

Oracle® Order Management

Open Interfaces, API, & Electronic Messaging Guide

Release 11*i*

Part No. B14446-01

September 2004

Oracle Order Management Open Interfaces, API, & Electronic Messaging Guide, Release 11i

Part No. B14446-01

Copyright © 1996, 2004 Oracle Corporation. All rights reserved.

Contributing Authors: Rajeev Bellamkonda, Charlene Chandonia, Kathleen Gahan, Aswin Kurella, Bernard Ladent, Nithya Lakshmanan, Jerome Mcfarland, Prakash Ojha, Gayatri Pendse, Sameer Phatarpekar, Alok Singh, Sumeet Rijhsinghani, Krishna Venkatesan, Anil Verma, Jessica Zhang, Manisha Nair, William Nelson, Alok Singh, Kannan Tarakad

Contributors: John Brazier, Elizabeth Looney, Jennifer Mosinski, Tom Myers, David Reitan, John Salvini

The Programs (which include both the software and documentation) contain proprietary information of Oracle Corporation; they are provided under a license agreement containing restrictions on use and disclosure and are also protected by copyright, patent and other intellectual and industrial property laws. Reverse engineering, disassembly or decompilation of the Programs, except to the extent required to obtain interoperability with other independently created software or as specified by law, is prohibited.

The information contained in this document is subject to change without notice. If you find any problems in the documentation, please report them to us in writing. Oracle Corporation does not warrant that this document is error-free. Except as may be expressly permitted in your license agreement for these Programs, no part of these Programs may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of Oracle Corporation.

If the Programs are delivered to the U.S. Government or anyone licensing or using the programs on behalf of the U.S. Government, the following notice is applicable:

Restricted Rights Notice Programs delivered subject to the DOD FAR Supplement are "commercial computer software" and use, duplication, and disclosure of the Programs, including documentation, shall be subject to the licensing restrictions set forth in the applicable Oracle license agreement. Otherwise, Programs delivered subject to the Federal Acquisition Regulations are "restricted computer software" and use, duplication, and disclosure of the Programs shall be subject to the restrictions in FAR 52.227-19, Commercial Computer Software - Restricted Rights (June, 1987). Oracle Corporation, 500 Oracle Parkway, Redwood City, CA 94065.

The Programs are not intended for use in any nuclear, aviation, mass transit, medical, or other inherently dangerous applications. It shall be the licensee's responsibility to take all appropriate fail-safe, backup, redundancy, and other measures to ensure the safe use of such applications if the Programs are used for such purposes, and Oracle Corporation disclaims liability for any damages caused by such use of the Programs.

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

Contents

Send Us Your Comments	xv
Preface.....	xvii
Audience for This Guide.....	xvii
How To Use This Guide	xvii
Conventions.....	xviii
Documentation Accessibility	xviii
Other Information Sources	xix
Online Documentation.....	xix
Related User's Guides.....	xx
Guides Related to All Products	xx
User Guides Related to This Product	xx
Installation and System Administration	xxiv
Other Implementation Documentation.....	xxv
Training and Support.....	xxvi
Do Not Use Database Tools to Modify Oracle Applications Data.....	xxvii
About Oracle.....	xxviii
Your Feedback	xxviii
 1 Integrating Your Systems	
Overview of Oracle Order Management APIs and Open Interfaces.....	1-2
Basic Business Needs	1-2
Oracle Order Management Interfaces	1-2
Inbound Open Interface Model	1-8

Components of an Open Interface	1-10
---------------------------------------	------

2 Oracle Order Management Open Interfaces and APIs

Integrating Oracle Order Management Using Order Import	2-4
Process Order Application Open Interface	2-69
Process Order API Features	2-70
Functional Overview.....	2-80
Processing the Sales Order business object.....	2-80
Setting Up the Process Order Procedure.....	2-84
Setting Up the Get_Order Procedure.....	2-93
Setting Up the Lock_Order Procedure	2-97
PL/SQL Record Structures.....	2-103
Process Order Usage	2-148
Integrating Oracle Order Management with Oracle Receivables and Invoicing	2-159
Basic Needs	2-159
Major Features.....	2-159
Invoicing of ATO Configurations.....	2-165
Understanding the Receivables Interface Tables	2-166

3 Release Management Open Interface

Understanding the Interface Tables	3-2
RLM_INTERFACE_HEADERS_ALL.....	3-2
RLM_INTERFACE_LINES_ALL.....	3-22

4 Oracle Shipping Execution Public APIs

Overview of API Information	4-2
Shipment Processing Using APIs	4-3
Shipping Transaction Form/Public API Correlation	4-3
Sample Flow Scenarios.....	4-9
API Package and Procedures Example	4-15
Package.....	4-15
Procedures	4-15
Script.....	4-24
Actions, APIs, and Parameters	4-26

Application Parameter Initialization	4-39
Trip Public Application Program Interface	4-40
Create_Update_Trip API Features	4-40
Functional Overview	4-40
Trip_Action API Features	4-46
Functional Overview	4-46
Stop Public Application Program Interface.....	4-49
Create_Update_Stop API Features	4-49
Functional Overview	4-49
Stop_Action API Features	4-56
Functional Overview	4-56
Deliveries Public Application Program Interface	4-59
Create_Update_Delivery API Features	4-59
Functional Overview	4-59
Delivery_Action API Features	4-70
Functional Overview	4-70
Generate_Documents API Features	4-76
Functional Overview	4-76
Setting Up the Generate_Documents API.....	4-77
Exceptions Application Program Interface	4-79
Exception_Action API Features	4-79
Functional Overview	4-79
Setting Up the Exception_Actions API.....	4-79
Get_Exceptions API Features	4-87
Functional Overview	4-87
Setting Up the Get_Exceptions API	4-87
Delivery Details Public Application Program Interface	4-90
Detail_To_Delivery API Features	4-90
Functional Overview	4-90
Split_Line API Features	4-94
Functional Overview	4-94
Update_Shipping_Attributes API Features	4-97
Functional Overview	4-97
Autocreate_Deliveries API Features	4-110
Autocreate_Del_Trip API Features	4-113

Container Public Application Program Interface	4-116
Create_Containers API Features	4-116
Functional Overview	4-116
Update Container API Features.....	4-120
Functional Overview	4-120
Auto_Pack API Features	4-128
Functional Overview	4-128
Container_Actions API Features	4-131
Functional Overview	4-131
Freight Costs Public Application Program Interface	4-134
Create_Update_Freight_Costs API Features	4-134
Functional Overview	4-134
Validate_Freight_Cost_Type API Features.....	4-141
Delete_Freight_Costs API Features.....	4-142
Functional Overview	4-142
Pick Release Application Program Interface	4-148
Create_Batch API Features	4-148
Functional Overview	4-149
Release_Batch API Features	4-160
Functional Overview	4-160
Migration from Open Interfaces to Public APIs	4-163

5 Oracle Advanced Pricing Open Interfaces

Agreements Public Application Program Interface	5-3
Functional Overview	5-3
Setting Up and Parameter Descriptions	5-3
Validation of Agreements Public API.....	5-22
Example of Agreements Public API.....	5-22
Attribute Mapping Application Program Interface	5-49
Functional Overview	5-49
Setting Up and Parameter Descriptions	5-50
Business Object for Modifier Setup Application Program Interface	5-60
Functional Overview	5-60
Setting Up and Parameter Descriptions	5-61
Validation of Business Object for Modifier Setup API.....	5-94

Example of Modifier Setup Application Program Interface	5-94
Business Object for Pricing Formulas Application Program Interface	5-133
Functional Overview.....	5-133
Setting Up and Parameter Descriptions	5-134
Validation of Business Object for Pricing Formulas API.....	5-144
Example of Pricing Formulas API.....	5-144
Business Object for Pricing Limits Application Program Interface	5-157
Functional Overview.....	5-157
Setting Up and Parameter Descriptions	5-158
Validation of Limits Public API.....	5-168
Example of Limits Public API.....	5-169
Get Currency Application Program Interface	5-177
Get Currency API Features	5-177
Functional Overview.....	5-177
Setting Up and Parameter Descriptions	5-177
Validation of Get_Currency API	5-178
Get Custom Price Application Program Interface	5-179
Get Custom Price API Features	5-179
Functional Overview.....	5-179
Setting Up and Parameter Descriptions	5-179
Validation of Get Custom Price API.....	5-181
Get Price List Application Program Interface	5-185
Get Price List API Features	5-185
Functional Overview.....	5-185
Setting Up and Parameter Descriptions	5-185
Validation of Get_Price_List API	5-186
Multi-Currency Conversion Setup Application Program Interface	5-187
Functional Overview.....	5-187
Setting Up and Parameter Descriptions	5-188
Validation of Multi-Currency Conversion API.....	5-194
Example of Multi-Currency Conversion API.....	5-194
Price List Setup Application Program Interface	5-199
Functional Overview.....	5-199
Setting Up and Parameter Descriptions	5-200
Validation of Price List Setup API	5-213

Example of Price List Setup API.....	5-213
Price List Setup Group Application Program Interface	5-240
Functional Overview	5-240
Setting Up and Parameter Descriptions	5-241
Validation of Price List Group API	5-254
Price Request Application Program Interface	5-255
Price Request API Features	5-255
Functional Overview	5-256
Setting Up and Parameter Descriptions	5-257
Validation of Price Request API	5-286
Example of Price Request Application Program Interface	5-286
Pricing Object Security - Check Function API.....	5-290
Functional Overview	5-290
Setting Up and Parameter Descriptions	5-291
Validation of Pricing Object Security API.....	5-292
Standard Validation	5-292
QP_ATTRIBUTES_PUB Application Program Interface	5-293
Functional Overview	5-293
QP_ATTR_MAPPING_PUB Application Program Interface	5-303
Functional Overview	5-303
Setting Up and Parameter Descriptions	5-303
Validation of Attribute Mapping API.....	5-307
Qualifiers Application Program Interface	5-308
Functional Overview	5-308
Setting Up and Parameter Descriptions	5-308
Validation of Qualifiers API.....	5-324
Example of Qualifiers API.....	5-324
Reverse Limits Application Program Interface	5-328
Reverse Limits API Features	5-328
Functional Overview	5-328
Setting Up and Parameter Descriptions	5-328
Validation of Reverse Limits API	5-330
Round Price Application Program Interface	5-333
Round Price API Features	5-333
Functional Overview	5-333

