these measures include out-of-stock indicator or promotional indicator. These measures are external measures to PrepDemand and should be configured in PrepDemand Common.
Second Aux Measures	

Creating a PrepDemand Solution Extension

Once all input measures are configured, perform the following steps in Configuration Tools and the Preprocessing Parameters utility.

1. From the Configuration Tools toolbar, select the Automation menu and then, from the Prepare Demand option, select Specify Parameters.

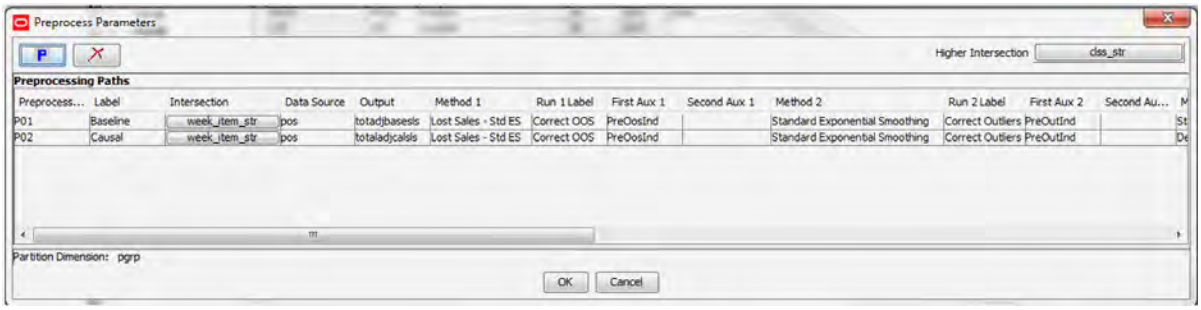
Figure 5–2 Configuration Tools: Prepare Demand



2. On the Preprocess Parameters utility, click **P**.

   A new preprocessing path is added, and it is assigned the next available level number. To specify the properties for the preprocessing path, see Edit Preprocessing Parameters for details. The Higher Intersection selection box allows you to specify the intersection of default preprocessing parameters.

Figure 5–3 Preprocess Parameters Utility



3. After finishing the configuration, click **OK** to start generating the preprocessing configuration.

Edit Preprocessing Parameters

Table 5–3 lists all of the Preprocessing Parameters. For details about preprocessing methods and associated parameters, see the Configuring the Preprocess Special Expression.

Table 5–3 Preprocessing Parameters

Preprocessing Path Parameters	Description
Preprocessing Path	The field is the system-assigned path number when a preprocessing path is created. This is a read-only parameter.
Label	The field is the level description that is viewed by the user once the domain is created.
Intersection	The intersection of the preprocessing input and output measures.
Data Source	The data source is the measure to be used as the input data (for example, POS) for the preprocessing.
Output	The output is the measure to store preprocessed result, which may serve as input to Newitem /RDF modules.
Method [n]	There are six fields for preprocessing method (method 1 through method 6). The plug-in provides a list to select a specific method each field. Each method is considered a run. The maximum number of runs allowed per path is six.
Run [n] Label	There are six fields to label preprocessing runs. One label per preprocessing method.
First Aux [n]	First Aux and Second Aux are fields to specify supporting measures per preprocessing method, such as seasonal profile, outlier indicator, outage indicator and promotion indicator. For each preprocessing method, the plug-in allows for two optional measures to be used. Some preprocessing methods need only one, others need none. If it is not needed, then leave the field empty. There are six First Aux fields and six Second Aux fields, one per method. Always populate the First Aux field first before using Second Aux. Refer to Table 5–4 for the First Aux and Second Aux supporting measures.
Second Aux [n]	

Table 5–4 lists the supporting measures for the First Aux and Second Aux preprocessing parameters.

Table 5–4 First Aux and Second Aux Supporting Measures.

Method	First Aux	Second Aux
Standard Median	not applicable	not applicable
Oracle Retail Median	not applicable	not applicable
Standard Exponential Smoothing	Outage	event flag
Lost Sales -StdES	Outage	event flag
Override	Deviation	reference
Increment	Deviation	reference
Clear	not applicable	not applicable
Deprice	Price	not applicable
Deseasonal	Seasonal Profile	not applicable

Deleting a Preprocessing Path

Deleting a preprocessing path causes the system-assigned enumerated values in the path name to renumber such that paths are in consecutive order, starting with preprocessing path 01. Deleting a preprocessing path may impact any solution configuration that uses a specific preprocessing output.

Caution: If the domain using the configuration has previously been installed, there is potential to lose data associated with a path that has been deleted or renumbered.

Perform the following steps to delete a preprocessing path:

1. On the Preprocessing Parameters utility, highlight the number of the path that you want to delete from the path window.
2. Click **X** to delete the path. The path is deleted.
3. Select **OK** to regenerate the solution with the changes to the PrepDemand configuration.

Edit PrepDemand GA Configuration

The PrepDemand autogeneration process creates all hierarchy dimensions, measures, rules and workbook template to support the essential PrepDemand functionality.

Note: It is recommended to leave the plug-in generated configuration alone and not to modify it manually.

Table 5–5 outlines the configuration restrictions:

Table 5–5 RDF GA Configuration Restrictions

Changes and Restrictions	Description
Prep Demand Solution Extension Name	The name assigned to the resulting PrepDemand solution after autogeneration occurs cannot be edited.
Major and Minor Components	Additional major components may be added to the Prep Demand. Additional minor components can only be added under the new major components. The Major and Minor components that are part of the GA configuration may not be edited. This restriction also applies to measure names and measure labels. Adding minor components to GA major components is forbidden.
Rules	Rule sets, rule groups, and rules that are part of the GA configuration may not be renamed. Existing rules that are part of the GA configuration may not be modified in any way.
Workbook Templates	New measures and rules cannot be added to the GA configuration workbook templates. No custom workbook template should be add to the module.

Set up the NewItem Solution

Note: This step can be skipped if you are not to configuring NewItem in your RDF configuration.

The NewItem module is designed to support the forecast for new item/store. RDF provides three approaches to forecast new item/store:

Forecast Approach	Description
Clone Sales History	Cloning allows users to generate forecasts for new items and locations by copying, or cloning history, from other items and stores. Users can map items or stores that have similar business cases, clone the historical data, and generate forecasts. Cloning provides the ability to generate forecasts based on historical data and promotional calendars.
Copy Forecast	Copy forecast allows users to generate forecasts for new items and locations by copying forecast/sales history from other items and stores. The new items and stores can be mapped to existing items and stores using the same approach as cloning.
Base Rate of Sales	Base rate of sales expects users to provide a sales rate per new item/store (average sales volume per time period). Seasonal pattern generated using aggregated level is applied to the sales rate to generate a seasonal forecast at item/store/week.

Both clone history and copy forecast require mapping of the new item/store. The New Item module provides tools to support the automatic and manual assignment of like item/store to new item/store. If the user can provide product attribute information, the new item can be automatically identified and provided a like item recommendation. If no product attribute information is available, the user has to assign like items manually. New store mapping is always done manually.

The implementer must decide that if a product attribute is available upfront, different workbook templates and rules are generated by plug-in based on that option.

RDF Cloud Service ships two different taskflow files with the packaged configuration:

- `taskflow.xml` for when the product attribute is available
- `taskflow.xml_noattribute` for when the product attribute is unavailable

If RDF Cloud Service is customized and configured with no attribute, then `taskflow.xml_noattribute` is renamed to `taskflow.xml`.

NewItem Pre-Configuration Data Requirements

The following describes the process of configuring NewItem.

There are two parameters within the NewItem configuration that may reference other measures that are configured external to the solution. Prior to configuring a NewItem solution, it is required that these measures already exist within the project:

- Clone Input
- Clone Output

The NewItem plug-in provides a list of existing non-string and non-boolean measures for users to select clone input and output measures.

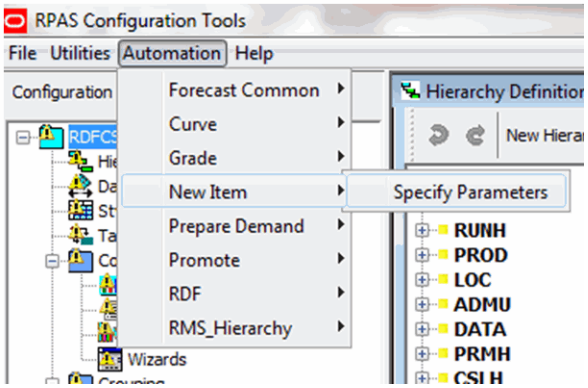
The clone input measure can be an output of PrepDemand plug-in, and the clone output measure can be the data source of RDF. These measures should be configured in Common module.

Generate New Item Solution

Perform the following steps to generate a New Item solution:

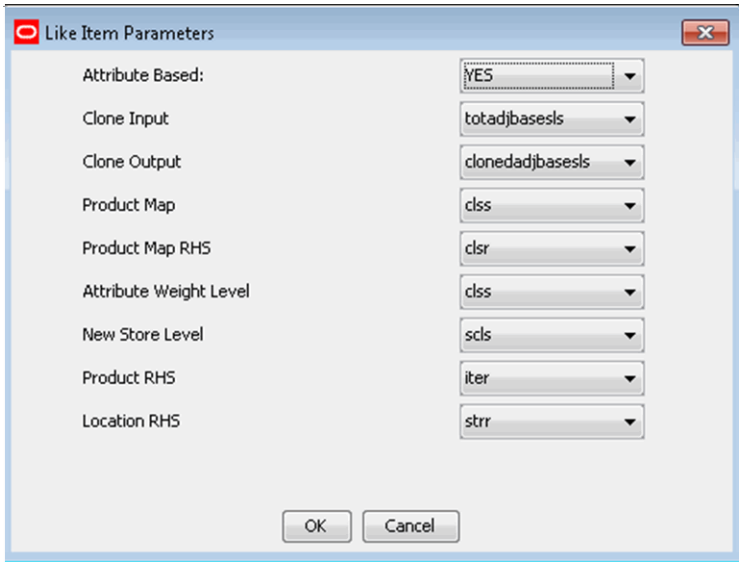
1. From the Configuration Tools toolbar, select the Automation menu and then, from the New Item option, select Specify Parameters.

Figure 5–4 Configuration Tools: New Item



2. From the Like Item Parameters utility, specify the properties for the New Item plug-in. Refer to Editing New Item Parameters for details.

Figure 5–5 Like Item Parameters



3. Click **OK** once editing is finished.

Editing New Item Parameters

Table 5–6 lists the New Item parameters available for editing.

Table 5–6 New Item Parameters

Parameter	Description
Attribute Based	This field indicates if product attribute is available.
Clone Input	This field is the input measure name for clone.
Clone Output	This field is the output measure name for clone.

Table 5–6 (Cont.) New Item Parameters

Parameter	Description
Product Map	This field specifies the range of the like item available to a new item. If the field is populated with clss, it means that only existing items under the same class as the new item are available as like item candidate. This parameter is only meaningful when product attribute is not available.
Product Map RHS	This is the corresponding Product RHS level of Product Map. If product map is clss, this field should be clsr. This parameter is only meaningful when product attribute is not available.
Attribute Weight Level	This field specifies the product level on which attribute weight is specified. If the field is set to clss, it means that the attribute weights can be different per class.
New Store Level	This field specifies the product level on which like store is assigned to new store. If the field is selected as scls, it means that the like store assignment can be different per subclass.
Product RHS	This field indicates for which product RHS level corresponds to New Item assignment's product level.
Location RHS	This field indicates for which location RHS level corresponds to new store assignment's location level.

Table 5–7 outlines the configuration restrictions:

Table 5–7 RDF GA Configuration Restrictions

Changes and Restrictions	Description
NewItem Solution Extension Name	The name assigned to the resulting NewItem solution after autogeneration occurs cannot be edited.
Major and Minor Components	Additional major components may be added to the Prep Demand. Additional minor components can only be added under the new major components. The major and minor components that are part of the GA configuration may not be edited. This restriction also applies to measure names and measure labels. Adding minor components to GA major components is forbidden.
Rules	Rule sets, rule groups, and rules that are part of the GA configuration may not be renamed. Existing rules that are part of the GA configuration may not be modified in any way.
Workbook Templates	New measures and rules cannot be added to the GA configuration workbook templates. No custom workbook template should be add to the module.

Configure the Curve Solution

Note: This step must be skipped if you are configuring for RDF Cloud Service. Curve is not included in the RDF Cloud Service package.

This step is only necessary if RDF needs to provide seasonal profile, spread profile, and so on using Curve. For RDF Cloud Service and most RDF configurations, this step can be skipped. For details about Curve solution setup, refer to Appendix G, "Curve Configuration Process."

Configure the RDF Solution

Forecast information is often required for items at the lowest levels in a hierarchy. Problems can arise when historic sales data for these items is too sparse and too noisy to identify clear selling patterns. In such cases, generating a reliable forecast requires aggregating sales data from a low level up to a higher level in the hierarchy. After a forecast is generated at the higher level, the resulting data can be allocated (spread) back down to the lower level. This is based on the lower level's relationship to the total at the aggregated level. Before the forecast data can be spread back down to a lower level, there must be an understanding of the relationship between the lower level and the higher level dimensions. Frequently, an additional forecast is generated at the low level to help determine this relationship. This low level is called the final forecast level. Forecast data at this level might be sufficient to generate reliable percentage-to-whole information, but the actual forecast numbers are more robust when they are generated at an aggregate level. This aggregate level from which forecast data is spread is referred to as the source forecast level.

Some high-volume items may possess sufficient sales data for robust forecast calculations directly at the final forecast level. In these cases, forecast data that is generated at an aggregate level and then spread down to lower levels can be compared to forecasts that are run directly at the low level. Comparing the two forecasts, each generated at a different hierarchy level, can be an invaluable forecast performance evaluation tool.

For causal forecasting, simple aggregation from lowest to aggregated level does not work because it is unrealistic to expect the same promotion activity schedule for every item/store under an aggregated level. The way to tackle the problem of sparse data is to chain the sales data and promotion calendar at each item/store per aggregated intersection. In this way, up to thousands of time series can be pooled together to generate averaged promotional effects at the aggregated level. The effects calculated at aggregated level are more reliable than the one calculated at final level. The effects at source level and final level can be combined with a user-specified weight to produce a final estimate. For a high volume item, the final level calculated promotional effects can have more weight than that of low volume items.

The RDF solution may include multiple final forecast levels. Different final forecast levels should have different purposes and parameters. In RDF CS configuration, two final forecast levels were configured. One is used to generation baseline forecast. The approved forecast from baseline forecasting serves as external baseline to another final level, which is configured to produce causal forecast. Forecast results must appear at some final level for the data to be approved and exported to other systems.

Using the RDF plug-in, final and source forecast levels are defined for the RDF solution.

Forecasting Pre-Configuration Data Requirement

There are several parameters within the RDF configuration that may reference other measures that are configured external to the solution. Prior to configuring an RDF solution, it is required that these measures already exist within the project.

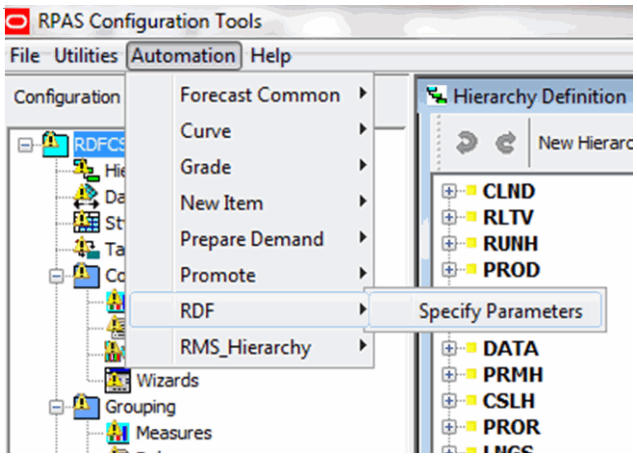
Parameter	Description
Source Data	The RDF plug-in populates a list with all non-Boolean and non-string measures that have been created in the project. This field is normally populated with the output of preprocessing or NewItem. If the level is a source level, and the source data is the same as the final level source data, then leave the source data empty. It will automatically aggregate from the final level data source. If you would prefer to use data input that is different from the aggregation of final level source data, then you can place a measure name in data source for that source level. This measure should be at the same intersection as the source level. If the source level is an HBI level, then the measure must be an `fnhbi` measure.
Plan Data	This measure should already exist if the Plan Data to be used to support Bayesian forecasting is being defined within another solution. The entry of this parameter is not required within the configuration, and it can be entered in the resulting domains.
Spreading Profiles Seasonal Profiles	If Curve is used to produce Spreading Profiles or Seasonal Profiles to support your Forecasting solution, these profiles should already have been configured in the Curve solution. If these profiles are being defined external to Curve, these measures should already exist within the project.

Configure RDF Solutions

Perform the following steps to generate an RDF solution.

1. From the Configuration Tools toolbar, select the Automation menu and then, from the RDF option, select Specify Parameters. The following steps outline the process for configuring RDF forecast levels.

Figure 5–6 Configuration Tools: RDF



2. Configure a forecast level:

   a. To configure a final forecast level:

      From the Forecasting Parameters utility, click the **F** icon. A new final level is added, and it is assigned the next available level number. Specify the properties for the final level. See Editing Forecast Level Parameters for details. For RDF Cloud Service, skip the columns that do not apply.

   b. To configure a source forecast level:

From the Forecasting Parameters utility, highlight the final level number in which the new source level will be associated from the Level window and then click the **S** icon. A new source level is added, and it is assigned the next available number. Specify the properties for the source level. See Editing Forecast Level Parameters for details.

Figure 5–7 Forecasting Parameters Utility

Editing Forecast Level Parameters

Table 5–8 lists the forecast level parameters available for editing.

Note: For more information on Source Level Forecasting, see the *Oracle Retail Demand Forecasting User Guide*.

Table 5–8 Forecast Level Parameters

Parameter	Applies to RDF Cloud Service	Description
Level Name	yes	The level name is the system-assigned level number when a forecast level is created. This is a read-only parameter.
Level Label	yes	The level label is the level description that will be viewed by the user once the domain is created. Level labels may not exceed 40 characters. It is recommended, but not required, that Level labels include the Level Name (the system-assigned level number). Within the Forecast Administration workbook, the Default Source Level may be edited. This list is populated with the Level Name for all levels that are associated with a final level. Since this value can also be specified within this configuration, this recommendation may not be necessary if changes to the Default Source Level are not expected within the application. RPAS automatically places parentheses ( ) around Forecast Level labels. The configuration specialist should not include these in the level label configuration or the installer will fail. An example of a Forecast Level label that would violate this requirement is (1:itm/str/week - Final). This example is acceptable: 1- item/str/week - Final. A hyphen '-' should not be used before or after the Forecast Level label. An example of a Forecast Level label that would violate this requirement is: -1:itm/str/week - Final-. This example is acceptable as: 1-itm/str/week – Final A colon ':' should not be used at all in the Level label. An example of a Level label that would violate this requirement is 1: itm/str/week-
Intersection	yes	The intersection is the hierarchy dimensions that define the forecasting level. If the product dimension is on the main trunk of the product hierarchy, it must be lower than the partition dimension. If the product dimension is on the alternative roll up of the product hierarchy, all the measures related to this level will be created as forced -non HBI measure.
Source Data	yes	Assigned only at the final level, the source data is the measure to be used as the input data (for example, POS) for the generation of forecasts. The values in this list are populated with all non-string and non-Boolean type measures that are configured in the project.
Periodicity	yes	Periodicity is the number of periods within the Calendar dimension, which are defined in the forecast level intersection. For example, if an intersection is defined at Week/item/store, the Periodicity value will be 52 (since there are 52 weeks within a year).
Forecast Method	yes	The Forecast Method window displays all forecast generation methods that may be defined for a forecast level. The Default Forecast Method is also determined here. When a level is causal level (causal enabled), no other method except `no forecast` can be defined. For additional information, see Selectable Forecast Methods.
Default Source Level	yes	Assigned only at the Final level, the Default Source Level is the primary level at which the aggregate, more robust forecast is run. The desired Source Level must first be created within the RDF configuration for it to be a selection in the list. For more information on Source Level Forecasting, refer to the *Oracle Retail Demand Forecasting User Guide*. If no source level is required, the final level should be selected.

Table 5–8 (Cont.) Forecast Level Parameters

Parameter	Applies to RDF Cloud Service	Description
Plan Data	yes	Assigned only at the final level, Plan Data (sales plans) provide details of the anticipated shape and scale of an item's selling pattern. Although not assignable at the source level, Plan Data can be specified for source level in forecast administration workbook. It is used in the same way as in the final level.This information is required when Bayesian or Load Plan is used as a Forecast Method. The value in this parameter is a measure name.
Group Assignment Measure	yes	Assigned only at Final level, the Group Assignment measure contains the measure that is used to store the time series grouping. After the domain build, the values (specified measure names) for the Group Assignment measure are stored in a measure.
Seasonal Profile	yes	A seasonal profile provides details of the anticipated seasonality of an item's selling pattern. The seasonal profile is required in conjunction with the Profile-based Forecast Method. The seasonal profile can be generated or loaded, depending on your configuration. The value in this parameter is a measure name.
Promo Lift Override Intersection	yes	Assigned only at the final level, the Promo Lift Override Intersection window displays the intersection level allowed to override the promotion lift.
Halo Spreading Profile Source	no	Assigned only at the final level, the Halo Profile Spreading Source specifies halo spreading profile used in calculating the halo promotion lifts. The value in this parameter is a measure name. For more information on the measure, refer to the section, Files Needed to Build the RDF Domain
Promo Halo Change Ratio Source	no	This measure is specified at an intersection higher than item/store. The content of the measure is visible if you roll up to All Products on the product hierarchy. At a lower intersection. the cell displays as a hash mark. This parameter contains the name of the measure that determines the percentage of the promotional lift that is going to increase demand of complimentary items due to the halo effect.
Promo Halo Max Change Ratio Source	no	This measure is specified at an intersection higher than item/store. The content of the measure is visible if you roll up to All Products on the product hierarchy. At a lower intersection, the cell displays as a hash mark. This parameter contains the name of the measure that determines an item's maximum allowed increase in sales due to halo. For instance, if the sales of an item for a given period are 15 units, and the maximum allowed percentage is 20%, the increase in sales due to halo for the period cannot exceed three units.
Cannibalization Spreading Profile Source	no	Assigned only at the final level, the Cannibalization Profile Spreading Source specifies the cannibalization spread profile used in calculating the cannibalization promotion lifts. The value in this parameter is a measure name. For more information on the measure, refer to the section,Files Needed to Build the RDF Domain.
Promo Cannibalization Change Ratio Source	no	This measure is specified at an intersection higher than item/store. The content of the measure is visible if you roll up to All Products on the product hierarchy. At a lower intersection, the cell displays as a hash mark. This parameter contains the name of the measure that determines the percentage of the promotional lift that is going to cannibalize related items.

Table 5–8 (Cont.) Forecast Level Parameters

Parameter	Applies to RDF Cloud Service	Description
Promo Cannibalization Max Change Ratio Source	no	This measure is specified at an intersection higher than item/store. The content of the measure is visible if you roll up to All Products on the product hierarchy. At a lower intersection, the cell displays as a hash mark. This parameter contains the name of the measure that determines an item's maximum allowed drop in sales due to cannibalization. For instance, if the sales of an item for a given period are 20 units, and the maximum allowed percentage is 20%, the drop in sales due to cannibalization for the period cannot exceed four units.
Price Elasticity Data	no	Assigned only at the final level, the Price Elasticity Data specifies the measure to be used as the input data (for example, rdfgamma) to calculate the regular price change lifts. The value in this parameter is a measure name. For more information on the measure, refer to the section,Files Needed to Build the RDF Domain.
Enabled Demand Transference	no	Assigned only at the final level, the Enabled Demand Transference window defines if demand transference is enabled. If the check box is not enabled, demand transference function will not be applied to generate the forecast.
Extra Week Indicator	yes	Assigned at both source level and final levels. The extra week indicator field should be populated with a measure name. This boolean measure indicates which week is the 53rd week. If the field is empty, then the 53 week processing is unavailable for that level.
Return Forecast Parameters Intermediate Intersection	yes	Assigned at the final level. This field should be populated with an intersection string. It is used to indicate the intermediate level of return forecast parameters. The return forecast parameters can be specified on three levels, default, intermediate and override. If this field is empty, then return forecast functionality is unavailable for this level.
Forecast Alert Intermediate Intersection	yes	Assigned at the final level only. This field holds an intersection string that indicates the alert parameters' intermediate intersection. The alert parameters can be specified on three intersections: default, intermediate and override. When this field is empty, all GA alerts in the level are unavailable.
Spreading Profile	yes	Assigned only at the source forecasting level, the Spreading Profile is used to spread source level forecasts down to the final forecast level. The value in this parameter is either empty, a measure name, or several measure names separated by commas. When the value is several measure names separated by commas, it is these measures that are multiplied together to create the spreading profile., separated by commas. If Curve is used to generate the static (manually approved) spreading ratios, this parameter should be populated with the Approved Profile measure. For example: apvp11 (this is the Approved Profile for Curve level 11).

Selectable Forecast Methods

The following is a list of forecast methods that may be selected. See the *Oracle Retail Demand Forecasting User Guide* for more information on each method.

- No Forecast
- Average
- Moving Average
- Simple

- Intermittent
- Simple/Intermittent
- Trend
- Additive Seasonal
- Multiplicative Seasonal
- Seasonal
- AutoES
- Causal

> **Note:** See About Causal Forecasting for special conditions for Causal methods.

- Bayesian
- Profile-based
- Load Plan
- Copy
- Moving Average
- Components

About Causal Forecasting

The Causal method should be selected as a valid method only for levels in which causal forecasting will be used.

When enabling Causal as a valid forecast method for a source level, note that RDF Promotion variables need to be provided at the same dimension, along the product and location hierarchies as the forecast level for which Causal forecasting is run (Final or Source). RDF Causal does not support aggregation of promotion variables along any hierarchies other than Calendar (Clnd). Aggregation of promotion variables along either or both of product or location hierarchies needs to be handled externally through configuration. Aggregation along the calendar hierarchy is support by RDF Causal, using specified aggregation and spread profiles. Refer to the *Oracle Retail Demand Forecasting User Guide* for details.

Deleting a Forecast Level

Deleting a forecast level will cause the system-assigned enumerated values in the Level Name to renumber such that levels are in consecutive order, starting with forecast level 01. Deleting a forecast level may impact any solution configuration that uses a specific level.

> **Caution:** If the domain using the configuration has previously been installed, there is potential to lose data associated with a level that has been deleted or has been renumbered.

Perform the following steps to delete a forecast level:

1. From the Forecasting Parameters utility, highlight the number of the level that you want to delete from the Level window.

2. Click **X**. The level is deleted. If you delete a final level, any source levels that are associated with it will also be deleted.
3. Click **OK** to regenerate the solution with the changes to the RDF configuration.

Edit the RDF GA Configuration

The autogeneration process creates hierarchies, measures, rules, and workbook templates that are required to support the essential RDF functionality. This base configuration is referred to as the GA Configuration. Certain changes to the GA Configuration are allowed. Once edits to the GA Configuration are made and the autogeneration process occurs again, valid changes to the configuration will be preserved. There is nothing in the RPAS Configuration Tools to prevent invalid changes from being made.

Table 5–9 outlines acceptable changes and restrictions:

Table 5–9 RDF GA Configuration Restrictions

Changes and Restrictions	Description
RDF Solution Extension Name	The name assigned to the resulting RDF solution after autogeneration occurs cannot be edited.
Major and Minor Components	Additional major components may be added to the Prep Demand. Additional minor components can only be added under the new major components. The major and minor components that are part of the GA configuration may not be edited. This restriction also applies to measure names and measure labels. Adding minor components to GA major components is forbidden.
Rules	Additional rule sets, rule groups, and rules may be added to the RDF GA configuration. This includes support for adding new rules to existing GA configuration rule groups. It is recommended that new rules added to the GA configuration rule groups include CUST (represents Custom) in the rule name. This allows for easy identification of rules that are not part of the GA configuration. Rule sets, rule groups, and rules that are part of the GA configuration may not be renamed. Existing rules that are part of the GA Configuration may not be modified in any way.
Workbook Templates	Additional workbook templates may be added to the RDF GA configuration. New measures and rules may also be added to the GA configuration workbook templates. This is done by adding new major and minor components, and adding new rules to existing rule groups in the GA configuration.

Configure the Promote Solution

Promote (Promotional Forecasting) is an optional add-on solution to RDF that allows for the setup of promotional and causal events, such as radio advertisements and holiday occurrences. It is only necessary to configure the promote solution if a forecast level setup in the RDF solution is causal enabled. Past sales data and promotional information are used together to forecast future demand.

Using the Promote plug-in, promotions are defined to be used within the Promote solution.

Create the Promote Solution Extension

Perform the following steps to create the Promote solution extension:

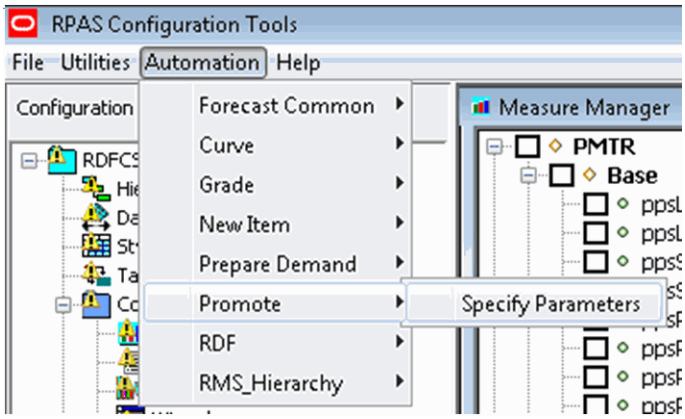
1. Open an existing configuration in which the RDF solution has already been defined.

   Note: Promote Automation must be run last - after RDF.

   Note: Promotion/causal forecasting levels are determined within the RDF Solution by selecting Causal as a valid Forecasting Method for source or final forecasting levels.

2. From the Configuration Tools toolbar, select the Automation menu and then, from the Promote option, select Specify Parameters.The following sections outline the process for configuring forecast levels.

Figure 5–8 Configuration Tools: Promote



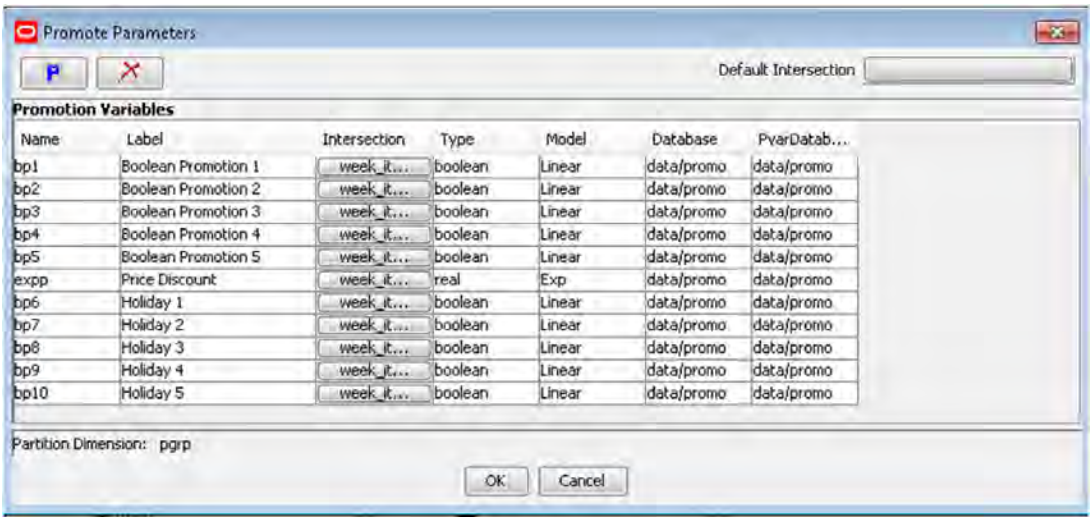
Create a Promotion

Perform the following steps to create a promotion:

1. From the Promote Parameters utility, click the **P** icon.

   A new promotion is added, and it is assigned a default promotion number for the Promotion Name (for example, P001).

Figure 5–9 Promote Parameters Utility



2. Specify the properties for the promotion. See Editing Promote Parameters for details.

Editing Promote Parameters

Table 5–10 lists the promotion parameters available for editing.

Table 5–10 Promote Parameters

Parameter	Description
Default Intersection	The Default Intersection is the intersection at which any new promotion will be defined. Editing the Default Intersection will not affect any existing promotions.
Promotion Name	The Promotion Name is the internal system identifier of the promotion. The system will initially assign a generic Promotion Name (P001), but this value may be overwritten. The Promotion Name may not be greater than four characters. The following characters may not precede or follow the name that is entered in this field: '( )' Example: (xmas) '-' Example: -xmas- The following must not be used at all in the Promotion Name: ':' Example: xmas:
Promotion Label	The Promotion Label is the description of the promotion that will be viewed by the user once the domain is created. Promotion Labels may not exceed 40 characters. The following characters may not precede or follow the label that is entered in this field: '( )' Example: (xmas) '-' Example: -xmas- The following must not be used at all in the Promotion Name: ':' Example: xmas:

Table 5–10 (Cont.) Promote Parameters

Parameter	Description
Promotion Intersection	Independent of the causal forecasting levels, the Promotion Intersection is the hierarchy dimension that defines the promotion. It is pre-populated with the value set in the Default Intersection at the time when the promotion is created.
Type	The Type is the data type of the promotion variable. Promotion Variables may be defined as Boolean or Real types. The value in this parameter defaults to Boolean.
Model	Model is the model type that the promotion variable is applied into. Model Types maybe defined as Linear or Exp. (Exponential). The value in this parameter defaults to Linear.
Database	The Database displays the database that will be used to store promotion variable information. The value in this parameter defaults to the data/promo database.
PvarDataBase	The PvarDataBase is the database used to store promotion variable information. The value in this parameter defaults to the data/promo database.

Setting Promotion Variable Type

Promotion Variable Type is defined through setting both Type and Model.

Note: Real Exponential and Real Linear promotion variables cannot be enabled at the same time for a given forecast level. Boolean with either Real Linear or Real Exponential promotion variables, is allowed.

When the Type is	And the Model is	Then the Promotion Variable Type is
Boolean	Linear	Boolean
Real	Linear	Real
Real	Exponential	Exponential

Note: After autogeneration completes, the following rules display as invalid; however, these should be ignored:

- PREF_PlHolder RuleGroup
- PREF_place Rule Group
- PRMA_place Rule Group
- PRPL_place

Deleting a Promotion

Perform the following steps to delete a promotion:

1. From the Promote Parameters utility, highlight the promotion that you want to delete from the configuration.
2. Click **X**. The promotion is deleted.

3. Click **OK** to regenerate the solution with the changes to the promotion configuration.
4. Patch the domain with the new configuration.

Edit the Promote GA Configuration

The Promote autogeneration process creates all hierarchy dimensions, measures, rules and workbooks to support the essential Promote functionality. The Promote plug-in allows for fewer options than in RDF for edits to the GA Configuration.

Table 5–11 outlines acceptable changes and restrictions:

Table 5–11 Promote GA Configuration Restrictions

Changes and Restrictions	Description
Promote Solution Extension Name	The name assigned to the resulting Promote solution after autogeneration occurs cannot be edited.
Major and Minor Components	Additional major components may be added to the Prep Demand. Additional minor components can only be added under the new major components. The major and minor components that are part of the GA configuration may not be edited. This restriction also applies to measure names and measure labels. Adding minor components to GA major components is forbidden.
Rules	Rule sets, rule groups, and rules that are part of the GA configuration may not be renamed. Existing rules that are part of the GA Configuration may not be modified in any way.
Workbook Templates	New measures and rules cannot be added to the GA configuration workbook templates.

Set up the Task Flow

The process of setting up task flow is roughly same in a RDF project as any other RPAS based product. There is a major difference for a configuration with an existing taskflow settings. When taskflow exists in the configuration, the plug-in automations may cause the workbook template field in the taskflow to be empty.

RDF Cloud Service ships two different taskflow files with the packaged configuration:

- `taskflow.xml` for when the product attribute is available
- `taskflow.xml_noattribute` for when the product attribute is unavailable

If RDF Cloud Service is customized and configured with no attribute, then `taskflow.xml_noattribute` is renamed to `taskflow.xml`.

It is recommended to perform the automation in the following steps to preserve the taskflow:

1. Save a copy of `taskflow.xml` in an existing configuration and open the configuration with RPAS ConfigTools.
2. Run automation of all necessary plug-ins, save the configuration and close it.
3. Copy the backed up `taskflow.xml` back into the configuration.
4. Reopen the saved configuration and make appropriate adjustment to the taskflow or other components.

Build an RDF Domain Using the RDF Configuration

The process of building a RDF domain using the RDF/RDF Cloud Service configuration is about the same as any other RPAS based applications. However, for pre-domain installation, it is necessary to generate the RDF internal hierarchy files and RDF internal measure data files through the plug-ins.

For domain patching, these internal data files need to be copied manually into the RPAS domain's input directory. The script, `rdf_build_domain.ksh` has been coded to take care of the previously listed details. It should be used for both domain building and domain patching.

If an implementer wishes to use their own custom scripts, ensure all necessary steps in `rdf_build_domain.ksh` are included in the custom scripts. For more information, refer to "rdf_build_domain.ksh" on page 7-42.

Best Practices of RDF Implementation: RDF Cloud Service

RDF Cloud Service configuration demonstrates the best practices that Oracle suggests when implementing RDF:

- Preprocessing is set up to produce different data sources for baseline forecasting and causal estimation.
- The number of forecast levels are limited to ten with a maximum of two final levels in RDF cloud configuration. The RDF plug-in will throw an error if this is broken. Baseline forecasting is configured on final level 1, with multiple source levels. The source levels are either on aggregated prod/location/week or custom-group/week. The custom-group assignment is provided by the user and is supposed to be loaded.
- In RDF Cloud Service, the number of promotions is limited to five Boolean promotions, one exponential promotion and five holiday/floating-events.The Promote plug-in will throw an exception if more promotions are configured for cloud-type configuration. Going beyond 11 causal variables does not show significant improvement in promotion forecast accuracy. Promotion effects estimation are set up to estimate on both final and source levels. The source level pooled estimation can leverage either aggregated PROD/LOC or custom-groups of item/stores. The effects from individual item/store and pooled estimations can be merged before applying to baseline.

Sending a Customer's Configuration to Oracle OCI for Domain Building and Patching

In RDF Cloud Service implementation, the configuration name must be RDFCS. It cannot be changed. An implementer is only allowed to perform the following steps:

1. Set up partition hierarchy and dimension.
2. Configure PrepDemand solution through the plug-in.
3. Configure NewItem solution through the plug-in.
4. Configure RDF solution through the plug-in.
5. Configure Promote Solution through the plug-in.
6. Configure the taskflow.

Configuration Files to Send

Oracle OCI does not require the whole configuration to be sent. OCI only needs the following four files from the configuration:

1. `rdf_plugins_<date>.tar.gz`
   a. This file should be a tar and gzip of the entire `RDFCS/plugins` folder. Ensure the `plugins` folder contains the following directories:

      Common

      PrepDemand

      NewItem

      RDF

      Promote

   b. To create this file, change directory to the `RDFCS` folder and issue the following command (where `<date>` is the current date):

      ```
      tar -cvf - plugins | gzip > rdf_plugins_<date>.tar.gz
      ```

2. `rdf_hiers_<date>.tar.gz`
   a. This file should be a tar and gzip of the required hierarchies to build the initial RDF domain. Ensure it contains the following files:

      clnd.csv.dat

      grph.csv.dat

      loc.csv.dat

      prod.csv.dat

      attr.csv.cat (If the New Item plug-in was configured to use attributes)

   b. There should be no directories or folders in the `rdf_hiers_<date>.tar.gz` file.
3. The modified `rdf-preprocess.ctl` (must be located in `$RPAS_HOME/bin`)
4. The modified `RDFCS/taskflow.xml`

Note: OCI uses these files and directories to regenerate the configuration and build/patch the domain. The `rdf-preprocess.ctl` is used for batch setup.

6

Advanced RDF Configurations

This chapter describes the advanced RDF configurations including:

- Daily Causal Implementation
- Configuring RDF for Short Life Cycle Merchandise
- Floating Annual Events and Holidays forecasting in RDF-lite

Daily Causal Implementation

Configuring Daily Causal, as opposed to using Daily Demand, has these advantages:

- Creates a more robust forecast
- Saves disk space by avoiding measures at the day dimension
- Performs much better in terms of runtime

To configure RDF for Daily Causal implementation, complete the following steps.

1. Create a Daily Final Causal level.
2. Set both Causal Calculation Intersection and Causal Data Source at the week level.
3. Causal External baseline is provided at week level.
4. Promotions can be provided at day or week level.
5. Provide a week to day profile (based on DOW) to spread external baseline to day level prior to system forecast output. If no spread profile is provided, the default of 1/7 is used. The profile can be created using a few weeks of daily data.
6. The same week to day profile (based on DOW) can be used to aggregate promotion from day to week. If no aggregation profile is provided, the default of 1/7 is used.