Setting Up and Parameter Descriptions	5-334
Validation of Round Price API	5-334
Validate_Price_list_Curr_code Application Program Interface	5-336
Validate_Price_list_Curr_code API Features	5-336
Functional Overview	5-336
Setting Up and Parameter Descriptions	5-336
Validation of Validate_Price_list_Curr_code API	5-336

6 Oracle Order Management EDI Transactions

Oracle Order Management	6-2
Inbound Purchase Order (POI/850/ORDERS)	6-4
Trading Partner Link to Oracle e-Commerce Gateway	6-4
Oracle e-Commerce Gateway Required Fields	6-4
TP_TRANSLATOR_CODE, TP_LOCATION_CODE (EDI Location Code)	6-6
Review Oracle e-Commerce Gateway Exceptions	6-6
Resolve Oracle e-Commerce Gateway Exceptions	6-6
Relevant Oracle Order Management Profiles and Setup Steps	6-6
Oracle Order Management Defaulting Functionality	6-7
Order Import Open Interface Data Levels	6-7
Line and Shipment Records	6-11
Original System Reference Data in all Order Import Tables	6-13
ORIG_SYS_DOCUMENT_REF	6-15
ORIG_SYS_LINE_REF	6-16
ORIG_SYS_SHIPMENT_REF	6-17
Alternative Original System Shipment Reference	6-18
Comment Text	6-22
Flags	6-22
Validate Mode Parameter in Concurrent Manager	6-23
Order Import Open Interface Data	6-24
OE_HEADERS_INTERFACE Table	6-32
Update Columns:	6-35
OE_LINES_INTERFACE Table	6-36
Update Columns:	6-38
Original System Reference Data:	6-39
Update Columns:	6-39

OE_PRICE_ADJS_INTERFACE Table.....	6-40
Original System Reference Data:	6-40
Update Columns:	6-40
Update Columns:	6-41
OE_RESERVTONS_INTERFACE Table	6-42
Update Columns:	6-43
OE_ACTIONS_INTERFACE Table.....	6-43
Update Columns:	6-44
Review Order Management Open Interface Exceptions.....	6-45
Resolve Order Management Open Interface Exceptions	6-45
Order Import Item Cross-Referencing.....	6-46
Inventory Tables	6-46
Cross Reference Types for the Customer Item Cross Reference Table	6-47
Cross Reference Types for the Inventory Item Cross Reference Table	6-48
Data in the e-Commerce Gateway Transaction	6-49
Table Searches	6-51
Miscellaneous Item Data.....	6-57
Inbound Purchase Order Change (POCI/860/ORDCHG)	6-58
Outbound Purchase Order Acknowledgment (POAO/855/ORDRSP) (POCAO/865/ORDRSP) 6-59	
Order Management Transaction Summaries	6-60
Inbound Purchase Order (POI/850/ORDERS).....	6-61
Inbound Purchase Order (POI/850/ORDERS).....	6-62
Inbound Purchase Order (POI/850/ORDERS).....	6-62
Inbound Purchase Order (POI/850/ORDERS).....	6-65
Inbound Purchase Order Change (POCI/860/ORDCHG)	6-73
Inbound Purchase Order Change (POCI/860/ORDCHG)	6-74
Inbound Purchase Order Change (POCI/860/ORDCHG)	6-74
Inbound Purchase Order Change (POCI/860/ORDCHG)	6-77
Outbound Purchase Order Acknowledgment (POAO/855/ORDRSP)(POCAO/865/ORDRSP) 6-85	
Outbound Purchase Order Acknowledgement (POAO/855/ORDRSP) (POCAO/865/ORDRSP) 6-86	
Outbound Purchase Order Acknowledgement (POAO/855/ORDRSP) (POCAO/865/ORDRSP) 6-88	
User Guide Update for Order Management Transactions	6-92

Purchase Order Inbound (POI) Program	6-92
Prerequisites	6-92
Purchase Order Inbound Program.....	6-93
Purchase Order Change Inbound (POCI) Program.....	6-95
Prerequisite Setup in Oracle Order Management.....	6-95
Prerequisites	6-95
Purchase Order Change Inbound	6-95
Purchase Order Acknowledgment Outbound (POAO) Program	6-97
Prerequisites	6-97
Purchase Order Acknowledgement Outbound Program.....	6-97
Purchase Order Change Acknowledgment Outbound (POCAO) Program.....	6-99
Prerequisites	6-99
ReadMe for November, 2000 Patch.....	6-102
Purchase Order Inbound (POI) Patch Readme	6-102
Additional Information for the March 2001 Patch	6-104
PATCH:.....	6-104
IMPORTANT:	6-104
RECORDS 4000: (Shipment level data)	6-106

7 Electronic Messaging Technical Information

What is RosettaNet?	7-2
Process_PO	7-4
Major Features.....	7-4
CONFIRM BOD	7-6
Message Details.....	7-7
Extension Tags	7-28
Acknowledge_PO	7-34
Overview.....	7-34
Major Features.....	7-34
Message Map.....	7-35
Seeded Workflow	7-49
Show_SalesOrder	7-50
Show_salesorder Overview	7-50
Major Features.....	7-50
Business Scenarios and Process Flow	7-51

Message Map	7-52
Message Map	7-59
Message Set Up	7-60
Seeded Workflow	7-67
Change_salesorder	7-68
Overview	7-68
Major Features	7-68
Message Map	7-71
Load/Delete Maps, Load/Delete DTD's	7-71
Message Details	7-79
Workflow Event Setup	7-80
Workflow (oexwfoa.wft)	7-81
Lookups	7-81
Inbound Change PO Request	7-83
Overview	7-83
Major Features	7-83
Message Map	7-87
Seeded Workflow for Inbound XML Messages	7-105
Message Details	7-108
Extension Tags	7-111
Cancel Purchase Order	7-117
Overview	7-117
Major Features	7-117
Business and Process Flow	7-120
CONFIRM BOD	7-120
Message Map	7-121
Seeded Workflow for Inbound XML Messages	7-128
Message Details	7-132
Extension Tags	7-135
Open Interface Tracking	7-140
Overview	7-140
Major Features	7-140
Business Scenarios and Process Flow	7-141
Messages	7-142
Event/Event Subscription	7-144

Seeded workflow OEEM/Open Interface Tracking.....	7-144
Event Parameters & Population	7-144

8 Electronic Messaging Implementation Considerations

Process_PO	8-2
Setup	8-2
Customer Setup	8-2
Installation	8-8
Message Map.....	8-9
Seeded Workflow for Inbound XML Messages	8-9
Sample Business Flow for 3A4 Process PO/3A4 Acknowledge PO.....	8-14
Acknowledge_PO	8-23
Setup	8-24
Customer Setup	8-24
Oracle XML Gateway Details	8-27
Show_SalesOrder	8-29
Setup	8-29
Customer Setup	8-29
Additional User Setup to Enable.....	8-33
User Procedures	8-34
Change_SalesOrder	8-43
Setup	8-43
Customer Setup	8-43
Data Archive and Purge Procedures	8-47
Change_PO	8-48
Setup	8-48
Customer Setup	8-48
Implementation Considerations.....	8-51
Message Map.....	8-51
Sample Business Flow for 3A4 Process PO/3A4 Acknowledge PO.....	8-53
Cancel_PO	8-63
Setup	8-63
Customer Setup	8-63
Implementation Considerations.....	8-68
Open Interface Tracking	8-68

Setup	8-68
Customer Setup.....	8-68
Integration Event – oracle.apps.ont.oi.xml_int.status.....	8-69
User Procedures	8-76

9 Order Management Suite Products

Release Management	9-2
Planning Schedule	9-2
Shipping Schedule	9-2
Shipping	9-3
Purpose Codes.....	9-3
Planning/Shipping Inbound.....	9-5
Transportation Execution.....	9-7
Tracking Message	9-8
DSNO	9-12

A Electronic Messaging Messages

Process_PO and Acknowledge_PO	A-2
Show_SalesOrder.....	A-3
Cancel_PO	A-4
Change SO.....	A-6
Inbound Change PO Request.....	A-7

Index

Send Us Your Comments

Oracle Order Management Open Interfaces, API, & Electronic Messaging Guide, Release 11i Part No. B14446-01

Oracle Corporation welcomes your comments and suggestions on the quality and usefulness of this document. Your input is an important part of the information used for revision.

- Did you find any errors?
- Is the information clearly presented?
- Do you need more information? If so, where?
- Are the examples correct? Do you need more examples?
- What features did you like most?

If you find any errors or have any other suggestions for improvement, please indicate the document title and part number, and the chapter, section, and page number (if available). You can send comments to us in the following ways:

- Electronic mail: mfgdoccomments_us@us.oracle.com
- FAX: 650-506-7294 Attn: Oracle Order Management Open Interfaces, API, & Electronic Messaging

If you would like a reply, please give your name, address, telephone number, and (optionally) electronic mail address.

If you have problems with the software, please contact your local Oracle Support Services.

Audience for This Guide

Welcome to Release 11i of the *Oracle Order Management Open Interfaces, API, & Electronic Messaging Guide*.

This guide assumes you have a working knowledge of the following:

- The principles and customary practices of your business area.
- ***Oracle Order Management***

If you have never used ***Oracle Order Management***, Oracle suggests you attend one or more of the ***Oracle Order Management*** training classes available through Oracle University.

- The Oracle Applications graphical user interface.

To learn more about the Oracle Applications graphical user interface, read the *Oracle Applications User's Guide*.

See Other Information Sources for more information about Oracle Applications product information.

How To Use This Guide

This guide contains the information you need to understand and use ***Oracle Order Management***.

- Chapter 1 gives you an overview of Order Management integration tools and explains how to use these tools to integrate Oracle Manufacturing products with one another and with non-Oracle systems.
- Chapter 2 contains information about Oracle Order Management.
- Chapter 3 contains information about Oracle Pricing.
- Chapter 4 contains information about Oracle Release Management.
- Chapter 5 contains information about Oracle Shipping.
- Chapter 6 contains information about Oracle EDI.
- Chapter 7 contains information about Electronic Messaging Technical details.

- Chapter 8 contains information about Electronic Messaging Implementation details.
- Chapter 9 contains information about other Order Management Suite products, Electronic Messaging details.
- Appendix A contains information about Electronic Messaging messages.

Conventions

The following conventions are used in this manual:

Convention	Meaning
.	Vertical ellipsis points in an example mean that information not directly related to the example has been omitted.
...	Horizontal ellipsis points in statements or commands mean that parts of the statement or command not directly related to the example have been omitted
boldface text	Boldface type in text indicates a term defined in the text, the glossary, or in both locations.
< >	Angle brackets enclose user-supplied names.
[]	Brackets enclose optional clauses from which you can choose one or none.

Documentation Accessibility

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

Accessibility of Code Examples in Documentation

JAWS, a Windows screen reader, may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, JAWS may not always read a line of text that consists solely of a bracket or brace.

Accessibility of Links to External Web Sites in Documentation

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

Other Information Sources

You can choose from many sources of information, including online documentation, training, and support services, to increase your knowledge and understanding of *Oracle Order Management*.

If this guide refers you to other Oracle Applications documentation, use only the Release 11*i* versions of those guides.

Online Documentation

All Oracle Applications documentation is available online (HTML or PDF).

- **Online Help** - The new features section in the HTML help describes new features in 11*i*. This information is updated for each new release of *Oracle Order Management*. The new features section also includes information about any features that were not yet available when this guide was printed. For example, if your administrator has installed software from a mini-packs an upgrade, this document describes the new features. Online help patches are available on MetaLink.
- **About Documentation** - The Oracle Order Management About Documentation contains new and changed features, software updates, upgrade considerations, new and changed setup steps, new and changed windows, and new and changed public APIs for the latest release of Oracle Order Management. The About document is available on Oracle MetaLink.
- **Readme File** - Refer to the readme file for patches that you have installed to learn about new documentation or documentation patches that you can download.

Related User's Guides

Oracle Order Management shares business and setup information with other Oracle Applications products. Therefore, you may want to refer to other user's guides when you set up and use *Oracle Order Management*.

You can read the guides online by choosing Library from the expandable menu on your HTML help window, by reading from the Oracle Applications Document Library CD included in your media pack, or by using a Web browser with a URL that your system administrator provides.

If you require printed guides, you can purchase them from the Oracle Store at <http://oraclestore.oracle.com>.

Guides Related to All Products

Oracle Applications User's Guide

This guide explains how to enter data, query, run reports, and navigate using the graphical user interface (GUI) available with this release of *Oracle Order Management* (and any other Oracle Applications products). This guide also includes information on setting user profiles, as well as running and reviewing reports and concurrent processes.

You can access this user's guide online by choosing "Getting Started with Oracle Applications" from any Oracle Applications help file.

User Guides Related to This Product

Oracle Applications User's Guide This guide explains how to enter data, query, run reports, and navigate using the graphical user interface (GUI) available with this release of Oracle Applications products. This guide also includes information on setting user profiles, as well as running and reviewing reports and concurrent processes.

You can access this user's guide online by choosing "Getting Started with Oracle Applications" from any Oracle Applications help file.

Oracle Applications Demonstration User's Guide This guide documents the functional storyline and product flows for Global Computers, a fictional manufacturer of personal computers products and services. As well as including product overviews, the book contains detailed discussions and examples across each of the major

product flows. Tables, illustrations, and charts summarize key flows and data elements.

Reference Manuals

Oracle Automotive Implementation Manual This manual describes the setup and implementation of the Oracle Applications used for the Oracle Automotive solution.

Oracle Applications Message Reference Manual This manual describes all Oracle Applications messages. This manual is available in HTML format on the documentation CD-ROM for Release 11*i*.

Oracle Project Manufacturing Implementation Manual This manual describes the setup steps and implementation for Oracle Project Manufacturing.

Oracle Self-Service Web Applications Implementation Manual This manual describes the setup steps for Oracle Self-Service Web Applications and the Web Applications dictionary.

Installation and System Administration

Oracle Alert User's Guide This guide explains how to define periodic and event alerts to monitor the status of your Oracle Applications data.

Multiple Reporting Currencies in Oracle Applications If you use the Multiple Reporting Currencies feature to record transactions in more than one currency, use this manual before implementing the Oracle Applications product. This manual details additional steps and setup considerations for implementation.

Multiple Organizations in Oracle Applications If you use the Oracle Applications Multiple Organization Support feature to use multiple sets of books for one product installation, this guide describes all you need to know about setting up and using the product with this feature.

Oracle Applications Implementation Wizard User's Guide If you are implementing more than one Oracle product, you can use the Oracle Applications Implementation Wizard to coordinate your setup activities. This guide describes how to use the wizard.

Oracle Applications Developer's Guide This guide contains the coding standards followed by the Oracle Applications development staff. It describes the Oracle Application Object Library components needed to implement the Oracle Applications user interface described in the *Oracle Applications User Interface Standards*. It also provides information to help you build your custom Developer/2000 forms so that they integrate with Oracle Applications.

Oracle Applications Flexfields Guide This guide provides flexfields planning, setup and reference information for the implementation team, as well as for users responsible for the ongoing maintenance of Oracle Applications product data. This manual also provides information on creating custom reports on flexfields data.

Oracle Applications Installation Manual for Windows Clients This guide provides information you need to successfully install Oracle Financials, Oracle Public Sector Financials, Oracle Manufacturing, or Oracle Human Resources in your specific hardware and operating system software environment.

Oracle Applications Upgrade Preparation Manual This guide explains how to prepare your Oracle Applications products for an upgrade. It also contains information on completing the upgrade procedure for each product. Refer to this manual and the *Oracle Applications Installation Manual* when you plan to upgrade your products.

Oracle Applications System Administrator's Guide This manual provides planning and reference information for the System Administrator.

Oracle Applications Concepts

This guide provides an introduction to the concepts, features, technology stack, architecture, and terminology for Oracle Applications Release 11i. It provides a useful first book to read before an installation of Oracle Applications. This guide also introduces the concepts behind Applications-wide features such as Business Intelligence (BIS), languages and character sets, and Self-Service Web Applications.

Installing Oracle Applications

This guide provides instructions for managing the installation of Oracle Applications products. In Release 11i, much of the installation process is handled using Oracle Rapid Install, which minimizes the time to install Oracle Applications, the Oracle8 technology stack, and the Oracle8i Server technology stack by automating many of the required steps. This guide contains instructions for using Oracle Rapid Install and lists the tasks you need to perform to finish your

installation. You should use this guide in conjunction with individual product user's guides and implementation guides.

Upgrading Oracle Applications

Refer to this guide if you are upgrading your Oracle Applications Release 10.7 or Release 11.0 products to Release 11*i*. This guide describes the upgrade process and lists database and product-specific upgrade tasks. You must be either at Release 10.7 (NCA, SmartClient, or character mode) or Release 11.0, to upgrade to Release 11*i*. You cannot upgrade to Release 11*i* directly from releases prior to 10.7.

Maintaining Oracle Applications

Use this guide to help you run the various AD utilities, such as AutoUpgrade, AutoPatch, AD Administration, AD Controller, AD Relink, License Manager, and others. It contains how-to steps, screenshots, and other information that you need to run the AD utilities. This guide also provides information on maintaining the Oracle applications file system and database.

Oracle Applications System Administrator's Guide

This guide provides planning and reference information for the Oracle Applications System Administrator. It contains information on how to define security, customize menus and online help, and manage concurrent processing.

Oracle Alert User's Guide

This guide explains how to define periodic and event alerts to monitor the status of your Oracle Applications data.

Oracle Applications Developer's Guide

This guide contains the coding standards followed by the Oracle Applications development staff. It describes the Oracle Application Object Library components needed to implement the Oracle Applications user interface described in the *Oracle Applications User Interface Standards for Forms-Based Products*. It also provides information to help you build your custom Oracle Forms Developer 6*i* forms so that they integrate with Oracle Applications.

Oracle Applications User Interface Standards for Forms-Based Products

This guide contains the user interface (UI) standards followed by the Oracle Applications development staff. It describes the UI for the Oracle Applications

products and how to apply this UI to the design of an application built by using Oracle Forms.

Installation and System Administration

Oracle Applications Concepts

This guide provides an introduction to the concepts, features, technology stack, architecture, and terminology for Oracle Applications Release 11*i*. It provides a useful first book to read before an installation of Oracle Applications. This guide also introduces the concepts behind Applications-wide features such as Business Intelligence (BIS), languages and character sets, and Self-Service Web Applications.

Installing Oracle Applications

This guide provides instructions for managing the installation of Oracle Applications products. In Release 11*i*, much of the installation process is handled using Oracle Rapid Install, which minimizes the time to install Oracle Applications, the Oracle8 technology stack, and the Oracle8*i* Server technology stack by automating many of the required steps. This guide contains instructions for using Oracle Rapid Install and lists the tasks you need to perform to finish your installation. You should use this guide in conjunction with individual product user's guides and implementation guides.

Upgrading Oracle Applications

Refer to this guide if you are upgrading your Oracle Applications Release 10.7 or Release 11.0 products to Release 11*i*. This guide describes the upgrade process and lists database and product-specific upgrade tasks. You must be either at Release 10.7 (NCA, SmartClient, or character mode) or Release 11.0, to upgrade to Release 11*i*. You cannot upgrade to Release 11*i* directly from releases prior to 10.7.

Maintaining Oracle Applications

Use this guide to help you run the various AD utilities, such as AutoUpgrade, AutoPatch, AD Administration, AD Controller, AD Relink, License Manager, and others. It contains how-to steps, screenshots, and other information that you need to run the AD utilities. This guide also provides information on maintaining the Oracle applications file system and database.

Oracle Applications System Administrator's Guide

This guide provides planning and reference information for the Oracle Applications System Administrator. It contains information on how to define security, customize menus and online help, and manage concurrent processing.

Oracle Alert User's Guide

This guide explains how to define periodic and event alerts to monitor the status of your Oracle Applications data.

Oracle Applications Developer's Guide

This guide contains the coding standards followed by the Oracle Applications development staff. It describes the Oracle Application Object Library components needed to implement the Oracle Applications user interface described in the *Oracle Applications User Interface Standards for Forms-Based Products*. It also provides information to help you build your custom Oracle Forms Developer 6i forms so that they integrate with Oracle Applications.

Oracle Applications User Interface Standards for Forms-Based Products

This guide contains the user interface (UI) standards followed by the Oracle Applications development staff. It describes the UI for the Oracle Applications products and how to apply this UI to the design of an application built by using Oracle Forms.

Other Implementation Documentation

Multiple Reporting Currencies in Oracle Applications

If you use the Multiple Reporting Currencies feature to record transactions in more than one currency, use this manual before implementing *Oracle Order Management*. This manual details additional steps and setup considerations for implementing *Oracle Order Management* with this feature.