> **Note:** Although the forecast is at the day level, the majority of the historical inputs are at weekly level. The daily measures are possibly some promotions (which are sparse), and very few weeks worth of daily demand to generate the DOW profile.

Configuring RDF for Short Life Cycle Merchandise

RDF is proven to work well for items with long life cycles. It also offers methods to forecast short life cycle, but these methods require additional input, other than the

historical demand. This input is a life cycle profile which is a pre-season estimate of how the items will sell.

Once the item starts selling (in-season), the life cycle is combined with the actual sales to create the forecast.

To configure RDF for Short Life Cycle Merchandise, complete the following steps.

1. Calculate a baseline forecast:
    a. If price information is available, use the deprice filter to deprice historical demand
    b. If promotion information is available, use the STD ES filter to depromote historical demand
    c. After demand has been cleaned, it can be used to create a life cycle profile.

       You can configure rules to achieve a life cycle profile, or use the Curve solution. In Curve, you specify which part of the history should be used, the intersection at which the profile is created, and the periods in the future, when the items will sell. Curve will handle the stretching/compressing and time-shifting of the profile to periods in the future. This profile can be transformed to units if the total buy for the items is known. It is common for short life cycle items to order a certain quantity, that sells over a certain period. The total quantity multiplied by the profile gives the number of units sold every period.

       This represents the pre-season forecast.
    d. Finally, the profile-based or Bayesian forecasting methods, combine the life cycle profile or the pre-season forecast with actual sales, to generate the in-season short life cycle forecast.

    If the merchandise is promotional, use Causal Effect Estimation to be configured to generate a causal forecast
2. Use Causal Effect Estimation:

    Use base RDF Causal functionality for data pooling.

    Data pooling allows promotion effects estimation at an agggregate level. The level/intersection can be the same as the level used to generate the life cycle curve in the previous steps. These effects are used for the new short life cycle merchandise.
3. Generating a Causal Forecast:

    Combine the baseline and promotional lifts calculated in Steps 1 and 2, respectively, to generate the final forecast.

Note: The aggregation level for generating the life cycle profile and the causal effects works best when it includes item/locations which share similar selling patterns. RDF has the ability to group item/locations based on attributes, and not always be tied to the hierarchy dimensions.

For example, you can write rules to group together all item/locations that started selling in the winter season and had a life cycle of three months. The life cycle profile created using these items, is probably more accurate (in terms of life cycle) than a profile created at, for instance, department level.

Floating Annual Events and Holidays forecasting in RDF-lite

In an RDF causal implementation, an annual event that does not occur in the same period every year, but has a spike in demand associated with it, is handled by associating a causal factor to it. The system then determines the associated lift, and applies it to the relevant point in time in the forecast horizon.

In an RDF-lite implementation, where there is no Promote module, these events may still exist. For example, an increase for sales leading to Easter, happens even if there is no promotion specifically associated with Easter (chocolate bunnies sell more during Easter even though they are not specifically promoted). Since Easter does not occur at the same time every year, the Easter spike appears randomly any time in March or April. The spike is baked in the seasonality of each item/store, making its prediction; timing and magnitude, inaccurate at best. RDF-lite can support floating events and holidays through extra configuration efforts. RDF GA configuration has configured an example to show how to support floating events forecasting without enabling the Promote module.

To configure RDF-lite for Floating Annual Events and Holidays forecasting, complete the following steps.

Note: Step 1 should be run whenever there is an update on floating event indicators.

Steps 2 and3 should be run weekly.

1. Twenty (20) floating event indicators are configured as Boolean measures at Item/Store/Week in PrepDemandCommon module. These indicators are expected to be loaded.

   Combined Weekly Floating Event lift Estimation is configured as a rule group named gen_floatlift in the PrepDemandCommon module. In this rule, the CausalEstimation special expression is used to estimate the floating event effects at Item/Store level using preprocessed sales history and floating event calendar.

   The input sales is corrected for out of stock, outliers and seasonal effects.

   The forecast150 special expression is used to transform the floating event effects at item/store into a combined weekly floating event lift. The lift ratio is 1 in normal week and it will be more or less than 1 when there is a floating event.

   This step can be invoked through running rdf_gen_float_lift.ksh

2. Configuring Baseline forecast in RDF. (Level 1 in RDF/RDFCS configuration). The input sales history is corrected for out of stock, outlier and promotion spike. Several source level are set up for the final level.
3. Configuring a final level (in RDF configuration, level 8) to run component based forecast. Refer to the *Oracle Retail Demand Forecasting User Guide* for more information about the component forecast method. The component baseline is pointing to the approved forecast from level 1. The component promotional lift is mapped to the combined floating weekly event lift calculated in Step 1.

7

Batch Processing

This chapter describes the various batch scripts and executable files provided by RDF and RDF Cloud Service.

Note: With this release, some script names have been changed. For more information, refer to Appendix J, "RDF Script Names."

Available Scripts

Table 7–1 shows which scripts are available for RDF and RDF Cloud Service.

Table 7–1 RDF and RDF Cloud Service Available Scripts

Script	RDF	RDF Cloud Service	Frequency
cpem_batch.ksh	Yes	No	Ad hoc
cpem_build_domain.ksh	Yes	No	Ad hoc
cpem_e_rdf.ksh	Yes	No	Ad hoc
cpem_load_measures.ksh	Yes	No	Ad hoc
rdf_auto_gen_config.ksh	Yes	Yes	Ad hoc
rdf_build_domain.ksh	Yes	Yes	Ad hoc
rdf_e_apcro.ksh	Yes	Yes	Ad hoc
rdf_e_cpem.ksh	Yes	No	Ad hoc
rdf_gen_float_lift.ksh	Yes	No	Ad hoc
rdf_gen_halo_lift.ksh	Yes	No	Ad hoc
rdf_load_bayesian_plan.ksh	No	Yes	Ad hoc
rdf_preprocess_dt.ksh	Yes	No	Ad hoc
rdf_repos.ksh	Yes	No	Ad hoc
rdf_upgrade_new_item_store_eport.ksh	Yes	No	Ad hoc
rdf_upgrade_new_item_store_load.ksh	Yes	No	Ad hoc
rdf_e_apcro_weekly.ksh	Yes	Yes	Supporting Scripts
rdf_environment.ksh	Yes	Yes	Supporting Scripts
rdf_functions.ksh	Yes	Yes	Supporting Scripts
rdf_batch.ksh	Yes	Yes	Weekly

Table 7–1 (Cont.) RDF and RDF Cloud Service Available Scripts

Script	RDF	RDF Cloud Service	Frequency
rdf_clone.ksh	Yes	Yes	Weekly
rdf_e_aip_appf.ksh	Yes	Yes	Weekly
rdf_e_aip_cumint.ksh	Yes	Yes	Weekly
rdf_e_appf.ksh	Yes	Yes	Weekly
rdf_e_rms.ksh	Yes	Yes	Weekly
rdf_fetch_input.ksh	Yes	Yes	Weekly
rdf_find_alerts.ksh	Yes	Yes	Weekly
rdf_gen_forecast.ksh	Yes	Yes	Weekly
rdf_load_hier.ksh	Yes	Yes	Weekly
rdf_load_measures.ksh	Yes	Yes	Weekly
rdf_new_item_store.ksh	Yes	Yes	Weekly
rdf_preprocess.ksh	Yes	Yes	Weekly
rdf_push_output.ksh	Yes	Yes	Weekly

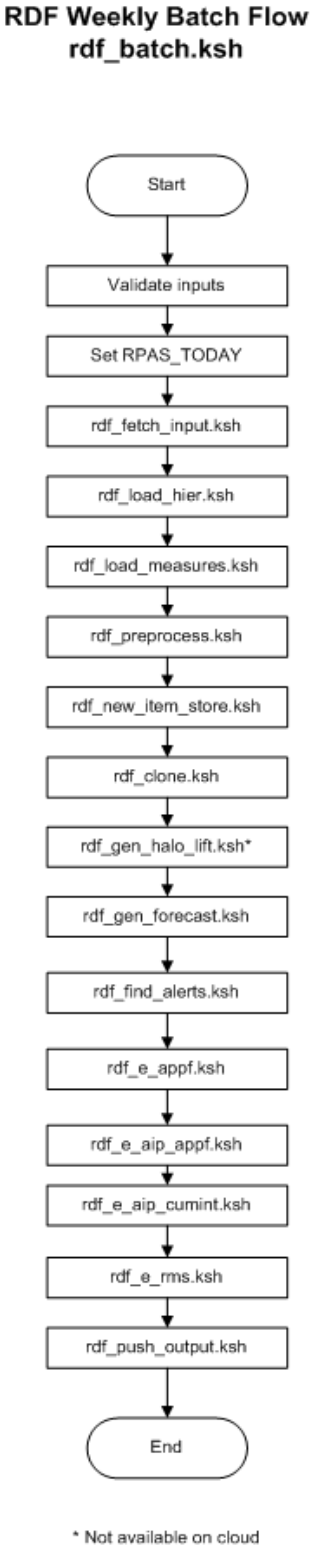
RDF Weekly Batch

RDF has a main control script, `rdf_batch.ksh`, which should be run on a weekly basis. This script is shared between RDF and RDF Cloud Services. The script encompasses everything to run the weekly RDF batch process.

Figure 7–1 displays the flow. All of the steps in the flow are optional depending on the input values to the script except for the **Validate Inputs** step.

The following sections describe each of the scripts in detail.

Figure 7–1 Weekly RDF Batch Process



RDF Batch Control Script

Script Name

rdf_batch.ksh

Domain Scope

This script should be run on the master domain.

Description

This is the main RDF batch script. It is expected to be run on a weekly basis. It includes all the steps expected to be run in a weekly RDF batch. Inputs to this task will be forecast levels to run (baseline, causal or both), RPAS_TODAY (optional), and which steps to run. Table 7–2 lists the weekly RDF batch steps.

Note: Steps noted by * are not available on RDF Cloud Service.

Table 7–2 Weekly RDF Batch Steps

Step	Step ID	Step Description	Script	Parameters Passed
1	not applicable	Set optional RPAS_ TODAY	not applicable	not applicable
2	FetchInputData	Fetch input data from staging area	rdf_fetch_input.ksh	-d <master domain path>
3	LoadAllHier	Load hierarchies	rdf_load_hier.ksh	-d <master domain path> -a 14
4	LoadAllMeasureData	Load measure data	rdf_load_measures.ksh	-d <master domain path>
5	PreprocessBatch	Preprocess batch	rdf_preprocess.ksh	-d <master domain path>
6	NewItemBatch	New item and new store batch	rdf_new_item_store.ksh	-d <master domain path>
7	CloningBatch	Clone	rdf_clone.ksh	-d <master domain path>
8	GenHaloBatch*	Generate halo effects*	rdf_gen_halo_lift.ksh	-d <master domain path> -l <final level>
9	ForecastBatch	Run forecast	rdf_gen_forecast.ksh	-d <master domain path> -l <final level>
10	Alerts	Find alerts	rdf_find_alerts.ksh	-d <master domain path>
11	ExportForecast	Export approved forecast	rdf_e_appf.ksh	-d <master domain path> -l <final level> -o appf<final level>.csv -e <export forecast levels>
12	ExportForecastAIP	Export approved forecast to AIP	rdf_e_aip_appf.ksh	-d <master domain path> -l <final level> -o sr0_frclvl2_<final level>.txt -s "S" -c "D"

Table 7–2 (Cont.) Weekly RDF Batch Steps

Step	Step ID	Step Description	Script	Parameters Passed
13	ExportCumIntAIP	Export interval to AIP	rdf_e_aip_cumint.ksh	-d <master domain path> -l <final level> -o sr0_fcterrlvl2_<final level>.txt -s "S"
14	ExportForecastRMS	Export forecast and interval to RMS	rdf_e_rms.ksh	-d <master domain path> -l <final level> -t "S"
15	PushOutputFiles	Push output files to staging area	rdf_push_output.ksh	-d <master domain path>

Required Arguments

Table 7–3 lists the required arguments.

Table 7–3 Required Arguments for rdf_batch.ksh

Parameter	Short Description	Valid Values	Description
-d	Master domain	Path to master domain	Path to master domain
-l	Final forecast levels	Comma separated list of final forecast levels to be run (for example: 01,07) OR baseline, causal, both	For the Run forecast and Generate halo effects steps, this parameter identifies which final levels will be run. If using baseline, causal, or both only one non-causal (baseline) final level and/or one causal final level must exist.

Optional Arguments

Table 7–4 lists the optional arguments.

Table 7–4 Optional Arguments for rdf_batch.ksh

Parameter	Short Description	Valid Values	Description
-e	Export forecast levels	Comma separated list of final forecast levels to be exported (for example: 01,07)	For all of the export steps, this parameter identifies which final levels will be exported. It defaults to the Final forecast levels value.
-i	Export intersection for 'Export approved forecast' step	Any intersection higher, at, or lower than the base intersection of the final level being exported.	This parameter sets the intersection at which the forecast is exported for the Export Approved Forecast step only. See the Export - Forecast task for details.

Table 7–4 (Cont.) Optional Arguments for rdf_batch.ksh

Parameter	Short Description	Valid Values	Description
-t	RPAS_TODAY	A date in YYYYMMDD format	Unless trouble-shooting, this should be left blank. It defaults to the RPAS_ TODAY environment variable or the current server date. If provided, the date needs to be within the range of the day dimension of the calendar hierarchy. Other considerations also apply.
-s	Step to run	See the Step.ID column in Table 7–2	If the –s argument is not supplied, all steps will be run. To run multiple steps, the –s argument must be passed multiple times (for example: -s ForecastBatch -s Alerts).

Optional Flags

None

Fetch Input Data from Staging Area Script

Script Name

rdf_fetch_input.ksh

Domain Scope

This script should be run on the master domain.

Description

This script fetches hierarchy and measure data files from the FTP incoming directory to the master domain input directory.

The script follows these steps:

1. Validate arguments.
2. Check if the sentinel file $INCOMING_FTP_PATH/COMMAND/COMPLETE exists.
3. If it does not exist, log informational message and skip to Step 8.
4. Change directory to $INCOMING_FTP_PATH.
5. Copy hierarchies
   a. Looks for files matching the pattern *.?(csv.?(hdr.))dat*(.*)
   b. Copies each file to <master domain path>/input directory.

      It does not copy pror or locr files. A warning is logged.
   c. If file copied is a prod or loc file, also copy the file as pror or locr respectively.

      An information message is logged.

   d. Removes each file from $INCOMING_FTP_PATH.

6. Copies measure files (ovr)
   a. Looks for files matching the pattern *.ovr
   b. Copies all files together to <master domain path>/input directory.
   c. Removes all *.ovr files in $INCOMING_FTP_PATH.
7. Copies measure files (rpl)
   a. Looks for files matching the pattern *.rpl
   b. Copies all files together to <master domain path>/input directory.
   c. Removes all *.rpl files in $INCOMING_FTP_PATH.
8. Log informational success message.

Additional Required Environment Variables

Table 7–5 lists additional required environment variables.

Table 7–5 Additional Required Environment Variables for rdf_fetch_input.ksh

Environment Variable	Valid Values	Description
INCOMING_FTP_PATH	Valid path	Path to FTP incoming files

Required Arguments

Table 7–6 lists the required arguments.

Table 7–6 Required Arguments for rdf_fetch_input.ksh

Parameter	Short Description	Valid Values	Description
-d	Master domain	Path to master domain	Path to master domain

Optional Arguments

None

Optional Flags

None

Load Hierarchies Script

Script Name

rdf_load_hier.ksh

Domain Scope

This script should be run on the master domain.

Description

This script will load all the hierarchy files specified in the rdf_load_hier.ctl file. Any hierarchy files to be loaded need to be present in the standard input directory. It is assumed that any necessary RETL transformations have already occurred before running this script.

The script follows these steps:

1. Validate arguments.
2. Parse the control file $RPAS_HOME/bin/rdf_load_hier.ctl.
3. Iterate over the hierarchies specified for cloud or on-premise, as appropriate.
4. For each hierarchy:
   a. If the load is optional, check for the data file using this pattern before calling loadHier:

      ```
      <hier> .?(csv.?(hdr.))dat*(.*)
      ```

      Log informational message if missing and continue with next hierarchy.
   b. Call loadHier for the specified hierarchy on the master domain using the following parameters:

Parameter	Value
-d	Path to the master domain.
-load	Current hierarchy.
-purgeAge	Purge age passed to script.
-forceNAConsistency	not applicable
-maxProcesses	$BSA_MAX_PARALLEL

5. Log informational success message.

Additional Required Environment Variables

None

Required Arguments

Table 7–7 lists the required arguments.

Table 7–7 Required Arguments for rdf_load_hier.ksh

Parameter	Short Description	Valid Values	Description
-d	Master domain	Path to master domain	Path to master domain
-a	Purge age	A non-negative integer	Purge age to pass to loadHier executable.

Optional Arguments

None

Optional Flags

None

Control File Format

The control file has three fields delimited by the colon ":" character. A line may be commented out using the hash "#"character as the first character of the line. Blank lines are not supported.

Table 7–8 lists the required arguments.

Table 7–8 Control File Format for rdf_load_hier.ksh

Field Number	Short Description	Valid Values	Description
1	Hierarchy ID		Hierarchy to load. For example, loc or prod.
2	On premise flag	N, O, Y	Used for on premise domains. N – Do not load O – Optional Y – Always load
3	Cloud flag	N, O, Y	Used for cloud domains. N – Do not load O – Optional Y – Always load

Load Measure Data Script

Script Name

rdf_load_measures.ksh

Domain Scope

This script should be run on the master domain.

Description

This script will load the measures specified in the rdf_load_measures.ctl file. The input files need to be present in the master domain input directory. It is assumed that any necessary RETL transformations have already occurred before the files are uploaded to the cloud.

The script follows these steps:

1. Validate arguments.
2. Parse the control file $RPAS_HOME/bin/rdf_load_measures.ctl..
3. Iterate over the measures specified for cloud or on-premise, as appropriate.
4. For each measure:
   a. Check to see if measure exists in the domain.

      If it does not, log a warning message and continue with next measure.

   b. Call loadmeasure with the following parameters:

Parameter	Value
-d	Path to the master domain.
-measure	Current measure.
-processes	$BSA_MAX_PARALLEL
-recordLogLevel	recordLogLevel passed to the script (if provided).

5. Log informational success message.

Additional Required Environment Variables

None

Required Arguments

Table 7–9 lists the required arguments.

Table 7–9 Required Arguments for rdf_load_measures.ksh

Parameter	Short Description	Valid Values	Description
-d	Master domain	Path to master domain	Path to master domain

Optional Arguments

Table 7–10 lists the optional arguments.

Table 7–10 Optional Arguments for rdf_load_measures.ksh

Parameter	Short Description	Valid Values	Description
-r	recordLogLevel	error, warning, information, and profile	Sets a logging level for record loading issues. If the logging level set at implementation time is less verbose than the record logging level, then record issues will not be logged. If utility's logging level is at same or higher verbosity as the record logging level, the record issues will be logged with the log indicator as set using this argument.

Optional Flags

None

Control File Format

The control file has three fields delimited by the colon ":" character. A line may be commented out using the hash "#"character as the first character of the line. Blank lines are not supported.

Table 7–11 lists the required arguments.

Table 7–11 Control File Format for rdf_load_measures.ksh

Field Number	Short Description	Valid Values	Description
1	Measure name		Measure to load. For example, pos or grpasnmt.
2	On premise flag	N, Y	Used for on premise domains. N – Do not load Y – Always load Note that loadmeasure does not fail if the file does not exist, so Y is effectively an optional load.

Table 7–11 (Cont.) Control File Format for rdf_load_measures.ksh

Field Number	Short Description	Valid Values	Description
3	Cloud flag	N, Y	Used for cloud domains. N – Do not load Y – Always load Note that loadmeasure does not fail if the file does not exist, so Y is effectively an optional load.

Preprocess Batch Script

Script Name

rdf_preprocess.ksh

Domain Scope

This script should be run on the master domain.

Description

This script will run the preprocessing rule group calc_oosoutlier from the PrepDemandCommon solution and the rule groups specified in the rdf_preprocess.ctl file. These rule groups should be the ones generated by the "Prepare Demand" configurations tools plug-in (for example, MergeAndRunP01, ppsPostRunP01).

The script follows these steps:

1. Validate arguments.
2. If the scalar boolean measure PreCalcOutlierto is set to true, run the calc_ oosoutlier rule group.
3. Parse the control file $RPAS_HOME/bin/rdf_ preprocess .ctl.
4. Iterate over the rule groups specified for cloud or on-premise, as appropriate.
5. For each rule group:
   a. Use mace to run the rule group on the local domains using para_spawn from BSA.
6. Log informational success message.

Additional Required Environment Variables

None

Required Arguments

Table 7–12 lists the required arguments.

Table 7–12 Required Arguments for rdf_preprocess.ksh

Parameter	Short Description	Valid Values	Description
-d	Master domain	Path to master domain	Path to master domain

Optional Arguments

None

Optional Flags

None

Control File Format

The control file has three fields delimited by the colon ":" character. A line may be commented out using the hash "#"character as the first character of the line. Blank lines are not supported.

Table 7–13 lists the required arguments.

Table 7–13 Control File Format for rdf_preprocess.ksh

Field Number	Short Description	Valid Values	Description
1	Measure name		Measure to load. For example, pos or grpasnmt.
2	On premise flag	N, O, Y	Used for on premise domains. N – Do not load O – Optional Y – Always load Note that loadmeasure does not fail if the file does not exist, so Y is effectively an optional load.
3	Cloud flag	N, O, Y	Used for cloud domains. N – Do not load O – Optional Y – Always load Note that loadmeasure does not fail if the file does not exist, so Y is effectively an optional load.

New Item and New Store Batch

Script Name

rdf_new_item_store.ksh

Domain Scope

This script should be run on the master domain.

Description

This script will create recommendations for new items if item attributes and weights exist.

The script follows these steps:

1. Validate arguments.
2. If the New Item plug-in was configured to use attributes:
   - Use mace to run these rule groups:

     NITC_bat_PreMst on the master domain.

     NITA_bat_PreRec on the local domains using para_spawn from BSA.

     NITA_bat_GenRec on the local domains using para_spawn from BSA.

NITA_bat_AutoApp on the local domains using para_spawn from BSA.

3. If the New Item plug-in was not configured to use attributes:
   - Use mace to run the rule group NITC_bat_PreMst on the master domain.
   - Use mace to run the rule group NITM_Bat_RunAll on the local domains using para_spawn from BSA.
4. Log informational success message.

Additional Required Environment Variables

None

Required Arguments

Table 7–14 lists the required arguments.

Table 7–14 Required Arguments for rdf_new_item_store.ksh

Parameter	Short Description	Valid Values	Description
-d	Master domain	Path to master domain	Path to master domain

Optional Arguments

None

Optional Flags

None

Clone

Script Name

rdf_clone.ksh

Domain Scope

This script should be run on the master domain.

Description

This task will run the rule groups clone_batch, clone_adjust, clone_adj_run.

The script follows these steps:

1. Validate arguments.
2. Use mace to run the rule group clone_batch on the local domains using para_spawn from BSA.
3. Use mace to run the rule group clone_adjust on the local domains using para_spawn from BSA.
4. Use mace to run the rule group clone_adj_run on the local domains using para_spawn from BSA.
5. Log informational success message.

Additional Required Environment Variables

None

Required Arguments

Table 7–15 lists the required arguments.

Table 7–15 Required Arguments for rdf_clone.ksh

Parameter	Short Description	Valid Values	Description
-d	Master domain	Path to master domain	Path to master domain

Optional Arguments

None

Optional Flags

None

Generate Halo Effects

Script Name

rdf_gen_halo_lift.ksh

Domain Scope

This script should be run on the master domain. This script is not available in RDF Cloud Service.

Description

This script is used to generate cross promotion halo lift effect. The script uses the last approved forecast and the cross promotion halo effect matrix from CPEM to calculate the halo lift effect ratio. This halo lift effect ratio will be used in the next forecast batch to calculate the cross promotion halo lift unit.

Note: If this script is not run, then no halo effects are incorporated in the forecast during the regular batch run, even when the Forecast Administration settings specify that halo lifts should be produced.

Note: Although not required, but highly recommended for the efficiency considerations is:

```
-finallevel {FinalLevelString}
```

If it is not specified, the script goes over all the final levels of the domain.

The script follows these steps:

1. Validate arguments.
2. Use mace to run the rule group HALO_pet on the master domain.
3. For the final forecast level passed or, if no final forecast level passed, for each final forecast level in the domain:
   a. If the halo measure for the level is not specified in promohaloxlxb Halo Spreading Profile Source, continue to next level. Informational message given.

b. If the intersection of the halo measure is not a higher base intersection, then use mace to run the rule group

HALO_m_base<level> on the master domain.

HALO_nla<level> on the local domains using para_spawn from BSA.

HALO_nm<level> on the master domain.

HALO_nlb<level> on the local domains using para_spawn from BSA.

c. If the intersection of the halo measure is a higher base intersection, then use mace to run the rule group:

HALO_m_base<level> on the master domain.

HALO_l_base<level> on the local domains using para_spawn from BSA.

HALO_hm<level> on the master domain.

4. Log informational success message.

Additional Required Environment Variables

None

Required Arguments

Table 7–19 lists the required arguments.

Table 7–16 Required Arguments for rdf_gen_halo_lift.ksh

Parameter	Short Description	Valid Values	Description
-d	Master domain	Path to master domain	Path to master domain

Optional Arguments

Table 7–17 lists the optional arguments.

Table 7–17 Optional Arguments for rdf_gen_halo_lift.ksh

Parameter	Short Description	Valid Values	Description
-l	Final forecast level	Final forecast level or all (for example, 01)	Level for which to calculate halo effects. Default is all.

Optional Flags

Table 7–18 lists the optional flags.

Table 7–18 Optional Flags for rrdf_gen_halo_lift.ksh

Parameter	Short Description	Description
-u	Usage	Show usage

Run Forecast

Script Name

rdf_gen_forecast.ksh

Domain Scope

This script should be run on the master domain.

Description

This task will run "PreGenerateForecast" on the master domain and then "generate" on all subdomains in parallel.

The script follows these steps:

1. Validate arguments.
2. Explicitly set RPAS_TODAY to the start date if provided. If not provided and RPAS_TODAY is not already set, RPAS_TODAY is set to the system date.
3. Create temporary xml input file for PreGenerateForecast based on input arguments.
4. Call PreGenerateForecast on master domain.
5. Run "generate" on the local domains using para_spawn from BSA.
6. Remove temporary xml files.
7. Log informational success message.

Additional Required Environment Variables

None

Required Arguments

Table 7–19 lists the required arguments.

Table 7–19 Required Arguments for rdf_gen_forecast.ksh

Parameter	Short Description	Valid Values	Description
-d	Master domain	Path to master domain	Path to master domain
-l	Final forecast level	Final forecast level (for example, 01)	Final forecast level to be forecasted

Optional Arguments

Table 7–20 lists the optional arguments.

Table 7–20 Optional Arguments for rdf_gen_forecast.ksh

Parameter	Short Description	Valid Values	Description
-o	Override	true, false	When override is false, the forecast is only generated if current time is later than the next run date in the domain. When the override is true, the forecast is generated regardless of the next run date. If not provided, the default value is false.
-s	Forecast start date	Date in YYYYMMDD format	This defaults to RPAS_TODAY or the current server date. If provided, the date needs to be within the range of the day dimension of the calendar hierarchy. It will export the forecast starting from this date.

Optional Flags

None

Find Alerts Script

Script Name
rdf_find_alerts.ksh

Domain Scope
This script should be run on the master domain.

Description
This script will run the alerts as specified in the rdf_find_alerts.ctl file.

The script follows these steps:

1. Validate arguments.
2. Parse the control file $RPAS_HOME/bin/rdf_find_alerts.ctl.
3. Iterate over the alerts specified for cloud or on-premise, as appropriate.
4. For each alert:
   a. If it is an alert category:

      Run alertmgr on the local domains using para_spawn from BSA using the following parameters:

Parameter	Value
-d	Path to the master domain.
-findAlerts	not applicable
-navigationThreshold	Value from rdf_find_alerts.ctl.
-categories	Alert category from rdf_find_alerts.ctl.

      Call alertmgr on the master domain with –sumAlerts flag.

   b. If it is an individual alert:

      Run alertmgr on the local domains using para_spawn from BSA using the following parameters:

Parameter	Value
-d	Path to the master domain.
-findAlerts	not applicable
-navigationThreshold	Value from rdf_find_alerts.ctl.
-alerts	Alert category from rdf_find_alerts.ctl.

      Call alertmgr on the master domain with –sumAlerts flag.

5. Log informational success message.

Additional Required Environment Variables
None

Required Arguments

Table 7–21 lists the required arguments.

Table 7–21 Required Arguments for rdf_find_alerts.ksh

Parameter	Short Description	Valid Values	Description
-d	Master domain	Path to master domain	Path to master domain

Optional Arguments

None

Optional Flags

None

Control File Format

The control file has three fields delimited by the colon ":" character. A line may be commented out using the hash "#"character as the first character of the line. Blank lines are not supported.

Table 7–22 lists the required arguments.

Table 7–22 Control File Format for rdf_find_alerts.ksh

Field Number	Short Description	Valid Values	Description
1	Alert name	An alert name or alert category	The name of an alert or alert category.
2	Run type	alerts or categories	Determines whether to run alertmgr with –alerts or –categories.
3	Alert threshold	Non-negative integer	Indicates the maximum number of alert hits for Find Next/Previous Alert functionality to remain operational in a workbook. If over that threshold, the Find Alert functionality will only work up to that number.
4	On premise flag	N, Y	Used for on premise domains. N – Do not load Y – Always load Note that loadmeasure does not fail if the file does not exist, so Y is effectively an optional load.
5	Cloud flag	N, Y	Used for cloud domains. N – Do not load Y – Always load Note that loadmeasure does not fail if the file does not exist, so Y is effectively an optional load.

Export Approved Forecast

Script Name

rdf_e_appf.ksh

Domain Scope

This script should be run on the master domain.

Description

This script exports the approved forecast of any final level from RDF in csv format. Export intersection is configurable but must be above, at, or below to the final level intersection. The order of the export file is always calendar dimension, product dimension, location dimension, and then forecast.

The script follows these steps:

1. Validate arguments.
2. Explicitly set RPAS_TODAY to the start date if provided. If not provided and RPAS_TODAY is not already set, RPAS_TODAY is set to the system date
3. Copy the Approved Forecast measure (appf<level>xb) to a temporary measure for export. It will copy those values in the measure which are greater than or equal to "now" and less than "now" plus the forecast length. This is done using mace on the local domains using para_spawn from BSA
4. Call exportMeasure on the master domain with the following parameters:

Parameter	Value
-d	Path to the master domain.
-out	$RDF_EXPORT_DIR/<file name> where <file name> is the value of the –o argument to the script.
-meas	Temporary measure name from step 3.
-intx	Export intersection.
-useDate	Only used if –e or –s flags were used when calling the script. If –e, end is passed. If –s, start is passed.
-processes	$BSA_MAX_PARALLEL

5. Exit.

Additional Required Environment Variables

None

Required Arguments

Table 7–23 lists the required arguments.

Table 7–23 Required Arguments for rdf_e_appf.ksh

Parameter	Short Description	Valid Values	Description
-d	Master domain	Path to master domain	Path to master domain
-o	Output file name	Any valid file name according to OS	No path should be used. The output file will be placed in $RDF_EXPORT_DIR.

Table 7–23 (Cont.) Required Arguments for rdf_e_appf.ksh

Parameter	Short Description	Valid Values	Description
-l	Final forecast level	Two digit number (for example, 01 or 07)	This parameter identifies which final level will be exported. Only one may be specified.

Optional Arguments

Table 7–24 lists the optional arguments.

Table 7–24 Optional Arguments for rdf_e_appf.ksh

Parameter	Short Description	Valid Values	Description
-f	Forecast start date	A date in YYYYMMDD format	This defaults to RPAS_TODAY or the current server date. If provided, the date needs to be within the range of the day dimension of the calendar hierarchy. It will export the forecast starting from this date.
-i	Export intersection	Valid RPAS intersection	Determines the intersection at which to export the data. The intersection must be above, at, or below the base intersection of the final level. It must use 4 characters for each dimension (i.e. itemstr_week). Defaults to the base intersection of the final forecast level if not provided.

Optional Flags

Table 7–25 lists the optional flags.

Table 7–25 Optional Flags for rrdf_e_appf.ksh

Flag	Short Description	Description
-e	End of period day flag	If calendar is in the export intersection, export it as end of period day instead of original dimension (i.e. 20170107 instead of W01_2017).
-s	Start of period day flag	If calendar is in the export intersection, export it as start of period day instead of original dimension (i.e. 20170101 instead of W01_2017).

Output File Format

The following table provides information about the output file data format (CSV).

Field	Format
Calendar ID	Alpha
Product ID	Alpha
Location ID	Alpha
Demand	Numeric

Export Approved Forecast to AIP

Script Name

rdf_e_aip_appf.ksh

Domain Scope

This script should be run on the master domain.

Description

This script exports the approved forecast from RDF for AIP in a flat file. The final level should be either at day/item/store or week/item/store for the script to work properly. The export intersection matches the final level intersection. If the final level is at week, the end of week position name for day is exported instead of the week position name. The order of the export file is always day, store, item, and then forecast.

The script follows these steps:

1. Validate arguments.
2. Explicitly set RPAS_TODAY to the start date if provided. If not provided and RPAS_TODAY is not already set, RPAS_TODAY is set to the system date.
3. Copy the Approved Forecast measure (appf<level>xb) to a temporary measure for export. It will copy those values in the measure which are greater than or equal to "now" and less than "now" plus the forecast length. This is done using mace on the local domains using para_spawn from BSA.
4. If exporting at the week level, calculate measure to map week position name to end of week day position name. It does this on the master domain.
5. Call exportData with the following parameters:

Parameter	Value
-d	Path to the master domain.
-out	$RDF_EXPORT_DIR/<file name> where <file name> is the value of the –o argument to the script.
-dim	<clnd dimension> <0 or mapping array> <clnd prefix>%-<col width>s 1 where <col width> is 9-(length of prefix)
-dim	STR 0 <str prefix>%-<col width>s 2 where <col width> is 20-(length of prefix)
-dim	ITEM 0 %-20s 3
-meas	<temp measure> %-8.4f 0.0 %-8.4f
-skipNa	anyna
-processes	$BSA_MAX_PARALLEL

6. Log informational success message.

Additional Required Environment Variables

None

Required Arguments

Table 7–26 lists the required arguments.

Table 7–26 Required Arguments for rdf_e_aip_appf.ksh

Parameter	Short Description	Valid Values	Description
-d	Master domain	Path to master domain	Path to master domain
-o	Output file name	Any valid file name according to OS	No path should be used. The output file will be placed in $RDF_EXPORT_DIR.
-l	Final forecast level	Two digit number (for example, 01 or 07)	This parameter identifies which final level will be exported. Only one may be specified.

Optional Arguments

Table 7–27 lists the optional arguments.

Table 7–27 Optional Arguments for rdf_e_aip_appf.ksh

Parameter	Short Description	Valid Values	Description
-f	Forecast start date	A date in YYYYMMDD format	This defaults to RPAS_TODAY or the current server date. If provided, the date needs to be within the range of the day dimension of the calendar hierarchy. It will export the forecast starting from this date.
-s	Store column prefix	A string of alphanu-meric characters. Normally S.	This prefix is prepended to the position name of the store dimension on export.
-c	Calendar column prefix	A string of alphanu-meric characters. Normally D.	This prefix is prepended to the position name of the day dimension on export.

Optional Flags

None

Output File Format

The following table provides information about the output file data format.

Field	Start	Width	Format
Day | EOW Day	1	9	Alpha
Product ID	9	20	Alpha
Location ID	29	20	Alpha
Forecast	49	8	Numeric

Export Interval to AIP

Script Name

rdf_e_aip_cumint.ksh

Domain Scope

This script should be run on the master domain.

Description

This script exports the cumulative interval for the first period of the forecast from RDF for AIP in a flat file. The final level should be either at day/item/store or week/item/store for the script to work properly. The export intersection matches the final level intersection except it does not include a calendar dimension. The order of the export file is always store, item, and then interval.

The script follows these steps:

1. Validate arguments
2. Explicitly set RPAS_TODAY to the start date if provided. If not provided and RPAS_TODAY is not already set, RPAS_TODAY is set to the system date.
3. Copy the Approved Cumulative Interval measure (appcumint<level>xb) to a temporary measure for export. It does this in two steps. First, it will copy those values in the measure which are equal to "now". Then, it is aggregated to another temporary measure that does not contain the calendar dimension. This is done using mace on the local domains using para_spawn from BSA.
4. Call exportData with the following parameters:

Parameter	Value
-d	Path to the master domain.
-out	$RDF_EXPORT_DIR/<file name> where <file name> is the value of the –o argument to the script.
-dim	STR 0 <str prefix>%-<col width>s 1" where <col width> is 20-(length of prefix)
-dim	ITEM 0 %-20s 2
-meas	<temp measure> %-20.6f 0.0 %-20.6f
-skipNa	anyna
-processes	$BSA_MAX_PARALLEL

5. Log informational success message.

Additional Required Environment Variables

None

Required Arguments

Table 7–28 lists the required arguments.

Table 7–28 Required Arguments for rdf_e_aip_cumint.ksh

Parameter	Short Description	Valid Values	Description
-d	Master domain	Path to master domain	Path to master domain
-o	Output file name	Any valid file name according to OS	No path should be used. The output file will be placed in $RDF_EXPORT_DIR.
-l	Final forecast level	Two digit number (for example, 01 or 07)	This parameter identifies which final level will be exported. Only one may be specified. Intersection of the final level should be day/item/store or week/item/store.

Optional Arguments

Table 7–29 lists the optional arguments.

Table 7–29 Optional Arguments for rdf_e_aip_cumint.ksh

Parameter	Short Description	Valid Values	Description
-f	Forecast start date	A date in YYYYMMDD format	This defaults to RPAS_TODAY or the current server date. If provided, the date needs to be within the range of the day dimension of the calendar hierarchy. It will export the forecast starting from this date.
-s	Store column prefix	A string of alphanumeric characters. Normally S.	This prefix is prepended to the position name of the store dimension on export.

Optional Flags

None

Output File Format

The following table provides information about the output file data format.

Field	Start	Width	Format
Product ID	1	20	Alpha
Location ID	21	20	Alpha
Forecast	41	20	Numeric

Export Forecast and Interval to RMS

Script Name

rdf_e_rms.ksh

Domain Scope

This script should be run on the master domain.

Description

This script exports the approved forecast and the first period of the approved cumulative interval from RDF for RMS in a flat file. Final level should be either at day/item/store or week/item/store for the script to work properly. Export intersection matches the final level intersection. If the final level is at week, the end of week position name for day is exported instead of the week position name. For information on how this script fits in the overall integration with RMS, refer to, Appendix A, "RPAS and RDF Integration with RMS."

Steps

The script follows these steps:

1. Validate arguments.
2. Explicitly set RPAS_TODAY to the start date if provided. If not provided and RPAS_TODAY is not already set, RPAS_TODAY is set to the system date.
3. Copy the Approved Forecast measure (appf<level>xb) to a temporary measure for export. It will copy those values in the measure which are greater than or equal to “now” and less than “now” plus the forecast length. This is done using mace on the local domains using para_spawn from BSA.
4. Copy the Approved Cumulative Interval measure (appcumint<level>xb) to a temporary measure for export. It will copy those values in the measure which are equal to “now”. This is done using mace on the local domains using para_spawn from BSA.
5. Export the data using exportData using –processes of $BSA_MAX_PARALLEL. After export, reformat the file to match the format RMS is expecting.
6. Log informational success message.

Additional Required Environment Variables

None

Required Arguments

Table 7–30 lists the required arguments.

Table 7–30 Required Arguments for rdf_e_rms.ksh

Parameter	Short Description	Valid Values	Description
-d	Master domain	Path to master domain	Path to master domain
-l	Final forecast level	Two digit number (for example, 01 or 07)	This parameter identifies which final level will be exported. Only one may be specified. Intersection of the final level should be day/item/store or week/item/store.
-t	Domain type	S, I	S is for sales, I is for issues. The only effect is in the file name of the output file.

Optional Arguments

Table 7–31 lists the optional arguments.

Table 7–31 Optional Arguments for rdf_e_rms.ksh

Parameter	Short Description	Valid Values	Description
-f	Forecast start date	A date in YYYYMMDD format	This defaults to RPAS_TODAY or the current server date. If provided, the date needs to be within the range of the day dimension of the calendar hierarchy. It will export the forecast starting from this date.
-w	Data width	[7..18]	Width of the columns for the Approved Forecast and Approved Interval in the output file. If not used, the default width is 14. It is always floating point to four decimal places.

Optional Flags

None

Output File Format

The following table provides information about the output file data format.

`${RDF_EXPORT_DIR}/d<s|i>demand.<forecast level> (demand at day)`

`${RDF_EXPORT_DIR}/w<s|i>demand.<forecast level> (demand at week)`

> **Note:** For items in the table noted by an asterisk, the Width of Demand and Standard Dev. Demand may be overridden with the `-w` parameter; stated values Demand width and Standard Dev. Demand start and width are based on default width of 14.

Field	Start	Width	Format
Day | EOW Day	1	8	Alpha
Product ID	9	25	Alpha
Location ID	34	20	Alpha
Demand	54	14*	Alpha
Std. Dev. Demand	68*	14*	Numeric (floating point, 4 decimal digits with decimal)

Push Output Files to Staging Area

Script Name

rdf_push_output.ksh

Domain Scope

This script should be run on the master domain.

Description

This script will push all files from the $RDF_EXPORT_DIR directory to the FTP outgoing directory.

The script follows these steps:

1. Validate arguments
2. Use `.scp` to copy all files to in $RDF_EXPORT_DIR to the FTP outgoing directory on the FTP server.
3. Create sentinel file COMMAND/COMPLETE in the FTP outgoing directory on the FTP server.
4. Remove all files from $RDF_EXPORT_DIR.
5. Log informational success message.

Additional Required Environment Variables

Table 7–32 lists additional required environment variables.

Table 7–32 Additional Required Environment Variables for rdf_push_output.ksh

Environment Variable	Valid Values	Description
FTP_SERVER	<server name>	FTP server name
OUTGOING_FTP_PATH	<path to directory>	Path on the FTP server

Required Arguments

Table 7–33 lists the required arguments.

Table 7–33 Required Arguments for rdf_push_output.ksh

Parameter	Short Description	Valid Values	Description
-d	Master domain	Path to master domain	Path to master domain

Optional Arguments

None

Optional Flags

None

Other Batch Processes

RDF has several other batch processes that are run on an occasional basis:

- RDF to APC-RO Integration
- Load Bayesian Plan
- Demand Transference Preprocessing
- Calculate Floating Event Effect

RDF to APC-RO Integration

Script Name

rdf_e_apcro.ksh

Supporting scripts:

- rdf_e_apcro_weekly.ksh
- dayweights.awk

- reformat_data.awk
- reformat_profile.awk
- reformat.awk

Domain Scope

This script should be run on the master domain.

Frequency

This script will be run extremely rarely. It should only need to be run when needing to set up a new APC-RO implementation.

Description

RDF's forecasts are important inputs for APC-RO. Forecasts are provided for initial load into APC-RO. A rolling 52 forecasts are generated and exported to APC-RO, each of the forecasts starts one week after another. The main purpose of the scripts is to generate RDF forecast in RDF GA domain and export the forecasts from a RDF domain and covert the forecasts into the format required by APC-RO.

Additionally, APC-RO provides to RDF a list of item/stores, and RDF only exports the forecasts for those item/stores to APC-RO.

Before running the script, the input datelist file needs to be created. The datelist file contains the list of desired forecast start date in the format of YYYYMMDD (for example, 20101130 and as shown in Example 3–1)

The forecast start dates need to be seven (7) days apart as shown in Example 3–1.

Example 7–1 Datelist File

```
20100101
20100108
20100114
20100120
```

The script follows these steps:

1. Validate arguments
2. If –fetch flag is used:
   a. Change directory to $INCOMING_FTP_PATH.
   b. Check if the sentinel file $INCOMING_FTP_PATH/COMMAND/COMPLETE exists.
   c. If it does not exist, log informational message and skip to step 3
   d. If –mask argument passed to script, copy the mask measure:

      Look for files matching the pattern <mask>.?(csv.?(hdr.))ovr*(.*)

      If none found, log informational message and skip to step 2e.

      Copy all files together to <master domain path>/input directory.

      Remove copied files from $INCOMING_FTP_PATH.
   e. If – DOWProfile argument passed to script, copy the DOWProfile measure:

      Look for files matching the pattern < DOWProfile> .?(csv.?(hdr.))ovr*(.*)

If none found, log informational message and skip to step 2f.

Copy all files together to <master domain path>/input directory.

Remove copied files from $INCOMING_FTP_PATH.

f. Copy datelist file:

Look the file name passed with the –datelist argument. Note, that only the file name is used for this step – the path is ignored.

If datelist is not found, log informational message and skip to step 2g.

Copy the datelist file to <master domain path>/input directory.

Remove copied files from $INCOMING_FTP_PATH.

g. Change directory to last directory.

3. If –load flag is used:

a. If –mask argument passed to the script call loadmeasure on the mask measure with –processes of $BSA_MAX_PARALLEL.

b. If –DOWProfile argument passed to the script call loadmeasure on the DOWProfile measure with –processes of $BSA_MAX_PARALLEL.

4. Remove the output file if it already exists.

5. Create temporary xml input file for PreGenerateForecast based on input arguments. Override is set to true.

6. For each date in the datelist file:

a. Clear out the ApprovedForecast (appf<level>xb) and Approved Cumulative Interval (appcumint<level>xb) measures.

b. Set RPAS_TODAY to the date in the datelist file.

c. Call PreGenerateForecast on master domain.

d. Run “generate” on the local domains using para_spawn from BSA.

e. Call rdf_e_apcro_weekly.ksh to create corresponding output file using a unique output file name.

Validate arguments.

Remove the output file if it already exists.

Export the newly generated Approved forecast, Approved Cumulative Interval and other outputs using temporary measures, various awk scripts, and exportMeasure.

f. Log informational message that data export is completed for the date.

7. Concatenate the results of all of the output files into one output file specified by the “-o” argument

8. Remove temporary output files.

9. If –push flag is used:

a. Use scp to copy the output file to the FTP outgoing directory on the FTP server.

b. Create sentinel file COMMAND/COMPLETE in the FTP outgoing directory on the FTP server.

c. Remove the output file from $RDF_EXPORT_DIR.

10. Exit.

Additional Required Environment Variables

Table 7–34 lists additional required environment variables.

Table 7–34 Additional Required Environment Variables for rdf_e_apcro.ksh

Environment Variable	Valid Values	Description
INCOMING_FTP_PATH	Valid path	Path to FTP incoming files. Only needed if –fetch flag is used.
FTP_SERVER	<server name>	FTP server name. Only needed if –push flag is used.
OUTGOING_FTP_PATH	<path to directory>	Path on the FTP server. Only needed if –push flag is used.

Required Arguments

Table 7–35 lists the required arguments.

Table 7–35 Required Arguments for rdf_e_apcro.ksh

Parameter	Short Description	Valid Values	Description
-d	Master domain	Path to master domain	Path to the master domain.
-o	Output path and file name	Any valid file name according to OS	Output path and file name
-l	Final forecast level	Two digit number (for example. 01 or 07)	This parameter identifies which final level will be exported. Only one may be specified. Intersection of the final level should be week/item/store.
-datelist	Forecast start date file name	Any valid file name according to OS	A file that contains a list of forecast start dates in format YYYYMMDD (for example, 20101130). If "Fetch datelist" option is not selected, the file must exist in the input directory of the master domain. If it is selected, it can either be in the input directory or in the $INCOMING_FTP_PATH.

Optional Arguments

Table 7–36 lists the optional arguments.

Table 7–36 Optional Arguments for rdf_e_apcro.ksh

Parameter	Short Description	Valid Values	Description
-DOWProfile	Day of week profile measure	A valid RPAS measure name	The optional day of week profile measure name to spread the date from week to day. (for example, dowprof)

Table 7–36 (Cont.) Optional Arguments for rdf_e_apcro.ksh

Parameter	Short Description	Valid Values	Description
-mask	Mask measure	A valid RPAS measure name	he measure name of a boolean measure at item/store intersection (for example, apcroexptmask). This is used to filter the export.
-sundayIndex	Sunday index	Integer from 1 to 7 inclusive	Sunday's position in the day of week (DOW) hierarchy. The default is 2

Optional Flags

Table 7–37 lists the optional flags.

Table 7–37 Optional Flags for rdf_e_apcro.ksh

Flag	Short Description	Description
-switch Order	Switch order flag	Flag to switch the order of merchandise and store in output (default is merchandise then store). If flag is passed, it will be store then merchandise.
-fetch	Fetch input data flag	If passed, the script will attempt to fetch the needed data (datelist file, mask data, DOW profile data) from $INCOMING_FTP_PATH and copy to the master domain input folder.
-load	Load measures flag	If passed, the script will attempt to load the mask and DOW profile measures into the domain from the master domain input folder.
-push	Push output file flag	If passed, the script will push the output file to the $OUTGOING_FTP_PATH on $FTP_SERVER.

Output File Format

The following table provides information about the output file data format.

Field	Format
EOW Day	Date
Item ID	Alpha
Store ID	Alpha
Forecast Demand	Numeric
Forecast Start Date	Date
Day 1 Weight	Number
Day 2 Weight	Numeric
Day 3 Weight	Numeric
Day 4 Weight	Numeric
Day 5 Weight	Numeric
Day 6 Weight	Numeric
Day 7 Weight	Numeric
Cumulative Interval	Numeric

Load Bayesian Plan

Script Name

rdf_load_bayesian_plan.ksh

Domain Scope

This script should be run on the master domain.This script is only available in RDF Cloud Service.

Frequency

This script will be run only when the Bayesian plan changes and either:

- The customer wishes to load the plans (rather than entering the plans through the workbooks).
- The customer wishes to spread the plans from the source level(s) to the final level.

Description

This task will optionally fetch the Bayesian plan measure data files from the FTP incoming directory to the master domain "input" directory, load the measures into the domain, and spread the plans from source level to final level.

The script follows these steps:

1. Validate arguments
2. If –fetch flag is used:
   a. Change directory to $INCOMING_FTP_PATH.
   b. Check if the sentinel file $INCOMING_FTP_ PATH/COMMAND/COMPLETE exists
   c. If it does not exist, log informational message and skip to step 3.
   d. If –mask argument passed to script, copy the mask measure:

      Look for files matching the pattern bayesianplan[0-9][0-9]*.ovr

      If none found, log informational message and skip to step 2e.

      Copy all files together to <master domain path>/input directory.

      Remove copied files from $INCOMING_FTP_PATH.
   e. Change directory to last directory.
3. If –load flag is used:
   a. Change directory to $INCOMING_FTP_PATH.
   b. Load any measure files matching the pattern bayesianplan[0-9][0-9]*.ovr with –processes of $BSA_MAX_PARALLEL and using the recordLogLevel passed to the script.
   c. Change directory to last directory.
4. If –spread flag is used:
   a. Change directory to $INCOMING_FTP_PATH.
   b. Use mace to run the countcalc_batch rule group on the local domains using para_spawn from BSA.

c. Use mace to run the bayplanspread rule group on the local domains using para_spawn from BSA.

5. Exit

Additional Required Environment Variables

Table 7–38 lists additional required environment variables.

Table 7–38 Additional Required Environment Variables for rdf_load_bayesian_plan.ksh

Environment Variable	Valid Values	Description
INCOMING_FTP_PATH	Valid path	Path to FTP incoming files. Only needed if –fetch flag is used.

Required Arguments

Table 7–39 lists the required arguments.

Table 7–39 Required Arguments for rdf_load_bayesian_plan.ksh

Parameter	Short Description	Valid Values	Description
-d	Master domain	Path to master domain	Path to the master domain.

Optional Arguments

Table 7–40 lists the optional arguments.

Table 7–40 Optional Arguments for rdf_load_bayesian_plan.ksh

Parameter	Short Description	Valid Values	Description
-r	Record log level	error, warning, information, and profile	Sets a logging level for record loading issues. If the logging level set at implementation time is less verbose than the record logging level, then record issues will not be logged. If utility's logging level is at same or higher verbosity as the record logging level, the record issues will be logged with the log indicator as set using this argument.

Optional Flags

Table 7–41 lists the optional flags.

Table 7–41 Optional Flags for rdf_load_bayesian_plan.ksh

Flag	Short Description	Description
-fetch	Fetch input data flag	If passed, the script will attempt to fetch the needed data from $INCOMING_FTP_PATH and copy to the master domain input folder.
-load	Load measures flag	If passed, the script will attempt to load the bayesian plan measures into the domain from the master domain input folder.
-spread	Spread the plan flag	If passed, the script will spread the Bayesian plan from the source level measures to the final level measure.

Demand Transference Preprocessing

Script Name

rdf_preprocess_dt.ksh

Domain Scope

This script should be run on the master domain.This script is not available in RDF Cloud Service.

Frequency

This script should be run when receiving new demand transference files from ORASE.

Description

This script generates adjusted weekly demand transference effects based on assortment multipliers per item/store, assortment multipliers decay factor and demand assortment multiplier applying period. The script uses the integrated assortment multipliers and the effective dates to calculate the time-phased assortment multipliers. The time-phased assortment multipliers are used in the forecast batch to calculate the demand transference lift unit. The script also integrates the base rate of sales for the new item demand forecast.

This script loads three measures based on the final level specified by the user. If no final level is specified, all the valid final levels will be iterated.

The following table lists the measures that will be used by the script where {valid_ final_ level} is the final level that you want to run on. The {valid_final_level} in the following measures will be replaced by the final level specified by you. If no final level is specified, this script will run on all the valid final levels.

Measure	Label
fmafstdt{valid_final_ level}xb	Forecast Start Date Override Baseline Forecast Final <level> - <intersection>
nitnwros	Base Rate of Sales
assmul{valid_final_ level}xb	Assortment Multipliers Baseline Forecast Final <level> - <intersection>

Note: If this script is not run, then no demand transference effects are incorporated in the forecast during the regular batch run, even though the Forecast Administration settings specify that demand transference lifts should be produced.

The script follows these steps:

1. Validate arguments
2.
   a. Check if demand transference is enabled for the level. If not, give informational warning message and go to next level.
   b. Load measures fmafstdt<level>xb and assmul<level>xb using BSA_MAX_ PARALLEL for number of processes.

c. Use mace to run the DT_PreProcess<level> rule group on the local domains using para_spawn from BSA.

d. Use mace to run the DT_F<level> rule group on the local domains using para_spawn from BSA.

3. Log informational success message.

Additional Required Environment Variables

None

Required Arguments

Table 7–42 lists the required arguments.

Table 7–42 Required Arguments for rdf_preprocess_dt.ksh

Parameter	Short Description	Valid Values	Description
-d	Master domain	Path to master domain	Path to master domain

Optional Arguments

Table 7–43 lists the optional arguments.

Table 7–43 Optional Arguments for rdf_preprocess_dt.ksh

Parameter	Short Description	Valid Values	Description
-l	Final forecast level	final forecast level or all (for example, 01)	Level for which to calculate demand transference. Default is all

Optional Flags

None

Calculate Floating Event Effect

Script Name

rdf_gen_float_lift.ksh

Domain Scope

This script should be run on the master domain.This script is not available in RDF Cloud Service.

Frequency

This script should be run on a monthly or quarterly basis or after new Floating Event Indicators have been loaded (preevnindt<event number>).

Description

This script should only be used in non-causal implementations. This script will calculate the lifts generated by floating events based on the Floating Event Indicator measures (preevnindt<event number>).

The script follows these steps:

1. Validate arguments

2. Use mace to run the genfloatingHBI rule group on the master domain. This is in the PrepDemandCommon solution in the configuration
3. Use mace to run the genfloating rule group on the local domains using para_spawn from BSA. This is in the PrepDemandCommon solution in the configuration.
4. Exit.

Additional Required Environment Variables

None

Required Arguments

Table 7–43 lists the required arguments.

Table 7–44 Required Arguments for rdf_gen_float_lift.ksh

Parameter	Short Description	Valid Values	Description
-d	Master domain	Path to master domain	Path to master domain

Optional Arguments

None

Optional Flags

None

Helper Scripts

RDF has two helper scripts to support the other scripts:

- RDF Environment Script
- RDF Functions Script

RDF Environment Script

Script Name

rdf_environment.ksh

Description

The scripts all source rdf_environment.ksh (except for the RETL integration scripts). The script, rdf_environment.ksh, sets various environment variables (if they are not already set) and sources two other scripts as shown in Table 7–45.

Table 7–45 Scripts Sourced by rdf_environment.ksh

Sourced Script	Description
bsa_common.sh	Invokes the Batch Scripting Architecture framework. Further details of this framework can be found in Oracle Retail Planning Batch Script Architecture Implementation Guide.
rdf_functions.ksh	Contains various helper functions used by the RDF scripts.

Table 7–46 lists the environment variables are set by rdf_environment.ksh if they are not already set in the environment or by the script that is sourcing rdf_environment.ksh.

Table 7–46 Scripts Set by rdf_environment.ksh (if not set another source)

Category	Environment Variable	Default Value	Description
Parallelization	BSA_MAX_PARALLEL	1	Parallelization variable used by the BSA framework. The maximum number of processes that can be started in parallel, from any spawning process.
	RPAS_PROCESSES	$BSA_MAX_PARALLEL	Parallelization variable used by some RPAS executables called by the RDF scripts.
Miscellaneous	BSA_ARCHIVE_DIR	$HOME	Archive directory used by the BSA framework. Currently not used by any RDF scripts.
	BSA_CONFIG_DIR	$HOME	Configuration directory used by the BSA framework. Currently not used by any RDF scripts.
	BSA_TEMP_DIR	$HOME	Directory used to store temporary files, for example: for sorting temporary space, for one-off files created during batch processes. This is used by both the BSA framework and RDF scripts directly.
	O_UNAME	See description.	Set to the result of calling the OS command uname. Used by scripts if there is OS specific processing. Note: This variable is always set to this value whether or not it was already set in the environment.
	RDF_WEEK_2_DATE_MEAS	WEEK2DATE	Used by some export scripts. Note: This variable is always set to this value whether or not it was already set in the environment.
	RPAS_INTEGRATION_HOME	${RPAS_HOME}/scripts/integration	This variable is used by some integration scripts. See Table A–1, " Environment Variables"
	TMP	/tmp	Temporary directory used by some scripts.

Table 7–46 (Cont.) Scripts Set by rdf_environment.ksh (if not set another source)

Category	Environment Variable	Default Value	Description
Logging	BSA_LOG_HOME	$HOME	The directory containing the log files generated by the BSA logging functionality.
	BSA_LOG_LEVEL	INFORMATION	Log level used by the BSA framework. The desired level of message logging. Used by both logging to screen and to log files. Valid values are PROFILE, DEBUG, INFORMATION, WARNING, ERROR and NONE.
	BSA_LOG_TYPE	1 (Text only)	Log type used by the BSA framework. The desired logging type. 1=Text Only. 2=XML Only. 3=Text & XML.
	BSA_SCREEN_ LEVEL	INFORMATION	Log level used by the BSA framework. The desired level of logging to the terminal. Valid values are PROFILE, DEBUG, INFORMATION, WARNING, ERROR, and NONE.
	log_path	See description.	Set to the path of the BSA log file. Used by some scripts to directly redirect some error messages to the log file when the BSA _call function cannot be used. NOTE: This variable is always set to this value whether or not it was already set in the environment.
	RPAS_LOG_LEVEL	$BSA_LOG_LEVEL	Log level used by the RPAS executables called by the RDF scripts. Valid values are all, profile, debug, audit, information, warning, error, none.

RDF Functions Script

Script Name

rdf_functions.ksh

Description

This script is sourced by the rdf_environment.ksh script which in turn is sourced by most of the RDF scripts. This script contains standard functions that are used by many scripts. The functions are detailed in Table 7–47.

Table 7–47 Functions for rdf_functions.ksh

Function	Argument	Argument Description	Function Description
_rdf_validate_ config_type	None	None	Validates that the RDF_CONFIG_TYPE environment variable is set to either 1 (on-premise) or 2 (cloud). Returns a NONZERO_EXIT code if not valid.
_rdf_get_config_ type	Master domain path	Path to master domain	Sets the RDF_CONFIG_TYPE environment variable to the value of the scalar measure Config Type (configType) in the domain. Then calls _rdf_validate_config_type. It assumes the master domain path has already been validated.
_rdf_set_config_ type	Config type	Should be 1 (on premise) or 2 (cloud)	Sets the RDF_CONFIG_TYPE environment variable to the value of the first argument. Then calls _rdf_validate_config_type. This should only be used when a domain is not available. If a domain is available, _rdf_get_ config_type should be called.
_rdf_validate_pp_ config_type	None	None	Validates that the RDF_PP_CONFIG_TYPE environment variable is set to either 1 (on-premise) or 2 (cloud). Returns a NONZERO_EXIT code if not valid.
_rdf_get_pp_ config_type	Master domain path	Path to master domain	Sets the RDF_CONFIG_TYPE environment variable to the value of the scalar measure preprocessing Config Type (ppsConfigType) in the domain. Then calls _rdf_validate_pp_ config_type. It assumes the master domain path has already been validated.
_rdf_set_pp_config_ type	Config type	Should be 1 (on premise) or 2 (cloud)	Sets the RDF_PP_CONFIG_TYPE environment variable to the value of the first argument. Then calls _rdf_validate_pp_ config_type. This should only be used when a domain is not available. If a domain is available, _rdf_get_pp_config_type should be called.
_rdf_validate_ master_domain	Master domain path	Path to master domain	Validates that the directory passed exists and that the domain is a master domain. If either test fails, it returns an INVALID_DOMAIN_ PATH error code.
_rdf_run_mace_ local_background	Master domain path Rule group	Path to master domain Rule group to run	Uses mace to run the rule group on the local domains using para_spawn from BSA. It assumes the master domain path has already been validated.
_rdf_run_mace_ expression_local_ background	Master domain path Rule expression	Path to master domain Rule expresssion to run	Uses mace to run the expression on the local domains using para_spawn from BSA. It assumes the master domain path has already been validated.
_rdf_run_mace_ master	Master domain path Rule group	Path to master domain Rule group to run	Uses mace to run the rule group on the master domain. It assumes the master domain path has already been validated.

Table 7–47 (Cont.) Functions for rdf_functions.ksh

Function	Argument	Argument Description	Function Description
_rdf_run_mace_ expression_master	Master domain path Rule expression	Path to master domain Rule expression to run	Uses mace to run the expression on the master domain using -processes of $BSA_ MAX_PARALLEL. It assumes the master domain path has already been validated.
_rdf_set_rdf_ export_dir_from_ domain	Master domain path	Path to master domain	Sets the RDF_EXPORT_DIR environment variable to be <master domain path>/from_ rdf. It also attempts to create the directory if it does not exist. It assumes the master domain path has already been validated. It only sets the variable if it is not already set - otherwise, it just makes sure the path exists.
_rdf_populate_ week_to_day_ measure	Master domain path	Path to master domain	Populates the week to day mapping date measure defined by the environment variable RDF_WEEK_2_DATE_MEAS (mapping to the last day in the week). The measure name is defined in rdf_environment.ksh. It is to be used for converting week IDs in export data to end-of-week dates to make it easier for integration with other applications. It assumes the master domain path has already been validated.

Implementation Scripts

RDF has two implementation scripts to support the implementation process:

- Run Plug-In Auto Generation
- Build RDF Domain

Run Plug-In Auto Generation

Script Name

rdf_auto_gen_config.ksh

Domain Scope

No domain is necessary for this script.

Description

This script will run the Configuration Tools plug-in automation from the command line so that it is not necessary to open the Configuration Tools. It will run the Prepare Demand, New Item, Curve, RDF, Promote, and Grade automation in that order for on premise domains. For cloud domains, it will run Prepare Demand, New Item, RDF, and Promote automation in that order.

Note: The `-clean` option should not be used if any customizations were made in any of the plug-in generated solutions.

The script follows these steps:

1. Validate arguments

2. Copy the taskflow.xml file to a temporary location.
3. If on cloud, copy the taskflow.xml_no_attribute file to a temporary location.
4. If on premise, run Curve:
   a. If –clean flag set, remove the Curve solution from the configuration.
   b. Call execPluginTask.sh for Curve:com.retek.labs.curve.plugin.installer.CurveCfgAutoGeneration on <configuration directory>/<configuration name>/<configuration name>.xml.
5. Run RDF:
   a. If –clean flag set, remove the RDF solution from the configuration.
   b. Call execPluginTask.sh for RDF:com.retek.labs.rdf.plugin.installer.RDFCfgAutoGeneration on <configuration directory>/<configuration name>/<configuration name>.xml.
6. Run Promote:
   a. If –clean flag set, remove the Promote solution from the configuration.
   b. Call execPluginTask.sh for Promote:com.retek.labs.promote.plugin.installer.PromoteCfgAutoGeneration on <configuration directory>/<configuration name>/<configuration name>.xml.
7. If on premise, run Grade:
   a. If –clean flag set, remove the Grade solution from the configuration.
   b. Call execPluginTask.sh for Grade:com.retek.labs.grade.plugin.GradeCfgAutoGeneration on <configuration directory>/<configuration name>/<configuration name>.xml.
8. Run New Item:
   a. If –clean flag set, remove the NewItem solution from the configuration.
   b. Call execPluginTask.sh for NewItem:com.retek.labs.newitem.plugin.installer.NewItemCfgAutoGeneration on <configuration directory>/<configuration name>/<configuration name>.xml.
9. Run Prepare Demand:
   a. If –clean flag set, remove the PrepDemand solution from the configuration.
   b. Call execPluginTask.sh for PrepDemand:com.retek.labs.preprocess.plugin.installer.PreprocessCfgAutoGeneration on <configuration directory>/<configuration name>/<configuration name>.xml.
10. Move the taskflow.xml file back from temporary location.
11. If on cloud, move the taskflow.xml_no_attribute back from temporary location.
12. Log informational success message.

Additional Required Environment Variables

None

Required Arguments

Table 7–48 lists the required arguments.

Table 7–48 Required Arguments for rdf_auto_gen_config.ksh

Parameter	Short Description	Description
-c	ConfigurationDir	Path of the parent directory of the configuration directory.

Optional Arguments

Table 7–49 lists the optional arguments.

Table 7–49 Optional Arguments for rdf_auto_gen_config.ksh

Parameter	Short Description	Valid Values	Description
-n	Configuration Name	Default is RDF	This is the configuration directory name.
-s	Configuration Type	Valid values are onpremise or cloud.	Default is onpremise.

Optional Flags

Table 7–50 lists the optional flags.

Table 7–50 Optional Flags for rdf_auto_gen_config.ksh

Flag	Short Description	Description
-clean	Clean flag	If used, the script will remove the solution from the configuration before calling the plug-in generation. This flag should not be used if any customizations were made in any of the plug-in generated solutions.

Build RDF Domain

Script Name

rdf_build_domain.ksh

Domain Scope

This script will be run when building or patching a domain.

Description

This script is an example script to show how an RDF domain can be built. This script does not support all of the features of rpasInstall. It does cover the options used most commonly for RDF. If this script is customized, the execPluginTask.sh calls should be kept or the domain may not build correctly.

The script follows these steps:

1. Validate arguments
2. If the configuration name is not supplied, set it to RDF if this is an onpremise call or RDFCS if it is a cloud call.
3. If the configuration directory is not set:
   a. If partition dimension is not set, set it to pgrp (Group).
   b. If the domain home is not set, set it to <current path>/domain.

4. If the configuration directory is set:
    a. Validate that the globaldomainconfig.xml file exists in the configuration directory.
    b. Set the domain home based on the contents of the globaldomainconfig.xml file.
5. If not a patch install, create the domain home directory if it does not exist.
6. Create hierarchy files and data files needed by the RDF solution.
    a. Call execPluginTask.sh for RDF:com.retek.labs.rdf.plugin.installer.InstallParameterDataGeneration with arguments "<configuration home>/<configuration name>/<configuration name>.xml" and input home.
    b. If patching, copy the files from input home to the master domain's input directory.
7. Create hierarchy files and data files needed by the Promote solution.
    a. Call execPluginTask.sh for Promote:com.retek.labs.promote.plugin.installer.PromotePosGenerator with arguments "<configuration home>/<configuration name>/<configuration name>.xml" and input home.
    b. If patching, copy the files from input home to the master domain's input directory.
8. If on-premise, create hierarchy files and data files needed by the Curve solution.
    a. Call execPluginTask.sh for Curve:com.retek.labs.curve.plugin.installer.InstallParameterDataGeneration with arguments "<configuration home>/<configuration name>/<configuration name>.xml" and input home.
    b. If patching, copy the files from input home to the master domain's input directory.
9. If on-premise, create hierarchy files and data files needed by the Grade solution.
    a. Call execPluginTask.sh for Grade:com.retek.labs.grade.plugin.GradeDataGenerator with arguments "<configuration home>/<configuration name>/<configuration name>.xml" and input home.
    b. If patching, copy the files from input home to the master domain's input directory.
10. Create hierarchy files and data files needed by the Prepare Demand (PrepDemand) solution.
    a. Call execPluginTask.sh for PrepDemand:com.retek.labs.preprocess.plugin.installer.PreprocessDataGenerator with arguments "<configuration home>/<configuration name>/<configuration name>.xml" and input home.
    b. If patching, copy the files from input home to the master domain's input directory.
11. Call rpasInstall to build the domain with the following parameters:
    a. If the configuration directory is not set:

Parameter	Value
Either: ■ -fullinstall ■ -patchinstall ■ -testinstall	Depends on whether –p or -t passed to build script
-ch	Configuration home
-cn	Configuration name
-in	Input home
-log	<log directory>/<log file>
-dh	Domain home
-verbose	not applicable
-p	Partition dimension
-updatestyles	not applicable
-rf	AppFunctions
-rf	RdfFunctions
-rf	ClusterEngine
-rf	LostSaleFunctions

b. If the configuration directory is set:

Parameter	Value
Either: ■ -fullinstall ■ -patchinstall ■ -testinstall	Depends on whether –p or -t passed to build script
-ch	Configuration home
-cn	Configuration name
-in	Input home
-log	<log directory>/<log file>
-configdir	Configuration directory
-verbose	not applicable
-updatestyles	not applicable
-rf	AppFunctions
-rf	RdfFunctions
-rf	ClusterEngine
-rf	LostSaleFunctions

12. Scan the build log for error messages.

13. If test install, skip to the last step.

14. Call rdf_load_measures.ksh on the master domain.

15. Log informational success message.

Additional Required Environment Variables

None

Required Arguments

None

Optional Arguments

Table 7–51 lists the optional arguments.

Table 7–51 Optional Arguments for rdf_build_domain.ksh

Parameter	Short Description	Valid Values	Description
-n	Configuration Name	Default is RDF	Default is RDF if on premise or RDFCS if on cloud.
-d	Domain home	Default is $PWD/../domain.	This is the path of the directory in which the domain will be created. Must be used with –r. Cannot be used with –o.
-c	Configuration home	Default is $PWD/../Domain Cfg/Version11.0/Promote.	This is the path to the directory containing the configuration.
-i	Input home	Default is $PWD/../Domain Build/to_rpas.	This is the directory containing the input files for the domain to be created.
-l	Log directory	Default is $PWD/..	Directory where the log file will reside.
-f	Log file	Default is build_domain.log.	This is the log file name.
-o	Configuration directory	This is the path to the directory containing the xml files used by RPAS.	This is a required argument if the user wants to supply globaldomainconfig.xml. The partition dimension (partitiondim) specified in globaldomainconfig.xml must match the Partition Dimension selected in the Forecast Common / Specify Configuration Details plug-in. Cannot be used with –d or –r.
-r	Partition dimension	Default is pgrp (Group)	This must match the Partition Dimension selected in the Forecast Common / Specify Configuration Details plug-in. Must be used with –d. Cannot be used with –o.
-s	Script type	Default is onpremise.	Valid values are onpremise or cloud.

Optional Flags

Table 7–52 lists the optional flags.

Table 7–52 Optional Flags for rdf_build_domain.ksh

Flag	Short Description	Description
-t	Test install flag	Run rpasInstall with the –testinstall flag.
-p	Patch install flag	Run rpasInstall with the –patchinstall flag.
-u	Usage flag	Displays the usage and then exits.

Other Scripts

RDF has several other scripts that are documented in other parts of this guide or in other guides as listed in Table 7–53.

Table 7–53 Other RDF Scripts

Script	Short Description	Reference Location
cpem_batch.ksh	CPEM batch script	Chapter 8, Cross Promotion Effects Module (CPEM)
cpem_build_domain.ksh	Script to build CPEM domain	Chapter 8, Cross Promotion Effects Module (CPEM)
cpem_e_rdf.ksh	Export CPEM results to RDF	Chapter 8, Cross Promotion Effects Module (CPEM)
cpem_load_measures.ksh	Load measures into CPEM domain	Chapter 8, Cross Promotion Effects Module (CPEM)
rdf_e_cpem.ksh	Export RDF data to CPEM	Chapter 8, Cross Promotion Effects Module (CPEM)
rdf_repos.ksh	Hierarchy conversion upgrade script for versions prior to 13.0.4.18	*Oracle Retail Demand Forecasting Installation Guide*
rdf_upgrade_new_item_store_export.ksh	Upgrade script for like item, like store and cloning from pre-15.0 to 15.0 and later	*Oracle Retail Demand Forecasting Installation Guide*
rdf_upgrade_new_item_store_load.ksh	Upgrade script for like item, like store and cloning from pre-15.0 to 15.0 and later	*Oracle Retail Demand Forecasting Installation Guide*

Executables

RDF calls these various executables from the scripts:

- PreGenerateForecast
- Generate
- RDFvalidate
- UpdateFnhbiRdf

PreGenerateForecast

PreGenerateForecast is an RDF executable that registers all measures with a birth date prior to forecast generation using generate. The first time PreGenerateForecast is run for a level, it registers the appropriate token measures for that level.
PreGenerateForecast may be run against the Master or a Local domain. At either level, the necessary measures to produce the batch forecast are registered across all domains.

PreGenerateForecast requires an input file in the form of an XML. The XML is configured with the following values:

Value	Description
FinalLevel	The Final Level Number that is used to generate the forecast.
OutputFile	The name of the resulting file located at the root of the domain after PreGenerateForecast is run. The OutputFile includes the values set for FinalLevel and Override in addition to the birth date. This date is the Forecast Generation Date, and it is passed to the domains when generate is run. The date is produced in the following format: yyyymmddHhhMmm (Example: 20050327H13M36). When this birth date is selected in the Forecast Approval wizard, it is viewed as: (03/27/2005 13:36).
Override	A True or False value. When generate is passed a True value, the Next Run Date is ignored, and the batch forecast uses today's date as the Next Run Date; and the batch is run. When generate is passed a False value, the batch forecast will run if the Next Run Date is the same as today's date.

Note: When the Run Batch template is used to generate the batch forecast, PreGenerateForecast is run automatically. Forecasts produced across Local domains using Run Batch cannot be aggregated in the Master domain because they do not share the same Forecast Generation Date.

PreGenerateForecast Usage

PreGenerateForecast -InputFile filename

InputFile is required.PreGenerateForecast must be called from either the master or the local domain directory since it does not have a –d option. The result of the script is the same.

The input file should be an XML file similar to Example 7–2:

Example 7–2 PreGenerateForecast Format

```
<Parameters>
   <Parameter>
      <Key>FinalLevel</Key>
      <Value>1</Value>
 </Parameter>

<Parameter>
    <Key>OutputFile</Key>
    <Value>MyOutput.xml</Value>
</Parameter>
<Parameter>
    <Key>Override</Key>
    <Value>true</Value>
    </Parameter>
</Parameters>
```

FinalLevel and OutputFile are required parameters of the XML file.

Override is an optional parameter of the XML file (default is **false**).

Other parameters may be included in the input XML file. They are passed through to the output XML file.

Return codes:

- 0 - Success (either ran pre-generate or did not need to run)
- 1 - Bad input
- 2 - Failure

To set the level of information provided, use -loglevel with values of: all, profile, debug, information, warning, error, or none. To disable timestamp header use -noheader.

Generate

Used to produce the batch forecast, generate is an RDF executable. This executable requires as an input, the OutputFile resulting from PreGenerateForecast and is called `generate.xml`.

This binary runs RDF's batch process. This executable, generate, can take two optional inputs: level and override.

Usage

```
generate -InputFile Filename
```

The `InputFile` is required. `generate` must be called from the local domain directory since it does not have a -d option.

Parameters

The following parameters setting are included in the input file:

- birth
- startdate
- finallevel
- override

The override input must be True or False. The defaulted value is False if this option is not included in the input file. When override is False, `generate.xml` only starts the batch process if current time is later than the next run date in the domain. When the override is True, `generate.xml` starts the batch forecast regardless of the next run date.

The generate binary invokes code in the BatchForecast library to run the batch process.

`finalLevel` and `birth` are required parameters of the XML file. `override` (**false**) and `StartDate` (Default Forecast Start Date) are optional parameters of the XML file (defaults in parentheses).

Return Codes

The return codes include:

- 0—Success (either ran generate or did not need to run)
- 1—Bad input
- 2—Failure

To set the level of information provided, use -loglevel with values of:

- all
- profile
- debug
- information
- warning
- error
- none

To disable timestamp header use -noheader.

The input file should be an XML file that looks similar to the following example:

Example 7–3 XML Format for generate

```
<Parameters>
    <Parameter>
         <Key>Birth</Key>
         <Value>20041027H11M52</Value>
</Parameter>
<Parameter>
       <Key>StartDate</Key>
       <Value>20041027</Value>
</Parameter>
<Parameter>
        <Key>FinalLevel</Key>
        <Value>1</Value>
</Parameter>
<Parameter>
        <Key>Override</Key>
        <Value>true</Value>
   </Parameter>
</Parameters>
```

RDFvalidate

`RDFvalidate` automatically runs during the domain install, and it can also be run at any time against a Master or one subdomain. If run against the Master Domain, it checks the master and all subdomains. If run against a subdomain, it checks the Master and only the subdomain (not all other subdomains). This function verifies that:

- If there is a partition dimension, it must be along the product hierarchy.
- Domains are cleanly partitioned, this means that for the partition dimension, there exists only one position in each local domain, whether partitioning along the main or an alternate (or branch) product hierarchy.
- All data, measures, and levels are defined properly based on the partition dimension.
- Causal parameters are properly defined based on final, source, and causal levels.

Usage

The usage of `rdfvalidate -d pathToDomain` is as follows:

Script Name

```
rdfvalidate -d pathToDomain
```

To get this usage text, use -?, -help, or -usage. To get the version of this utility, use -version. To set the level of information provided, use -loglevel with values of: all, profile, debug, information, warning, error, or none. To disable timestamp header use -noheader.

RDF Validation

Table 7–54 displays the validation performed internally by the plug-in and the RDFvalidate utility.

Table 7–54 Internal Validation Performed by the Plug-in and RDFvalidate utility

Validation Area	Steps
Hierarchies and Dimensions	a. Verify day dimension exists on calendar hierarchy
	b. If there is a partition dimension, it must be along the product hierarchy.
For Final Levels	a. Intersection (fintxlxb) ■ Cannot be blank ■ Must be at or below all source level intersections ■ Must be at or below the partition dimension on the partition branch
	b. Seasonal profile (seasprofxlxb) can be either: ■ Blank ■ Measure name (only one) ■ Must be valid measure ■ Should be of type real ■ Measure intersection must be equal to the level intersection
	c. Source data (datasrcxlxb) must be a measure name (only one) ■ Must be a valid measure ■ Should be of type real ■ Measure intersection must be at or below the final level intersection
	d. Plan data (r fplanxlxb) must be either: ■ Blank ■ Measure name (only one) ■ Must be valid measure ■ Should be of type real ■ Measure intersection must be equal to the final level intersection

Table 7–54 (Cont.) Internal Validation Performed by the Plug-in and RDFvalidate utility

Validation Area	Steps
For Source Levels	**a.** Intersection (fintxlxb) ■ Cannot be blank ■ Must be at or above final level intersection ■ Must contain a dimension from the partition hierarchy ■ Must be either: ■ At or below the partition dimension on the partition branch. ■ On a branch of the partition hierarchy. If on a branch of the partition hierarchy, also check if domains are cleanly partitioned (executable only). This means for the branched dimension on the partition hierarchy, each position for that dimension can exist in only one sub-domain.
	b. Seasonal profile (seasprofxlxb) can be either: ■ Blank ■ Measure name (only one) ■ Must be valid measure ■ Should be of type real ■ Measure intersection must be equal to the level intersection
	c. Spreading profiles (sprdprofxlxb) ■ Can only be blank if source level intersection equals final level intersection ■ Must be comma-separated list of Curve levels and measure names (can be mixed) ■ If Curve level, must be a valid Curve level (final profile) ■ If measure: ■ Must be a valid measure ■ Should be of type real ■ Measure intersection must be at or higher than final level

Executable Only Table 7–55 displays the validation performed internally by the RDFvalidate utility.