Multiple Organizations in Oracle Applications

This guide describes how to set up and use *Oracle Order Management* with Oracle Applications' Multiple Organization support feature, so you can define and support different organization structures when running a single installation of *Oracle Order Management*.

Oracle Workflow Guide

This guide explains how to define new workflow business processes as well as customize existing Oracle Applications-embedded workflow processes. You also use this guide to complete the setup steps necessary for any Oracle Applications product that includes workflow-enabled processes.

Oracle Applications Flexfields Guide

This guide provides flexfields planning, setup and reference information for the *Oracle Order Management* implementation team, as well as for users responsible for the ongoing maintenance of Oracle Applications product data. This manual also provides information on creating custom reports on flexfields data.

Oracle eTechnical Reference Manuals

Each eTechnical Reference Manual (eTRM) contains database diagrams and a detailed description of database tables, forms, reports, and programs for a specific Oracle Applications product. This information helps you convert data from your existing applications, integrate Oracle Applications data with non-Oracle applications, and write custom reports for Oracle Applications products. Oracle eTRM is available on Metalink

Oracle Manufacturing APIs and Open Interfaces Manual

This manual contains up-to-date information about integrating with other Oracle Manufacturing applications and with your other systems. This documentation includes APIs and open interfaces found in Oracle Manufacturing.

Oracle Applications Message Reference Manual

This manual describes all Oracle Applications messages. This manual is available in HTML format on the documentation CD-ROM for Release 11i.

Training and Support

Training

Oracle offers a complete set of training courses to help you and your staff master *Oracle Order Management* and reach full productivity quickly. These courses are organized into functional learning paths, so you take only those courses appropriate to your job or area of responsibility.

You have a choice of educational environments. You can attend courses offered by Oracle University at any one of our many Education Centers, you can arrange for our trainers to teach at your facility, or you can use Oracle Learning Network (OLN), Oracle University's online education utility. In addition, Oracle training professionals can tailor standard courses or develop custom courses to meet your needs. For example, you may want to use your organization structure, terminology, and data as examples in a customized training session delivered at your own facility.

Support

From on-site support to central support, our team of experienced professionals provides the help and information you need to keep **Oracle Order Management** working for you. This team includes your Technical Representative, Account Manager, and Oracle's large staff of consultants and support specialists with expertise in your business area, managing an Oracle8i server, and your hardware and software environment.

Do Not Use Database Tools to Modify Oracle Applications Data

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

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

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

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

About Oracle

Oracle Corporation develops and markets an integrated line of software products for database management, applications development, decision support, and office automation, as well as Oracle Applications, an integrated suite of more than 160 software modules for financial management, supply chain management, manufacturing, project systems, human resources and customer relationship management.

Oracle products are available for mainframes, minicomputers, personal computers, network computers and personal digital assistants, allowing organizations to integrate different computers, different operating systems, different networks, and even different database management systems, into a single, unified computing and information resource.

Oracle is the world's leading supplier of software for information management, and the world's second largest software company. Oracle offers its database, tools, and applications products, along with related consulting, education, and support services, in over 145 countries around the world.

Your Feedback

Thank you for using *Oracle Order Management* and this user's manual.

Oracle values your comments and feedback. At the end of this guide is a Reader's Comment Form you can use to explain what you like or dislike about *Oracle Order Management* or this user's manual. Send electronic mail to **mfgdoccomments_us@oracle.com**.

Integrating Your Systems

This chapter gives you an overview of the Order Management Suite use of Oracle Applications integration tools and explains how to use these tools to integrate Order Management products with one another and with your existing non-Oracle systems.

Oracle Applications integration tools are powerful, flexible tools that enable you to capture data from other Oracle and non-Oracle applications, define necessary format conversions, and direct data to your Oracle Order Management products. Topics covered in this chapter are:

- [Overview of Oracle Order Management APIs and Open Interfaces](#) on page 1-2

Overview of Oracle Order Management APIs and Open Interfaces

Oracle Order Management products provide a number of open interfaces so you can link them with non-Oracle applications, applications you build, applications on other computers, and even the applications of your suppliers and customers.

The purpose of this chapter is to help you understand the general model Oracle Order Management products use for open application interfaces. Other chapters in this book provide specific information on how to use each of the open interfaces. Additional functional information on these interfaces is available in each product's user's guide. Additional technical information on these interfaces is available in each product's Technical Reference Manual.

Basic Business Needs

The Oracle Order Management Suite's product APIs and open interfaces provide you with the features you need to support the following basic business needs:

- Connect to data collection devices. This lets you collect material movement transactions such as receipts, issues, quality data, movements, completions, and shipments. This speeds data entry and improves transaction accuracy.
- Connect to other systems — such as finite scheduling packages, computer-aided design systems, custom and legacy manufacturing systems — to create integrated enterprise wide systems.
- Connect to external systems — such as the customer's purchasing system and the supplier's order entry system — to better integrate the supply chain via electronic commerce.
- Control processing of inbound data imported from outside Oracle applications.
- Validate imported data to ensure integrity of Oracle Order Management products.
- Review, update, and resubmit imported data that failed validation.
- Export data from Oracle Order Management products

Oracle Order Management Interfaces

Open Interface Architectures

Oracle Order Management products have three different methods to import and export data:

- Interface Tables
- Interface Views (Business Views)
- Function Calls or Programmatic Interfaces (Processes)

Interface Tables

Interface tables in Oracle Order Management applications provide a temporary storage area for loading information from an external source. After the information is loaded, concurrent programs are executed to validate the information and then to apply the information to the base product tables.

The benefit of an interface table is that it provides a repository where records can be processed and if errors are found it can be edited and resubmitted.

In the Oracle Order Management family of applications you should never write directly to the product's tables. An Oracle Applications validation step is always required. You may achieve this either by loading information into the interface tables and submitting a provided concurrent program to validate and process the information or by using a function call to a programmatic interface.

Interface Views (Business Views)

Views simplify the data relationships for easier processing, whether for reporting or data export. Oracle Order Management products have defined *business views* that identify certain areas of key business interest. You can access this data using your tool of choice. The OE_ORDER_HEADERS_BV is an example of a key business view.

Product views are defined in the Technical Reference Manuals. The view definitions also briefly describe how they are used.

Function Calls or Programmatic Interfaces (Processes)

As an alternative to the two step process of writing to an interface table and executing a program to process the table data, many Oracle Order Management interfaces support direct function calls. A calling application can pass appropriate parameters and execute a public function to invoke the application logic.

The benefit of a function call is that the integration is real time, as opposed to interface tables where the integration is batch.

Summary: Beyond Published Interfaces

The Oracle Cooperative Applications Initiative references many third party products which provide import and export capabilities and allow loose to tight

integration with legacy systems, other supplier systems, and so on. Contact your Oracle consultant for more information about system integration.

Current Documentation For Open Interfaces

Below are the actual names of the tables, views, and modules:

Table 1–1 Table Key

Key	Key
Data Flow Direction	<i>Inbound</i> means into Oracle Order Management; <i>Outbound</i> means out from Oracle Order Management
Iface Man	The interface is documented in detail in the <i>Oracle Order Management APIs and Open Interfaces Manual</i>
TRM	The tables, views, or modules are described in the product’s Technical Reference Manual

Table 1–2 Oracle Order Management APIs/Open Interfaces

Interface/API Name	Data Flow Direction	Table, View, Process, or Procedure	Iface Man	TRM	Table, View, Module Name, or Procedure Name
ONT	ONT	ONT	ONT	ONT	ONT
Order Import	Inbound	Table	Yes	Yes	OE_HEADERS_IFACE_ALL OE_LINES_IFACE_ALL OE_RESERVTONS_IFACE_ALL OE_CREDITS_IFACE_ALL OE_PRICE_ADJS_IFACE_ALL OE_LOTSERIALS_IFACE_ALL OE_ACTIONS_IFACE_ALL
Process Order	Inbound	Process	No	Yes	OE_ORDER_PUB.PROCESS_ORDER
QP	QP	QP	QP	QP	QP

Table 1–2 Oracle Order Management APIs/Open Interfaces

Interface/API Name	Data Flow Direction	Table, View, Process, or Procedure	Iface Man	TRM	Table, View, Module Name, or Procedure Name
Agreement Public Application Program Interface	Inbound/Outbound	Procedure	Yes	Yes	OE_PRICING_CONT_PUB.PROCESS_AGREEMENT OE_PRICING_CONT_PUB.GET_AGREEMENT OE_PRICING_CONT_PUB.LOCK_AGREEMENT
Attribute Mapping Application Program Interface	Inbound/Outbound	Procedure	Yes	Yes	QP_ATTR_MAPPING_PUB.BUILD_CONTEXTS
Business Object for Modifier Setup Application Program Interface	Inbound/Outbound	Procedure	Yes	Yes	QP_MODIFIERS_PUB.PROCESS_MODIFIERS QP_MODIFIERS_PUB.GET_MODIFIERS QP_MODIFIERS_PUB.LOCK_MODIFIERS
Business Object for Pricing Formulas Application Program Interface	Inbound/Outbound	Procedure	Yes	Yes	QP_PRICE_FORMULA_PUB.LOCK_PRICE_FORMULA QP_PRICE_FORMULA_PUB.PROCESS_PRICE_FORMULA QP_PRICE_FORMULA_PUB.GET_PRICE_FORMULA
Business Object for Pricing Limits Application Program Interface	Inbound/Outbound	Procedure	Yes	Yes	QP_LIMITS_PUB.PROCESS_LIMITS QP_LIMITS_PUB.GET_LIMITS QP_LIMITS_PUB.LOCK_LIMITS
Get Currency Application Program Interface	Inbound/Outbound	Procedure	Yes	Yes	QP_GET_CURRENCY
Custom Runtime Sourcing Application Program Interface	Inbound/Outbound	Procedure	Yes	Yes	QP_RUNTIME_SOURCE
Get_Attribute_Text Application Program Interface	Inbound/Outbound	Procedure	Yes	Yes	QP_UTIL_PUB