Table 7–55 RDFvalidate Utility

#	Executable Only	Steps
1	Domains are Cleanly Partitioned	**a.** Verify that there is only one partition dimension per subdomain.
2	For Final and Source Levels	**a.** Causal Aggregation Profile (aggxlxb) values should be either: ■ Blank ■ Measure name (one only) ■ Should be a valid measure ■ Should be of type real ■ The intersection of the measure must be at or above final level
		b. Causal Calculation Intersection (calcintxlxb) values should be either: ■ Blank ■ Intersection Intersection must be valid: ■ Must contain the calendar dimension ■ Must be at or above level intersection
		c. Causal Data Source (calcdtsrcxlxb) values should be either: ■ Blank ■ Measure name (one only) ■ Should be a valid measure ■ Should be of type real ■ The intersection of the measure must be at or above level intersection
		d. CausalHigher Intersection (cslhint) values should be either: ■ Blank ■ Intersection ■ Must be valid intersection ■ Must not contain the calendar dimension ■ Must contain a dimension from the partition hierarchy. ■ Must be at or above level intersection ■ Must be either: ■ At or below the partition dimension on the partition branch. ■ On a branch of the partition
	Note: If on a branch of the partition hierarchy, also check if domains are cleanly partitioned (executable only). This means that for the branched dimension on the partition hierarchy, each position for that dimension can exist in only one sub-domain.	

Table 7–55 (Cont.) RDFvalidate Utility

#	Executable Only	Steps
2. (continued)	For Final and Source Levels (continued)	**e.** Causal Spread Profile (spreadxlxb) values should be either: ■ Blank ■ Measure name (one only) ■ Should be a valid measure ■ Should be of type real ■ The intersection of the measure must be at or above final level
		f. Deseasonalized Demand Array (ddemandxlxb) values should be either: ■ Blank ■ Measure name (one only) ■ Should be a valid measure ■ Should be of type real ■ The intersection of the measure must be the level intersection less the calendar dimension
3.	For Final Levels only	**a.** Default History Start Date (defhstdt) values should be either: ■ Blank ■ A date within the calendar
		b. Forecast Start Date (dfxlxb) values should be either: ■ Blank ■ A date within the calendar

Promote Validation Plug-in and Executable

1. Hierarchies and Dimensions:
   Check whether or not PTYP, FLVL, and PROM exist in Data Hierarchy. If not, create them.
2. Promotion Names:

   Check if promotion names have 1 to 4 characters.
3. Causal levels must be at or below the partition dimension on the partition branch.

UpdateFnhbiRdf

UpdateFnhbiRdf is required after Generate is run if an alternate hierarchy dimension from the Product hierarchy is used as a dimension in a forecast level. It performs the following functionality:

- Checks that certain measures are cleanly partitioned
- Copies corresponding cells (based on the partition) from each sub-domain to the master domain
- Runs automatically with the Run Batch wizard
- After ensuring that the FNHBI (Forced non-Higher Based Intersections) measures are cleanly partitioned, UpdateFnhbiRdf copies corresponding cells (based on the

partition dimension) from each sub-domain into the master domain

Usage

The usage of `UpdateFnhbiRdf -d pathToDomain -InputFile filename` is as follows:

Script Name

`UpdateFnhbiRdf -d pathToDomain -InputFile filename`

To get this usage text, use `-?`, `-help`, or `-usage`. To get the version of this utility, use `-version`. To set the level of information provided, use `-loglevel` with values of: `all`, `profile`, `debug`, `information`, `warning`, `error`, or `none`. To disable timestamp header, use `-noheader`.

The `InputFile` format expected is as printed by the usage information. The timestamp or the birth key will have to be the same as the one output by pregenerateForecast, that is used by `generate.xml`.

8

Cross Promotion Effects Module (CPEM)

Cross Promotion Effects Module (CPEM) is used to generate the crossing promotion effects between merchandise and location. There are two types of cross effects that are calculated at corresponding levels:

- Cannibalization effects
- Halo effects

The Cannibalization level and Halo level are to define at which product and location levels the Cannibalization or Halo cross effects are produced. The Halo level should be higher than Cannibalization level along hierarchy.

> **Note:** The Cross Promotion Effects Module is not available for RDF Cloud Service.

RDF was launched with Promotional Halo and Cannibalization capabilities in January 2013. The functionality has had limited adoption due to various factors including necessary data required to support reliable results, market maturity and implementation complexity. In an attempt to share lessons learnt with our experience so far, we would recommend the following to our customers considering implementing this functionality:

Starting Conditions

Both Halo and Cannibalization deal with the cross item effects of running promotions, that is, the impact of promoting an item on the sales of other related items. The most important inputs for estimating these cross effects are a robust baseline forecast and self-promotion effects from RDF Causal. Hence, we recommend that CPEM be considered for implementation only after RDF Causal has been up and live for a 12-18 month period, delivering reliable baseline and causal forecasts to drive business decisions.

- **Replacement SKUs** — All historical SKUs that are replacement or promotional variants of each other need to be grouped together into a Plan SKU. The Plan SKU level needs to be used within RDF for estimating both baseline forecasts and causal forecasts.
- **Appropriate pre-processing** — That corrects for missing sales periods and other factors that might bias the estimated baseline forecast and hence impact that quality of CPEM output.

Set Up for Cannibalization

Careful configuration and pre-implementation set up is required for Cannibalization to run. It is important to note that Cannibalization (though configurable) is designed to run at an item-group level (as defined in the following list) and not the individual SKU level.

Note: Cannibalization needs to stay inside a local domain.

- **Item-group or L1 Set Up** — This is the level at which Cannibalization is estimated. SKUs have to be grouped (outside the system and fed in). These are groups of items that are typically promoted together. For example, in the yogurt Category, SKUs of a particular brand, size, fat content but different flavor variants. An attribute analysis exercise, should inform this L1-grouping. What are the key combinations of attributes that need to be the same for all items in an L1-group and what attributes can be different? It is important to note that these attributes will vary significantly, from Category to Category.
- **Cannibalization-group Or L2 Set Up** — L2 is the level within which Cannibalization is estimated. Only item group or L1 within a Cannibalization group are analyzed for possible Cannibalization, when one or more items are promoted. This could be a grouping of one or more classes or sub-classes. Note that L2 grouping needs to be a roll up of L1 groupings, that is, every L1 item-group needs to cleanly map to one and only one L2 grouping (many to one mapping from L1 to L2).
- **L1-pairing Set Up** — In addition to set up of L1 or Item-groups, RDF's CPEM also needs to be told of legal/possible L1-pairs. This requires careful analysis on a Category by Category basis. For example. SKUs within a certain regular price range could be considered Cannibalistic and hence valid L1-pairs. It is recommended that this analysis be conducted carefully, ideally in conjunction with Customer Decision Trees (CDTs) to ensure the correct pairings are set up for each Category.
- **Co-promotion Effects are Turned Off** — A system flag that effectively looks for conditions when both L1 pairings are promoted at the same time and accounts for it appropriately.

Halo effects

Current functionality is designed for a limited use case for businesses with minimal promotional activity. The solution aims to estimate Halo effects by observing the impact on sales of other subclasses when a subclass is promoted. As we vetted this with a broad set of retail use cases with significant promotional activities, we recognized that this functionality is not adequate. We are revisiting our approach to estimating Halo effects and welcome partnerships with retailers.

CPEM is a standalone RPAS instance and its hierarchies including calendar, product, location, and RHS product, are the same as the ones in RDF and all measure data files are imported from RDF. RDF prepares the CPEM needed measure data files based on the Cannibalization and Halo level, so CPEM and RDF should have exactly the same Cannibalization and Halo levels set for CPEM to work. In other words, if you change the cannibalization or Halo level in RDF (CPEM), then you should make the same changes in CPEM (RDF).

Functionality

CPEM performs the following functionality:

- Define and override the cross promotion estimation related parameters in the Effect Estimation Administration workbook.
- Run the CPEM batch script to produce both Cannibalization and Halo promotion effects. The CPEM determines the regression based on the promotion, baseline sales, sales, and price information to produce the cross effects.
- Review and approve the produced crossing promotion effects in the Effect Estimation Review and approval workbook.
- Export the approval cross promotion effects into RDF.

Input

CPEM is a standalone RPAS instance and it is a single domain.

Hierarchy Files

The hierarchy files of Product, Calendar, location and RHS Product are the same as the RDF domain.

1. Calendar (CLND) Hierarchy File
2. Merchandise or Product (PROD) Hierarchy File
3. Location (LOC) Hierarchy File
4. RHS Merchandise or Product (PROR) Hierarchy File

Measures Data Files

The measures data files needed in CPEM are imported from RDF as listed in Table 8–1.

Table 8–1 CPEM Measure Data Files imported by RDF

Measure Name	Intersection	Description
slscann	Cannibalization Level + week	The aggregated weekly sales history at cannibalization level.
slsbaselinecann	Cannibalization Level + week	The aggregate weekly baseline sales history at cannibalization level.
normprccann	Cannibalization Level + week	The weekly normalized price at cannibalization level.
promoindcann	Cannibalization Level + week	Weekly Promotion indicator at cannibalization level.
slsHalo	Halo Level + week	The aggregated weekly sales history at Halo level.
slsbaselineHalo	Halo Level + week	The aggregate weekly baseline sales history at Halo level.
normprcHalo	Halo Level + week	The weekly normalized price at Halo level.
promoindHalo	Halo Level + week	Weekly Promotion indicator at Halo level.

Output

CPEM produces the cross promotion effects and then outputs the cross promotion cannibalization effects and Halo effects as listed in Table 8–2.

Table 8–2 Measure Data Files Output by CPEM

Measure Name	Intersection	Description
FinCannEff	Cannibalization Level + RHS Cannibalization Product	The cross promotion cannibalization effects.
FinHaloEff	Halo Level + RHS Halo Product	The cross promotion Halo effects.

Installation Considerations

The following sections provide an overview of CPEM installation.

Note: For additional information, refer to the chapter, "CPEM Installation" in the *Oracle Retail Demand Forecasting Installation Guide*.

Installation Dependencies

RPAS and RDF must be installed before setting up and configuring CPEM. For information on installing these products, refer to the *Oracle Retail Predictive Application Server Installation Guide* and *Oracle Retail Demand Forecasting Installation Guide*.

Environment Setup

Before downloading the installation package to the UNIX server, a central directory structure to support the environment needs to be created. This central directory is referred to as `<cpem_directory>`. Set `<cpem_directory>` to the full path name to the $CPEM_HOME variable.

CPEM Installer

The CPEM installer performs the following functions:

- Downloads the configuration and batch scripts into the `<cpem_directory>/config` and `<cpem_directory>/bin` directories
- Downloads a set of sample hierarchy and data files into the `<cpem_directory>/input` directory
- Builds a sample domain at `<cpem_directory>/domain/cpem`

Custom Domain Build

To do a custom build of a domain:

1. Update the `globaldomainconfig.xml` file with the correct domain paths.
2. If needed, update the default environment variables in `environment.sh`.
3. Run the `cpem_build_domain.ksh` script: `./ cpem_build_domain.ksh`

CPEM Taskflow for the RPAS Fusion Client

The CPEM installation software enables you to install the taskflow and online help files for the RPAS Fusion Client. In order to install the taskflow files, the RPAS Fusion Client must already be installed. For more information on installing the RPAS Fusion Client, see the *Oracle Retail Predictive Application Server Installation Guide*.

During the RPAS Fusion Client installation, the installer automatically sets up the RPAS domain connection configurations in the `ProfileList.xml` file. If you choose to set up the domain connection after the installation or set up an additional domain, you must manually set up the connection. For more information, see the *Oracle Retail Predictive Application Server Administration Guide for the Fusion Client*.

Batch Processing

CPEM has a batch script to generate the cross promotion effects and export the final effects. The following batch scripts are available in CPEM:

- CPEM batch script
- Cross effect export script

Running the Batch Process

Before running the batch script, set these two system variables:

- CHECK_PROMO_DIFF: It is recommended to set to YES. YES turns off the co-promotion. When set to NO, RDF considers the co-promotion case.
- PROMO_THRESHOLD: A numeric value for the promotion threshold. Only the week the promotion value is greater or equal then this threshold will be treated as "on promotion".

To generate the cross effects, run the `cpem_batch.sh` script in the domain.

```
./cpem_batch.sh
```

To export the cross promotion effects, run the `exportCPEMtoRDF.ksh` script in the domain

```
./cpem_e_rdf.ksh
```

CPEM Scripts

This section describes CPEM scripts including:

- Build CPEM domain
- Load CPEM Measure Data
- Export RDF Data to CPEM
- Export CPEM Data to RDF
- CPEM Batch

> **Note:** With this release, some script names have been changed. For more information, refer to Appendix J, "RDF Script Names."

Build CPEM domain

Script Name

cpem_build_domain .ksh

Frequency

This script runs when building or patching a CPEM domain.

Description

This script is an example script to show how a CPEM simple domain can be built. This script does not support all of the features of rpasInstall. It does cover the options used most commonly for CPEM.

Steps

These are the steps of the script:

1. Validate arguments.
2. If the configuration name is not supplied, set it to CPEM.
3. If the domain home is not set, set it to <current path>/../domain.
4. If not a patch install, create the domain home directory if it does not exist.
5. Call rpasInstall to build the domain with the following parameters:

Parameter	Value
either: ▪ -fullinstall ▪ -patchinstal ▪ -testinstall	Depends on whether –p or -t passed to build script
-ch	Configuration home
-cn	Configuration name
-in	Input home
-log	<log directory>/<log file>
-dh	Domain home
-verbose	not applicable
-updatestyles	not applicable
-rf	AppFunctions
-rf	RdfFunctions

6. Scan the build log for error messages.
7. If test install, skip to the last step.
8. Call cpem_load_measures.ksh on the domain.
9. Log informational success message.

Additional Required Environment Variables

None

Required Arguments

None

Optional Arguments

Parameter	Short Description	Description
-n	Configuration name	Default is CPEM.

Parameter	Short Description	Description
-d	Domain home	Default is $PWD/../domain. This is the path of the directory in which the domain will be created.
-c	Configuration home	Default is $PWD/../config. This is the path to the directory containing the configuration.
-i	Input home	Default is $PWD/../input. This is the directory containing the input files for the domain to be created.
-l	Log directory	Default is $PWD/.. Directory where the log file will reside. -fLog fileDefault is build_domain.log. This is the log file name.

Optional Flags

Flag	Short Description	Description
-t	Test install flag	Run rpasInstall with the –testinstall flag.
-p	Patch install flag	Run rpasInstall with the –patchinstall flag.
-u	Usage flag	Displays the usage and then exits.

Load CPEM Measure Data

Script Name

cpem_load_measures.ksh

Domain Scope

This script should be run on the CPEM domain.

Description

This script will load the measures needed for CPEM. The input files need to be present in the domain input directory.

Steps

These are the steps of the script:

1. Validate arguments
2. For the measures cannitbasesls, cannitsls, cannpromoflag, cannpromoprice, itembrand, itempcksz, itemsz, itemuom, iterbrand, iterpcksz, itersz, and iteruom call loadmeasure with the following parameters:

Parameter	Value
-d	Path to the CPEM domain
-measure	Current measure
-processes	$BSA_MAX_PARALLEL

Parameter	Value
-recordLogLevel	recordLogLevel passed to the script (if provided)

3. If the directory <current directory>/../translations/CPEM exists and there are files that start with "r_" in it, then:
   a. Copy the files to <CPEM domain>/input directory.
   b. Call loadmeasure on the following files (using the same parameters as above): r_dimlabel, r_hierlabel, r_measdescripti, r_measlabel, r_measpicklist, r_msglabel, r_wbtglabel, r_wbtlabel.
4. Exit.

Additional Required Environment Variables

None

Required Arguments

Parameter	Short Description	Valid Values	Description
-d	CPEM domain	Path to CPEM domain	Path to the CPEM simple domain.

Optional Arguments

Parameter	Short Description	Valid Values	Description
-r	recordLogLevel	error, warning, information, and profile	Sets a logging level for record loading issues. If the logging level set at implementation time is less verbose than the record logging level, then record issues will not be logged. If utility's logging level is at same or higher verbosity as the record logging level, the record issues will be logged with the log indicator as set using this argument.

Optional Flags

None

Export RDF Data to CPEM

Script Name

rdf_e_cpem.ksh

Domain Scope

This script should be run on the RDF master domain.

Description

This script exports a pre-determined list of measures from RDF for use by CPEM.

Steps

These are the steps of the script:

1. Validate arguments.
2. Use mace to run the Xpromo_preproc rule group on the local domains using para_spawn from BSA.
3. Export measures at the specified intersections to the specified files names to the destination path using exportMeasure.

Measure Name	File Name	Export Intersection
itbrand	itembrand.csv.ovr	item
itpcksize	itempcksz.csv.ovr	item
itsize	itemsz.csv.ovr	item
ituom	itemuom.csv.ovr	item
pos	cannitsls.csv.ovr	weekitemstr_
pvarxlpprc	cannpromoprice.csv.ovr	weekitemstr_
rdfpromoind	cannpromoflag.csv.ovr	weekitemstr_
regprice07xb	cannpromoprice.csv.ovr	weekitemstr_
totadjbasesls	cannitbasesls.csv.ovr	weekitemstr_

4. Copy the following files:

Original file name	Copied file name
itembrand.csv.ovr	iterbrand.csv.ovr
itempcksz.csv.ovr	iterpcksz.csv.ovr
itemsz.csv.ovr	itersz.csv.ovr
itemuom.csv.ovr	iteruom.csv.ovr

5. Log informational success message.

Additional Required Environment Variables

None

Required Arguments

Parameter	Short Description	Valid Values	Description
-d	RDF master domain	Path to RDF master domain	Path to the RDF master domain.
-out	Output file path	Any valid directory name according to OS	

Optional Arguments

None

Optional Flags

None

Export CPEM Data to RDF

Script Name

cpem_e_rdf.ksh

Domain Scope

This script should be run on the CPEM domain.

Description

This script exports a pre-determined list of measures from CPEM for use by RDF.

Steps

These are the steps of the script:

1. Validate arguments.
2. Use mace to run the Xpromo_preproc rule group on the local domains using para_spawn from BSA.
3. Export measures at the specified intersections to the specified files names to the destination path using exportMeasure.

Measure Name	File Name	Export Intersection
FinChgRatioHalo	halochgratio.csv.ovr	rgn_clsr
FinMaxChgRatioHalo	halomaxratio.csv.ovr	rgn_clss
FinSpProfileHalo	apphaloeff.csv.ovr	clssrgn_clsr
itemchgratioCann	cannchgratior.csv.ovr	rgn_iter
itemmaxchgratioCann	cannmaxratio.csv.ovr	rgn_item
itemspprofileCann	appcanneff.csv.ovr	itemrgn_iter

4. Convert the file cannchgratior.csv.ovr into cannchgratio.csv.ovr by swapping the first two fields.
5. Remove the temporary file cannchgratior.csv.ovr.
6. Log informational success message.

Additional Required Environment Variables

None

Required Arguments

Parameter	Short Description	Valid Values	Description
-d	CPEM domain	Path to CPEM domain	Path to the CPEM domain.
-out	Output file path	Any valid directory name according to OS	

Optional Arguments
None

Optional Flags
None

CPEM Batch

Script Name
cpem_batch.ksh

Domain Scope
This script should be run on the CPEM domain.

Description
This script runs the rule groups to calculate the cross promotional effects.

Steps
These are the steps of the script:

1. Validate arguments.
2. Calls mace to run the following rule groups in the specified order on the CPEM domain:
   a. pre_data
   b. Calc_Markdownsls
   c. pre_process
   d. pre_cann
   e. batch_cann
   f. Batch_Cann_Ratio
   g. Batch_Cann_Prof
   h. pre_halo
   i. batch_halo
   j. Batch_Halo_Ratio
   k. Batch_Halo_Prof
   l. Batch_export
3. Log informational success message

Additional Required Environment Variables
None

Required Arguments

Parameter	Short Description	Valid Values	Description
-d	CPEM domain	Path to CPEM domain	Path to the CPEM domain.

Optional Arguments

None

Optional Flags

None

9

AutoSource

The AutoSource binary may be used to determine the optimal source level for a product/location. For the final level specified, AutoSource produces a forecast using each source level. The source level that produces the best PAE (Percent Absolute Error) for a time series is selected as the Optimal Source Level. The AutoSource results may be accessed by the user through the Forecast Maintenance workbook. If the Optimal Source Level is to be used for a product/location, the `Use Optimal Source` parameter should be set to True.

The AutoSource binary invokes code in the BatchForecast library to run the batch process. AutoSource can take four inputs: mode (required), finallevels (required), today, and timelimit (required for ONCEONLY and CYCLE mode).

The AutoSource binary does the following:

- Provides a starting Source Level recommendation for new forecasting customers. The recommended Source Level can be applied to the Final Level, which would allow the user to be focused on other tuning activities.
- Is helpful for existing customers that are starting to forecast new businesses. AutoSource can be included as an activity in the customer's forecasting roll-out process.
- Is useful for merchandise groups that have shifting demand patterns due to business or market changes such as pricing and marketing strategy changes, or product realignment.

AutoSource uses the forecast horizon to compute the PAE (Percent Absolute Error). If the forecast horizon is changed from the default of 13 weeks, AutoSource starts forecasting that number of weeks back. For example, if you have a forecast horizon of 52 weeks, AutoSource starts its analysis 53 weeks before today. This approach can disallow Winters and Seasonal models if sufficient calendar is not available. If the forecast horizon is 52 weeks, you should have at least 3.5 years of history for AutoSource to be able to perform all of its analysis.

Unlike Generate, there is no interim forecast calculation in AutoSource. Instead, AutoSource attempts to generate an AutoES result at the final level, then uses that result to perform the source level spreads.

Note: If the time series data is dense enough at the final level, the spread is not based on a linear contribution to the source. It will not maintain the source shape, and it will make recommendations based on such spreads.

AutoSource makes an initial recommendation to all the product/location combinations with sufficient data to perform analysis. Subsequent PAE calculation and comparison only occurs to these product/location combinations. The product/location combination without sufficient data (total sales = 0 during history region or total sales = 0 during forecast evaluation region) will not get any recommendation.

Inputs to AutoSource Binary

AutoSource is invoked from a script or the command line. The binary inputs are detailed in Table 9–1 and Table 9–2:

Table 9–1 AutoSource Binary Input Descriptions

Binary Inputs	Description		Example
-d	Relative or absolute path to domain		-d /cygdrive/c/Domains/RDF/ldom2
-mode	Includes the following options:		
	RESTART	Resetting measures, such that the next run starts without prior information. This option does not actually kick off any source level optimization run. Use this option when a clean run is desired, and then run AutoSource with one of the following next modes.	-mode RESTART
	CYCLE	If AutoSource doesn't complete an optimization run due to the time limit, the next time it is run it picks up where it left last time. For instance, if there are 10 source levels and during one run AutoSource only evaluated 3 source levels, then the next time it runs it optimizes source levels 4 and up. CYCLE without a time limit will never finish. Once the last source level was evaluated, AutoSource starts with the first level again.	-mode CYCLE
	ONCEONLY	AutoSource completes the run or stops when the time limit is up	-mode ONCEONLY
-flvllist	A list of the final forecast levels to be optimized		-flvllist 1

Table 9–2 AutoSource Optional Binary Input Descriptions

Optional Binary Inputs	Description	Example
[-today]	Specifies the date when AutoSource stops the evaluation of the forecast error. The evaluation starts at the date given by today minus the number of periods specified in the forecast length. Hence the time interval over which AutoSource evaluates the forecast error is [today - forecast length, today]. The date should be in the RPAS format stored in the dim_day array.	-today D20010101
[-timelimit]	Time, in minutes allowed AutoSource to run. It is optional for RESTART mode, but required for ONCEONLY and CYCLE mode. It needs to be greater than 0 to allow AutoSource to run.	-timelimit 10,000
[-noclear]	This is a flag indicating if temporary information should be deleted. If not specified, the temporary information is deleted.	-noclear

Example 9–1 AutoSource Binary Example 1

```
AutoSource -d . -mode RESTART -flvllist 01,06 -today 20020101 -timelimit 10
AutoSource -d . -mode ONCEONLY -flvllist 01,06 -today 20020101 -timelimit 10
```

Example 9–2 AutoSource Binary Example 2

```
AutoSource -d . -mode CYCLE -flvllist 01,06 -today 20020101 -timelimit 10
```

If only running AutoSource periodically, then use the RESTART and ONCEONLY modes. If the run exceeds the time limit during a RESTART run, then ONCEONLY should be run. If you want to start from the beginning, RESTART and ONCEONLY should be run again.

If AutoSource is scheduled as part of the daily cron job, use CYCLE. CYCLE runs RESTART and then ONCEONLY consecutively.

Refer to the *Oracle Retail Demand Forecasting User Guide* for specifics pertaining to the Forecast Maintenance Workbook and picking optimal levels.

Note: For item/stores that are new or highly seasonal, AutoSource may not return the best recommendation since new items may not have an adequate sales history length and highly seasonal items may only sell for a short period during the year. For these items, you should not set the AutoSource recommendation as default at the final level. Only use AutoSource recommendations for item/stores that have an adequate sales history.

AutoSource Measures

The following AutoSource measures are available in the Forecast Maintenance workbook.

Optimal Source Levels

Displayed only at final levels, a value is populated in this field if AutoSource has been run on the final level. The AutoSource executable evaluates all levels associated to a final level and returns the Source Level that yields the optimal forecast results or lowest error.

Pick Optimal Level

Set only at final levels, a check mark in this field indicates that the batch forecast should use the Optimal Source Level selected by AutoSource.

The final level measure Optimal Source Levels is used for reference. The RDF user can view the optimal Source Level that was determined by AutoSource. This Source Level was chosen by generating forecasts at all Source Levels and determining the lowest forecast error (PAE) at the final level.

If the user would like to use the Optimal Source Level during forecast generation they can set the Pick Optimal Level Boolean measure to True.

If Pick Optimal Level is set to True, when forecast generation is run, the optimal Source Level is used. The Forecast Method set at the optimal Source Level and the additional associated forecast parameters is also used.

Usage

AutoSource -d pathToDomain -mode RESTART/ONCEONLY/CYCLE -flvllist lvlx,lvly

[-today] todayString (the same format as in dim_day)

[-timelimit] minutes (optional for RESTART mode, but required for ONCEONLY and CYCLE mode)

[-noclear]

To get this usage text, use -?, -help, or -usage.

To get the version of this utility, use -version.

To set the level of information provided, use -loglevel with values of: all, profile, debug, information, warning, error, or none.

To disable the timestamp header, use -noheader.

The mode input must be one of RESTART, CYCLE, or ONCEONLY.

The flvllist must be a comma separated list of final levels.

The today input must be the same format as in dim_day, For example, YYYYMMDD.

The timelimit is in minutes.

- RESTART: This mode initializes the system in preparation for a new Autosource batch process.
- ONCEONLY: This mode runs the Autosource batch process until it completes or until the timelimit has been reached (whichever comes first).

> **Note:** In order to run in ONCEONLY mode, RESTART mode has to be run first.

- CYCLE: This mode continuously runs the Autosource batch process by first running the RESTART mode, and then running ONCEONLY. The CYCLE mode allows the Autosource batch process to always use the latest data in determining the optimal source level for a prod/loc.

> **Note:** For Autosource usage examples, refer to Example 9–1 and Example 9–2.

10

Forecast Approval Alerts

This chapter describes Forecast Approval Alert configuration.

About Alerts

Alerts can be configured through the RPAS Configuration Tools or can be manually registered in the domain. The alert expressions require familiarity with the RPAS rule functions. Registering an alert with the alert category of FORECAST_APPROVAL allows RDF to use the alert expression during the batch forecasting process to determine if a time series is automatically approved. When this category of alert is registered, the pick lists for Default Approval Method (in Forecast Administration) and the Approval Method Override (in Forecast Maintenance) are updated to include the label of the alert.

Note that the label of the alert when displayed in the list has certain special characters replaced with a space. These characters are:

- comma
- colon
- right parenthesis
- left parenthesis

You can select the alert for any product/location.

The following steps are an example of Forecast Approval Alert configuration using the example domain that is provided in the release package:

Prerequisite: Build Global Domain

1. Option 1:

   Run PreGenerateForecast or Generate

   Option 2:

   Use regTokenMeasure to Manually Register Any Token Measures Needed to Support the Alert Expression
2. Register the Alert Measure
3. Register the Expression for the Forecast Approval Alert

Prerequisite

Build Global Domain

Using the Mock Install Configuration, build the global domain environment.

Step 1 (Option 1)

Run PreGenerateForecast or Generate

If using a pristine global domain or simple domain environment, token measures have yet to be registered in the domains. Since you do not know the specific birth date at configuration time, token measures allow for measures with birth dates (a time stamp applied during the batch) to be evaluated. The token measure that we are using in this example is System Forecast for level 1 (sf01). The registration of the token measures can be accomplished by running PreGenerateForecast (in a global domain environment) or Generate (in a simple domain environment). This removes the need to manually run `regTokenMeasure`.

Step 1 (Option 2)

Use regTokenMeasure to Manually Register Any Token Measures Needed to Support the Alert Expression

If you prefer to manually register the token measures, the regTokenMeasure must be run with -FNHBI option if in a global domain environment. This allows the token measures to have different values across subdomains. The token measure requires a value to the measure while registering. In Example 10–1, the token measure is registered in the Master Domain and are made to be equal to pos (Weekly Sales) since pos has the same base intersection (item/store/week) and data type (real) as the System Forecast for level 1.

Example 10–1 regTokenMeasure

```
C:\Domains\RDF>regTokenMeasure -d . -add sf01=pos -fnhbi
```

Note: Do not perform this step if the batch has already been generated since the batch will have automatically registered sf01.

Step 2

Register the Alert Measure

The next step in the process is to register the alert measure in the Master Domain. In Example 10–2, an alert with the name of rdf_a1_1 with label of Alert1level1 is being registered.

Example 10–2 Register the Alert Measure

```
C:\Domains\RDF>regmeasure -d $DOMAIN_DEST_DIR -add "rdf_a1_1" -label
"Alert1level1" -baseint "itemstr_" -db "data/myalerts" -type boolean -navalue
False
```

Step 3

Register the Expression for the Forecast Approval Alert

The alertmgr utility is used to register the alert and the alert expression. In the Example 10–3, the alert expression first filters out time series with low volume sales (items with forecasts less than three units). This alert compares the System Forecast in the first three weeks in the forecast horizon with last approved forecast for the same three weeks. If the values are within a 33% range, the full forecast horizon is set to automatic approval, otherwise the Alert is triggered. This is all done in batch, so the Alert Manager is not necessary to apply the alert. For intersections that do not qualify for automatic approval, the Approval Comment on the Approval Worksheet in the Forecast Approval workbook contains *refused by alert*. You may use the Alert Manager to insert this alert into the workbook to display the intersections that have the alert flag set to True.

Example 10–3 alertmgr Utility

```
C:\Domains\RDF> alertmgr -d . -register "rdf_a1_1" -category "FORECAST_APPROVAL"
-categoryLabel "Alert1level1" -expression "rdf_a1_
1=if(tssum(@sf01,index([clnd].[week],flookup(lfsXLXB.level([data].[flvl]+[prod].to
p),[data].[flvl].[flvl01])),
index([clnd].[week],flookup(lfsXLXB.level([data].[flvl]+[prod].top),[data].[flvl].
[flvl01]))+3)>=3.0,
abs(1-tssum(@sf01,index([clnd].[week],flookup(lfsXLXB.level([data].[flvl]+[prod].t
op),[data].[flvl].[flvl01])),
index([clnd].[week],flookup(lfsXLXB.level([data].[flvl]+[prod].top),[data].[flvl].
[flvl01]))+3)/(tssum(lappf01XB,index([clnd].[week],flookup(lfsXLXB.level([data].[f
lvl]+[prod].top),[data].[flvl].[flvl01]))+3,index([clnd].[week],flookup(lfsXLXB.le
vel([data].[flvl]+[prod].top),[data].[flvl].[flvl01])))+0.001))>.33,false)"
```

11

Adding New Local Domains

This chapter provides an overview on adding new local domains to an existing RDF global domain. New local domains can be added using the RPAS reconfigGlobalDomainPartitions utility. It is important to keep in mind that as new local domains are added, they must be added such that the RDF partitioning requirements continue to be met. This means each new local domain can only contain one position along the partition dimension.

When new local domains are added, the following additional scripts must be run, which are located in the /bin directory of $RPAS_HOME:

loadCurveParameters.ksh

This script is used to load the Curve data parameter measures including Profile Data Source, Default Source Profile, Default Profile Approval Method, Training Window Method, and Normal Value. This action is typically performed within the plug-ins at domain creation time, however, when you add a new local domain to an existing domain environment, the plug-ins are not run, and therefore this script performs that action manually.

Usage:

loadCurveParameters -d fullPathToDomain -s fullPathToNewSubdomain

loadRDFParameters.ksh

This script is used to load the RDF data parameter measures including Default Required Method, Default Source Level, Data Plan, Seasonal Profile, and Spreading Profile. This action is typically performed within the plug-ins at domain creation time, however, when you add a new local domain to an existing domain environment, the plug-ins are not run, and therefore this script performs that action manually.

Usage

```
loadRdfParameters -d fullPathToDomain -s fullPathToNewSubdomain
```

12

Internationalization

Internationalization is the process of creating software that can be translated more easily. Changes to the code are not specific to any particular market.

Oracle Retail applications have been internationalized to support multiple languages.

Refer to "Internationalization Considerations for RETL" for information about internationalization and RETL.

Translation

Translation is the process of interpreting and adapting text from one language into another. Although the code itself is not translated, components of the application that are translated include the following:

- Graphical user interface (GUI)
- Error messages

The following components are not translated:

- Documentation (online help, release notes, installation guide, user guide, operations guide)
- Batch programs and messages
- Log files
- Configuration tools
- Reports
- Demonstration data
- Training materials

The user interface has been translated into the following languages:

- Brazilian Portuguese
- Chinese (Simplified)
- Chinese (Traditional)
- Croatian
- Dutch
- French
- German
- Greek

- Hungarian
- Italian
- Japanese
- Korean
- Polish
- Russian
- Spanish
- Swedish
- Turkish

Note: For information about adding languages for the first time or for translation information in general, see the *Oracle Retail Predictive Application Server Administration Guide for the Classic Client*.

A

RPAS and RDF Integration with RMS

This appendix addresses RMS integration between RPAS and RDF.

Integration Approach with RMS

The strategy for the extraction of foundation data from Retail Merchandising System (RMS) is for the extract programs (RMSE) to provide flat files in a generic format. For each solution that uses this data, transformation scripts are used to reformat the data as needed to produce a file suitable for loading into the application. For the instances of data coming from RPAS to non-RPAS applications, extract programs are specific to the application in need of the data. Other scripting languages are then used (Perl or AWK) to perform additional data formatting.

This appendix summarizes the following:

Specifics on the usage of RMS extract programs (RMSE's) within the RDF transformation programs are beyond the scope of this document. See the *Oracle Retail Merchandising System Operations Guide* for more information on the RMS extract programs.

Note: For integration compatibility information, see the *Oracle Retail Predictive Application Server Installation Guide.*

Environment Variable Setup

In addition to any variables identified in the RMS integration documentation, the transformation or extract programs or both require the environment variables listed in Table A–1.

Table A–1 Environment Variables

Environment Variable	Description
$FROM_RPAS	The staging area for the data extract out of RPAS. This directory should be located at the same level as the root of the RPAS domain. For example, if the domain RDF is located in Domains directory. (example: `/Domains/RDF`), then $FROM_RPAS should be located at the same level as RDF (example: `/Domains/from_rpas`).
$RDF_HOME	Identifies the location of the root of the RFX directory. The RFX directory packaged with the ARPOPlatform should be added to the location RFX directory packaged with the RMS RETL programs.
$RI_RMSVERSION	Identifies the version of RMS. If this variable is not set, the integration scripts assume an RMS version of 13. Set the value of this environment variable to 13.
$RPAS_INTEGRATION_HOME	Identifies the location of the integration scripts when `/common/header.ksh` is run. This variable is used for all integration scripts packaged with the ARPOPlatform except those included in RFX (see "$RDF_HOME").
$TO_RPAS	The staging area for the data to be loaded into RPAS. This directory should be located at the same level as the root of the RPAS domain. For example, if the domain RDF is located in Domains directory. (example: `/Domains/RDF`), then $TO_RPAS should be located at the same level as RDF (example: `/Domains/to_rpas`).
LIBPATH	To run RETL, include these items in the LIBPATH variable: $RETL_JAVA_HOME/jre/bin:$RETL_JAVA_HOME/jre/bin/classic:$RFX_HOME/bin:$LIBPATH
PATH	To run RETL, include these items in the PATH variable: $RETL_JAVA_HOME/bin:$RETL_JAVA_HOME/jre/bin:$RFX_HOME/lib:$RFX_HOME/bin:$PATH
PREFERRED_LANG	This value indicated the language that the calendar labels are shown The default is english (_en), but any value from `$RDF_HOME/resources/SupportedLanguages.txt` where *column 4 = Yes* is supported, provided the corresponding retl_msgs.* file is present.
RDF_SCHEMA_DIR	This directory contains the RFX or RETL schemas. This variable is used when converting the `Unassigned` value in the schemas to a locale specific language. The default (and recommended) value is `$RDF_HOME/rfx/schema`.

Table A–1 (Cont.) Environment Variables

Environment Variable	Description
RESOURCE_DIR	This directory contains resource information to support multi-language capability for calendar-related fields as well as the `Unassigned` value present in some schema. The default (and recommended) value is `$RDF_HOME/resources`.
RETL_JAVA_HOME	The directory that contains Java. See the latest version of the *Oracle Retail Extract, Transform, and Load Programmer's Guide* for the version of Java that corresponds to the version of RETL used.
RFX_HOME	The directory that holds the RETL executable files.
RMS_RPAS_HOME	This variable needs to be set in order to maintain compatibility with RMS. The required value is `$RDF_HOME`.
RPAS_ALERTNAVIGATIONTHRESHOLD	This variable controls alerts and specifies the override for the default alert navigation threshold. The RPAS alertmgr api takes navigation threshold as an argument. If this navigation threshold is not passed into the api as an argument, then the value of the environment variable RPAS_ALERTNAVIGATIONTHRESHOLD is used. If RPAS_ALERTNAVIGATIONTHRESHOLD is not set then a default value of 5000 is used.
UPGRADE_BACKUP_DIR	This directory is used to backup existing schema information prior to converting the `Unassigned` value in the schemas to a locale specific language. The default (and recommended) value is `$RDF_HOME/schema_backup`.

RDF Transformation Programs

This section describes the RDF transformation scripts.

Common Program for All Transformations

The `rdft.ksh` script runs all of the necessary data extraction and transformation scripts (`rmse_*.ksh` and `rdft_*.ksh`, respectively) that are needed to produce the files to be loaded into RPAS/RDF/Planning. Most of these scripts are run in parallel (as background jobs).

Usage

```
rdft.ksh  [-x]  [-c] [-sd startDate] [-ed endDate] [-d dir]
```

Arguments

The following table lists and describes the arguments for these scripts.

Argument	Description
-x	This option causes the running of the RMS data extraction wrapper (`rmse.ksh`) to be skipped.
`-c`	This option causes FILE_DATE in rmse_config.env to be set to the current date instead of using VDATE.
`-sd`	This option sets the start date for optionally filtering out records based on date. Records with dates prior to this date are excluded from loading into RDF. The date needs to be in the format YYYYMMDD.

Argument	Description
ed	This option sets the end date for optionally filtering out records based on date. Records with dates after this date are excluded from loading into RDF. The date needs to be in the format YYYYMMDD.
`-d`	This option causes all programs run by `rdft.ksh` to be obtained from the *dir* directory.

Transformations of Merchandise Hierarchy Data

The `rdft_merchhier.ksh` script is the primary script used to build the data for RPAS from the RMS Merchandise Hierarchy tables. The schema used to produce the output file depends on the attributes and differentiator settings in RMS:

Case	Settings	Outcome
1	If PROD_ATTRIBUTES_ACTIVE = false and DIFFS_ACTIVE = false	Then `rdft_merchhier.base.schema` is used to produce the file. In this case, attributes and diff fields are not included in the merchandise hierarchy file.
2	If PROD_ATTRIBUTES_ACTIVE = true and DIFFS_ACTIVE = false	Then `rdft_merchhier.attributes.schema` is used to produce the file. This schema must be manually edited to support a specific attribute model and must be kept in sync with `rmse_attributes.schema` and `rmse_attributes.ksh`.
3	If PROD_ATTRIBUTES_ACTIVE = false and DIFFS_ACTIVE = true	Then `rdft_merchhier.schema` is used to produce the file. In this case, diff fields are included in the merchandise hierarchy file.
4	If PROD_ATTRIBUTES_ACTIVE = true and DIFFS_ACTIVE = true	Then an error results. In this release, the combination of diffs and attributes is not supported.

Intermediate schema and scripts which may be used (depending on configuration options) to produce the merchandise hierarchy file:

- `rdft_diff.domain.schema`
- `rdft_merchdiff.domain.schema`
- `rdft_merchhier_diff_trans.ksh`
- `rdft_merchhier_split_by_domain.ksh`
- `rdft_clean_partition.ksh`

Additional merchandise hierarchy support for issue domains is provided in `rdft_item_loc.ksh`. This script is designed to produce a full item list for issues domains, only containing items that exist in the warehouses.

Note: Issues-specific data transformation functionality is triggered based on the issues setting in RMS (ISSUES_ACTIVE must be set to True).

The script `rdft_clean_partition.ksh` was designed to mimic some of the functionality of local domains prior to local domains being supported by RPAS. This script is not relevant to versions of RDF 12.0 or later. It is retained for use by customers on earlier versions of RDF.

RPAS Position Names

RPAS position names must consist only of alphanumeric characters.

Invalid position names are created when using any other characters including:

- Spaces
- Punctuation
- Non-alphanumeric characters

Invalid position names are not allowed in RPAS and are rejected by loadHier. RMS does allow non-alphanumeric characters, though, in the fields that are transformed into RPAS position names, including DIFF fields. These non-alphanumeric characters must be screened out or transformed by the customer before they can be loaded into RPAS. The GA scripts do not perform this function.

Transformations of Location Hierarchy Data

The `rdft_orghier.ksh` script is the primary script used to build the location data file needed for RPAS from the RMS Organizational Hierarchy Table.

The following constants may be modified in the script based on location hierarchy data requirements:

Modifiable Constant	Description
COMPANY_NAME	The label for the company position to be populated in the file.
COMPANY_ID	The name for the company position to be populated in the file.
STORE_CLASS_CONCAT	When set to True, causes the STORE_CLASS to be concatenated on the left of the STORE_CLASS_DESCRIPTION field in the final Store data output file.
ADD_AT_SIGN_TO_WH_DESC	When set to True, causes the WHSE_NAME field in the Warehouse output file to have an *at sign* "@" prefix.
LONG_WAREHOUSE_ RECORDS	When set to True, the Warehouse output records consists of 16 fields. If it is False, the records contain only four fields, WH, WHSE_NAME, COMPANY, and CO_NAME.

Intermediate schemas which may be used (depending on configuration options) to produce the location hierarchy file:

- `rdft_issues.schema`
- `rdft_issues_long.schema`
- `rdft_orghier_store.schema`

> **Note:** Issues-specific data transformation functionality is triggered based on the issues setting in RMS (ISSUES_ACTIVE must be set to True).

Transformations of Calendar Hierarchy Data

The `rdft_calhier.ksh` script transforms the Calendar Hierarchy data extracted from RMS for loading into RPAS.

Configuration inputs to the `rdft_calhier.ksh` script include:

- **DATE_PREF** - The path to the file that contains text indicating whether the format of the Date Description field are mm/dd/yyyy or dd/mm/yyyy. See the *Oracle Retail Merchandising System Operations Guide* for date format options.
- **LAST_DOW** - The path to the file that contains a day of week name or abbreviation indicating which day of the week is considered to be the end of the week for the fiscal calendar being used at this installation.

The RMS to RDF calendar integration has been enhanced to allow calculation of several new quantities. These new quantities are present in `rdft_calhier.dat`, and include:

- dos (Day of Season)
- woy (Week of Year)
- wos (Week of Season)

In order for these values to calculate properly, the `rdft_clndhier.ksh` script looks to two files, `rfx/etc/first_day_of_season.txt` and `rfx/etc/first_week_of_season.txt`. Each file contains a single number representing the day or the week number from where the season starts.

For example, suppose the data coming from RMS is starting from the 2nd day of the 3rd week of season, then the `first_day_of_season.txt` has two (2) and `first_week_of_season.txt` has three (3) as starting day and starting week of the season.

If the files are missing then the season is assumed to start with first day of the first week of the season.

Transformations of Daily Sales and Issues Data

The `rdft_daily_sales.ksh` script produces the daily sales and issues data files based on regular, promotion, clearance, and issues.

The following constant may be modified in the script based on data requirements:

- `DOM_START_COL`

  Defines the starting column position of the Domain ID in the RETL output schema. This is needed by `rdft_merchhier_split_by_domain.ksh` to split the files by domain ID. If the `OUTPUT_SCHEMA` file is modified, the value of `DOM_START_COL` may also require modification from the default value.

Intermediate schemas which may be used (depending on configuration options) to produce either or both of the sales or issues data file:

- `rdft_daily_sales.schema`

> **Note:** Issues-specific data transformation functionality is triggered based on the issues setting in RMS (ISSUES_ACTIVE must be set to True).

Transformations of Weekly Sales and Issues Data

The `rdft_weekly_sales.ksh` script produces the weekly sales and issues data files based on regular, promotion, clearance and issues.

The following constant may be modified in the script based on data requirements

- `DOM_START_COL`

Defines the starting column position of the Domain ID in the RETL output schema. This is needed by `rdft_merchhier_split_by_domain.ksh` to split the files by domain ID. If the `OUTPUT_SCHEMA` file is modified, the value of `DOM_START_COL` may also require modification from the default value.

Intermediate schemas which may be used (depending on configuration options) to produce either or both of the sales or issues data files:

- `rdft_weekly_sales.schema`

Note: Issues-specific data transformation functionality is triggered based on the issues setting in RMS (ISSUES_ACTIVE must be set to True).

Transformations of Store Open Date Data

The `rdft_open_date.ksh` script produces the Store/Warehouse Opening Date data file.

Intermediate schema used to produce the store open date data files:

- `rdft_open_date.schema`

Transformations of Store Close Date Data

The `rdft_close_date.ksh` script produces the Store/Warehouse Closing Date data file.

Intermediate schema used to produce the store closing date data files:

- `rdft_close_date.schema`

Transformations of Out-of-stock Indicator Data

The `rdft_outofstock.ksh` script produces the Store and Warehouse (issues) out-of-stock indicator data extracted from RMS.

Intermediate schema and scripts which may be used (depending on configuration options) to produce the out-of-stock indicator data file:

- `rdft_outstock_split_by_domain.awk`
- `rdft_outofstock.schema`
- `rdft_outofstock_issues.schema`
- `rdft_outofstock_sales.schema`

RDF Transformation Matrix

The following matrix identifies the transformation scripts and schemas used for each the hierarchy and data files produced for RDF:

Table A–2 RDF Transformation Scripts and Schema

Directory	Script or Schema Name	Merchandise Hierarchy	Location Hierarchy	Calendar	Daily Sales & Issues	Weekly Sales & Issues	Out-of-stock Indicator	Store Open Dates	Store Close Dates
rfx/lib	rdft_merchhier_diff_trans.ksh	X							
	rdft_merchhier_split_by_domain.ksh	X							
	rdft_outofstock_split_by_domain.ksh						X		
rfx/schema	rdft_close_date.schema								X
	rdft_daily_sales.schema				X				
	rdft_diff.domain.schema	X							
	rdft_merchierdiff.domain.schema	X							
	rdft_merchier.attributes.schema	X							
	rdft_merchhier.base.schema	X							
	rdft_merchhier.domain.schema	X							
	rdft_merchhier.schema	X							
	rdft_open_date.schema							X	
	rdft_orghier_issues.schema	X							
	rdft_orghier_issues_long.schema	X							
	rdft_orghier_strore.schema	X							
	rdft_outofstock.schema						X		
	rdft_outofstock_issues.schema						X		
	rdft_outofstock_sales.schema						X		
	rdft_weekly_sales.schema					X			

Table A–2 (Cont.) RDF Transformation Scripts and Schema

Directory	Script or Schema Name	Merchandise Hierarchy	Location Hierarchy	Calendar	Daily Sales & Issues	Weekly Sales & Issues	Out-of-stock Indicator	Store Open Dates	Store Close Dates
rfx/src	rdft_ksh	X	X	X	X	X	X	X	X
	rdft_calhier.ksh			X					
	rdft_clean_partition.ksh	X							
	rdft_close_date.ksh								X
	rdft_daily_sales.ksh								
	rdft_item_loc.ksh	X							
	rdft_merchhier.ksh	X							
	rdft_open_date.ksh							X	
	rdft_orghier.ksh	X							
	rdft_outofstock.ksh						X		
	rdft_weekly_sales.ksh					X			

Loading Transformed RMS Data into RDF

This section describes the loading of transformed RMS data into RDF.

renameRETLFiles.ksh

After the transformation of the RMS hierarchy and history data, the data files must be combined and renamed to match the appropriate measure names expected by RDF. This is accomplished by a script, `renameRETLFiles.ksh`. Usage of this script is optional, but recommended. The script reads the data files produced by the RETL transformation, combines and renames the data files, and writes the output to the domain's input folder. Existing data in the domain's input folder is backed up prior to the writing the new files.

Usage

```
renameRETLFiles.ksh -dataDir <data dir> -domInput <domain's input dir>
```

Arguments

The script has two required arguments:

Argument	Description
-dataDir	Required. The path to the data directory. This directory contains the RMS data transformed by the RDF RETL scripts. This is typically $RDF_HOME/data.
domInput	Required. The input directory of the domain in which the transformed data is to be loaded. The directory is typically the input directory under the domain root.

Common Programs for Extracts

This section describes the common programs for extracts.

config.ksh

The `config.ksh` script is a configuration directory that requires both the RMS version being integrated and the backup action to be defined.

Arguments

The following optional arguments are available:

Argument	Description
Name of the domain	Defaults to directory name
Number of the domain	Defaults to the 2 last digits of the directory name
Format of timestamp attached to logs and processed input files	Defaults to: (date +"%b%d%a%I%M%p") (example: Aug02Thu0111PM)
Data Drop	Defaults to ../../to_rpas
Data Export	Defaults to ../../from_rpas
Log Drop	Defaults to ./logs
Error Drop	Defaults to ./err
Reclass Data	Defaults to ../reclass_data

functions.ksh

The `functions.ksh` script file contains ksh functions that are used by scripts in [DOM]/scripts. It should be sourced, not run in order to preserve environment variables.

header.ksh

The `header.ksh` script file should be run at the beginning of any implementation-specific script to setup function libraries, environment, and platform-specific routines.

Extract of Forecast Data for RMS

This section describes the scripts that extract forecast data for RMS.

rdf_e_rms.ksh

The `rdf_e_rms.ksh` script extracts forecast demand value and standard deviation (cumulative interval) from an RDF domain.For more information, refer to Chapter 7, "Batch Processing."

Load of Extracted Forecast Data and Standard Deviations to RMS

This section describes the load of extracted forecast data and standard deviations to RMS.

rmsl_forecast.ksh

The `rmsl_forecast.ksh` script pulls the daily/weekly forecast items into RMS.

During the loading of each domain file the following steps are performed:

1. Truncate the partition in the RMS forecast table which corresponds to the domain ID.

   Note: Partition names should always be in the format: `[tablename]_[domainID]`

2. Append a domain field and insert the `domain_id` into each record.
3. Load the forecast data into the RMS forecast table.

Example A–1 rmsl_forecast.ksh Format

```
rmsl_rpas_forecast.ksh daily | weekly
```

Intermediate schemas which may be used (depending on configuration options) to produce the forecast data files:

- `rmsl_forecast_daily.schema`
- `rmsl_forecast_weekly.schema`

Extract of Diff Profile Data for Allocation

This section describes the extract of diff profile data for Allocation.

profile_e_alloc.ksh

The `profile_e_alloc.ksh` script extracts Curve diff profiles for use by Allocation.

Arguments

The script accepts the following arguments:

Argument	Description
-p	<Profile Number>
-m	<Mask Measure> (Optional mask; only positions for which the mask value is non-NA are exported.)
-w	<Data Width> ([7...18], defaults to 12)
-d	<Domain> (defaults to current directory)
-n	<Domain Number> (defaults to last two digits of domain)

Output file

`${RPAS_EXPORT}/d1<Product Level>.<Domain Number>`

Note: Where Product Level is the Aggregation intersection's Prod dimension.

The following table provides information about the output file data format.

Field	Start	Width	Format
Product ID	1	25	Alpha
Location ID	26	20	Alpha
Diff ID (optional)	46	36	Alpha
Quantity	82	12*	Numeric (floating point, 4 decimal digits, no decimal)*
Std. Dev. Demand	68*	12*	Numeric (floating point, 4 decimal digits with decimal)
* Quantity width may be overridden with the -w parameter.			

Note: The following must be defined in the shell environment prior to calling this script:

- RPAS_HOME
- RPAS_INTEGRATION_HOME

Extract of Store Grade Data for RMS

This section describes the extract of Store Grade data for RMS.

grade_e_rms.ksh

The `grade_e_rms.ksh` script extracts store grades for use by RMS.

Arguments

The script accepts the following arguments:

Argument	Description
-t	<Timestamp> (YYMMDDTTTT). This value corresponds to the timestamp of the Cluster Membership measure (clpm+<Timestamp>) to be extracted
-d	<Domain> (defaults to current directory)
-n	<Domain Number> (defaults to last two digits of domain)

Output File

`${RPAS_EXPORT}/gr<Timestamp>.<Domain Number>`

Note: The following must be defined in the shell environment prior to calling this script:

- RPAS_HOME
- RPAS_INTEGRATION_HOME

The following tables list the output file data formats:

Header Records
FHEAD
Line ID Number
GRADU

Detail Records	Data Type	Width	Description
FDETL	String	5	Always 'FDETL'
Line ID	Number	10	Line Sequence Identifier (generated by the script)
Grade Group ID	Number	8	This value corresponds to the first 8 characters of the Cluster Run Name measure (clnam+<user-defined name>) set by the user in the Generate Cluster wizard in Grade. For integration with RMS, the Cluster Run Name must be populated with only numeric characters.
Grade Group	String	120	This value corresponds to the first 120 characters of the Cluster Run Name measure (clnam+<user-defined name>) set by the user in the Generate Cluster wizard in Grade.
Grade Store	Number	10	Valid Grade Store derived from the point mapping measure (clpm+<user-defined name>). For integration with RMS, the Cluster Run Name must be populated with only numeric characters.
Grade ID	Number	10	Valid Grade ID derived from the point mapping measure (clpm+<user-defined name>). For integration with RMS, the Cluster Run Name must be populated with only numeric characters.

Detail Records	Data Type	Width	Description
Grade Name	String	120	Valid Grade Name, also derived from the point mapping measure (clpm+<user-defined name>).

Footer Records
Line ID Number
FDETL Line Total Number

RDF Extract Matrix

The following matrix identifies the extract scripts and schemas used for each the data files produced for RMS.

Directory	Script or Schema Name	Forecasts and Standard Deviations	Diff Profiles
common	config.ksh		
	functions.ksh	X	
	header.ksh	X	X
curve	profile_e_alloc.ksh		X
grade	grade_e_rms.ksh		
plan	Plan_e_alloc.ksh		
	Plan_e_price.ksh		
	Plan_e_plcblwdm.ksh		
	Plan_e_ploblwdm.ksh		
rdf	rdf_e_rms.ksh	X	
	rmsl_forecast.ksh	X	
	rmsl_forecast_daily.schema	X	
	rmsl_forecast_weekly.schema	X	

Internationalization Considerations for RETL

These sections describe internationalization considerations for RETL:

- Calendar Data
- Unassigned Value in Schemas

Calendar Data

Calendar data (that is names of weeks and months) is translatable into Oracle supported languages. Enabling this feature requires setup of these two environment variables:

- export RESOURCE_DIR=$RDF_HOME/resources
- export PREFERRED_LANG=_en

Note that for PREFERRED_LANG, the default is english (_en), but any value from `$RDF_HOME/resources/SupportedLanguages.txt` where *column 4 = Yes* is supported, provided the corresponding retl_msgs.* file is present.

Once these variables are setup, then the translation takes place for RETL. No separate script is required.

Unassigned Value in Schemas

The `Unassigned` value in schemas can be converted to a locale specific equivalent. To enable this feature, set these environment variables:

- export RESOURCE_DIR=$RDF_HOME/resources
- export PREFERRED_LANG=_en
- export RDF_SCHEMA_DIR=$RDF_HOME/rfx/schema
- export UPGRADE_BACKUP_DIR=$RDF_HOME/schema_backup

Note that for PREFERRED_LANG, the default is english (_en), but any value from `$RDF_HOME/resources/SupportedLanguages.txt` where *column 4 = Yes* is supported, provided the corresponding retl_msgs.* file is present.

After the environment variables are set, then run this script: `$RDF_HOME/rfx/etc/update_schema_nullvalues.ksh`.

This script backs up existing schemas into UPGRADE_BACKUP_DIR and then replaces the Unassigned value with a locale specific equivalent. This update preserves field widths.

B

Configuring the Cluster Procedure

Clustering may be used to provide insight into how various parts of a retailer's operations can be grouped together. Typically a retailer may cluster stores over item sales to create logical groupings of stores based upon sales of particular products. This provides increased visibility to where products are selling, and it allows the retailer to make more accurate decisions in merchandising. Beyond this traditional use of clusters, the Cluster is flexible enough to cluster any business measure based on products, locations, time, promotions, customers, or any hierarchy configured in the solution.

> **Note:** The syntax is slightly different than the standard RPAS functions and procedures that are described in the "Rule Functions Reference Guide" section of the *Oracle Retail Predictive Application Server Configuration Tools User Guide*.

The two approaches available for clustering are BreakPoint and Cluster, or the BaNG approach. See the *Oracle Retail Grade User Guide* for details on these two approaches. The following sections explain the specifics for configuring clustering.

Cluster Requirements

The following libraries must be registered in any domains that will use the Cluster solution extension:

- AppFunctions
- ClusterEngine

Using the Cluster Procedure

The following notes are intended to serve as a guide for configuring the Cluster procedure within the RPAS Configuration Tools:

1. See the section, "Syntax Conventions" on page B-2, for the appropriate syntax for calling this procedure. Parameter labels must always be used.
2. If the ClusterEngine is not registered with the Configuration Tools, this rule will remain red, which indicates that it is invalid because the RPAS JNI cannot validate it at this point in time. Therefore, there is no validation for this rule. Refer to the Grade documentation for the appropriate input parameters and output measures. Make sure to register the ClusterEngine with the Configuration Tools. It is recommended that you register the ClusterEngine when creating the domain to avoid potential issues.

3. Make sure that the resultant measures are at the right intersection levels by using the information based on the input and output parameters.
4. The Cluster procedure is a multi-result procedure, which means that it can return multiple results with one procedure call within a rule. In order to get multiple results, the resultant measures must exist, and the specific measure label must be used on the left-hand-side (LHS) of the procedure call. The resultant measure parameters must be comma-separated in the procedural call.
5. You must configure/register all required input measures.
6. Be sure to create load and commit rules for the input measures. The RPAS JNI cannot validate the Cluster procedure call, so all input measures must exist within other rules in the rule set in order for them to be available for selection in the Workbook Tool.
7. You must use the latest version of RPAS to build the domain. You will get the following message in the log because the Cluster function is not validated:

   Warning: unable to parse new expression (Unknown special expression: Cluster)

   This message is okay.

   Registering ClusterEngine will eliminate this error from occurring.
8. After the domain build, use the regfunction RPAS utility to register the Grade library. The library, which is located in the $RPAS_HOME/applib directory, is libClusterEngine.so. Do not specify the lib or .so file extension for the function name with the regfunction utility.

Example B–1 ClusterEngine

regfunction -d /domains/D01 -l ClusterEngine

9. Use the Mace command to run the Cluster rule with the rule group (for instance, grade_batch).

Syntax Conventions

The following table displays the syntax conventions used in this document.

Indicator	Definition
[…]	All options listed in brackets are optional.
{…|…}	Options listed in "{}" with "|" separators are mutually exclusive (either/or).
{…,…}	Options listed in "{}" with "," separators way are a complete set.
Bold	Labels.
Italics	Italics indicate a temporary placeholder for a constant or a measure.
Italics/meas	This indicates that the placeholder can be either a constant or a measure.
BoldItalics	This indicates a numeric placeholder for the dynamic portion of a label. Usually a number from 1 to N.
Normal	Normal text signifies required information.
Underlined	This convention is used to identify the function name.

Cluster Syntax

The syntax for using the Cluster or BaNG algorithm is shown in the following examples. The input and output parameter tables explain the specific usage of the parameters names use in the procedure.

Example B–2 Generic Example

```
POINTMEMBERSHIP: MEMBERMEAS, CENTROID: CENTROIDMEAS [, DISTFROMCENTROID:
DISTCENTDMEAS, COHESION: COHESIONMEAS, CLUSTERPORTION: CLPORTMEAS, CENTROIDTOAVG:
C2AVGMEAS, CLOSESTCLUSTER: CLOSCLUSTMEAS, CLOSESTCLUSTERDIST: CLOSCLUSTDISTMEAS]
<-Cluster(MEASURE: MEASMEAS, METHOD: METHOD, NUMCLUSTERS: NUMCLUST, CLUSTERHIER:
CLUSTHIER, CLUSTEROVERHIER: CLUSTOVERHIER [, BYGROUPDIMS: BYGROUPDIM, AGGMETHOD:
AGGTYPES])
```

Example B–3 Sample - Cluster with Minimum Information:

```
POINTMEMBERSHIP:MEMB, CENTROID:CENT<-Cluster(MEASURE:RSAL, METHOD:"BANG",
NUMCLUSTERS:5, CLUSTERHIER:"PROD", CLUSTEROVERHIER:"LOC")
```

Syntax for Calculate Cluster Statistics (CalculateClusterStatistics)

Example B–4 Generic Example

```
CENTROID: CENTROIDMEAS, [DISTFROMCENTROID: DISTCENTDMEAS, COHESION: COHESIONMEAS,
CLUSTERPORTION: CLPORTMEAS, CENTROIDTOAVG: C2AVGMEAS, CLOSESTCLUSTER:
CLOSCLUSTMEAS, CLOSESTCLUSTERDIST: CLOSCLUSTDISTMEAS]
<-CalculateClusterStatistics(MEASURE: MEASMEAS, POINTMEMBERSHIP: MEMBERMEAS,
CLUSTERHIER: CLUSTHIER, CLUSTEROVERHIER: CLUSTOVERHIER [, BYGROUPDIMS: BYGROUPDIM,
AGGMETHOD: AGGTYPE])
```

Example B–5 Sample - Cluster Statistics with Minimum Information

```
CENTROID:CENT<-CalculateClusterStatistics(MEASURE:RSAL, POINTMEMBERSHIP:MEMB,
CLUSTERHIER:"PROD", CLUSTEROVERHIER:"LOC")
```

Configuration Parameters and Rules

Input Parameters

The following table provides the input parameters for the Cluster procedure and special expressions.

Parameter Name	Description
POINTMEMBERSHIP	This is an input parameter for CalculateClusterStatistics and bpstatistics. Its intersection should be the dimension being clustered and all by group dimensions from other hierarchies. The values state which positions are assigned to which cluster index. Data Type: Integer Required: Yes

Parameter Name	Description
MEASURE	The measure you are trying to cluster. It must have at least two dimensions. Data Type: Real Required: Yes
METHOD	Determines which clustering algorithm to use. Valid values are BANG (preferred) or KMEANS. Data Type: Real Required: Yes - for Cluster.
NUMCLUSTERS	For each by group partition, the maximum number of clusters. Data Type: Integer Required: Yes - for Cluster.
CLUSTERHIER	The hierarchy that contains the dimension to cluster. The results will give you clusters of positions in this dimension. Data Type: String Required: Yes
CLUSTEROVERHIER	The hierarchy that contains the dimension to cluster over. The algorithm uses the positions in this dimension as the co-ordinates when clustering. Data Type: String Required: Yes
BYGROUPDIMS	The algorithm generates clusters one by group combination at a time. Provide the by group intersection. Data Type: String Required: No
AGGMETHOD	The algorithm aggregates the measure data up to the appropriate level. If AGGMETHOD is specified, it will use it; otherwise, it will use whatever is defined on the measure. Data Type: String Required: No

Output Parameters

The following table provides the output parameters for the Cluster procedure.

Parameter Name	Description
POINTMEMBERSHIP	Its intersection should be the dimension being clustered and all by group dimensions from other hierarchies. The values state which positions are assigned to which cluster index. Data Type: Integer Required: Yes - output for bpcluster
CENTROID	Its intersection should be the cluster dimension, the dimension being clustered over and all by group dimensions from other hierarchies. The values are the average of all points in the cluster. Data Type: Real Required: Yes

Parameter Name	Description
DISTFROMCENTROID	Its intersection should be the dimension being clustered and all by group dimensions from other hierarchies. The values are the squared Euclidean distance from that point to its centroid. Data Type: Real Required: No
COHESION	Its intersection should be the cluster dimension and all by group dimensions. The value is the ratio of points in this cluster versus all clusters. Data Type: Real Required: No
CLUSTERPORTION	Its intersection should be the cluster dimension and all by group dimensions. The value is the ratio of points in this cluster versus all clusters. Data Type: Real Required: No
CENTROIDTOAVG	Its intersection should be the cluster dimension, the dimension being clustered over and all by group dimensions from other hierarchies. The values are the ratio of the centroid to the average of all points. Data Type: Real Required: No
CLOSESTCLUSTER	Its intersection should be the cluster dimension and all by group dimensions. The value is the nearest cluster index. Data Type: Integer Required: No
CLOSESTCLUSTERDIST	Its intersection should be the cluster dimension and all by group dimensions. The values are the squared Euclidean distance from the centroid of the cluster to the centroid of the closest cluster. Data Type: Real Required: No

Syntax for Break Point Cluster (bpcluster)

Example B–6 Generic Example

```
POINTMEMBERSHIP <- bpcluster(SOURCEMEASNAME, CONFIGURATIONMEASNAME, CONFIGNAME [,
GROUPBYINT] )
```

Example B–7 Sample - Break Point Cluster with Minimum Information

```
MEMB<-bpcluster(RSAL, GCFG, "GCFG01", "CHN_PGRP")
```

Syntax of Break Point Cluster Statistics (bpstatistics)

Example B–8 Generic Example

```
CENTROID, DISTANCE <- bpstatistics(POINTMEMBERSHIP, SOURCEMEASNAME [, GROUPBYINT]
)
```

Example B–9 Sample - Break Point Cluster Statistics with Minimum Information

```
CENTROID, DISTANCE <- bpstatistics(MEMB, RSAL, "CHN_PGRP" )
```

Configuration Parameters and Rules

Input Parameters

The following table provides the input parameters for the bpcluster and bpstatistics procedures and special expressions.

Parameter Name	Description
SOURCEMEASNAME	The measure you are trying to cluster. Data Type: Real Required: Yes
CONFIGURATIONMEASNAME	Measure defined at Cluster/Configuration intersection. It contains the thresholds for the breakpoint calculation. Data Type: Real Required: Yes - for bpcluster.
CONFIGNAME	Breakpoint Configuration that will be used to produce the grades. (Refer to the *Oracle Retail Grade User Guide* for details on Breakpoint configuration and administration.) Data Type: String Required: Yes
GROUPBYINT	The algorithm generates clusters by group combination one at a time. Provide the by group intersection. Data Type: String Required: No

Output Parameters

The following table provides the output parameters for the bpcluster and bpstatistics procedures and special expressions.

Parameter Name	Description
POINTMEMBERSHIP	Its intersection should be the dimension being clustered and all by group dimensions from other hierarchies. The values state which positions are assigned to which cluster index. Data Type: Integer Required: Yes - output for bpcluster
CENTROID	Measure defined at Cluster/Configuration intersection. It contains the thresholds for the breakpoint calculation. Data Type: Real Required: Yes - for bpcluster.
DISTANCE	This is the point distance of member from centroid. Data Type: Real Required: Yes

C

Configuring the Clone Procedure

Cloning allows users to generate forecasts for new items and locations by copying, or cloning history, from other items and stores. Users can map items or stores that have similar business cases, clone the historical data, and begin generating forecasts. Cloning provides the ability to generate forecasts based on historical data and promotional calendar.

The Clone Syntax section contains the specifications and syntax for configuring the Forecast procedure.

The clone procedure can be set up to clone sales history, promotion history.

This appendix details these topics:

Clone Requirements

The following libraries must be registered in any domains that will use the clone solution extension:

- RdfFunctions

Using the CloneMeasure Procedure

The following notes provide information about CloneMeasure functionality.

- Refer to the appropriate input parameters and output measures when using the CloneMeasure procedure.
- The PROD and LOC hierarchies are required by the CloneMeasure expression. If CLND exists, it must be the innermost hierarchy.
- Cloning supports up to three like items or three sister stores with contribution percentages for each of these items or stores.
- An adjustment ratio can be defined to modify the level of the CloneMeasure history for the new product or location.
- Like items are auto recommended after you run the new item recommendation script and it is reviewed/approved by using new item review and maintenance workbooks. Sister stores are manual selected by you when using new store workbooks.

- The cloning (copying) of historical data is performed as part of the batch process using the CloneMeasure special expression.
- The CloneMeasure special expression is generic enough that it can be used to copy not just history, but forecast parameters, Casual histories and more, using the CloneMeasure special expression.
- The input parameters include a replicate mode(1 is clone), a substitute method measure, a source array, up to six map measures, six contribution measures, two Adjustment Ratio Measures, and a destination array.
- The source and destination array must be at the same intersection, as validated by the special expression. The intersection will be where cloning is performed.
- The map, contribution measures must be at the same intersections as the source and destination arrays.
- The map arrays and contribution measures are optional; at least one of each is required. The number of contribution measures should be equal to the number of map measures.
- Two additional optional start and end date measures can be passed, which specify the start end and end date indexes of the cloning process.
- Similar restrictions for the intersection of the Start and End Date Index measures apply as mentioned with map measures above.
- The index should be an index along a calendar dimension equal to the Calendar Dimension along Source and Destination array. For example, if the Source and Destination Arrays are at the item/store/week level intersection, then the Start and end date index measures should contain Index values of the Week dimension.
- If these values are not passed, they will default to calendar hierarchy Start and End dates.
- When running a CloneMeasure at a higher intersection, you need to create a mask measure. If you create a mask, do so at the same intersection as your SRC and DEST measures less the Calendar dimension if the SRC and DEST are at a higher intersection.
- In some instances, it is preferable that the CloneMeasure expression is run separately for items and locations. Perform the following procedure to run the CloneMeasure expression separately for items and locations:

  1. Run the CloneMeasure expression for items.
  2. Update the CloneMeasure source measure with the results from the item cloning run.

     For example, if the initial source measure for the cloning run was **source_measure**, and the item cloning adjustments are stored in the **item_cloning_adjustments** measure. Then the update could be achieved by an expression of the type:

     source_measure = source_measure + item_cloning_adjustments
  3. Run the CloneMeasure expression for locations with the same **source_measure** as source measure.
  4. Aggregate the various demand components to build the forecast source.

Cloning Real or Integer Measures

Since up to three parent items and three parent stores can be specified, the number of mapping measures can consist of up to nine combinations of item/stores (# of parent items specified x # pf parent stores specified) and corresponding % contributions.

Example C–1 Calculating Real or Integer Values Using the Clone Expression

The following example illustrates how real or integer values are calculated using the clone expression.

Item1		
Item2	Item3	Item4
20%	20%	60%
STR1		
STR2	STR3	STR4
20%	50%	30%

The Special Expression will calculate mappings and contributions as follows:

- Item2/STR2 at 4%
- Item2/STR3 at 10%
- Item2/STR4 at 6%
- Item3/STR2 at 4%
- Item3/STR3 at 10%
- Item3/Str4 at 6%
- Item4/STR2 at 12%
- Item4/STR3 at 30%
- Item4/Str4 at 18%

Example C–2 Subset of Possible Values Provided

If only a subset of values is populated, then the clone expression performs its calculations as shown in this example.

Item1	
Item2	Item3
20%	80%
STR1	
STR2	STR3
50%	50%

The Special Expression will calculate mappings and contributions as follows:

- Item2/STR2 at 10%
- Item2/STR3 at 10%
- Item3/STR2 at 40%

- Item3/STR3 at 40%

Example C–3 Subset of Possible Values Provided with Adjustment Ratio

If only a subset of values is populated and an Adjustment Ratio is defined, then the clone expression performs its calculations as shown in this example.

Item1	
Item2	Item3
20%	80%
Adjustment Ration = 1	
STR1	
STR2	STR3
50%	50%
Adjustment Ration = 0.5	

The Special Expression will calculate mappings and contributions as follows:

- Item2/STR2 at 5% (=20% x 0.5x50%)
- Item2/STR3 at 5%
- Item3/STR2 at 20% (=80%x0.5x50%)
- Item3/STR3 at 20%

Clone Syntax

The syntax for using the clone procedure is shown in the following examples. The input and output parameter tables explain the specific usage of the parameters names use in the procedure.

Example C–4 Generic Example for Cloning Real or Integer Measures:

DEST:DEST_MEASURE <-CloneMeasure(REPLMOD:1,SUBMETH:SUBMETHOD_MEASURE,SRC:SOURCE_MEASURE,

SKUMAP1: SKUMAPMEAS1, SKURATIO: SKUCONTRIBUTION1, SKUMAP2: SKUMAPMEAS2, SKURATIO2:SKUCONTRIBUTION2, SKUMAP3:SKUMAPMEAS3, SKURATIO3: SKUCONTRIBUTION3, SKUADJRATIO: SKUADJUSTMENTRATIOMEAS

STRMAP1:STRMAPMEAS1, STRRATIO1:STRCONTRIBUTION1, STRMAP2:STRMAPMEAS2, STRRATIO2:STRCONTRIBUTION2, STRMAP3:STRMAPMEAS3, STRRATIO3:STRCONTRIBUTION3, STRADJRATIO:STRADJUSTMENTRATIOMEAS [, STARTINDEX:STARTINDEX]

[, ENDINDEX:ENDINDEX]

Example C–5 Sample of Clone Function with Real or Integer Measures:

DEST:clnadj<- CloneMeasure

(REPLMOD:1,SUBMETH:fcpsubm,SRC:prebasesls,SKUMAP1:nitapplkitm1,SKURATIO1:nitappcnt1,SKUMAP2:nitapplkitm2,SKURATIO2:nitappcnt2,SKUMAP3:nitapplkitm3,SKURATIO3:nitappcnt3,SKUADJRATIO:nitappadj,STRMAP1:nstapplkstr1,STRRA

TIO1:nstappcnt1,STRMAP2:nstapplkstr2,STRRATIO2:nstappcnt2,STRMAP3:nstapplkstr3,STRRATIO3:nstappcnt3,STRADJRATIO:fcpadj)

Configuration Parameters and Rules

This section describes the configuration parameters and rules for clones.

Input Parameters

Table C–1 provides the input parameters for the clone procedure and special expressions.

Table C–1 Input Parameters for the Clone Procedure and Special Expressions

Parameter Name	Description
1	Replicate mode 1 - Clone
SUBMETHOD_MEASURE	Substitute method for new items or new stores. 9 - New Item Clone 10 - New Store Clone
SOURCE_MEASURE	The source measure used for cloning. The source array and destination array need to be at the same intersection, which will be validated by the special expression. This is the intersection in which cloning is performed. Data Type: Integer, Real Required: Yes
SKUMAPMEAS1	The first SKU measure that is mapped to the new item/store. Data Type: String Required: Yes At least one map measure is required when using the clone function. SKUMAPMEAS1 needs to be populated with position ID of clone item.
SKUCONTRIBUTION1	The percentage of data used when cloning historical data. Used to weigh or assign importance of item or store map. Data Type: Real Required: Yes—for Real or Integer cloning. Ignored for Boolean and String cloning. At least one contribution measure is required Real or Integer cloning.
SKUMAPMEAS2	The second SKU measure that is mapped to the new item/store. Data Type: String Required: Yes— for Integer, Real, Boolean, or String cloning. Ignored for Date cloning.
SKUCONTRIBUTION2	The percentage of data used when cloning historical data for SKUMAPMEAS2. Used to weigh or assign importance of item or store map. Data Type: Real Required: Yes Ignored for Boolean and String cloning.

Table C–1 (Cont.) Input Parameters for the Clone Procedure and Special Expressions

Parameter Name	Description
SKUMAPMEAS3	The third SKU measure that is mapped to the new item/store. Data Type: String Required: Yes— for Integer, Real, Boolean, or String cloning. Ignored for Date cloning.
SKUCONTRIBUTION3	The percentage of data used when cloning historical data for SKUMAPMEAS3. Used to weigh or assign importance of item or store map. Data Type: Real Required: Yes Ignored for Boolean and String cloning.
SKUADJUSTMENTRATIOMEAS	A value greater than zero (0) used to adjust the level of history calculated at the item level. Data Type: Real Required: Yes
STRMAPMEAS1	The first STR measure that is mapped to the new item/store. Data Type: String Required: Yes STRMAPMEAS1 needs to be populated with position ID of clone store.
STRCONTRIBUTION1	The percentage of data used when cloning historical data for STRMAPMEAS1. Used to weigh or assign importance of item or store map. Data Type: Real Required: Yes — for Real or Integer cloning. Ignored for Boolean and String cloning.
StrMapMeas2	The second STR measure that is mapped to the new item/store. Data Type: String Required: Yes— for Integer, Real, Boolean, or String cloning. Ignored for Date cloning.
StrContribution2	The percentage of data used when cloning historical data for STRMAPMEAS2. Used to weigh or assign importance of item or store map. Data Type: Real Required: Yes Ignored for Boolean and String cloning.
StrMapMeas3	The third STR measure that is mapped to the new item/store. Data Type: String Required: Yes— for Integer, Real, Boolean, or String cloning. Ignored for Date cloning.

Table C–1 (Cont.) Input Parameters for the Clone Procedure and Special Expressions

Parameter Name	Description
StrContribution3	The percentage of data used when cloning historical data for STRMAPMEAS3. Used to weigh or assign importance of item or store map. Data Type: Real Required: Yes Ignored for Boolean and String cloning.
STRADJUSTMENTRATIOMEAS	A value used to adjust the level of history calculated at the store level. Data Type: Real Required: Yes
STARTINDEX	The index value for the Week dimension to be used as the starting point to clone data. Data Type: Integer Required: No
ENDINDEX	The index value for the Week dimension to be used as the ending point to clone data. Data Type: Integer Required: No
DEST_MEASURE	An array containing the item or location data is being cloned. Data Type: Integer, Real Required: Yes Data type and Base Intersection of this measure need to be the same as that of the SOURCE_MEASURE.

Clone Special Expression

This section describes special expressions for clones.

ClonePostProc

The ClonePostProc special expression can be used to adjust the cloned sales history based on actual sales of new items. When a new item start selling, the actual sales and the clone sales can be used to calculate an adjustment ratio. The cloned sales is multiplied with the adjustment ratio so that the scale of cloned sales is close to actual sales.

Calculation

*adjust cloned sales = pow(total actual sales from history start to t period forward /total cloned sales over t period backward from history start,1-alpha) * cloned sales*

Example C–6 Clonepostproc Syntax

DEST:clnadj<-ClonePostProc(SALESHIST:pos, NPERSSLSTHRSHLD:clnslsprdXLXB,

MASKMEAS:clnAdjMask, NPERSCALCRAT:clnclcprdX

LXB, ADJRATIO:clnalphaXLXB)

Table C–2 Parameters for the ClonePostProc Special Expression

Parameter Name	Description
DEST	host cloned sales at prod/loc/clnd before special expression run. After special expression run, it host the adjusted sales: Data Type: Real Required: Yes The Data Type and Base Intersection of this measure needs to be the same as SALESHIST.
SALESHIST	Sales measure contains actual sales. Based on prod/loc/calendar Data Type: Real Required: Yes
NPERSSLSTHRSHLD	Maximum number of periods having actual sales. Based on prod/loc, When the sales history period is more than this threshold. no adjust is made. Data Type: Integer, Real Required: Yes
MASKMEAS	A Boolean measure at prod/loc. It indicates which times eries is eligible for adjustment. Data Type: Boolean Required: Yes
NPERSCALCRAT	An integer measure indicate the number of period to used pre and after actual sales history start to calculate adjustment. Data Type: Integer, Real Required: Yes
ADJRATIO	The alpha number in the calculation formula. A real measure based on prod/loc. Data Type: Real Required: Yes Valid alpha is between 0-1. When alpha=0 or1, no adjustment is made. When alpha is close to 0, the adjusted sales is close to the cloned sales. When alpha is close to 1, the adjusted sales is close to the actual sales. It needs to be on the same intersection as NPERSCALCRAT and NPERSSLSTHRSHLD.

D

AppFunctions

The AppFunctions library supports a number of functions and special expressions, most of which are for internal use, and not recommended for customer use, except TransformSpread, which achieves higher performance than its peer in RPAS.

Supported Functions

The following functions are supported by AppFunctions:

AsDouble
clndstart
addperiods
datediff
datefn
dateToString
doubletoint
hiername
union_int_low
union_int_high
ispopulated
merge
parseDate
parseDateFromPosition
range
resizeCal

Supported Special Expression Functions

The following special expression functions are supported by AppFunctions:

activeindex
copyseries
maskedAgg
rangese
RepIndInRangeExpr
ScaleExpr
TransformSpread

TransformSpread

The TransformSpread special expression function converts data across hierarchies using different spreading flavors: transformProp, transformRepl, and transformEven.

Syntax

<target1, [target2, …, targetn]> <- TransformSpread (<transform spreading flavor>, <map>, Source [<hierarchy>].[<dimension>], Desination [<hierarchy>].[<dimension>]

< input1> [, < input2> … ,< inputn>])

Input Parameters

Table D–2 provides the input parameters for the TransformSpread function.