Table 1–2 Oracle Order Management APIs/Open Interfaces

Interface/API Name	Data Flow Direction	Table, View, Process, or Procedure	Iface Man	TRM	Table, View, Module Name, or Procedure Name
Get Custom Price (Used in Formulas Setup) Application Program Interface	Inbound/Outbound	Procedure	Yes	Yes	QP_CUSTOM.GET_CUSTOM_PRICE
Get_Price_For_Line Application Program Interface	Inbound/Outbound	Procedure	Yes	Yes	QP_PREQ_PUB
Get Price List Currency Application Program Interface	Inbound/Outbound	Procedure	Yes	Yes	QP_UTIL_PUB
Get Price List Application Program Interface	Inbound/Outbound	Procedure	Yes	Yes	QP_GET_PRICE_LIST
Multi-Currency Conversion Setup Application Program Interface	Inbound/Outbound	Procedure	Yes	Yes	QP_CURRENCY_PUB.PROCESS_CURRENCY
Price List Setup	Inbound/Outbound	Procedure	Yes	Yes	QP_PRICE_LIST_PUB.PROCESS_PRICE_LIST QP_PRICE_LIST_PUB.GET_PRICE_LIST QP_PRICE_LIST_PUB.LOCK_PRICE_LIST
Price List Setup Group Application Program Interface	Inbound/Outbound	Procedure	Yes	Yes	QP_PRICE_LIST_GRP.PROCESS_PRICE_LIST
Price Request Application Program Interface	Inbound/Outbound	Procedure	Yes	Yes	QP_PREQ_GRP.PRICE_REQUEST
Pricing Data Bulk Loader API	Inbound/Outbound	Procedure	Yes	Yes	QP_BULK_LOADER_PUB.LOAD_PRICING_DATA
Pricing Object Security CHECK_FUNCTION	Inbound/Outbound	Procedure	Yes	Yes	QP_SECU_VIEW QP_SECU_UPDATE

Table 1–2 Oracle Order Management APIs/Open Interfaces

Interface/API Name	Data Flow Direction	Table, View, Process, or Procedure	Iface Man	TRM	Table, View, Module Name, or Procedure Name
Qualifiers Application Program Interface	Inbound/Outbound	Procedure	Yes	Yes	QP_QUALIFIER_RULES_PUB.PROCESS_QUALIFIER_RULES QP_QUALIFIER_RULES_PUB.LOCK_QUALIFIER_RULES QP_QUALIFIER_RULES_PUB.GET_QUALIFIER_RULES QP_QUALIFIER_RULES_PUB.COPY_QUALIFIER_RULES
Reverse Limits Application Program Interface	Inbound/Outbound	Procedure	Yes	Yes	QP_UTIL_PUB
Round Price Application Program Interface	Inbound/Outbound	Procedure	Yes	Yes	QP_ROUND_PRICE
Validate_Price_list_Curr_code Application Program Interface	Inbound/Outbound	Procedure	Yes	Yes	QP_VALIDATE_PRICE_LIST_CURR_CODE
RLM	RLM	RLM	RLM	RLM	RLM
RLM_INTERFACE_HEADERS	Inbound	Table	Yes	Yes	RLM_INTERFACE_HEADERS
RLM_INTERFACE_LINES	Inbound	Table	Yes	Yes	RLM_INTERFACE_LINES
WSH	WSH	WSH	WSH	WSH	WSH
Trip Public API	Inbound	Procedure	Yes	No	WSH_TRIPS_PUB (Procedure package)
Stop Public API	Inbound	Procedure	Yes	No	WSH_TRIP_STOPS_PUB (Procedure package)
Deliveries Public API	Inbound	Procedure	Yes	No	WSH_DELIVERIES_PUB (Procedure package)
Exceptions Public API	Inbound	Procedure	Yes	No	WSH_EXCEPTIONS_PUB (Procedure package)
Delivery Details Public API	Inbound	Procedure	Yes	No	WSH_DELIVERY_DETAILS_PUB (Procedure package)

Table 1–2 Oracle Order Management APIs/Open Interfaces

Interface/API Name	Data Flow Direction	Table, View, Process, or Procedure	Iface Man	TRM	Table, View, Module Name, or Procedure Name
Container Public API	Inbound	Procedure	Yes	No	WSH_CONTAINER_PUB (Procedure package)
Freight Costs Public API	Inbound	Procedure	Yes	No	WSH_FREIGHT_COSTS_PUB (Procedure package)
Pick Release Application Program Interface API	Inbound	Procedure	Yes	No	WSH_PICKING_BATCHES_PUB (Procedure package)

Inbound Open Interface Model

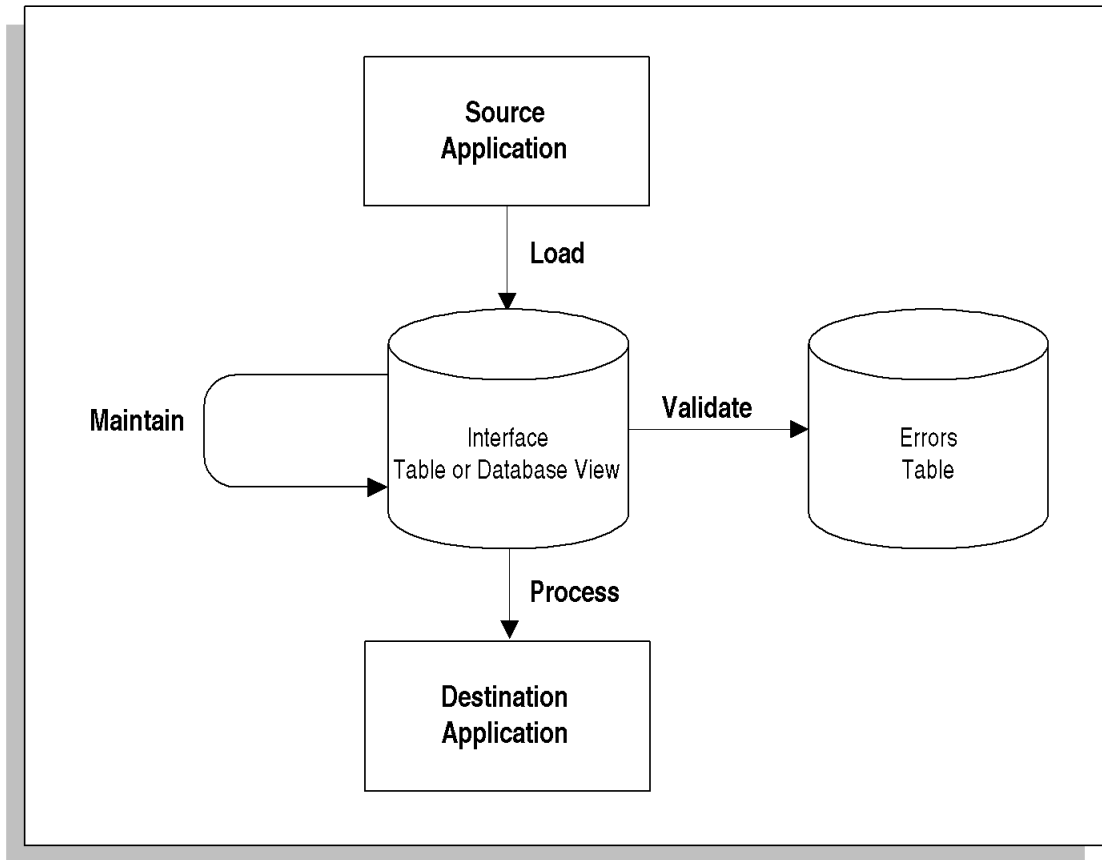
Oracle Order Management products provide both inbound and outbound interfaces. For inbound interfaces, where these products are the destination, interface tables as well as supporting validation, processing, and maintenance programs are provided. For outbound interfaces, where these products are the source, database views are provided and the destination application should provide the validation, processing, and maintenance programs.

Discussion of Inbound Interfaces Only

In this manual, we discuss only inbound interfaces in detail. You can find information about the tables, views, and processes involved in outbound interfaces in the product’s Technical Reference Manual. Note that the Technical Reference Manuals do *not* contain detailed, narrative discussions about the outbound interfaces.

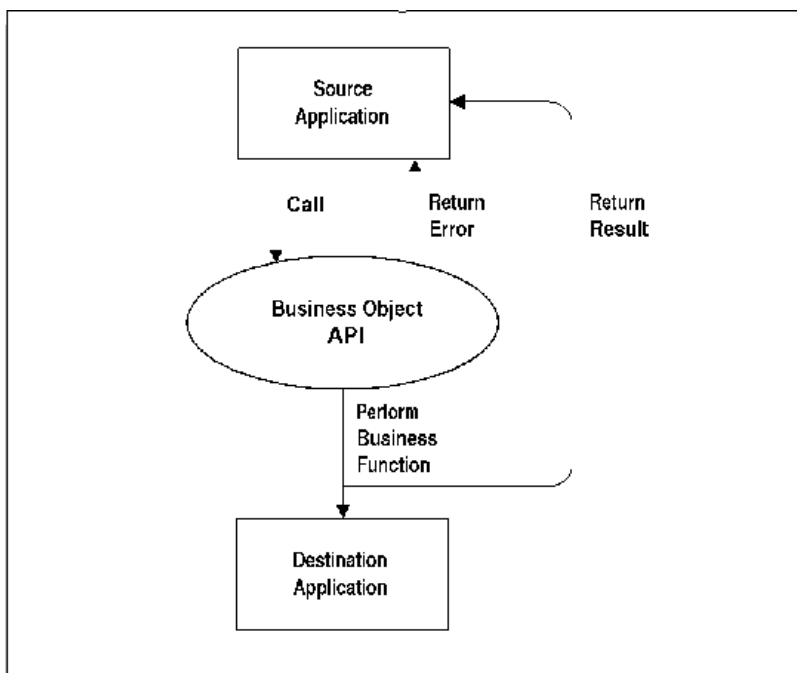
Open Interface Diagram

The general model for open application interfaces is as follows:

Figure 1–1 Open Interface Diagram**Open Application Programmatic Interface (API) Diagram**

The model used by APIs such as the Service Request interfaces (Oracle Service) is as follows:

Figure 1–2 Open Application Programmatic Interface (API) Diagram



Components of an Open Interface

There are a number of components that are generally common to all open interfaces. These components are described below. However, all open interfaces do not include every component, and in some cases the component may be implemented slightly differently than described below.

Source Application

You obtain data from a source application to pass on to a destination application for further processing and/or storage. Typically the data has completed processing in the source application before being passed.

Oracle Applications products are the source for outbound interfaces.

Destination Application

You send data to a destination application so that the application can perform further processing and/or storage.

Oracle Order Management products are the destinations for inbound interfaces.