Table D–1 Input Parameters for the TransformSpread Function

Parameter Name	Description
Transform spreading flavor	Spreading flavor map
Source	[<hierarchy>].[<dimension>]
Destination	[<hierarchy>].[<dimension>]
Input1, [input2,...,inputn]	Input measures from which data are spread

Output Parameter

Table D–2 provides the output parameter for the TransformSpread function.

Table D–2 Output Parameter for the TransformSpread Function

Parameter Name	Description
target1, [target2, …, target]	Measures into which the source measures are spread to.

Example D–1 TransformSpread Function

```
mace -d . -run -expression "output1,output2,output3
<-TransformSpread(\"repl\",map,[CLSH].[CLST],[LOC].[STR],input1,input2,input3)"
```

E

Configuring the Preprocess Special Expression

This appendix details these topics:

About the Preprocess Module

The purpose of the Oracle Retail Preprocess module, which may also be referred to as Preprocessing, is to correct past data points that represent unusual sales values that are not representative of a general demand pattern. Such corrections may be necessary when an item is out of stock and cannot be sold, which usually results in low sales. Preprocessing will adjust for stock out for both the current week and the following week because it assumes that the out of stock indicators represent end of week stock out. Data Correction may also be necessary in a period when demand is unusually high. The Preprocess module allows you to automatically make adjustments to the raw POS (Point of Sales) data so that subsequent demand forecasts do not replicate undesired patterns that are caused by lost sales or unusually high demand.

The Preprocess Syntax section contains the specifications and syntax for configuring the Preprocess function in the RPAS Configuration Tools. There is an RPAS multi-return function named preprocess and one RPAS special expression named preprocess. The special expression provides better performance; however, it only works in the batch mode. The multiple return function preprocess works in both batch mode and workbook mode. The syntax is exactly the same in both modes, except that procedures use <- instead of = in the expression.

Note: The syntax is slightly different than the standard RPAS functions and procedures that are described in the Rule Functions Reference Guide section of the *Oracle Retail Predictive Application Server Configuration Tools User Guide*.

Preprocess Requirements

The following libraries must be registered in any domains that will use the Preprocess solution extension:

- AppFunctions
- PreprocesssFunctions

Configuration Restrictions

The following restrictions apply to use the Preprocess function/procedure:

- An underscore (_) character may not be used in any measure names and rules unless the measures and rules are to be expanded using the RDF or Curve solution's classification scheme.

The classifications apply the AppFunctions and are as follows:

- _F: Expand measures and rules across final levels
- _S: Expand measures and rules across source levels
- _B: Expand measures and rules across birth dates

Preprocess Parameter/Model Dependencies

The following models require that the stated measure is to be provided.

- Bayesian model—Plan measure required.
- Profile model—Profile measure required.

Using the Preprocess Function

The following notes are intended to serve as a guide for configuring the Preprocess function within the RPAS Configuration Tools:

- The Preprocess function is an RPAS C++ special expression. In order to get multiple results, the resultant measures must be configured in the Measure Tool, and the specific measure label must be used on the left-hand side (LHS) of the function call. The resultant measure parameters must be comma-separated in the function call as in the example.
- Because different filtering methods require different input parameters, it is necessary that every input parameter (measure or constant) must be accompanied by the corresponding label. All of the input measure parameters must be configured and registered before the function call. The input parameters must be comma-separated in the function call as in the example.
- The Preprocess function library must be registered after the domain build by using the regfunction RPAS utility.
- The Preprocess function required all the input and output measures using the same intersections. Mixed input/output measure intersections should be aligned to the same calculation intersection with other RPAS function/procedure before calling the Preprocess function. The same procedure can be carried out to the resultant measures to spread or aggregate them to the designated intersections.
- Because of the limitation that the same measure cannot simultaneously appear on both left-hand side and right-hand side, the implementation of the CLEAR filter requires the user to provide a LSOVER_REF measure (a duplication of the

previously calculated LSOVER measure) when you try to retain the results on certain time series but clear the others by providing a mask measure (TSMASK_DENSE). The LSOVER_REF is not required when the results for all the time series need to be cleared.

- The LSTODAY measure is used to specify the end date for the filter processing. It only accepts the index number for the end date along the calendar dimension as valid input. If it is desired that the string position name to be used for the end date specification, the available RPAS time dimension translation function index can be used to do the name-index conversion before calling the Preprocess function.
- The LSTODAY input parameter is designed to be a measure rather than a constant to provide more flexibility. Current implementation only allows one global LSTODAY index value to be used in processing all the time series. To specify the end date, you just need to populate its value for the first time series, and this index will be applied to all the other time series.
- The index value in the LSTODAY measure starts from 0.
- FLP_FIRST and FLP_LAST are the resultant measures to be used for the First-Last-Populated Location calculation. They do not have the calendar dimension, and each of their cell values represent the indices for the first and last populated locations along the calendar dimension from the first time series up to the current time series, respectively.
- TSMASK_DENSE is a Boolean input measure without calendar dimension to specify which time series is going to be processed and which is not. For filtering methods other than the CLEAR method, the true value means that it will be processed if the popcount for the current time series is larger than the hard-coded threshold value. Otherwise, it will not be processed. The false value means that the current time series will not be processed. If the TSMASK_DENSE measure is not specified, all the time series will be processed and the internal hard-coded threshold value will not be considered. For the CLEAR filtering method, the true value means that the previously calculated results for the current time series will be cleared and the false value means the results will be retained. If the TSMASK_DENSE measure is not specified, all the results will be cleared.
- For all the input measures that do not have the calendar dimension, such as UP_ADJ_RATIO and DELTA, you can use a constant as input. In this case, the constant value will be applied to all the time series.

Preprocess Syntax

The syntax for using the Preprocess is shown in the following examples. The input and output parameter tables explain the specific usage of the parameters names use in the function/procedure.

Example E–1 Generic Example 1:

LSOVER: LSOVERMEAS, LS: LSMEAS, [, TSALERT: TSALERTMEAS, SERVICE_LEVEL: SERVICELEVELMEAS, STOCK_LEVEL: STOCKLEVELMEAS, FLP_FIRST: FLPFIRSTMEAS, FLP_LAST: FLPLASTMEAS] <- preprocess(SRC: SRCMEAS, LSTODAY: LSTODAYMEAS, NPTS: NPTSMEAS [, MIN_TSALERT: MINTSALERTMEAS, OUTAGE: OUTAGEMEAS, TSMASK_DENSE: TSMASKMEAS, UP_ADJ_RATIO: UPADJMEAS, DOWN_ADJ_RATIO: DOWNADJMEAS, REFERENCE: REFMEAS, DEVIATION: DEVMEAS {, WINDOW: WINDOWMEAS |, WINDOW1: WINDOW1MEAS, WINDOW2: WINDOW2MEAS, WINDOW3: WINDOW3MEAS, WINDOW4: WINDOW4MEAS, WINDOW5: WINDOW5MEAS} {,

ALPHA: ALPHAMEAS, NPAST: NPASMEAS, NFUT: NFUTMEAS} {, NSIGMA_MIN: NSIGMA_MINMEAS, NSIGMA_MAX: NSIGMA_MAXMEAS |, NSIGMAOUT_MIN: NSIGMAOUT_MINMEAS, NSIGMAOUT_MAX: NSIGMAOUT_MAXMEAS, NSIGMAADJ_MIN: NSIGMAADJ_MINMEAS, NSIGMAADJ_MAX: NSIGMAADJ_MAXMEAS} {, FRCST_MIN: FRCST_MINMEAS, HIST_MIN_FS: HIST_MIN_FSMEAS} {, PRICE: PRICEMEAS, INVENTORY: INVENTORYMEAS, HIST_MIN_MD: HISTMINMDMEAS}, DELTA: DELTAMEAS, LSOVER_REF: LSOVERREFMEAS]

Example E–2 Generic Example 2:

LSOVER: LSOVERMEAS, LS: LSMEAS, [, TSALERT: TSALERTMEAS, SERVICE_LEVEL: SERVICELEVELMEAS, STOCK_LEVEL: STOCKLEVELMEAS, FLP_FIRST: FLPFIRSTMEAS, FLP_LAST: FLPLASTMEAS] <-preprocess(SRC: SRCMEAS, LSTODAY: LSTODAYMEAS, NPTS: NPTSMEAS [, MIN_TSALERT: MINTSALERTMEAS, OUTAGE: OUTAGEMEAS, TSMASK_DENSE: TSMASKMEAS, UP_ADJ_RATIO: UPADJMEAS, DOWN_ADJ_RATIO: DOWNADJMEAS, REFERENCE: REFMEAS, DEVIATION: DEVMEAS {, WINDOW: WINDOWMEAS |, WINDOW1: WINDOW1MEAS, WINDOW2: WINDOW2MEAS, WINDOW3: WINDOW3MEAS, WINDOW4: WINDOW4MEAS, WINDOW5: WINDOW5MEAS} {, ALPHA: ALPHAMEAS, NPAST: NPASMEAS, NFUT: NFUTMEAS} {, NSIGMA_MIN: NSIGMA_MINMEAS, NSIGMA_MAX: NSIGMA_MAXMEAS |, NSIGMAOUT_MIN: NSIGMAOUT_MINMEAS, NSIGMAOUT_MAX: NSIGMAOUT_MAXMEAS, NSIGMAADJ_MIN: NSIGMAADJ_MINMEAS, NSIGMAADJ_MAX: NSIGMAADJ_MAXMEAS} {, FRCST_MIN: FRCST_MINMEAS, HIST_MIN_FS: HIST_MIN_FSMEAS} {, PRICE: PRICEMEAS, INVENTORY: INVENTORYMEAS, HIST_MIN_MD: HISTMINMDMEAS}, DELTA: DELTAMEAS]

Example E–3 Sample 1:

LSOVER:LSOVER1, LS:LS1, TSALERT:TSALERT1 <- preprocess(SRC:POS, METHODID:MTHID, LSTODAY:TODAY1, NPTS:NPTS, WINDOW:WIN)

Example E–4 Sample 2:

LSOVER:LSOVER1, LS:LS1, TSALERT:TSALERT1 <- preprocess(SRC:POS, METHODID:MTHID, LSTODAY:TODAY1, NPTS:NPTS, WINDOW:WIN)

Configuration Parameters and Rules

Input Parameters

Table E–1 provides the input parameters for the Preprocess procedure.

Table E–1 Input Parameters for the Preprocess Procedure

Parameter Name	Description
SRC	The source data. Data Type: Real Required: Yes
METHODID	The filtering method ID. Data Type: Real Required: Yes

Table E–1 (Cont.) Input Parameters for the Preprocess Procedure

Parameter Name	Description
LSTODAY	The end date for filter processing. Data Type: Real Required: Yes
NPTS	The number of points into history that will be filtered. Data Type: Real Intersection: Required: Yes
MIN_TSALERT	The threshold value used to set off TSALERT. Data Type: Real Required: No
OUTAGE	The outage indicator. Data Type: Boolean Required: No
TSMASK_DENSE	A Boolean value to specify which time series will be processed. Data Type: Boolean Required: No
UP_ADJ_RATIO	The upward adjustment ratio that will be applied on LS. Data Type: Real Required: No Default value: 1.0* * If the measure is not specified, the default value will be applied to each of the time series to be processed.
DOWN_ADJ_RATIO	The downward adjustment ratio that will be applied on LS. Data Type: Real Required: No Default value: 1.0* * If the measure is not specified, the default value will be applied to each of the time series to be processed.
REFERENCE	Reference will be used for source data substitution. Data Type: Real Required: No
DEVIATION	The standard deviation for confidence interval calculation by Forecast Sigma filters. Data Type: Real Required: No
WINDOW	Filter window length for Standard Median filter. Data Type: Real Required: No Default value: 13

Table E–1 (Cont.) Input Parameters for the Preprocess Procedure

Parameter Name	Description
WINDOW1	First round filter window length for Oracle Retail Median filter. Data Type: Real Required: No Default value: 13
WINDOW2	Second round filter window length for Oracle Retail Median filter. Data Type: Real Required: No Default value: 19
WINDOW3	Third round filter window length for Oracle Retail Median filter. Data Type: Real Required: No Default value: 7
WINDOW4	Forth round filter window length for Oracle Retail Median filter. Data Type: Real Required: No Default value: 5
WINDOW5	Fifth round filter window length for Oracle Retail Median filter. Data Type: Real Required: No Default value: 11
ALPHA	The exponential coefficient used to evaluate past and future velocities. Data Type: Real Required: No Default value: 0.2
NPAST	The maximum number of historical points to calculate past velocity. Data Type: Real Required: No Default value: 5
NFUT	The maximum number of historical points to calculate future velocity. Data Type: Real Required: No Default value: 5
NSIGMA_MIN	The number of standard deviations for lower bound calculation. Data Type: Real Required: No Default value: 3.0

Table E–1 (Cont.) Input Parameters for the Preprocess Procedure

Parameter Name	Description
NSIGMA_MAX	The number of standard deviations for upper bound calculation. Data Type: Real Required: No Default value: 3.0
FRCST_MIN	The forecast lower bound for Forecast Sigma filters. Data Type: Real Required: No Default value: 0.1
HIST_MIN_FS	The minimum number of historical points required for Forecast Sigma filters. Data Type: Real Required: No Default value: 5
NSIGMAOUT_MIN	The number of standard deviations for lower outlier calculation. Data Type: Real Required: No Default value: 3.0
NSIGMAOUT_MAX	The number of standard deviations for upper outlier calculation. Data Type: Real Required: No Default value: 3.0
NSIGMAADJ_MIN	The number of standard deviations for lower bound calculation. Data Type: Real Required: No Default value: 1.5
NSIGMAADJ_MAX	The number of standard deviations for upper bound calculation. Data Type: Real Required: No Default value: 1.5
DELTA	Ratio of reference will be used to copy or increase for OVERRIDE and INCREMENT filters. Data Type: Real Required: No Default value: 1.0* * If the measure is not specified, the default value will be applied to each of the time series to be processed.
LSOVER_REF	Data will be used to override SRC. Used by CLEAR filter only. Data Type: Real Required: No

Table E–1 (Cont.) Input Parameters for the Preprocess Procedure

Parameter Name	Description
POA	Partial Outage Flag. Used by STD ES LS filter only. If set to *False*, the period immediately following an out-of-stock period will not be adjusted. The default value is *True*. Data Type: Boolean Required: No
EVENT_FLAG	Used by STD ES and STD ES LS filters. If a value is *True*, the given period is not used in the calculation of the past or future velocities. Data Type: Boolean Required: No
STOP_AT_EVENT FLAG	Used by STD ES and STD ES LS filters. This parameter determines which periods are included in the calculation of past/future velocities. If the flag is set to *True*, then the algorithm only includes periods before the first event flag or event indicator. If the flag is set to *False*, then all available, non-flagged periods, within the windows defined by nfut and npast, are used in the calculation of the past and future velocities. The default setting for the flag is *False*
SRCPOPCUT	The minimum number of non-zero sales data points in the preprocessing window. If the non-zero sales data points number is less than this parameter, then the time series is not processed. Data Type: Numeric Required: No. If this parameter is not provided, it defaults to 3 in the code.

Output Parameters

Table E–2 provides the output parameters for the Preprocess function/procedure.

Table E–2 Output Parameters for the Preprocess Procedure

Parameter Name	Description
LSOVER	Adjusted source data. It is the Primary Result. LSOVER = SRC + LS Data Type: Real Required: Yes
LS	The adjustment on the source data. Data Type: Real Required: Yes
TSALERT	Boolean flag set to *True* when more than MIN_TSALERT number of data points have been modified. Data Type: Boolean Required: No
SERVICE_LEVEL	SERVICE_LEVEL = SRC / LSOVER Data Type: Real Required: No

Table E–2 (Cont.) Output Parameters for the Preprocess Procedure

Parameter Name	Description
STOCK_LEVEL	Used by Mark Down filter only. Data Type: Real Required: No
FLP_FIRST	First populated position. Used by FLP filter only. Data Type: Real Required: No
FLP_LAST	Last populated position. Used by FLP filter only. Data Type: Real Required: No

Lost Sales Method/Model List

Table E–3 provides the numeric value assigned to the forecast model/model list.

Table E–3 Numeric Values Assigned to the Lost Sales Model/Model List

Model	Numeric Value	Comments
MEDIAN5	0	Oracle Retail Median. Required input parameters: None Optional input parameters: ■ WINDOW1 ■ WINDOW2 ■ WINDOW3 ■ WINDOW4 ■ WINDOW5
MEDIAN1	1	Standard Median. Required input parameters: None Optional input parameters: WINDOW
OVERRIDE	2	Override Required input parameters: REFERENCE Optional input parameters: DELTA
INCREMENT	3	Increment. Required input parameters: REFERENCE Optional input parameters: DELTA
ES_LT	4	Standard ES. Required input parameters: OUTAGE Optional input parameters: ■ ALPHA ■ NPAST ■ NFUT ■ EVENT_FLAG

Table E–3 (Cont.) Numeric Values Assigned to the Lost Sales Model/Model List

Model	Numeric Value	Comments
LS_ES_LT	9	Lost Sales—Standard ES. Required input parameters: OUTAGE Optional input parameters: ■ ALPHA ■ NPAST ■ NFUT ■ EVENT_FLAG ■ POA
FRCST_SIGMA	14	Forecast and standard deviation algorithm. Required input parameters: ■ REFERENCE ■ DEVIATION Optional input parameters: ■ NSIGMA_MAX ■ NSIGMA_MIN ■ FRCST_MIN ■ HIST_MIN_FS
FRCST_SIGMA_EVENT	15	Forecast and standard deviation algorithm with Event. Required input parameters: ■ OUTAGE ■ REFERENCE ■ DEVIATION Optional input parameters: NSIGMAOUT_MAX ■ NSIGMAOUT_MIN ■ NSIGMAADJ_MAX ■ NSIGMAADJ_MIN ■ FRCST_MIN ■ HIST_MIN_FS
CLEAR	17	Clear—clears specified result measures. Required input parameters: None Optional input parameters: ■ TSMASK_DENSE ■ LSOVER_REF
NO_FILT	19	No filtering. Required input parameters: None Optional input parameters: None
DEPRICE	22	Remove pricing effects Required input parameters:price, maximum price Optional input parameters:none

Table E–3 (Cont.) Numeric Values Assigned to the Lost Sales Model/Model List

Model	Numeric Value	Comments
FLP: first populated location	20	Required input: src and method

Preprocess Filtering Methods

This section details the following Preprocess filtering methods:

- Standard Median
- Oracle Retail Median
- Standard Exponential Smoothing
- Lost Sales—Standard Exponential Smoothing
- Forecast Sigma
- Forecast Sigma Event
- Override
- Increment
- Clear
- DePrice

Standard Median

Standard Median is recommended for getting data baselines on long time ranges when promo indicators are not available.

A standard median filter implementation:

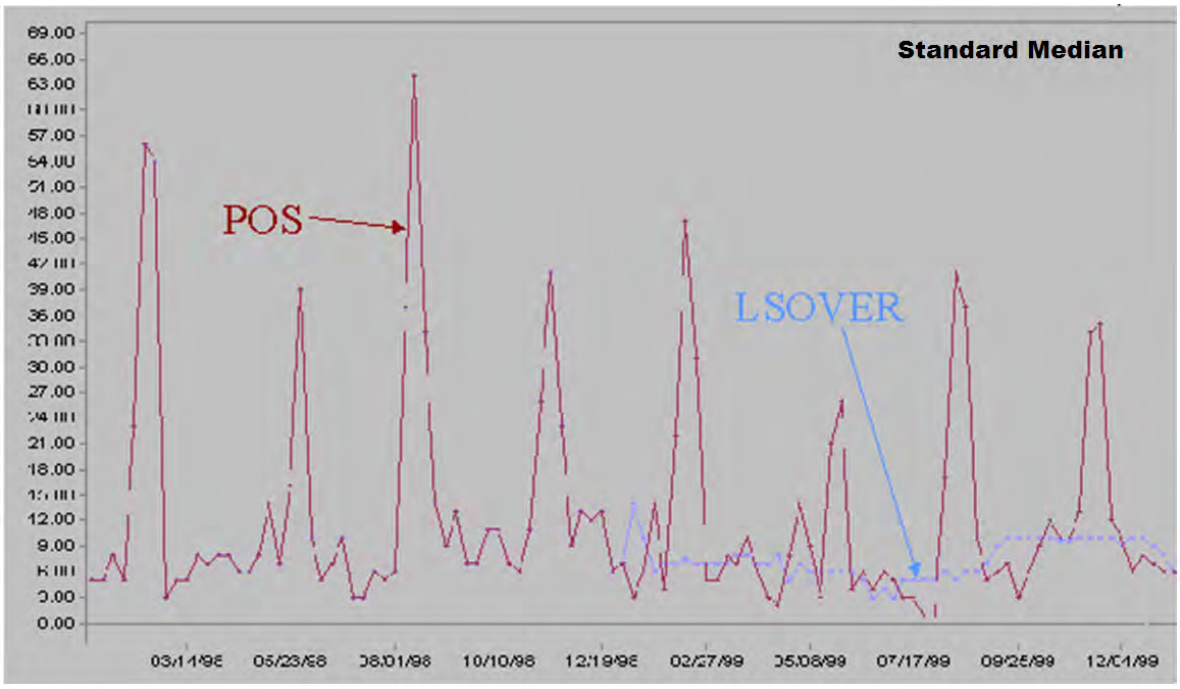
- Does not take outage information as an input.
- Can use one optional parameter: window length.

Mathematical Formulation

LSOVER(t) = median value of SRC over [t-window/2, t+window/2],

Where: window is the parameter window length of the filter.

Figure E–1 Standard Median with Window = 13 points



Example E–5 Standard Median with Window = 13 points

LSOVER:lsover1, LS:ls1, TSALERT:tsalert1 <- preprocess(SRC:pos, METHODID:mthid, LSTODAY:today1, NPTS:npts, WINDOW:win)

Oracle Retail Median

Oracle Retail Median is recommended for getting data baselines on long time ranges when promo indicators are not available.

Oracle Retail Median provides the following features:

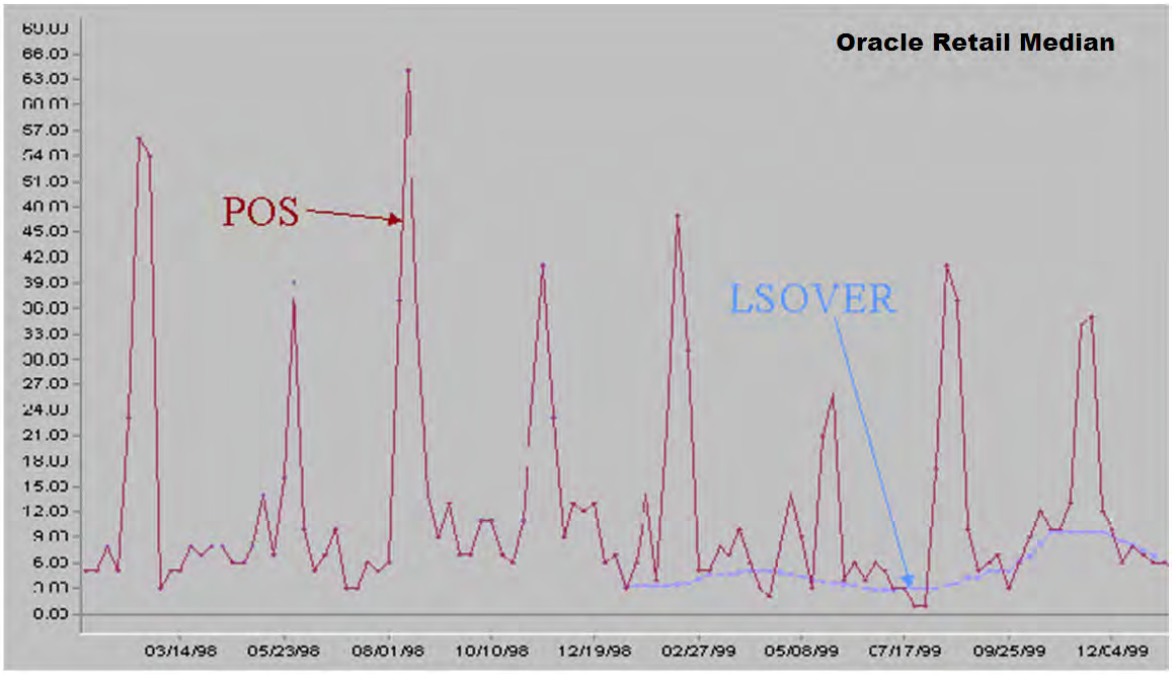
- A sophisticated median filter that takes trends into consideration and improves side effects over the standard median filter. It makes five standard median filter passes.
- Does not take outage information as an input.
- Can accept five optional parameters: window length for each pass.

Mathematical Formulation

1. The first two passes recursively apply the standard median filter. The result is denoted by MEDIAN_2(t). The one-step difference of MEDIAN_2(t) is calculated. That is, DIFF_1(t) = MEDIAN_2(t) - MEDIAN_2(t-1). Then, the standard median filter is applied to DIFF_1(t). The result is denoted by MEDIAN_DIFF_1(t).
2. Using MEDIAN_DIFF_1(t), a first smoothed version (that is, baseline) of the source data is calculated at the third step: SMOOTH_1(t) = SMOOTH_1(t-1) + MEDIAN_DIFF_1(t) on points where the absolute deviation of SRC(t) over its mean is larger than half of the global absolute standard deviation. Otherwise, SMOOTH_1(t) = SRC(t).

3. To prepare for the fourth pass, the one-step difference of SMOOTH_1(t) is calculated. That is, DIFF_2(t) = SMOOTH_1(t) - SMOOTH_1(t-1). An average version of DIFF_2(t) is calculated using the standard median filter. The result is denoted by AVG_DIFF_2(t). The result of the fourth pass is SMOOTH_2(t) = SMOOTH_2(t-1) + AVG_DIFF_2(t).
4. Finally, LSOVER(t) is the result of applying the standard median filter to SMOOTH_2(t).

Figure E–2 Oracle Retail Median with Default Parameters



Example E–6 Oracle Retail Median with Default Parameters

LSOVER:lsover1, LS:ls1, TSALERT:tsalert1 <- preprocess(SRC:pos, METHODID:mthid, LSTODAY:today1, NPTS:npts, WINDOW1:win, WINDOW2:win2, WINDOW3:win3, WINDOW4:win4, WINDOW5:win5)

Standard Exponential Smoothing

Standard Exponential Smoothing (Std ES) removes spikes (such as promotional promo, temporary price changes, and so on), as well as filling the gaps (out of stock, unusual events such as a fire or hurricane).

Input: An Event Indicator that indicates which periods should be preprocessed.

Optional Parameters:

The following table details the optional parameters for Standard Exponential Smoothing.

Optional Parameters	Description
ES (Exponential Smoothing)	The alpha parameter that determines the weight put on observations of periods included in the calculations.

Optional Parameters	Description
Number of future periods (nfut)	The number of periods after an outage periods that are considered in the calculation of the future velocity. Note that if during these periods an event flag or a event indicator is on, the particular period is excluded from the calculation.
Number of past periods (npast)	The number of periods before an outage periods that are considered in the calculation of the past velocity. Note: When calculating the past velocity and the first period in the preprocessing window is flagged, then the past velocity is calculated using earlier periods outside the preprocessing window. Note that if during these periods an event flag or a event indicator is on, the particular period is excluded from the calculation.
Event flag	This parameter indicates if a period should be excluded from the calculation of past/future velocities.
Stop at event flag	This parameter determines which periods are included in the calculation of past/future velocities. If the flag is set to True, then the algorithm only includes periods before the first event flag or event indicator. If the flag is False, then all available, non-flagged periods, within the windows defined by nfut and npast, are used in the calculation of the past and future velocities. The default setting for the flag is False.

Mathematical Formulation

Std ES is the standard Exponential Smoothing filter. It preprocesses a subset of points as predetermined by an input measure. For every contiguous sequence of points to adjust, say between *tf* and *tl*, a past velocity and a future velocity are calculated using an exponentially weighted average. For the points between *tf* and *tl*, the adjustment is calculated as a linear interpolation of the past and future velocities.

Figure E–3 Standard Exponential Smoothing Calculations

$$Past\_Velocity = \frac{\sum_{i=1}^{np}(1-\alpha)^{i-1} * SRC(t_f - i)}{\sum_{i=1}^{np}(1-\alpha)^{i-1}}$$

$$Future\_Velocity = \frac{\sum_{i=1}^{nf}(1-\alpha)^{i-1} * SRC(t_l + i)}{\sum_{i=1}^{nf}(1-\alpha)^{i-1}}$$

$$LSOVER(t) = Past\_Velocity + \frac{Future\_Velocity - Past\_Velocity}{t_l - t_f + 2} * (t - t_f + 1), \forall t \in [t_f, t_l]$$

Where:

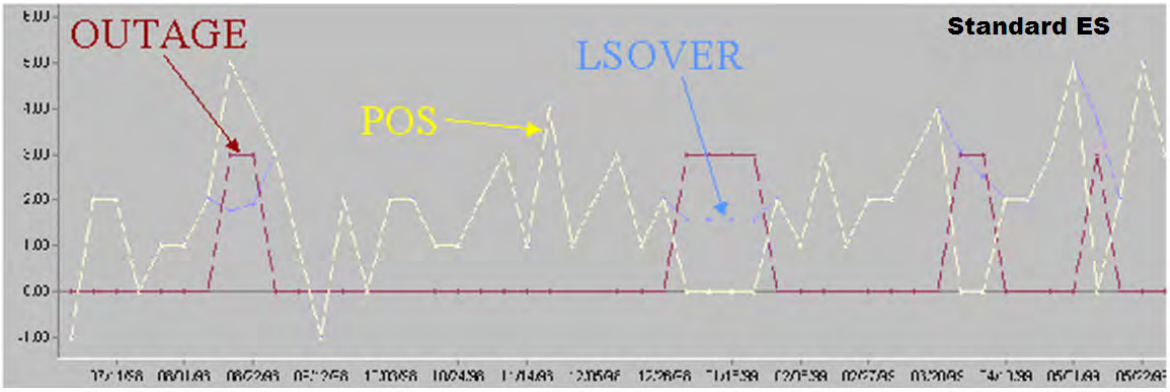
α is the exponential coefficient used to evaluate past and future velocities.

np is the maximum number of historical points to calculate past velocity.

nf is the maximum number of future points to calculate future velocity.

Figure E–4 Standard Exponential Smoothing

Std ES with α = 0.2, np = 2 weeks, and nf = 2 weeks



Example E–7 Standard Exponential Smoothing

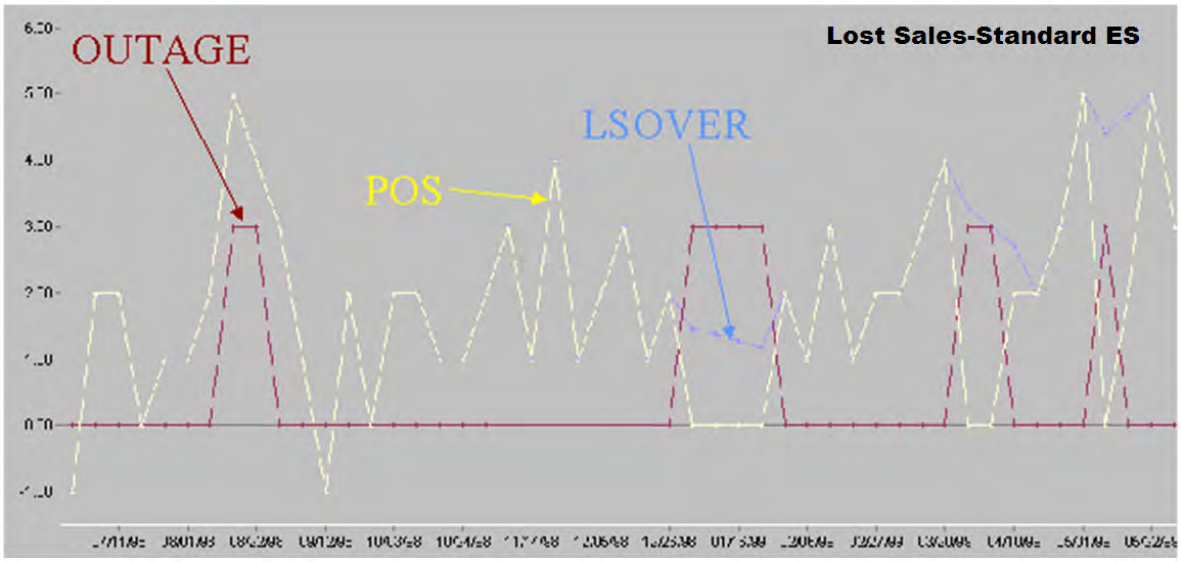
LSOVER:lsover1, LS:ls1, TSALERT:tsalert1 <- preprocess(SRC:pos, METHODID:mthid, LSTODAY:today1, NPTS:npts, OUTAGE:outage1, ALPHA:alpha, NPAST:npast, NFUT:nfut)

Lost Sales—Standard Exponential Smoothing

Lost Sales—Standard Exponential Smoothing functions like Std ES with two exceptions. First, it only adjusts lost sales (that is, negative spikes). Second, it can adjust not only the out-of-stock period but also the period immediately following such a period (partial outage period).

Figure E–5 Lost Sales—Standard Exponential Smoothing

Lost Sales -- Std ES with α = 0.2, np = 2 weeks, and nf = 2 weeks



Example E–8 Lost Sales—Standard Exponential Smoothing

LSOVER:lsover1, LS:ls1, TSALERT:tsalert1 <- preprocess(SRC:pos, METHODID:9, LSTODAY:today1, NPTS:30, OUTAGE:outage1, ALPHA:0.2, NPAST:5, NFUT:5,EVENT_FLAG: event_flag_measure, POA:TRUE)

Forecast Sigma

Forecast Sigma is recommended for removing recent spiky data points when approved forecasts and approved confidence intervals are available on the filtering window, but spike indicators are not available. This method is based on the principle that if a data point significantly deviates from an approved forecast, this data point is likely to be an unusual event that should be overridden in the source measure (POSOVER) used by the forecasting engine. It is adjusted by bringing the override value within some bounds of the approved forecast as defined by a proportional coefficient scalar of the forecasts' standard deviation.

Forecast Sigma provides the following features:

- Does not take outage information as an input
- Requires two parameters:
  - Approved forecast array
  - Approved standard deviation array of forecast
- Can accept four optional parameters:
  - Number of standard deviations for upper bound
  - Number of standard deviations for lower bound.
  - Forecast lower bound
  - Minimum item history (# points) required for filtering

Mathematical Formulation

This method relies on approved forecasts with their corresponding confidence intervals. It adjusts the points that are far (as defined by a multiple of the forecast

standard deviation) from their corresponding previously approved forecasts by bringing the override values to their closest confidence interval bounds.

IF # historical points < MinHist **THEN**
LSOVER(t) = SRC(t)
ELSE IF forecast(t) < MinFrcst **THEN**
forecast(t) = MinFrcst **AND** σ = MinFrcst
ELSE IF $\sigma = 0$ **THEN**
IF forecast(t) < 1.0 **THEN**
σ = forecast(t)
ELSE $\sigma = \sqrt{\text{forecast(t)}}$
IF SRC(t) > forecast(t) + nsu*σ **THEN**
LSOVER(t) = forecast(t) + nsu*σ
ELSE IF SRC(t) < forecast(t) – nsl*σ **THEN**
LSOVER(t) = forecast(t) - nsl*σ
ELSE LSOVER(t) = SRC(t)

Where:

- nsu is the number of standard deviations for upper bound.
- nsl is the number of standard deviations for lower bound.
- MinFrcst is the forecast lower bound.
- MinHist is the minimum item history (# points) required for filtering.

Example E–9 Lost Sales—Forecast Sigma with nsu = 3, nsl = 3, minFrcst = 0.1 and minHist = 5 weeks

LSOVER:LSOVER1, LS:LS1, TSALERT:TSALERT1 <- preprocess(SRC:POS, METHODID:mthid, LSTODAY:TODAY1, NPTS:npts, REFERENCE:forecast1, DEVIATION:dev1, NSIGMA_MIN:nsigma_min, NSIGMA_MAX:nsigma_max, FRCST_MIN:0.1, HIST_MIN_FS:hist_min_fs)

Forecast Sigma Event

This is similar to Forecast Sigma. It takes an outage (for instance, event) indicator to further process.

Mathematical Formulation

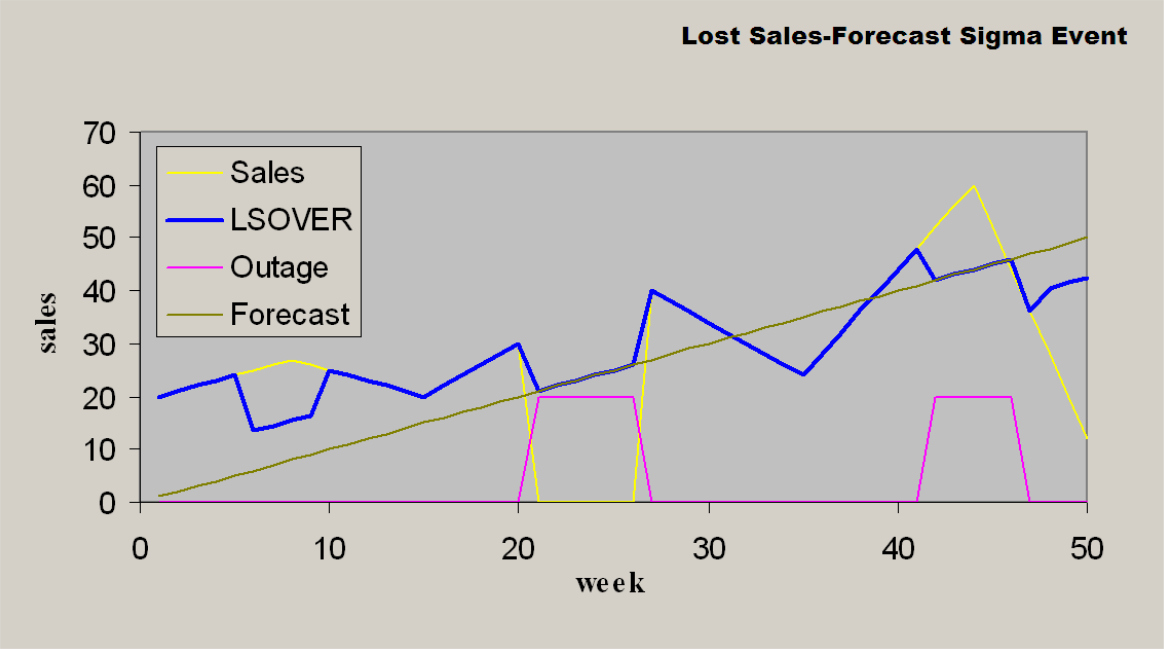
When the outage/event mask is ON:

LSOVER(t) = forecast(t)

When the outage/event mask is OFF:

If the data points that are outside of the outliers calculated through NSIGMAOUT_MIN and NSIGMAOUT_MAX, they will be brought into the confidence interval bounds, which are defined through NSIGMAADJ_MIN and NSIGMAADJ_MAX.

Figure E–6 Lost Sales—Forecast Sigma Event



Lost Sales—Forecast Sigma Event with nsigmaout_min = 3, nsigmaout_max = 3, nsigmaadj_min = 1.5, nsigmaadj_max = 1.5,

minFrcst = 0.1 and minHist = 5 weeks

Example E–10 Lost Sales—Forecast Sigma Event

LSOVER:LSOVER1, LS:LS1, TSALERT:TSALERT1 <- preprocess(SRC:POS, METHODID:mthid, LSTODAY:TODAY1, NPTS:npts, OUTAGE:outage1, REFERENCE:forecast1, DEVIATION:dev1, NSIGMAOUT_MIN:nsigmaout_min, NSIGMAOUT_MAX:nsigmaout_max, NSIGMAADJ_MIN:nsigmaadj_min, NSIGMAADJ_MAX:nsigmaadj_max, FRCST_MIN:frcst_min, HIST_MIN_FS:hist_min_fs)

Override

This method overrides the destination measure with the source measure that is adjusted by the adjustment percentage according to the mask. It is recommended for filling data gaps when an existing reference measure exists as a default value.

Override provides the following features:

- It is a simple data copy of a given percentage of the reference data to copy from.
- This may or may not take outage (for instance, event) info as an input to mask the operation.
- Requires two parameters:
  - Reference measure to copy data from

  - Source measure for the original data
- Can accept one optional parameter, Ratio of reference to actually copy.