Interface Table

For inbound interfaces, the interface table is the intermediary table where the data from the source application temporarily resides until it is validated and processed into an Oracle Order Management product. The various types of interface columns, are listed below:

Identifier Columns Identifier columns uniquely identify rows in the interface table and provide foreign key reference to both the source and destination applications. For example, typical identifier columns for a move transaction would identify:

- The source application
- The row's unique identifier in the source application
- The destination application's unique identifier.

Control Columns Control columns track the status of each row in the interface table as it is inserted, validated, errored, processed, and ultimately deleted. Additional control columns identify who last updated the row and the last update date.

Data Columns Data columns store the specific attributes that the source application is sending to the Oracle Order Management products.

Required Columns Required columns store the minimum information needed by the Oracle Order Management products to successfully process the interface row.

Some columns are conditionally required based on the specifics of the interface.

Derived Columns Derived columns are created by the destination product from information in the required columns.

Optional Columns Optional columns are not necessarily required by Oracle products but can be used for additional value-added functionality.

Errors Table

For inbound interfaces, the errors table stores all errors found by the validation and processing functions. In some cases, the errors table is a child of the interface table. This allows each row in the interface table to have many errors, so that you can manage multiple errors at once. In other cases, the errors are stored in a column within the interface table, which requires you to fix each error independently.

Database View

Database views are database objects that make data from the Oracle Order Management source products available for selection and use by destination applications.

Oracle Order Management products provide predefined views of key data that is likely to be used by destination applications.

For example, Oracle Cost Management provides work in process valuation and transaction distribution database views for use by other cost reporting destination products.

Load Function

For inbound interfaces, the load function is the set of programs that selects and accumulates data from the source application and inserts it into Oracle Order Management interface tables. The programming languages and tools used in the load function are highly dependent on the hardware and system software of the source application.

For example, if you are passing data between an Oracle based source application and an Oracle destination application, you would likely use a tool such as Pro*C or PL/SQL since these tools work in both environments. If you are bringing data from a non-Oracle based application into a product's interface table, you would likely use a procedural language available on the source application to select the data and convert it into an ASCII file. Then you could use SQL*Loader to insert that file into the destination product's interface table.

For outbound interfaces, the load function is the SQL that creates the database view.

Validate Function

The validate function is the set of programs that Oracle Applications destination products use to insure the integrity of inbound data. In the source application, you can typically validate data upon entry using techniques such as forms triggers, not

null columns, data types, and so on. However, since Oracle Applications products may not be the source of this data, validation programs ensure data integrity.

In addition, the validate function can derive additional columns based on the required columns and foreign key relationships with other data elsewhere in the Oracle destination application.

The validation programs check the interface table for rows requiring validation, then validate and update each row indicating either validation complete or errors found. If errors are found, validation programs need to write errors to the destination application's errors table or to the interface table's error column.

When an Oracle application product is the source product, the destination application should provide the validate function.

Process Function

The process function is a set of programs that processes the data from the interface table into the Oracle destination product. The specific processing performed varies by application. For open transaction interfaces, the processing generally includes recording transaction history, updating inventory and order balances, and charging costs.

Interfaces typically let you control both the frequency and the number of validated rows that the processing programs attempt to process. Upon successful completion of processing, the process function should delete the processed row from the interface table.

On occasion, the process function may need to insert rows into the errors table.

When an Oracle application product is the source, the destination application should provide the process function.

Maintain Function

The maintain function is generally accomplished from a window within an Oracle application. Most of these windows allow you to query, update, and resubmit interface records that have failed validation. You can generally use these windows to query unprocessed or unvalidated rows and check their current status.

In the case where there is no formal maintain function, you can use SQL*Plus to query and update the errored interface table rows.

When an Oracle application is the source application, the destination application should provide the maintain function.

2

Oracle Order Management Open Interfaces and APIs

This chapter contains information about Oracle Order Management Open Interfaces and public APIs.

- [Integrating Oracle Order Management Using Order Import](#) on page 2-4
- [Overview](#) on page 2-4
- [Prerequisites and Set-Up](#) on page 2-5
- [Transaction Sources](#) on page 2-11
- [Import Types](#) on page 2-13
- [Data Values and Options](#) on page 2-14
- [Validations](#) on page 2-15
- [Order Import Statistic Concurrent Program](#) on page 2-17
- [Request Submission](#) on page 2-17
- [Processing Results](#) on page 2-20

Oracle Order Management Interface Tables and Column Descriptions

- [OE_CUSTOMER_INFO_IFACE_ALL](#) on page 2-21
- [OE_HEADERS_IFACE_ALL](#) on page 2-25
- [OE_HEADERS_IFACE_ALL Conditional Settings](#) on page 2-38
- [OE_LINES_IFACE_ALL](#) on page 2-40
- [OE_LINES_IFACE_ALL Conditional Settings](#) on page 2-55
- [OE_PRICE_ADJS_IFACE_ALL](#) on page 2-58
- [OE_PRICE_ATTs_IFACE_ALL](#) on page 2-61
- [OE_CREDITS_IFACE_ALL](#) on page 2-63
- [OE_CREDITS_IFACE_ALL Conditional Settings](#) on page 2-64

-
- [OE_LOTSERIALS_IFACE_ALL](#) on page 2-65
 - [OE_RESERV_TNS_IFACE_ALL](#) on page 2-66
 - [OE_ACTIONS_IFACE_ALL](#) on page 2-68
 - [Process Order Application Open Interface](#) on page 2-69
 - [Process Order API Features](#) on page 2-70
 - [Functional Overview](#) on page 2-80
 - [Setting Up the Process Order Procedure](#) on page 2-84

Process Orders Entities and Associated Tables

- [Setting Up the Get_Order Procedure](#) on page 2-93

Process Orders Entities and Associated Tables

- [Setting Up the Lock_Order Procedure](#) on page 2-97

Process Orders Entities and Associated Tables

- [PL/SQL Record Structures](#) on page 2-103

Process Orders Entities and Associated Tables

- [Header_Val_Rec_Type](#) on page 2-111
- [Header_Adj_Rec_Type](#) on page 2-114
- [Header_Adj_Val_Rec_Type](#) on page 2-118
- [OE_PRICE_ADJ ASSOCS](#) on page 2-120
- [OE_SALES_CREDITS](#) on page 2-122
- [Line_Val_Rec_Type](#) on page 2-135
- [Process Order Usage](#) on page 2-148
- [CREATE operation](#) on page 2-148
- [UPDATE operation](#) on page 2-151
- [DELETE operation](#) on page 2-153
- [Process Order and Action Requests](#) on page 2-154
- [Book the order](#) on page 2-154
- [Apply Automatic Attachments](#) on page 2-155
- [Apply Hold](#) on page 2-155

-
- [Release Hold](#) on page 2-156
 - [Delink Config](#) on page 2-157
 - [Match and Reserve](#) on page 2-157
 - **[Integrating Oracle Order Management with Oracle Receivables and Invoicing](#) on page 2-159**
 - [Major Features](#) on page 2-159
 - [Invoicing of ATO Configurations](#) on page 2-165
 - [Understanding the Receivables Interface Tables](#) on page 2-166
 - [Invoicing Rules](#) on page 2-163
 - [Credit Method for Accounting Rule](#) on page 2-164
 - [Credit Method for Installments](#) on page 2-164
 - [Internal Sales Orders](#) on page 2-164
 - [Invoicing Attributes](#) on page 2-165

Integrating Oracle Order Management Using Order Import

Overview

Prior to this release, Order Management would analyze Order Import interface tables for related records to determine the optimum record processing order; the analysis used to occur each time the Order Import concurrent program was submitted. With this release of Order Management, the analysis of the Order Import interface tables no longer occurs for each submission of the Order Import concurrent program; you must manually submit the Order Import Statistics concurrent program prior to submitting the Order Import concurrent program if you wish to optimize interface record processing.

Order Import, like the rest of Oracle Applications 11*i*, uses the Cost Based Optimizer of the database for optimizing queries. The Cost Based Optimizer uses generated statistical information to optimize queries. The Order Import Statistics concurrent program gathers statistics that will be used by the cost based optimizer. This concurrent program should be run after data is populated into the interface tables.

See: Order Import Concurrent Program.

Order Import is an Order Management Open Interface that consists of open interface tables and a set of APIs. Order Import can import new, change, and completed sales orders or returns from other applications such as a legacy system. The orders may come from any source such as EDI transactions that are processed by the Oracle e-Commerce Gateway or internal orders created for internal requisitions developed in Oracle Purchasing or returns.

Order Import features include validation and defaulting, processing constraint checks, applying and releasing of order holds, scheduling of shipments, then ultimately inserting, updating or deleting the orders in the base Order Management tables. Order Management checks all the data during the import process to ensure its validity within Order Management. Valid transactions are then converted into orders with lines, reservations, price adjustments, and sales credits in the base Order Management tables.

You can use the Order Import Correction window to examine the order and optionally correct data if it fails the import process. You can use the Error Message window to determine if your data failed to import.

Each time you run Order Import, Order Management produces a summary of information letting you know of the total number of orders that Order Import evaluates, and succeeded or failed.

Prerequisites and Set-Up

Before using this program to import orders, you should:

- Set up every aspect of Order Management that you want to use with imported orders, including customers, pricing, items, and bills.
- Define and enable your Order Import sources using the Order Import Sources window.
- Determine if you should submit the Order Import Statistics concurrent program.

Order Management Parameters

The following Order Management parameter affects the operation of the Order Import program:

- *OM: Item Validation Organization*--Determines the organization used for validating items and bill of material structures.

Profile Options

- *OM: Reservation Time Fence*--This profile option controls automatic reservations during scheduling.
- *OM: Apply Automatic Attachments*--This profile option determines whether rule-based attachments are applied without user intervention.
- *OM: Schedule Line on Hold*-- This profile decides if Order Management scheduling should schedule lines that are on hold.
- *OM: AutoSchedule*-- This profile decides whether the order line should be automatically scheduled or not.
- *OM: Authorized to Override ATP*-- This profile provides the authorization to users to override the scheduling results.
- *OM: Unique Order Source, Orig Sys Document Ref Combination for Each Customer*-- This profile determines whether or not customer information will be utilized to distinguish orders with the same reference information.

Items and Bills

Order Management uses the same customer, item pricing, and bill attribute validation and logic for imported orders as for orders entered in the Sales Orders window.