Mathematical Formulation

This method uses the following parameters:

- A source measure that can be any measure in the system as long as it has the same intersection as the destination measure
- A reference measure that can be any measure in the system as long as it has the same intersection as the destination measure
- A destination measure that can be any measure in the system as long as it has the same intersection as the source measure
- A mask that is a Boolean measure that has the same intersection as the source and destination measures
- An adjustment percentage

This method overrides the destination measure with the source measure adjusted by the adjustment percentage according to the mask:

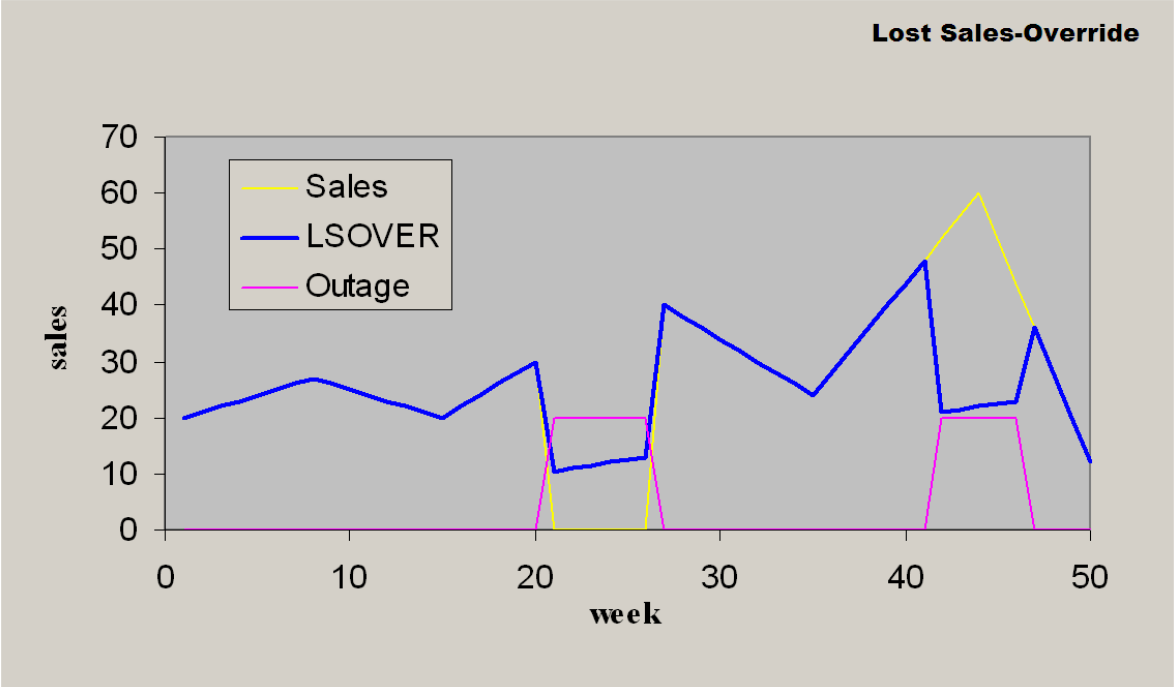
Let:

- S(i) is the value in cell (i) of the source measure
- R(i) is the value in cell (i) of the reference measure
- D(i) is the value in cell (i) of the destination measure
- M(i) is the value of cell (i) of the mask
- a is an adjustment percentage

The result of the override method is:

- D(i) = a * R(i) if M(i) is *True*
- D(i) = S(i) if M(i) is *False*

Figure E–7 Lost Sales—Override with Delta = 0.5



Example E–11 Lost Sales—Override with Delta = 0.5

LSOVER:lsover1, LS:ls1, TSALERT:tsalert1 <- preprocess(SRC:pos, METHODID:mthid, LSTODAY:today1, NPTS:npts, REFERENCE:ref1, OUTAGE:outage1, DELTA:delta1)

Increment

This method increments or decrements the destination measure by the source measure, which is adjusted by the adjustment percentage according to the mask. It is recommended for updating outliers or data gaps when an existing reference measure exists as a default adjustment.

Increment provides the following features:

- It is a simple data increment of a given percentage of the reference data to copy from.
- It may or may not take outage information (for example, event) as an input to mask the operation.
- Has one required parameter, Reference measure to increment by.
- Can accept one optional parameter, Ratio of reference to actually increment by.

Mathematical Formulation

This method uses the following inputs:

- A source measure that can be any measure in the system as long as it has the same intersection as the destination measure.
- A reference measure that can be any measure in the system as long as it has the same intersection as the destination measure.

- A destination measure that can be any measure in the system as long as it has the same intersection as the source measure.
- A mask that is a Boolean measure that has the same intersection as the source and destination measures.
- An adjustment percentage.

This method increments or decrements the destination measure by the source measure, which is adjusted by the adjustment percentage according to the mask.

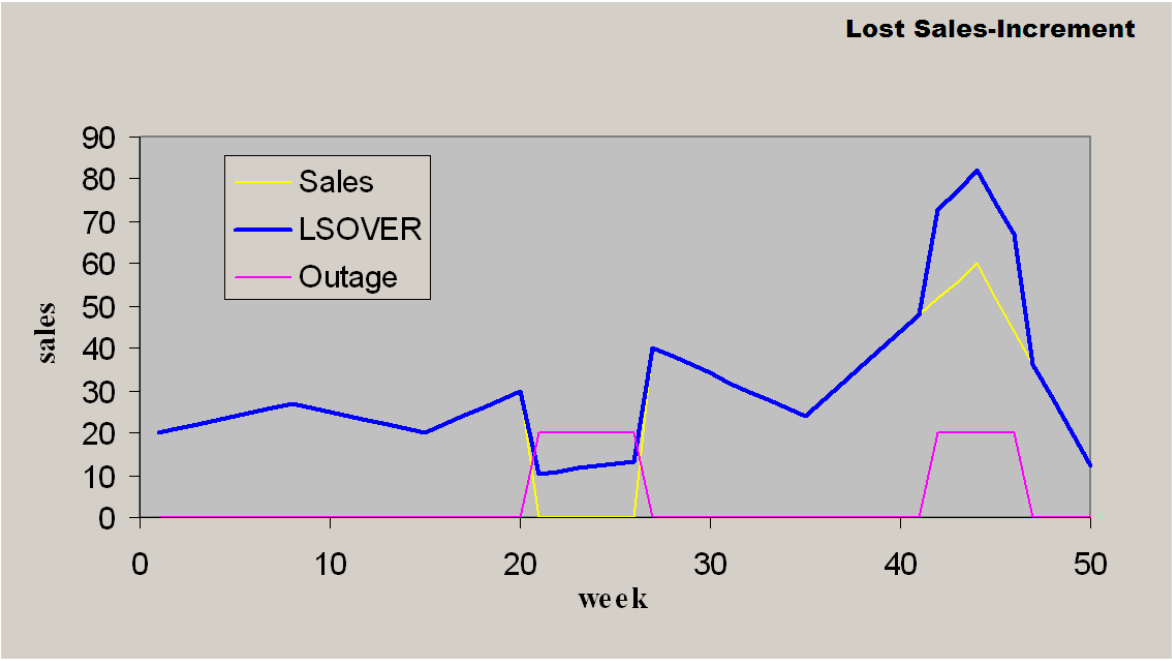
Let:

- S(i) is the value in cell (i) of the source measure
- R(i) is the value in cell(i) of the reference measure
- D(i) is the value in cell (i) of the destination measure
- M(i) is the value of cell (i) of the mask
- a is an adjustment percentage (can be between (-100%) and (+100%))

The result of the reduction method is:

- D(i) = S(i) + a * R(i) if M(i) is *True*
- D(i) = S(i) if M(i) is *False*

Figure E–8 Lost Sales—Increment with Delta = 0.5



Example E–12 Lost Sales—Increment with delta = 0.5

LSOVER:lsover1, LS:ls1, TSALERT:tsalert1 <- preprocess(SRC:pos, METHODID:mthid, LSTODAY:today1, NPTS:npts, REFERENCE:ref1, OUTAGE:outage1, DELTA:delta1)

Clear

This is used for canceling the effect of some former preprocessing adjustments.

Clear provides the following features:

- Does not take outage information as an input.
- May or may not take time series mask (does not have calendar dimension) input to retain results for certain time series.
- If time series mask is specified, one duplicated LSOVER measure must be provided in addition to the original LSOVER measure.

Mathematical Formulation

IF TimeSeriesMask is provided & TimeSeriesMask = false **THEN**

LSOVER(t) = LSOVER_REF(t)

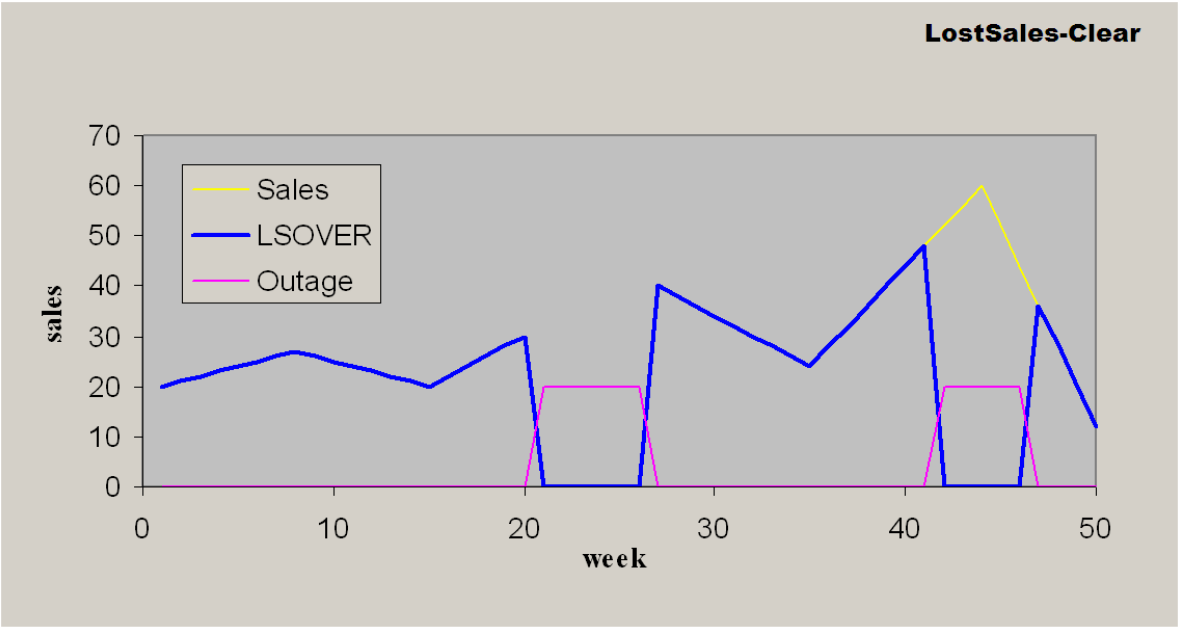
LS(t) = LSOVER_REF(t) - SRC(t)

ELSE

LSOVER(t) = 0

LS(t) = 0

Figure E–9 Preprocess—Clear with TS_Mask



Example E–13 Clear All

LSOVER:LSOVER1, LS:LS1, TSALERT:TSALERT1 <- preprocess(SRC:POS, METHODID:mthid, LSTODAY:TODAY1, NPTS:npts)

Example E–14 Partial Clear with Mask Input

LSOVER:LSOVER1, LS:LS1, TSALERT:TSALERT1 <- preprocess(SRC:POS, METHODID:mthid, LSTODAY:TODAY1, NPTS:npts, TSMASK_DENSE:tsMask1, LSOVER_REF:lsoverref1)

DePrice

A filter removing pricing effects:
smoothed=original*(price/maxprice)*(price/maxprice)

LSOVER:LSOVER1, LS:LS1 <-preprocess(SRC:POS, METHODID:mthid, LSTODAY:TODAY1, NPTS:npts, TSMASK_DENSE:tsMask1,PRICE:price.MAX LSOVER_REF:lsoverref1)

F

Customizing Hooks for the RDF Generate Utility and Curvebatch

RDF and Curve provide a number of hooks for running customized computation at certain points of the batch process.

Generate is an RDF utility that runs the RDF batch, such as generating profiles, generating source level forecasts, and approving forecasts. Often, some customized computation is needed during the running of *generate.* For instance, after the generation of forecast and prior to the forecast approval, you may want to run some rules to compute the value of the approval alerts.

Curvebatch is a Curve utility that runs profile generation batch, such as range data source, generate source level profiles, and merge profiles.

This appendix details these topics:

Hooks

There are several hooks provided in the RDF Generate and Curvebatch utilities so that custom expressions can be run at various predefined phases of the batch.

The hooks listed in the following tables are defined in the appcust.xml file.

Phases and Hooks in RDF

Each phase of the batch cycle begins and ends with a hook that links to the next phase. Table F–1 lists and describes the RDF batch phases along with its hooks. These hooks are provided in the RDF Generate utility and defined in the appcust.xml file as shown in Figure F–1.

Table F–1 RDF Phases and Hooks

Phase	Phase Action	This Hook...	Is Run...
Initialization	Prepares initializing environment for forecast generation	preinit	before initialization starts
		postinit	after initialization
Forecast Generation	Generates forecast	pregen	before generate forecast
		postgen	after generate forecast

Table F–1 (Cont.) RDF Phases and Hooks

Phase	Phase Action	This Hook...	Is Run...
Like Functionality Performance	Runs a like item sister store functionality	prelikets	Before likets function
		postlikets	After likets function
Forecast Adjustment	Automatic adjustment for forecast	preadjust	Before adjust forecast
		postadjust	After adjust forecast
Alert Execution	Runs alerts	prealert	Before generate alerts
		postalert	After generate alerts
Forecast Approval	Automatic approval of forecast	preapprove	Before forecast approval
		postapprove	After forecast approval

Phases and Hooks in Curve

Each phase of the batch cycle begins and ends with a hook that links to the next phase. Table F–2 lists and describes the Curve batch phases along with its hooks. These hooks are provided in the Curvebatch utility and defined in the appcust.xml file as shown in Figure F–2.

Table F–2 Curve Phases and Hooks

Phase	Phase Action	This Hook...	Is Run...
Ranging	Ranging the source data based on training window	prerangesource	Before rangeDataSource is run
		postrangesource	After rangeDataSource is run
Profile Generation	Generate profile at all source levels	prerunsource	One time before SourceLevel:run is iteratively performed on all source levels of the profile
		postrunsource	One time after SourceLevel:run is iteratively performed on all source levels of the profile
Profile Merger	Merge profiles	premerge	Before merge is run
		postmerge	After merge is run
Profile Reshape	Reshape profiles	prereshape	Before reshape is run
		postreshape	After reshape is run
Profile Renormalize	Renormalize profiles	prerenormalize	Before renormalize is run
		postrenormalize	After renormalize is run
Profile Clip	Clip profiles based on phase start and end dates	prephaseclip	Before phaseClip is run
		postphaseclip	After phaseClip is run
Profile Approval	Automatic approvals of profiles	preapprove	Before approve is run
		postapprove	After is approve run

About appcust.xml

The hooks listed in "Hooks" on page F-1 are defined in the appcust.xml file. For the setup directory of each domain, appcust.xml must be included.

The format of the RDF appcust.xml file is shown in Figure F–1.

The format of the Curve appcust.xml file is shown in Figure F–2.

Figure F–1 Format of appcust.xml for RDF

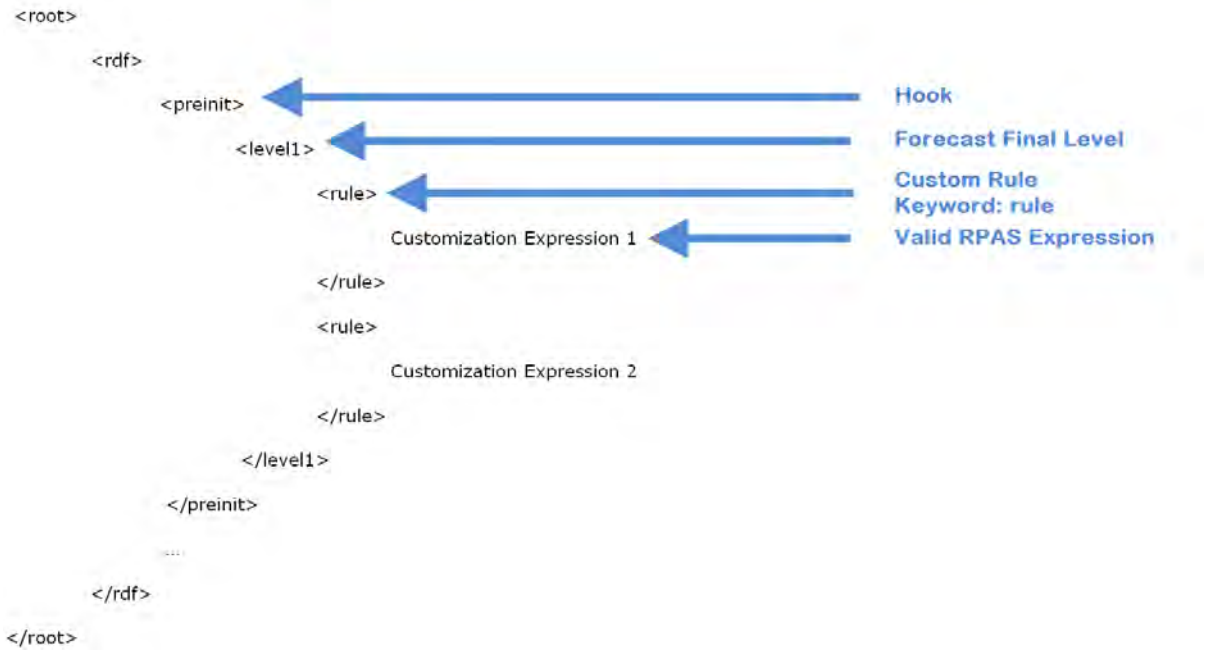
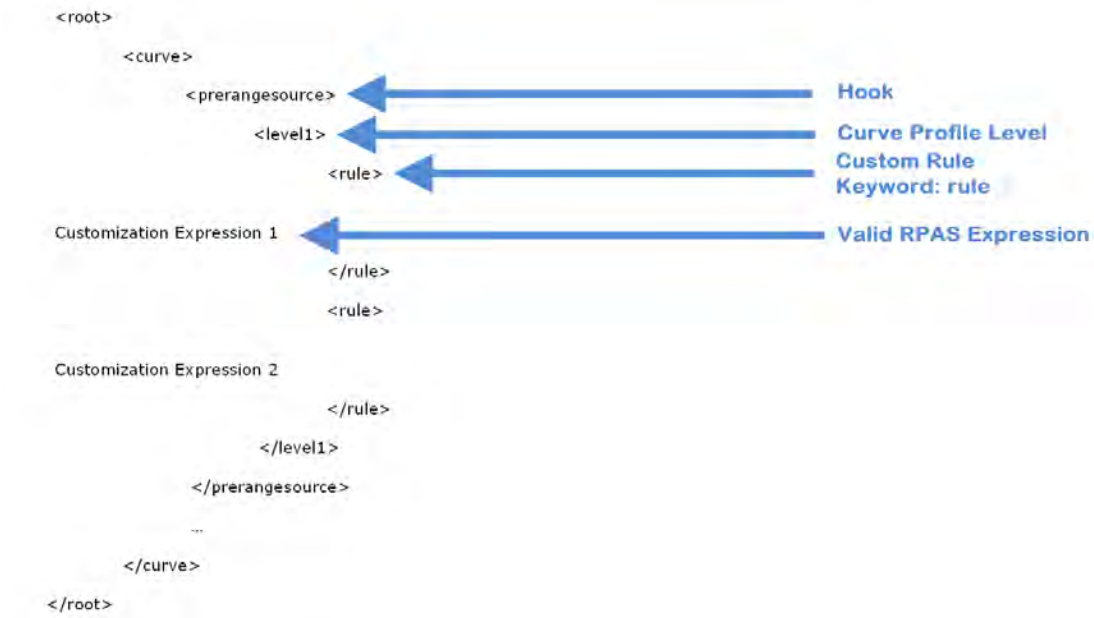


Figure F–2 Format of appcust.xml for Curve



G

Curve Configuration Process

Curve is an RPAS solution that is used to generate ratios from historical data at user-specified intersections. The profiles generated by Curve can be used for various purposes:

- To convert the organization-level assortment plans into base level weekly sales forecasts
- For generating seasonal forecasts, daily forecasts, or new product forecasting using lifecycle profiles

RDF requires profiles (created by Curve) to determine how a source level forecast (for instance, Item/Chain/Week) is spread down to the execution or final level (for instance, Item/Store/Day). Profiles are generated using historical data and phase definitions that are based on the system configuration. Using the Curve Plug-In, profiles are defined to support the Curve solution.

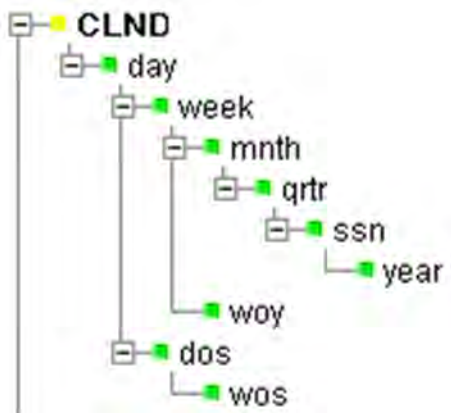
Note: For information on building the Curve domain, refer to the *Oracle Retail Predictive Application Server Installation Guide*.

Curve Hierarchy Configuration Requirements

The following dimensions are required to support different seasonal profiles. If the following defined types of profiles are not required, these hierarchy dimensions may not be necessary:

- **dos** (day of season) - A dimension off day, dos is used to support seasonal profiles that are normalized to day. This profile should use the Daily Seasonal profile type.
- **wos** (week of season) - A dimension off day or dos, wos is used to support seasonal profiles normalized to week. This profile can use the Store Contribution, Product Profile, or User Defined profile types.
- **woy** (week of year) - A dimension off week or year, woy is used to support weekly seasonal profiles normalized to year. This profile can use the Store Contribution, Product Profile, or User Defined profile types.

Figure G–1 Additional Hierarchy Requirements for Season Profiles

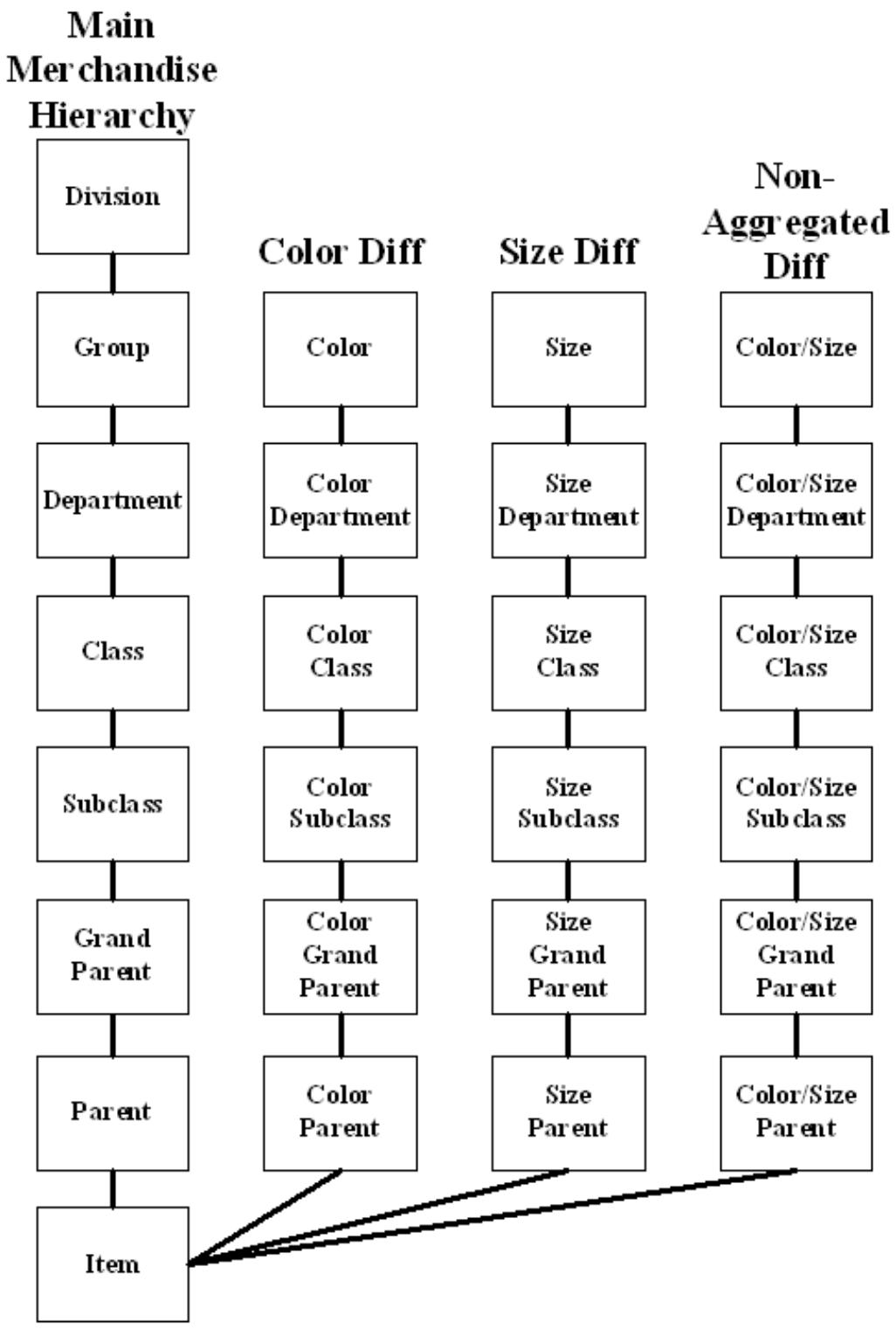


Configuring Curve Differentiator Dimensions (optional) and Merchandise Hierarchy Requirements to Support RMS

Configuring Differentiator dimensions (also referred to as Diff dimensions) within the merchandise hierarchy is optional. Differentiator dimensions allow the merchandise dimensions to be distinguished based on an alternative attribute property such as Color, Size, Flavor, or other attributes properties that are required to support your merchandising needs.

Differentiator dimensions are the combination of each Differentiator and a dimension that is created off of the lowest dimension in the merchandise hierarchy (item). This only goes up as high as Department. The mock install configuration released with Curve is configured with an example of a Differentiator branch along the merchandise hierarchy; however, up to 10 Differentiator branches may be configured. The following diagram provides an example of how a Differentiator branch may be configured.

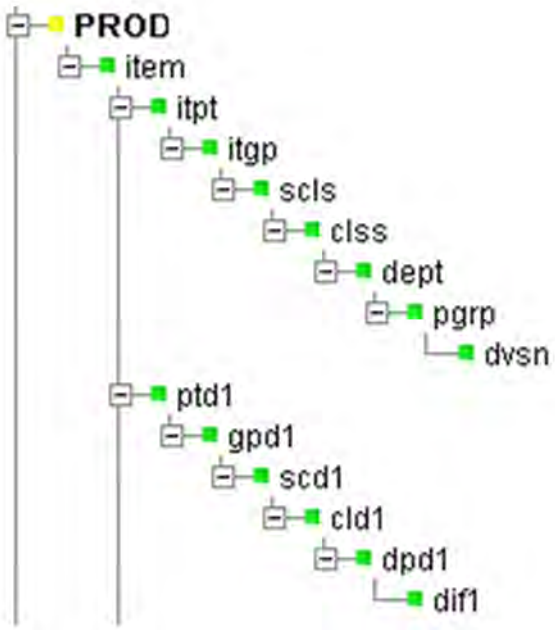
Figure G–2 Differentiator Branch



If Curve is to be integrated with RMS /Allocation, the Diff dimensions configured in the RPAS Configuration Tools must map to the same Diff dimensions that are or will be configured in the RMS/ Allocation hierarchies. Allocation also requires Non-Aggregated Differentiator dimensions. These dimensions allow for Diff Dimensions to be combined (as shown in the diagram above) and allow for Curve to generate profiles to support Allocation. Within Allocation, these Non-Aggregated Differentiators are represented by Diffs with Aggregation Indicators set to *No*.

Within the merchandise hierarchy, which is also required to support RMS / Allocation, is the itpt (Item Parent) dimension off the item (Item) dimension. And off of the itpt dimension, add itgp (Item Grandparent) dimension. The other aggregate dimensions above item, should be dimensions beginning off of itgp. Figure G–3 illustrates the GA configuration of the merchandise hierarchies that is configured using Item Parent and Item Grandparent in addition to a Differentiator branch:

Figure G–3 Merchandise Hierarchy Configuration



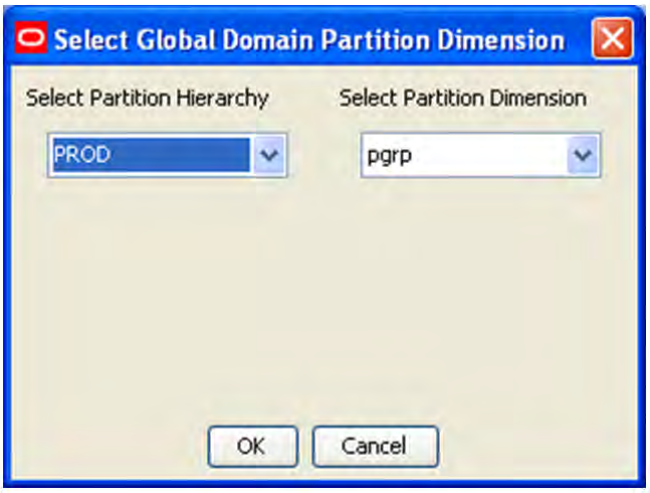
In addition to the above example, profiles 30 through 43 in the mock installations provided in the release packages are diff profile configurations that may be used to support the generation of spreading ratios for RMS / Allocation.

Creating a Curve Solution

To create a Curve solution:

1. Open an existing configuration in which hierarchies (for example, product, location, and calendar) have already been defined.
2. From the Configuration Tools toolbar, select the **Automation** menu. If installing a Global Domain environment, go on to Step 3. If installing a Simple Domain environment, go on to Step 4.
3. Select **Forecast Common**, and then select **Specify Partition Dimension**. The Select Global Domain Partition Dimension dialog box opens.

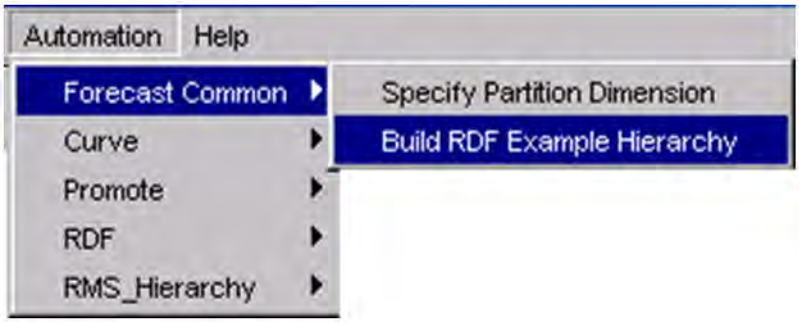
Figure G–4 Select Global Domain Partition Dimension



Note: To access this dialog, the configuration must already be defined as a Global Domain environment. This is performed by selecting **Workspace Properties** from the File menu and selecting the **GlobalDomain** option.

4. Perform the following:
5. From the **Select Partition Hierarchy** list, select the hierarchy in which the domains will be partitioned.
6. From the Select Partition Dimension list, select the appropriate partition dimension.
7. Select **OK**.
8. Optional: The Forecast Common plug-in may also be used to create an example hierarchy configuration. Select **Forecast Common - Build RDF Example Hierarchy**. The resulting hierarchy configuration is the same hierarchy that is used for the mock installation configurations provided in the release packages.

Figure G–5 Build RDF Example Hierarchy



9. From the Automation menu, select **Curve - Specify Parameters**.

Figure G–6 Specify Parameters

Configuring Profiles

The following sections provide information on configuring profiles:

Configuring a Final Profile

Configuring Source Level Attributes

Editing Profile Properties

Configuring a Final Profile

To create a final profile:

1. On the Curve Parameters utility, click the **F** icon.
2. A new final profile is added and is assigned the next consecutive number starting with 01.
3. Specify the properties for the final profile. See Editing Profile Properties for details.

Configuring Source Level Attributes

To create a source level:

1. On the Profile and Source Level window, highlight the final profile number in which a source will be created.
2. Click the **S** icon. A new source profile is added and is assigned the next consecutive number.
3. Specify the properties for the Source Level. See Editing Profile Properties for details.

Editing Profile Properties

The following sections describe how to edit profile properties.

Profile Name

The Profile Name is the system-assigned level number when a Final Profile or Source Level is created. This is a read-only field.

Profile Label

The Profile Label is the profile description that will be viewed by the user once the domain is created.

- Level Labels may not exceed forty characters.
- It is recommended (but not required) that Profile Labels include the Profile Name, which is the system-assigned profile number. There are two reasons for this:
- The Profile Name is referenced in the RDF configuration to specify Spreading Profiles. Curve requires that profiles 1 through 9 be referenced as 01, 02,…, 09 when being specified as a Spreading Profile in the RDF configuration.
- The Default Source Profile parameter in the Profile Administration workbook is a pick-list that is populated with the Profile Name of each source level configured for the final profile being viewed in the workbook. If the Default Source Profile set within this configuration, it not expected to change within the domain(s). This recommendation may not be necessary for consideration.
- RPAS automatically puts **( )** around Profile Labels. The configuration specialist should not include these in their level label configuration, or the installer will fail. An example of a Profile Label that would violate this requirement is (01 - chn->str-Final). It is acceptable as 01 - chn->str-Final.
- '**-**' should not be used before or after the Profile Label. An example of a Profile Label that would violate this requirement is -01-chn->str-Final-. It is acceptable as 01 - chn->str-Final.
- '**:**' should not be used at all in the Profile Label. An example of a Profile Label that would violate this requirement is 01: chn->str Final.

Profile Type

Assigned on the final profile, the Profile Type is a pick-list of profile types that are used to determine the profile algorithm and validation required by the profile level. Profile Types are represented with pre-defined configuration information.

Profile Types That Share The Same Profile Algorithm

The following Profile Types share the same profile algorithm. The rationale for providing different types that have the same behavior is strictly to remind the user of the intent of the profile while using the Profile Administration workbook:

Profile Types	Description
Store Contribution Profile	The Store Contribution Profile is used to determine the data relationship between stores to aggregate dimensions in the location hierarchy.
Hourly Profile	The Hourly Profile is used to determine the spreading ratios from aggregate dimensions to the hour, hour of day, or hour of week dimensions. To configure Hourly Profile, you must set the following: ■ The root dimension of the calendar hierarchy must be hour dimension. From configure tools, both RPAS Name and Tool Name for hour dimension have to be *HOUR*. Otherwise, the code will not work. ■ The position format for the configuration must be changed to *HOU%YEAR%MO%DAY%HR* and the calendar hierarchy file must match that format. ■ The training window start and phase start date for hourly profile will be mapped to the first hour of the day. ■ The training window end and phase end date for hourly profile will be mapped to the last hour of the day.
Daily Profile	The Daily Profile is used to determine the data relationship between a given day to the week in which it belongs.

Profile Types	Description
Product Profile	The Product Profile is used to determine the data relationship between any two dimensions along the product hierarchy.
Size Profile	The Size Profile is used to determine the data relationship between any dimension in the size hierarchy and any dimension in the product hierarchy. A size hierarchy must be defined to use this profile type.
User Defined Profile	The User Defined Profile may be used to support any profile configuration.

Profile Types with Unique Behavior

The following Profile Types have unique behavior:

Profile Types	Description
Diff Profile	Diff Profiles are used to determine spreading ratios from aggregate dimensions in the Product hierarchy to diff dimensions. Used to support the spreading of data in RMAS Allocation, Diff Profiles exhibit the same behavior as the previous profile types. However, unique to Diff Profiles is special validation of the relationship between the defined diff dimensions to dimensions along the main branch of the Product hierarchy. See the *Oracle Retail Demand Forecasting Implementation Guide* for more information on validation criteria.
Daily Seasonal Profile	The Daily Seasonal Profile is used to determine the data relationship between a given day of the week to aggregate dimensions in the calendar hierarchy. This profile type uses training window data to compute the profile. The resulting profile is then clipped to fit within the defined phase window.
Life Cycle Profile	The Life Cycle Profile uses data along a user-defined training window, and then stretches or shrinks data to fit a user-defined phase window.
Profile Intersection	The Profile Intersection is the intersection at which an intermediate profile is calculated. This intermediate profile is then replicated down or aggregated up to the Stored Intersection. If the Store Intersection is the same as the Profile Intersection, the values in intermediate profile are copied to the Stored Intersection. The Profile Intersection must be lower than the Aggregation Intersection. If the profile is being used as the Spreading Profile in RDF, this Profile Intersection should be the same as the Final Forecast Level. Once the Profile Intersection is entered at the Final Profile level, the Stored Intersection for both the Final and Source (if created) will populate with the same value. These may be overwritten if necessary. **Note:** If installing a Global Domain environment, all intersections configured to support a profile MUST include a dimension at or below the partition dimension.
Aggregation Intersection	The Aggregation Intersection is the intersection at which the profile will sum to one (or 100%). If the profile is being used as the Spreading Profile in RDF, this Aggregation Intersection should be the same as the Source Forecast Level. Once the Aggregating Intersection is entered, the Approval Intersection will populate with the same value for both the Final and Source (if created). This may be overwritten if necessary. **Note:** If installing a Global Domain environment, all intersections configured to support a profile MUST include a dimension at or below the partition dimension.

Profile Types	Description
Approval Intersection	Assigned only at the Final Profile, the Approval Intersection is the intersection at which the profile is approved. Approval Intersection should be above or equal to the Aggregation Intersection. If the profile is being used as the Spreading Profile in RDF, this Approval Intersection should be the same as the Aggregation Intersection. The Approval Intersection may be pre-populated with the value set for the Aggregation Intersection. This may be overwritten if necessary. **Note:** If installing a Global Domain environment, all intersections configured to support a profile MUST include a dimension at or below the partition dimension.
Stored Intersection	The Stored Intersection is the destination intersection of the profile. The intermediate profile produced at the Profile Intersection is either replicated down to or aggregated up to the Stored Intersection. If the Store Intersection is the same as the Profile Intersection, the values in intermediate profile are copied to the Stored Intersection. The Stored Intersection should not be greater than the Aggregation Intersection. If the profile is being used as the Spreading Profile in RDF, this Stored Intersection should be the same as the Profile Intersection. The Stored Intersection may be pre-populated with the value set for the Profile Intersection. This may be overwritten if necessary. **Note:** If installing a Global Domain environment, all intersections configured to support a profile MUST include a dimension at or below the partition dimension.
Default Source	Assigned only at the Final Profile, the Default Source is the primary Source Level that will be used in the calculation of the Final Profile. The desired Source Level must be created before it is an option in this pick-list.
Source Data	Assigned only at the Final Profile, the Source Data is the measure to be used as the input data (for example, POS) for the generation of profiles. The values in this pick-list are populated with all measures configured external to the RDF, Curve, and Promote solution extensions. If the profile is to be used to support the dynamic generation of spreading ratios (Spreading Profile) in the RDF batch forecast process, no value in Source Data should be specified.

Editing the Curve GA Configuration

The autogeneration process creates hierarchies, measures, rules, and workbook templates that are required to support the essential Curve functionality. This base configuration is referred to as the GA Configuration. Certain changes to the GA Configuration are allowed. Once edits to the GA Configuration are made and the autogeneration process occurs again, valid changes to the configuration will be preserved. There is nothing in the RPAS Configuration Tools to prevent invalid changes from being made.

> **Note:** When a custom taskflow exists in the configuration, the plug-in automation may cause the workbook template field to be blank. If this occurs, save the configuration and then re-open the configuration and the workbook template field is populated.

The following outlines acceptable changes and restrictions:

Item	Description
Curve Solution Extension Name	The name assigned to the resulting Curve solution after autogeneration occurs cannot be edited.
Major and Minor Classes	Additional Major components may be added to the Curve. Additional minor components can only be added under the new Major components. The Major and Minor components that are part of the GA Configuration may not be edited. This restriction also applies to Measure Names and Measure Labels. Adding minor components to GA major components is forbidden.
Rules	Additional Rule Sets, Rule Groups, and Rules may be added to the Curve GA Configuration. This includes support for adding new Rules to existing GA Configuration Rule Groups. It is recommended that new Rules that are added to the GA Configuration Rule Groups include cust (represents Custom) in the Rule Name. This allows for easy identification of Rules that are not part of the GA Configuration. Rule Sets, Rule Groups, and Rules that are part of the GA Configuration may not be renamed. Existing Rules that are part of the GA Configuration may not be modified in any way.
Workbook Templates	Additional Workbook Templates may be added to the Curve GA Configuration. As well, new Measures and Rules may be added to the GA Configuration Workbook Templates. This is done by adding new Major and Minor classes, and adding new Rules to existing Rule Groups in the GA Configuration.

Deleting a Profile Level

Deletion of a profile level will cause the system-assigned enumerated values in the Profile Name to renumber such that levels are in consecutive order starting with profile level 01. Deleting a profile level may impact any solution configuration that uses a specific profile level. For example, the following parameters within an RDF Solution configuration may be affected if profile levels are deleted or renumbered:

- Seasonal Profile
- Spreading Profile

If the domain using the configuration has been previously installed, there is potential to lose data associated to a level that has been deleted or renumbered.