You need to define items using Oracle Inventory for items to be orderable via Order Import. You also need to define bills of material in Oracle Bills of Material for models if you have any complex items that customers can order in various configurations.

Order Import provides the ability to import an item specified in the following supplier, customer or generic formats:

- Supplier Specific Internal Part number
- Customer Specific Item number
- Generic (depending on what you have set up in Oracle Inventory as cross-references):
 - CLEI (Common Language Equipment Identifier)
 - EAN (European Article Number) code
 - ISBN (International Standard Book Number)
 - JAN (Japanese Article Number) code
 - UPC (Universal Product code) code

Customer Relationships

Site Level Customers and Sites for Bill_To, Ship_To and Deliver_To will be validated based on the Customer Relationships profile.

This parameter has three setting:

Single Customer: Site Can only belong to the Sold to Customer. The Site level Customer cannot be different than the Sold to Customer.

Related Customers: Site Can only belong to the Sold to Customer or a Customer Related to the Sold To Customer . The Site level Customer can be different than the Sold to Customer but the relationship should exist between the site level customer and the Sold to Customer.

All Customers: Site Can only belong to the Sold to Customer or any other Customers. The Site level Customer can be different than the Sold to Customer and it is not required that relationship should exist between the site level customer and the Sold to Customer.

When checking for relationship we also check the appropriate usage in the Relationship also.

For example, customer "A" can be a valid related BillTO customer to Customer "B," only if the relationship exists and the bill_to_Flag is checked in the relationship.

Similarly for ShipTO and Deliver_to we check the Ship_To flag.

Add Customer

Order Management provides the capability to add a new customer, the address and contacts using Order Import. The table OE_CUSTOMER_INFO_IFACE_ALL needs to be populated for this.

Based on the data available in the OE_HEADERS_IFACE_ALL the data from the table OE_CUSTOMER_INFO_IFACE_ALL is processed to add a new customer. This table can also be used to import a new address only or contact information of an existing Customer.

The link between the header interface record and the customer interface record is made through the column CUSTOMER_INFO_REF of the table OE_CUSTOMER_INFO_IFACE_ALL and any one the columns mentioned below. The link depends on the CUSTOMER_INFO_TYPE_CODE. When the order import is used to create a new customer (new account/address/contact etc), the CUSTOMER_INFO_TYPE_CODE should be either ('ACCOUNT', 'ADDRESS', or 'CONTACT') to denote the type of information in the record. The following table explains the linkage between the various header interface table columns and the column CUSTOMER_INFO_TYPE_CODE and the usage of the combination.

Table 2–1 OE_HEADERS_IFACE_ALL

OE_HEADERS_IFACE_ALL	Usage	CUSTOMER_INFO_TYPE_CODE	Result
Orig_Sys_Customer_Ref		ACCOUNT	New Customer account will get created
Orig_Ship_Address_Ref	SHIP_TO	ADDRESS	New Ship to address will get created
Orig_Bill_Address_Ref	BILL_TO	ADDRESS	New Bill to address will get created
Orig_Deliver_Address_Ref	DELIVER_TO	ADDRESS	New Deliver to address will get created

Table 2–1 OE_HEADERS_IFACE_ALL

OE_HEADERS_IFACE_ALL	Usage	CUSTOMER_INFO_TYPE_CODE	Result
Sold_to_Contact_Ref	SOLD_TO	CONTACT	New Sold to Contact will get created
Ship_to_Contact_Ref	SHIP_TO	CONTACT	New Ship to Contact will get created
Bill_to_Contact_Ref	BILL_TO	CONTACT	New Bill to Contact will get created
Deliver_to_Contact_Ref	DELIVER_TO	CONTACT	New Deliver to Contact will get created

The following profiles affect the add customer functionality.

1. OM: Add Customer (Order Import)

This will decide whether the customer can be imported using order import. This must be set to “ALL” for importing new customer account and to ‘Address and Contact only ’ for importing addresses and contacts alone. The eligible values for the profile are as follows:

N = None

P = Address and Contact only

Y = All

2. OM: Email Required on New Customers

This will decide whether the records in the oe_customer_info_iface_all should have the email or not. If the profile is set to “Yes” then you must enter a not null value in the field EMAIL_ADDRESS of the table OE_CUSTOMER_INFO_IFACE_ALL.

3. HZ: Generate Party Number = ‘YES’ or NULL

This will decide whether the records in the oe_customer_info_iface_all should have the party number or not. If the profile is set to Yes then you must enter a not null value in the field PARTY_NUMBER of the table OE_CUSTOMER_INFO_IFACE_ALL.

4. HZ: Generate Contact Number = ‘YES’ or NULL.

If this is set to 'N' then Order Import program will generate the Contact Number. If this is set to 'YES' or NULL then it will be generated later while creating the Contact.

5. HZ: Generate Party Site Number = 'YES' or NULL.

This will decide whether the records in the oe_customer_info_iface_all should have the site number or not. If the profile is set to Yes then you must enter a not null value in the field SITE_NUMBER of the table OE_CUSTOMER_INFO_IFACE_ALL..

The value of the profile option 'Automatic Customer Numbering' and 'Automatic Site Numbering' from the AR System Options are also important. If the flags are unchecked in the option, then the following fields of the table OE_CUSTOMER_INFO_IFACE_ALL must have a not null value.

CUSTOMER_NUMBER if Automatic Customer Numbering is unchecked

LOCATION_NUMBER if Automatic Site Numbering is unchecked

Order Import will respect the setting of the OM System Parameter – Customer Relationships (Setup=>Parameters) and create customer-to-customer relationships if needed and allowed. The parameter values will be interpreted as follows:

Customer Relationships = 'Single Customer'

The Ship_To, Deliver_To and Bill_To addresses must belong to the Sold_To customer on the order. If an attempt is made to import an order with a Bill_To, Deliver_To or Ship_To from a different customer than the Sold_To, Order Import will raise an error and the order will not be imported.

Customer Relationships = 'Related Customers'

If a new customer is added for a Ship_To, Bill_To, or Deliver_To address, the appropriate relationship will be created in the TCA (Trading Community Architecture)

Customer Relationships = 'All Customers'

A new customer can be added, but no relationship creation occurs.

Importing an order using a new Customer information will require you to add and also to remove some data from the headers interface table depending on the kind of information being imported. For example if the a customer account alone is to be imported then the Orig_Sys_Customer_Ref needs to be populated and the sold_to_org/sold_to_org_id should be populated as null.

The some or all of the following data is to be populated in oe_headers_iface_all table,

Orig_Sys_Customer_Ref
Orig_Ship_Address_Ref
Orig_Bill_Address_Ref
Orig_Deliver_Address_Ref
Sold_to_Contact_Ref
Ship_to_Contact_Ref
Bill_to_Contact_Ref
Deliver_to_Contact_Ref

And some or all of the following data needs to be removed from the oe_headers_iface_all tables depending upon the information being imported.

sold_to_org / sold_to_org_id
invoice_to_org / invoice_to_org_id
ship_to_org / ship_to_org_id

Against this the oe_custome_info_iface_all table needs to be populated. The number of records being populated will depend on what is being imported. For example while importing only the customer account only one record will be populated and while importing account, address and contacts three or more records will be populated.

If the customer addresses are to be different for ship, bill, and deliver, then three different records need to be populated. Whether an address is for ship , bill, or deliver, is decided by the following columns of the oe_customer_info_iface_all table-

IS_SHIP_TO_ADDRESS,
IS_BILL_TO_ADDRESS,
IS_DELIVER_TO_ADDRESS

These take the values 'Y,' 'N' or null. So if the particular record needs to be only a bill to address, then the IS_BILL_TO_ADDRESS should be set as 'Y' and the rest as 'N' or null. If the same address needs to be Bill to , ship to and deliver to Address then the flag should be set as 'Y' against all the three columns.

The link between the Account record and the address or contact record is through the column `PARENT_CUSTOMER_REF`. This should be populated in the address or contact record with the `CUSTOMER_INFO_REF` of the account record. For example,

If the `CUSTOMER_INFO_REF` in the record with the `CUSTOMER_INFO_TYPE_CODE` as 'ACCOUNT' is 'XXXX' then the `PARENT_CUSTOMER_REF` for the record with the `CUSTOMER_INFO_TYPE_CODE` as 'ADDRESS' or 'CONTACT' should be 'XXXX.'

Defaulting Rules

You can setup your defaulting rules which allow you to default columns in the same way as for orders entered online. You can pass the column value Null to Order Import if you want the defaulting rules to populate the column. However, if the column is defined as Not Null or Mandatory column and the defaulting rules fail to default the column, for any reason, Order Import displays an error message without importing the order.

Transaction Sources

Importing from External Systems

You can import orders with any external source defined in the Define Document Sequences window.

Note: You cannot specify a value in the `DROP_SHIP` column of the `OE_HEADERS_IFACE_ALL`. If you enter a value in this column, the Order Import concurrent program will fail.

Internal Sales Orders

Oracle Purchasing uses Order Import to transfer requisitions for internally sourced products to Order Management. Once imported, the internal sales orders are processed as regular sales orders.

Returns

Returns can be imported like a regular sales order. Order Management utilizes workflow activities to import returns.

Special Considerations for importing Return orders

Creation of a non-referenced RMA

To import a return order for a non referenced return, you must:

- Populate all required attributes for creating a return order
- Use an order category of RETURN or MIXED for the Order Header record.
- Populate all required attributes for creating a return order lines.
 - For Order Line Record, you cannot specify a value for the line category_code column. You need to populate the column ordered_quantity with a negative value.
 - Line_type_id is optional, provided a default line_type has been defined for the specified Order Type.)
 - Additionally, you will need to populate the column reason_code for all return lines. Valid values are those values defined for the Order Management QuickCode CREDIT_MEMO_REASON.

Note: If the Reason Code is not provided, Order Import errors out. However we do not currently validate whether Reason Code is entered correctly. Users need to enter Reason Code when they populate the interface table, instead of the *definition* of the reason. For example: if you enter 'No reason provided' (definition) as reason but the system is expecting 'Not provided' (code), that is why the reason was not properly translated. If you enter 'Reason Code' and try again that works as expected.

- You can send the ordered_quantity as a negative on the line record and the line will be treated/created as an RMA line.

Creation of a Referenced RMA (if you want to return an existing outbound line) If you create a referenced RMA, you should copy the Header Record for the return from the referenced Order header record, modifying the Order Type to category RETURN or MIXED (order_type_id from oe_transaction_types_all).