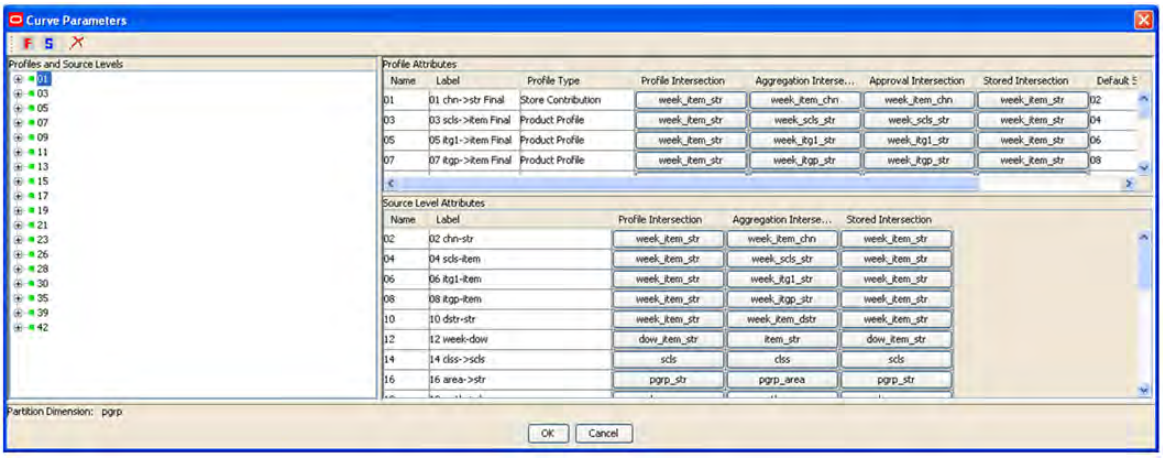
To delete a level:

1. On the Profiles and Source Level window, highlight the number of the profile you want to delete.
2. Click the **X** icon. The profile is deleted. If you delete a final profile, any source profiles that are associated with it will also be deleted.
3. Select **OK** to regenerate the solution with the changes to the cluster configuration.

Curve Plug-In Example

Figure G–7 shows the Curve Parameters plug-in.

Figure G–7 Curve Parameters Window



H

Grade Configuration Process

Grade is a clustering tool that provides insight into how various parts of a retailer's operations can be grouped together. Typically, a retailer may cluster stores over item sales to create logical groupings of stores based on sales of particular products. This provides increased visibility to where products are selling, and it allows the retailer to make more accurate decisions in merchandising. Beyond this traditional use of clusters, Grade is flexible enough to cluster any business measure based on products, locations, time, promotions, customers, or any hierarchy configured in the solution.

Grades/clusters used within the Grade Solution are defined using the Grade Parameters utility in the RPAS Configuration Tools.

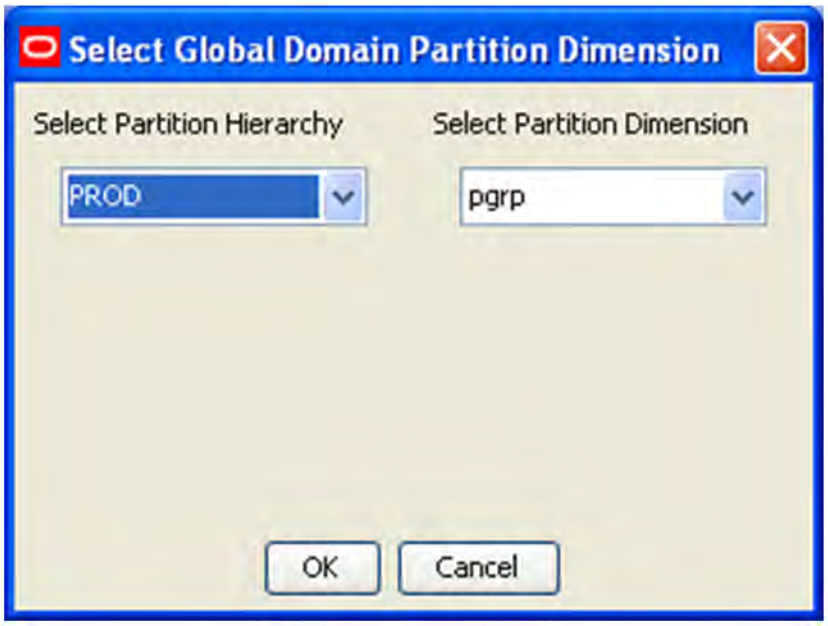
Note: For information on building the Grade domain, refer to the *Oracle Retail Predictive Application Server Installation Guide*.

Creating a Grade Solution Extension

To create a Grade solution extension:

1. Open an existing configuration in which hierarchies (for example, Product, Location, and Calendar) have already been defined.
2. From the Configuration Tools toolbar, select the **Automation** menu. If installing a Global Domain environment, go on to Step 3. If installing a Simple Domain environment, go on to Step 4.
3. Select **Common** and then the **Specify Partition Dimension**. The Select Global Domain Partition Dimension window opens. Select the hierarchy in which the domains will be partitioned, and then the Partition Dimension. Select **OK**, and continue with Step 4.

Figure H–1 Select Global Domain Partition Dimension Window



Note: To access this dialog, the configuration must already be flagged as a Global Domain environment. To do this, select **File - Configuration Properties**. The Configuration Properties window opens. Select the **GlobalDomain** option and click **OK**.

4. Optional: The Common plug-in may also be used to create an example hierarchy configuration. Select **Common**, and then **Build RMS Example Hierarchy.** The resulting hierarchy configuration is the same hierarchy that is used for the mock installation configurations provided in the release packages.
5. Open the Function Library Manager and add the **ClusterEngine** library.
6. From the Grade option, select **Specify Clusters**. The following sections outline the process for configuring profiles.

Create Clusters

On the Grade Parameters utility, enter the Maximum Number of Clusters that will be required to support any Clustering/Grading process.

1. Click the **Create Clusters** icon.
2. The Cluster and Label parameters will update to reflect the number of clusters specified.
3. Specify the properties for the clusters. See Editing Grade Parameters for details.

Editing Grade Parameters

Edit Grade parameters:

- **Cluster** — the system assigned Cluster Name. This value cannot be edited.

- **Label** — the description of the cluster/grade that will be viewed by the user once the domain is created.

Cluster Labels may not exceed 40 characters.

The following characters may not precede or follow the label that is entered in this field:

- '( )' Example: (cluster01)
- '-' Example: -cluster01-

A colon (:) may not be used in the Cluster Label field.

Example: cluster01:

Example H–1 Grade Parameters

```
cluster01:
```

Autogenerating Hierarchies, Measures, Rules and Workbook Templates

The following is the process to autogenerate the hierarchies, measures, rules, and workbook templates that are required by Promote to support the promotion configuration entered in the Promote plug-in:

- On the Grade Parameters utility, click **OK**.
- The system automatically generates:
- **Hierarchies** - The CLSH hierarchy will be created with a clst dimension. The GRCH hierarchy will be created with the grcd dimension.
- **Measures** - All measures necessary to support the base Grade solution will be created.
- **Rules** - Only the rules and rule groups necessary to support the installation of the Grade solution are visible in the configuration. A special code is used within the domain to create rules as needed for cluster generation and workbook templates.
- **Workbook Templates** - All pre-defined workbook templates to support the base Grade solution will be created; however, only the worksheets necessary to support the domain installation are visible. Additional processes within the application handles the creation of additional worksheets based on the user's selections in the workbook template wizards.

Note: You may continue to make changes to the Grade plug-in configuration and the autogeneration process may be repeated as often as needed prior to the installation.

Note: After autogeneration completes, the following rules will display as invalid; however, these should be ignored:

Rule: clad_l1

Rule: clad_l2

Rule: clev_l2

Rule: clev_l3

Rule: clev_l4

Rule: clev_l5

Rule: clev_l6

Rule: clev_l7

Rule: clev_l8

Rule: clev_l9

RuleGroup: clad_load

Rule Group: clad_refresh

Rule Group: clrev_load

Rule Group: clrev_refresh

Adding or Deleting Clusters in the Configuration

Follow this process if you need to add clusters to the configuration or remove clusters from the configuration:

1. On the Grade Parameters utility, enter the new Maximum Number of Clusters that will be required to support any Clustering/Grading process.
2. Click the **Create Clusters** icon.

   The Cluster and Label parameters will update to reflect the number of clusters specified.
3. Click **OK** to regenerate the solution with the changes to the cluster configuration.

Editing the Grade GA Configuration

The Grade autogeneration process creates all hierarchy dimensions and measures to support the essential Grade functionality; however, only the minimum rules, workbook templates, and worksheets required to support the domain installation are visible in the configuration. Additional processes within the application handles the creation of rules and workbook template worksheets.

Note: When a custom taskflow exists in the configuration, the plug-in automation may cause the workbook template field to be blank. If this occurs, save the configuration and then re-open the configuration and the workbook template field is populated.

Note: This limitation allows for fewer options than in RDF and Curve for edits to the GA Configuration.

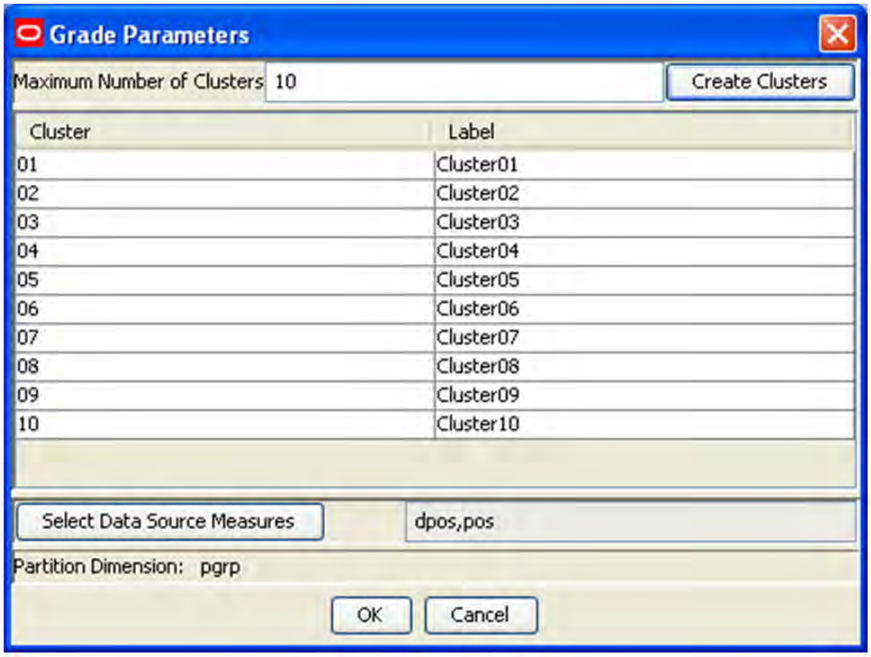
Acceptable changes and restrictions are outlined as follows:

Item	Description
Grade Solution Extension Name	The name assigned to the resulting Grade solution after autogeneration occurs cannot be edited.
Major and Minor Classes	Additional Major components may be added to the Grade. Additional minor components can only be added under the new Major components. The Major and Minor components that are part of the GA Configuration may not be edited. This restriction also applies to Measure Names and Measure Labels. Adding minor components to GA major components is forbidden.
Rules	Additional Rule Sets, Rule Groups, and Rules may be added to the Grade GA Configuration. This includes support for adding new Rules to existing GA Configuration Rule Groups. New Rules that are added to the GA Configuration Rule Groups should include cust (represents Custom) in the Rule Name, which makes it easy to identify Rules that are not part of the GA Configuration. Rule Sets, Rule Groups, and Rules that are part of the GA Configuration may not be renamed. Existing Rules that are part of the GA Configuration may not be modified in any way.
Workbook Templates	Additional Workbook Templates may be added to the Grade Configuration; however, new Measures and Rules can only be added to the Configuration Workbook Template worksheets that are visible in the configuration.

Grade Example

Figure H–2 provides an example of the Grade Parameters utility.

Figure H–2 Grade Parameters Window



CPEM Calculations

Cross Promotional Effects Module (CPEM) is a data mining solution that determines promotional Cannibalization, or Halo relationships, or both between items or groups of items. The special expressions detailed in this appendix calculate cross promotional effects.

This appendix details these topics:

- RdfPromoCrossEffectExpr
- RDF Configuration Options for CPEM

RdfPromoCrossEffectExpr

The promo cross effect special expression calculates the cross promotion Halo or Cannibalization effects.

RdfPromoCrossEffectExpr Syntax

The syntax for using the RdfPromoCrossEffectExpr special expression is shown in Example I–1. The input and output parameter tables (Table I–1 and Table I–2 explain the specific usage of the parameters names used in the special expression.

Example I–1 Syntax for RdfPromoCrossEffectExpr

```
CROSS_EFFECTS:RtnCannEff,STD_ERR:CannStdErr <-RdfPromoCrossEffectExpr(DES_
SLS:LogDesnCannSls,NORM_PRICE:LogCannNorPrc,PROMO_VAR:LogCannProVar,CROSS_
LIFT:LogCannLift,CROSS_MASK:CannMask,CROSS_TYPE:EffType,HIST_
START:HistStrIdx,HIST_END:HistEndIdx)
```

Table I–1 Input Parameters for the RdfPromoCrossEffectExpr Special Expression

Parameter Name	Description
DES_SLS	Aggregated deseasonalized sales at the Halo / Cannibalization level Cannibalization) Data Type: Real Required: Yes Default value: 0
NORM_PRICE	Normalized regular price at the Halo / Cannibalization level Data Type: Real Required: Yes Default value: 0

Table I–1 (Cont.) Input Parameters for the RdfPromoCrossEffectExpr Special

Parameter Name	Description
PROMO_VAR	Aggregated promotion variables at the Halo / Cannibalization level Data Type: Real Required: Yes Default value: 0
CROSS_LIFT	Aggregated cross product sales lift at the Halo / Cannibalization level Data Type: Real Required: Yes Default value: 0
CROSS_MASK	The cross promotion calculation mask to specify whether the cross effects will be calculated at Product/RHS Product/Location Data Type: Boolean Required: Yes Default value: 0
CROSS_TYPE	Cross Promotion Type : 1(Cannibalization),2(Halo) Data Type: Integer Required: Yes Default value: 0
HIST_START	History Start Date Index Data Type: Integer Required: Yes Default value: 0
HIST_END	History End Date Index Data Type: Integer Required: Yes Default value: 0
RUN_MASK	This indicator runs the SE for the given product and location. Data Type: Boolean Required: Yes Default value: True

Table I–2 Output Parameters for the RdfPromoCrossEffectExpr Special Expression

Parameter Name	Description
CROSS_EFFECTS	Cross effects (Halo or Cannibalization) Data Type: Real Required: Yes Default value: 0

Table I–2 (Cont.) Output Parameters for the RdfPromoCrossEffectExpr Special

Parameter Name	Description
STD_ERR	Standard Error (This Parameter is useless in this release and it is reserved for future enhancement) Data Type: Real Required: Yes Default value: 0

RDF Configuration Options for CPEM

CPEM gives unconstrained estimates of cross promotional elasticities, and verifies that the values are within some preset boundaries.

However, it is only at application time, in RDF, that the actual magnitude of the cross promotional lift can be calculated.

There may be several occasions when additional checks could be introduced. For instance, when an item is on promotion, it usually experiences a spike in sales. However, it may also receive Halo lift from a driver item as well. Appropriately, the cross effect should not be larger than the self lift. Preferably, the cross lift should not be larger than ten percent of the self lift.

RPAS and RDF offer the tools and hooks to implement logic preventing or alerting these types of situations.

Adjusted Forecast Example 1

An example of how the forecast can be adjusted is achieved by implementing the following pseudo code in the RPAS rules, and packaging it in the appcust.xml hook.

The stage where the rules are run can be pre-approved.

Pseudo code:

- if an item is on promotion
- if halo lift > self lift * threshold then
- halo lift = self lift * threshold

These rules prevent the halo lift from being larger than a certain percentage of the self lift for a promoted item.

Adjusted Forecast Example 2

Another example of how the forecast can be adjusted with the following pseudo code in the RPAS rules, and packaging it in the appcust.xml hook.

Pseudo code:

- if an item is not on promotion
- if abs(cannibalization effect) > baseline forecast * threshold then
- cannibalization effect = - baseline forecast * threshold

These rules prevent the cannibalization effects from dragging down the forecast too much.

Similar logic can be implemented to trigger an exception, rather than adjust the forecast, so that you can review it and determine whether or not to act.

In the pre-alert stage of the appcust.xml hook, the following logic alerts you that the cross lift is close to the self lift.

Alert logic:

- if an item is on promotion
- if halo lift > self lift * threshold then
- trigger high_halo_alert

CPEM Configuration Points

There are a few configuration points for CPEM, which define intersections at which the solution mines for Cannibalization and Halo effects, as well as supporting measures. This is handled thru labeled intersections, and the following table gives the label - level assignment for the two main ones.

Table I–3 Level - Labeled Intersection Assignment

Level	Labeled Intersection
Cannibalization	#CannLvl#
Halo	#HaloLvl#

There are additional labeled intersections in CPEM, but they all depend on the two main ones.

Note: Changing the main labeled intersections may require changing the dependent labeled intersections as well, to match the Cannibalization and Halo intersections

If the Halo level is class/region, a dependent labeled intersection may be class/region/week.

If the Halo level is changed to class/area, the dependent intersection must be changed accordingly, that is, to class/area/week.

Configuration Challenges

Finding the correct level for mining Halo and Cannibalization is not an easy task. Mining Halo is difficult because the entire merchandise space needs to be searched.

Cannibalization is difficult because the correct level is hard to determine. For instance, item is too low, but subclass may be too high.

Hierarchical Example

In the following example the item level may be too low, but Small Yogurts may be too high because of no visibility to brand or flavor.

Yogurt (Category) -> Small Yogurts (Sub Category) -> Dannon Light ‘n Fit 8 oz. (Item parent) -> Dannon Light 'n Fit NF 8 oz.:Raspberry (item)

In this example, the right level may be Item parent.

Note: The cannibalization level is recommended at the item group level. For additional information, refer to the *Oracle Retail Demand Forecasting Implementation Guide*.

J

RDF Script Names

With the introduction of RDF Cloud Service, many of the scripts used in RDF on premise were changed for consistency and ease of implementation. Also, rather than setting the processes or log level by script, the scripts use the default value from rdf_environment.ksh or the BSA_MAX_PARALLEL and BSA_LOG_LEVEL values exported before the scripts are run.

Script Name Changes

Table J–1 lists the script names that were changed from the 15.0 RDF release to the current release of RDF on premise.

Table J–1 RDF Script Name Changes

Script Name in 15.0 RDF	Script Name in 15.0.1 (and later) RDF On Premise
buildCPEM.sh	cpem_build_domain.ksh
buildRDF.sh	rdf_build_domain.ksh
cpem_batch.sh	cpem_batch.ksh
dtPreProcessingBatch.sh	rdf_preprocess_dt.ksh
exportCPEMtoRDF.ksh	cpem_e_rdf.ksh
exportRDFtoAPCRO.ksh	rdf_e_apcro.ksh
exportRDFtoAPCROWeekly.ksh	rdf_e_apcro_weekly.ksh
exportRDFtoCPEM.ksh	rdf_e_cpem.ksh
gen_floatlift.ksh	rdf_gen_float_lift.ksh
genHaloLift.sh	rdf_gen_halo_lift.ksh
loadCpemMeasures.sh	cpem_load_measures.ksh
loadRDFMeasures.sh	rdf_load_measures.ksh
newitemstr.ksh	rdf_new_item_store.ksh
newitemUpgradeExport.ksh	rdf_upgrade_new_item_store_export.ksh
newitemUpgradeLoad.ksh	rdf_upgrade_new_item_store_load.ksh
RDFPreProcessingBatch.sh	rdf_preprocess.ksh
repos.ksh	rdf_repos.ksh

K

Configuring the Forecast150 and CausalEstimate

This appendix describes how to configure the Forecast150 procedure that generates the forecast and estimates the promotion effects.

About the Forecast and the CausalEstimate Procedures

Previously RDF used only the Forecast procedure to generate both the forecast and to estimate the promotion effects. Now the Forecast procedure is divided into two procedures:

Forecast150 procedure — generates the forecast for all forecast methods

CausalEstimate procedure — calculates the promotion effects at both final or source level.

The Forecast150 Procedure Syntax section contains the specifications and syntax for configuring the Forecast150 procedure.

Forecast150 Procedure

The following sections detail the Forecast150 procedure for generating the forecast for all forecast methods.

Forecast150 Requirements

The following libraries must be registered in any domains that will use the Forecast150 solution extension:

- AppFunctions
- RdfFunctions

Forecast150 Parameter/Model Dependencies

The following models require that the stated measure is to be provided.

- Bayesian model — Plan measure required
- Profile model — Profile measure required
- Causal Model — The estimated promotion effects and causal baseline

Using the Forecast150 Procedure

The following notes are intended to serve as a guide for configuring the Forecast150 procedure within the RPAS Configuration Tools:

- Refer to the appropriate input parameters and output measures when using the Forecast150 procedure.
- The resultant measure (frcstout) should be at the same intersection as your history measure (pos). This will be the base intersection of the final level.
- The Forecast150 procedure is a multi-result procedure, meaning that it can return multiple results with one procedure call within a rule. In order to get multiple results, the resultant measures must be configured in the Measure Tool and the specific measure label must be used on the left-hand-side (LHS) of the procedure call. The resultant measure parameters must be comma-separated in the procedural call.
- The startdatemeas that specifies the forecast start date needs to be periodically updated (every week or so) by configuring rules.
- The forecast methods are specified using the mask measure. This is an int measure. Refer to Table K–3, " Forecast Method/Mode List" for the expected values of this measure for each forecast method.

Forecast150 Procedure Syntax

The syntax for using the Forecast150 procedure is shown in Example K–1.

Example K–1 Generic Example

```
FORECAST: FORMEAS [, INT: INTMEAS, CUMINT:CUMINTMEAS, PEAKS:PEAKSMEAS,
CHMETHOD:METHMEAS, CHLEVEL:LVLMEAS, CHTREND:TRENDMEAS, ALERTS:ALERTSMEAS]
<-FORECAST(MASK:MEASKMEAS, FORCSTSTART:STARTDATE, FORECASTLENGTH:FORECASTLENGTH,
HISTORY:HISTORYMEAS, PERIOD:PERIOD [
[,{PROMO_0:PROMO0,PROMOEFF_0:PROMOEFF0,PROMOEFFTYPE_0:PROMOEFFTYPE0, } … {,PROMO_
N:PROMON,PROMOEFF_N:PROMOEFFN,PROMOEFFTYPE_N:PROMOEFFTYPE_N, PROMOEFFTYPE_N:
PROMOEFFTYPEN, } POVLPEFF:OVERLAPEFFS,POVLPCOM:OVERLAPCOMBIN,
PROMOOVERLAPMTHD:OVERLAPMETHOD,PROMOPVALUE:OVERLAPPVALUE,PEFFMETHOD:PROMOEFFMETHOD
],
HISTSTART: HISTSTARTMEAS,MINWINTERS:MINWINTERSMEAS, MINHOLT: MINHOLTMEAS,
MINCROSTON:MINCROSTON, MAXALPHA:MAXALPHA, MAXWINTERSALPHA:MAXWINALPHA,
MAXPROFILEALPHA:MAXPROFILEALPHA, BAYESALPHA:BAYESALPHA, TRENDDAMP:TRENDDAMP,
{VALID_DD:VALID_DD, DDPROFILE:DDPROFILE }, PLAN:PLAN, PROFILE:PROFILE,
AGGPROFILE:AGGPROF, SPREADPROFILE:SPREADPROF, BAYESIAN_HORIZ,BAYESIAN_HORIZ,
CAPS:CAPSMEAS, CAPRATIOS:CAPRATIOSMEAS, USECAPPING:USECAPPING,
MINCAPHIST:MINCAPHIST, PLANINT:PLANINTMEAS, PLANCUMINT:PLANCUMINTMEAS,
CAPINTERVALS:CAPINTERVALS]
```

Configuration Parameters and Rules

The Table K–1, " Input Parameters for the Forecast150 Procedure" and Table K–2, " Output Parameters for the Forecast 150 Procedure" explain the specific usage of the parameters names used in the procedure. Table K–3, " Forecast Method/Mode List" provides the expected values of the mask measure for each forecast method.

Table K–1 Input Parameters for the Forecast150 Procedure

Parameter Name	Description
aggprofile	The celendar aggregation profile. Data Type: real Multiple Allowed: No Required: No
baycapratio	The Bayesian Cap Ratio Data Type: Real Multiple Allowed: No Required: No
bayesalpha	The maximum Bayesian alpha value. Data Type: Real Multiple Allowed: No Required: No
bayesian_horiz	The horizon to which the Bayesian adjust is applied. Data Type: Integer Multiple Allowed: No Required: No
capintervals	Indicator whether cap the interval Data Type: Real Multiple Allowed: No Required: No
capratios	Cap ratio for each time series. Data Type: Boolean Multiple Allowed: No Required: No
caps	Caps for each time series. Data Type: Real Multiple Allowed: No Required: No
causalextbaseline	Baseline for Causal Forecast Data Type: Real Multiple Allowed: No Required: No
ddprofile	De-seasonalized demand measure used only for profile-based forecasting. Data Type: Real Multiple Allowed: No Required: No
extraweekmth	Extra Week Process Method Data Type: Integer Multiple Allowed: No Required: No

Table K–1 (Cont.) Input Parameters for the Forecast150 Procedure

Parameter Name	Description
extraweeksrc	Extra Week Data Source Data Type: Boolean Multiple Allowed: No Required: No
fallbackmth	Fall Back Forecast Method Data Type: Integer Multiple Allowed: No Required: No
forecastlength	The length of the forecast. Data Type: Integer Multiple Allowed: No Required: Yes
frcststart	The forecast start date. Data Type: Datetime or String Multiple Allowed: No Required: No
history	The input measure the forecast is based on. Data Type: Real Multiple Allowed: No Required: Yes
histstart	The historical start date Index. Data Type: Integer Multiple Allowed: No Required: No
intcapratiolower	Lower Interval Cap Ratio Data Type: Real Multiple Allowed: No Required: Yes
intcapratioupper	Upper Interval Cap Ratio Data Type: Real Multiple Allowed: No Required: Yes
mask	Array that identifies what forecast method is used for each time series. Refer to Table K–4, " Numeric Values Assigned to the Forecast Model/Model List". Data Type: Integer Multiple Allowed: No Required: Yes

Table K–1 (Cont.) Input Parameters for the Forecast150 Procedure

Parameter Name	Description
maxalpha	The maximum alpha value. Data Type: Real Multiple Allowed: No Required: No
maxholtgamma	The maximum gamma value for Holt Method. Data Type: Real Multiple Allowed: No Required: No
maxprofilealha	The maximum Alpha value for Profile Method. Data Type: Real Multiple Allowed: No Required: No
maxwintersalpha	The maximum Alpha value for Winter Method. Data Type: Real Multiple Allowed: No Required: No
maxwintersgamma	The maximum Gamma value for Winter Method. Data Type: Real Multiple Allowed: No Required: No
mincaphist	The minimum number of weeks before capping can be used. Data Type: Real Multiple Allowed: No Required: No
mincroston	The minimum Croston history. Data Type: Integer Multiple Allowed: No Required: No
minholt	The minimum Holt history. Data Type: Integer Multiple Allowed: No Required: No
minwinters	The minimum Winters history. Data Type: Integer Multiple Allowed: No Required: No
movingaveragewindowlength	The moving average window of Average method. Data Type: Integer Multiple Allowed: No Required: No

Table K–1 (Cont.) Input Parameters for the Forecast150 Procedure

Parameter Name	Description
peffmethod	The promotion effect method Data Type: Integer Multiple Allowed: No Required: No
period	The forecasting period for calculating seasonal coefficients. Data Type: Integer Multiple Allowed: No Required: Yes
plan	The Plan measure. This measure's intersection may not be higher than the intersection of a corresponding forecast source level. Data Type: Real Multiple Allowed: No Required: No
plancumint	The cumulative Interval of the plan associated with the plan (PARAMETER forecast); Bayesian only. Data Type: Real Multiple Allowed: No Required: No
planint	The interval of the plan associated with the plan (PARAMETER forecast); Bayesian only. Data Type: Real Multiple Allowed: No Required: No
povlpcom	The overlapping promotion combination Data Type: String Multiple Allowed: No Required: No
povlpeff	The overlapping promotion combined effects Data Type: Real Multiple Allowed: No Required: No
profile	The Seasonal Profile measure. Data Type: Real Multiple Allowed: No Required: No
promo_	The Promo variable measure (one for each promotion). Data Type: Integer Multiple Allowed: Yes Required: No

Table K–1 (Cont.) Input Parameters for the Forecast150 Procedure

Parameter Name	Description
promoeff_	The calculated promotional effects (one per promotion). Data Type: Real Multiple Allowed: Yes Required: No
promoefftype_	The calculated promotional effects type (Linear or Exponential) Data Type: Integer Multiple Allowed: Yes Required: No
promooverlapmthd	The overlapping promotion method Data Type: Integer Multiple Allowed: No Required: No
promopvalue	The overlapping adjust factor Data Type: Real Multiple Allowed: No Required: No
seasonalindexsmooth	Seasonal Index Smooth factor in AWinter Data Type: Real Multiple Allowed: No Required: No
spreadprofile	The profile to spread to final forecast level. Data Type: Real Multiple Allowed: No Required: No
trenddamp	The trend damping parameter. Data Type: Real Multiple Allowed: No Required: No
usecapping	A Boolean measure that indicates whether capping is applied. Data Type: Boolean Multiple Allowed: No Required: No
valid_dd	The maximum non-zero history to use de-seasonalized demand value for seasonal profile based forecasting. Data Type: Integer Multiple Allowed: No Required: No
wintersmode	Winter Mode Data Type: Integer Multiple Allowed: No Required: No

Table K–2 Output Parameters for the Forecast 150 Procedure

Parameter Name	Description
forecast	Forecast output. Data Type: Real Multiple Allowed: No Required: Yes
ape	Forecast APE output. Data Type: Real Multiple Allowed: No Required: Yes
std	Forecast STD output. Data Type: Real Multiple Allowed: No Required: Yes
peaks	Peaks, which are used for calculating baseline of the forecast. Data Type: Real Multiple Allowed: No Required: No
chmethod	Selected method. Refer to Table K–4, " Numeric Values Assigned to the Forecast Model/Model List". Data Type: Integer Multiple Allowed: No Required: No
chlevel	ES level. Data Type: Integer Multiple Allowed: No Required: No
chtrend	ES trend. Data Type: Real Multiple Allowed: No Required: No
chalpha	ES alpha. Data Type: Real Multiple Allowed: No Required: No
chgamma	ES Gamma. Data Type: Real Multiple Allowed: No Required: No
alerts	A high-level forecast alert generated by the forecast engine. Data Type: Boolean Multiple Allowed: No Required: No

Table K–2 (Cont.) Output Parameters for the Forecast 150 Procedure

Parameter Name	Description
povlpflag	The overlapping indicator in Forecast Data Type: Boolean Multiple Allowed: No Required: No

Table K–3 Forecast Method/Mode List

Model	Numeric Value
AUTO ES	1
SIMPLE	2
HOLT	3
WINTERS	4
CASUAL	5
AVERAGE	6
NO FORECAST	7
COPY	8
CROSTON	9
M. WINTERS	10
A. WINTERS	11
SIMPLE CROSTON	12
BAYESIAN	13
LOADPLAN	14
PROFILE	15
MOVING AVERAGE	17
COMPONENTS	19

Forecast Method/Model List

Table K–4 provides the numeric value assigned to the forecast model/model list.

Table K–4 Numeric Values Assigned to the Forecast Model/Model List

Model	Numeric Value
AUTO ES	1
SIMPLE	2
HOLT	3
WINTERS	4
CASUAL	5
AVERAGE	6
NO FORECAST	7
COPY	8

Table K–4 (Cont.) Numeric Values Assigned to the Forecast Model/Model List

Model	Numeric Value
CROSTON	9
M. WINTERS	10
A. WINTERS	11
SIMPLE CROSTON	12
BAYESIAN	13
LOADPLAN	14
PROFILE	15
MOVING AVERAGE	17

CausalEstimate Procedure

The following sections detail the CausalEstimate procedure which estimates the promotion effects at both final level and source level. The final level is usually item/store/calendar, the source level is higher than the final level. While estimate the promotion effects at source level, RDF pools all the data points at the final level, then estimate the promotion effects.

CausalEstimation Procedure Syntax

The syntax for using the CausalEstimation procedure is shown in Example K–2.

Example K–2 CausalEstimation Procedure Syntax

```
POVLPCOM:POVLPCOM,POVLPEFF:POVLPEFF,PEFFMETHUSED:PEFFMETHUSED,PROMOEFF_0:PROMOEFF_
0,{PROMOEFF_N:PROMOEFF_
N},APE:APE,STD:STD<-CausalEstsimation(HISTORY:HISTORY,HISTSTART:HISTSTART,HISTEND:
HISTEND,MASK:MASK,POOLINGELIGIBLEMASK:POOLINGELIGIBLEMASK,MAXB:MAXB,MINB:MINB,KEEP
CLAMPEDMAXB:KEEPCLAMPEDMAXB,AGGPROF:AGGPROF,PROMOOVERLAPMTHD:PROMOOVERLAPMTHD,ITEM
GROUP:ITEMGROUP,NUMOFEXTRAPOINTS:NUMOFEXTRAPOINTS,CAUSALDATASRCTHR:CAUSALDATASRCTH
R,PROMO_0:PROMO_0,PROMOTYPE_0:PROMOTYPE_0,PROMOEFFTYPE_0:PROMOEFFTYPE_0,PROMONEG_
0:PROMONEG_0,PROMOOVER_0:PROMOOVER_0,{PROMO_N:PROMO_N,PROMOTYPE_N:PROMOTYPE_
N,PROMOEFFTYPE_N:PROMOEFFTYPE_N,PROMONEG_N:PROMONEG_N,PROMOOVER_N:PROMOOVER_N})
```

Configuration Parameters and Rules

The Table K–5, " Input Parameters for CausalEstimation" and Table K–6, " Output Parameters for CausalEstimation" explain the specific usage of the parameters names used in the CausalEstimation procedure.

Table K–5 Input Parameters for CausalEstimation

Parameter Name	Description
history	The input measure the forecast is based on. Data Type: Real Multiple Allowed: No Required: Yes

Table K–5 (Cont.) Input Parameters for CausalEstimation

Parameter Name	Description
histstart	The historical start date. Data Type: DateTime Multiple Allowed: No Required: Yes
histend	The historical enddate. Data Type: DateTime Multiple Allowed: No Required: Yes
mask	Run Mask Indicator Data Type: Boolean Multiple Allowed: No Required: Yes
poolingeligiblemask	If item/store eligible mask for pooling Data Type: Boolean Multiple Allowed: No Required: Yes
maxb	The maximum ratio between beta and baseline. Data Type: Real Multiple Allowed: No Required: No
minb	The minimum ratio between beta and baseline. Data Type: Real Multiple Allowed: No Required: No
keepclampedmaxb	Determines whether variables exceeding minb are clamped or values are dropped and regression is re-run. Data Type: Real Multiple Allowed: No Required: No
aggprof	Aggregation profile from the low calendar to higher lelvel. Data Type: Real Multiple Allowed: No Required: No
promooverlapmthd	The Promotion Overlapping method to deal with overlapping promotion. Data Type: Integer Multiple Allowed: No Required: No

Table K–5 (Cont.) Input Parameters for CausalEstimation

Parameter Name	Description
itemgroup	The item/store to group mapping Data Type: boolean Multiple Allowed: No Required: No
numofextrapoints	The number of extra data points before/after promotion periods Data Type: Integer Multiple Allowed: No Required: No
causaldatasrcthr	The data dales value threshold for promotion variabble. Data Type: Real Multiple Allowed: No Required: No
promo_	The Promo variable measure (one for each promotion). Data Type: Integer Multiple Allowed: Yes Required: No
promotype_	The promo type measure (one for each promotion). Data Type: Integer Multiple Allowed: Yes Required: No
promoefftype_	The calculated promotional effects type (Linear or Exponential) Data Type: Integer Multiple Allowed: Yes Required: No
promoneg_	The indicator whether the negative promo effects allow. Data Type: Boolean Multiple Allowed: Yes Required: No
promoover_	The promo effect override measure (one for each promotion). Data Type: Boolean Multiple Allowed: Yes Required: No

Table K–6 Output Parameters for CausalEstimation

Parameter Name	Description
povlpcom	The overlapping promotion combination Data Type: String Multiple Allowed: No Required: No

Table K–6 (Cont.) Output Parameters for CausalEstimation

Parameter Name	Description
povlpeff	The overlapping promotion combined effects Data Type: Real Multiple Allowed: No Required: No
peffmethused	The promotion effect method Data Type: Integer Multiple Allowed: No Required: No
promoeff_	The calculated promotional effects (one per promotion). Data Type: Real Multiple Allowed: Yes Required: No
ape	Forecast APE output. Data Type: Real Multiple Allowed: No Required: Yes
std	Forecast STD output. Data Type: Real Multiple Allowed: No Required: Yes

L

Configuring the Similarity Score Calculation

This appendix describes the procedure to configure the Similarity Score Calculation.

The approach to estimate item to item similarity is to estimate the fraction of attributes that match between a new item and a potential like item. When calculating similarity score, it is important that item to item similarity calculations take into account attribute weights. The similarity calculations to follow consume user supplied attribute ranks to derive attribute weights. You are allowed to override attribute weights if such weights are available readily from an upstream system

Similarity Score Calculation Requirements

The following libraries must be registered in any domains that will use the clone solution extension:

- .RdfFunctions

Similarity Score Calculation Procedure

The following sections detail the Similarity Score Calculation procedure.

Using the Similarity Score Calculation Procedure

The following notes provide information about Similarity Score Calculation functionality.

- Refer to the appropriate input parameters and output measures when using the Similarity Score Calculation procedure:
- A mask measure is used to define when Similarity Score Calculation is performed. When the mask measure is True, a Similarity Score Calculation is performed; setting the mask to False stops the Similarity Score Calculation process. A business rule may be defined (using RPAS rules) to set the mask measure to False when it is desired to stop calculating similarity score for the new item/like item.
- There are two destination arrays of Similarity Score Calculation. One is summarized score between a new item and a like item. Another is detail score for each attribute between a new item and a like item.
- Attribute weight need to provide for each attribute as input parameter of Similarity Score Calculation expression.
- Attribute Type need to provide for each attribute. The valid attribute type are: string or numeric. For numeric attribute, Attribute Tau need to provide as input parameter. It served as threshold to see if a new item is same as a like item on a numeric attribute.

Similarity Score Calculation Syntax

The syntax for using the Similarity Score Calculation procedure is shown in Example L–1 The input and output parameter tables explain the specific usage of the parameters names use in the procedure.

Example L–1 Similarity Score Calculation

```
SIM_SCORE:nitsimsco, ATTR_SCORE:nitattsco <- newitemsim(SIM_MASK:nitsimmsk, PROD_
ATTR:nitattval, ATTR_TYPE:nitattvaltyp, ATTR_WEIGHT:nitclattwgt, ATTR_
TAU:nitclatttau)
```

Configuration Parameters and Rules

Table L–1 provides the input parameters for the Similarity Score Calculation procedure and special expressions.

Table L–1 Input Parameters for the Similarity Score Calculation Procedure

Parameter Name	Description
SIM_MASK	Mask of pair of new item and like item (pre calculated by rule expression)
PROD_ATTR	Product attribute values for both new items and like items
ATTR_TYPE	Product attribute values type (string or numeric)
ATTR_WEIGHT	Attribute weight for new items
ATTR_TAU	Tau of numeric attribute for new items
SIM_SCORE	Output Array contains similarity score between new item and like item
ATTR_SCORE	Output Array contains similarity score between new item and like item for each attribute

M

Appendix: Configuring Seasonal Profiles

This appendix describes several options to create a seasonal profile for Curve.

Create a Seasonal Profile Option 1

This option is the most desirable because the numbers are not replicated to item/store.

> **Note:** A source level is still necessary, even if it is at the same intersection as the final level.

- Create a final level at an intersection, for example: class/region/woy.
- Create a source level at an intersection, for example: class/region/woy.
- Specify training start and end dates.
- There is no need for phase start and end dates
- Run the Curve batch.

Create a Seasonal Profile Option 2

This option is less desirable because the class/region numbers are replicated to item/store making the profile an (unnecessarily) dense measure.

- Create a final level at an intersection, for example: item/store/woy.
- Create a source level at an intersection, for example: class/region/woy.
- Specify training start and end dates.
- There is no need for phase start and end dates
- Run the Curve batch.

Create a Seasonal Profile Option 3

This option is the least desirable because the class/region numbers are replicated to item/store making the profile an (unnecessarily) dense measure and it requires the most setup.

- Create a final level at an intersection, for example: item/store/week.
- Create a source level at an intersection, for example: class/region/woy.
- Specify training start and end dates.

- Specify phase start and end dates.
- Run the Curve batch.