For the Order Line record, populate the following attributes only:

1. line_category_code: RETURN
2. return_context: ORDER
3. return_attribute1: header_id from the referenced order.

4. `return_attribute2`: `line_id` from the referenced order line.
5. `calculate_price_flag`: Set it to *P* if you want to retain the original price, the flag to *Y* if you want to reprice the RMA line.
6. `line_type_id`: Assign a `line_type_id` from RMA line. `Line_type_id` is optional, provided a default `line_type` has been defined for the specified Order Line Type.)
7. `return_reason_code`: Populate a reason code from lookup_type *CREDIT_MEMO_REASON*
8. For sales credit info please populate the header_level/line level sales credits details from the referenced order.

Import Types

Configurations

Order Management provides you with the ability to import ATO and PTO configurations. For EDI orders, you can import valid and invalid configurations, however, you will not be able to book orders with invalid configurations. Top model line ref need to be populated for all lines including the model.

line for importing configurations

Changes

You can import changes to orders that have been imported by passing all changed data through Order Import. You can update or delete orders, order lines, price adjustments, and sales credits. You can use change sequence numbers to control the sequence of changes you want to make to orders.

Order Status

You can import new, booked or closed orders. If an order is imported with an entry status of Booked (`OE_HEADERS_IFACE_ALL.BOOKED_FLAG=Y`) the result after import is that a Action Request of `BOOK_ORDER` is initiated. You may also pass the Action Request to `BOOK_ORDER`; both methods are supported.

Order Import ensures that all required fields for entry or booking are validated appropriately as the orders are imported. Order Import imports the order in as Entered and attempts to book it. If any of the required fields for a booked order are not supplied, Order Management retains the order in the Entered status and notifies you of the error.

Line Sets

You can import grouped order lines, called sets, based on certain common attributes for a new or existing order. You can also add a line to an existing set. You will need to provide the set ID or name in the Order Import tables. If that set already exists, the line will be included in the set. However, if the set does not already exist, a new set will be created and the line will be added to the set. In addition, if any line attribute, which is also a set attribute, does not match with the set attribute value, the set attribute value will overwrite the line attribute.

Workflows

You can import an order within any valid order workflow activity. The order must be at the initial activity of Entered, Booked, or Closed. Orders imported using Order Import cannot be in the middle of a workflow activity.

Data Values and Options

Manual and Automatic Pricing

You can indicate whether you want to manually enter prices for imported orders or allow Order Management to automatically price the order. You can use automatic pricing or manual pricing for your imported orders. If you want to use automatic pricing, you should set the column `OE_LINES_INTERFACE.CALCULATE_PRICE_FLAG` to Y, and define all your pricing setup including discounts, promotions, surcharges, free goods, etc. in Oracle Pricing and Order Management. However, if you want to use the manual pricing, you should set the column `OE_LINES_INTERFACE.CALCULATE_PRICE_FLAG` to N. In this case, you should define all your discounts as line level, overridable, and not automatic.

Note: Order Import now allows you to import Coupons, Promotions, Surcharges, and Pricing Attributes.

Pricing Agreements

You can specify an agreement name if you want to order against a specific customer agreement for an order or order line.

Scheduling

Order Import enables you to reserve orders as they are imported, using the same rules as online order entry. If the scheduling request is unsuccessful on an imported order, the order will still be imported, and the scheduling exceptions can be viewed

in the Error Messages of the Order Import Corrections window. You can use Schedule, Unschedule, Reserve or Unreserve as values for scheduling actions.

To override the line, Schedule Ship Date or Arrival Date has to be populated on the line record. The dates populated are based on the Ordered Date Type. If the Ordered Date Type is 'Ship' then populate the ship date otherwise populate the Arrival Date.

Validations

Process Order Interface (API)

The Process Order Interface is the central application process interface (API) provided by Order Management to perform all common operations such as inserting, updating, deleting, and validating an order or order line. The API also performs the scheduling and returns a promise date. This API is called by Order Import.

Order Import passes one order, with all lines and other entities, at a time to the Process Order Interface, along with the operations that need to be completed on the order or line such as, inserting or updating an order or line. Errors at any line or entity level will cause the order to fail the importing of the entire order. In addition, Order Import processes only those orders and lines which are not rejected and do not have the ERROR_FLAG column set to Y from previous processes.

Attachments

Order Management applies any automatic attachments to imported orders that meet your automatic note criteria based on the setting of the *OM: Apply Automatic Attachments* profile option.

Credit Checking

Order Management performs credit checking on all imported orders or changes, according to the credit checking rules you have defined in Order Management.

Defaulting Rules

You can pass the column value Null to Order Import if you want the defaulting rules to populate the column. However, if the column is defined as Not Null or Mandatory column and the defaulting rules fail to default the column, for any reason, Order Import displays an error message without importing the order.

Holds and Releases

Order Management automatically applies all holds to imported orders and order lines that meet hold criteria. Order Import allows you to apply holds on imported orders for review, just as you would for orders entered through the Sales Orders window. You can also apply holds or release holds using the actions interface table.

Price Comparisons

Order Import performs a price comparison on your imported orders. For example, if you provide a selling price and also want the system to calculate a price, Order Import warns you of the differences, if any, between the two prices as discrepancies. The warning can be viewed in the Error Message window of the Order Import Corrections window.

If there is a difference between your selling price and the system calculated price, Order Import raises a warning of the difference. Order Import saves your customer-provided value for the selling price in a column on the order line table, so you can have visibility to what your customer sent in.

Note: You cannot copy or interface an order line having a price list with a currency code different from the existing or newly created order header's currency code. An error message will be displayed in the Process Messages window

Payment Term Comparison

Order Import performs payment term comparisons. If there is a difference between your payment terms, Order Import raises a warning of the difference. Order Import saves your customer-provided value for payment terms in a column on the order line table so that you can have visibility to what your customer sent in.

Processing Constraints

Order Import checks the processing constraints you have defined in Order Management to assure that any operation such as insert, update, and delete are acceptable by your security standards. Order Import displays an error message if it encounters a processing constraint that has been violated.

Corrected Data

Once the data is corrected, the ERROR_FLAG for the record is updated to N. You can set the REJECT_FLAG to Y for headers and line in case your data cannot be corrected by using the Order Import Corrections window.

Order Import Statistic Concurrent Program

The Order Import Statistics concurrent program performs a table analysis of all interface tables related to Order Import for determining optimum record processing should the Order Import concurrent program be submitted. You can choose to submit this program (or not) prior to each submission of the Order Import concurrent program. If you normally process a similar number of interface records, you typically do not need to submit this program prior to submitting the Order Import concurrent program.

There are no parameters for the submission of the Order Import Statistics concurrent program. See: *Oracle Order Management User's Guide*, Order Import Statistics Concurrent Program

Request Submission

You can submit a request by selecting Order Import Request. You can run the Order Import process in the validation-only mode.

This mode allows the transaction to be validated against all the Order Management rules but not pass valid transactions to the base Order Management tables. If you choose you can run production transactions in validation-only mode for a preview of exceptions. Make necessary corrections to the transactions in the Order Import window, then choose the Validate button to perform a validation check. The validation-only mode may also facilitate testing of transactions through Order Import even in a production environment to take advantage of all the setup is the production environment.

Parameters

The Order Import program provides the following parameters:

- Order Source
Choose a specific Order Import source that you have defined in the Order Import Sources window so that only records with that source are processed, or leave this parameter blank so that all enabled sources are processed for data existing in the interface tables.
- Order References
You can enter the System Document Reference if you want to run Order Import for a specific order.
- Validate Only (Yes/No)

Choose whether to validate only the data in the interface tables. If Yes, the order will be validated, but not imported into the base orders tables. The default value is No. In the Order Import Concurrent Program window, the “Instances” parameter is also displayed. This parameter determines the maximum number of child processes you wish Order Import to spawn.

Purge Open Interface Data Concurrent Program

The Purge Open Interface Data concurrent program allows you to delete records that are no longer required from the interface tables.

Parameters

The Order Import program provides the following parameters:

·View Name (required)

Choose a view to be purged. When you choose a particular view, any child views are automatically purged. For instance, if you select Headers Interface, all the child records of the deleted Headers Interface records from the Lines Interface will also be purged. Note that this concurrent program can also be used to purge Acknowledgment or Open Interface Tracking data.

Customer Number

You can optionally specify a Customer Number so that only records with the Sold_To_Org_Id or Sold_To_Org corresponding to the Customer Number are deleted.

Order Source

You can optionally specify an Order Import source that you have defined in the Order Import Sources window so that only records with that source are deleted.

·Order Reference To/Order Reference From

You can optionally enter a System Document Reference range so that only records with order references within that range are deleted.

Order Import Corrections Window

The Order Import window consists of the Find and Summary windows. The Find window allows you to find orders to be imported based on certain attributes such as Request ID, Order Source, Original System Document Reference, Sold To Org ID, and Change Sequence.

The Summary windows displays order headers, lines, sales credits, price adjustments, lot serials, reservations and action requests information. You have the ability to remove columns from the folder.

The Order Import window displays all orders or selected orders based on the criteria given in the Find window. You can modify the orders here. The orders that have errors display in red.

You can insert, update, and delete the orders and lines in the interface tables. You can update one or multiple orders or lines at the same time through the Summary window. You can also mark an order or a line to be rejected by setting the REJECTED flag. There are separate windows for the header and line level data. These windows have related fields grouped as tabs.

Buttons

- *Lines*: Displays line level information for orders.
- *Discounts*: Displays discount information for orders.
- *Validate*: Validates the data but does not import it. Only the selected orders will be validated and performed online.
- *Import*: Imports the orders. The data is validated before being imported. If an error is encountered while importing, the order will be rejected and the error messages can be viewed by choosing the Errors button. Only the selected orders will be imported and the import is performed online. If an order is successfully imported, it also gets deleted from the interface tables. If you attempt to re-query the window, you will not be able to view that order in the Order Import Corrections window.
- *Errors*: Displays all the errors encountered while importing. The error messages are stored context sensitive. If you choose the Errors button from the Order Headers region, all the errors for that order are displayed. If you choose the Errors button from the Lines region, all the errors are displayed for that line. If you encountered errors while importing orders, you can also fix these errors in the window and try importing the order again. You can navigate from the Errors window to the Order Headers or Lines region where the error has occurred.
- *Actions*: Displays order actions for orders.
- *Sales Credits*: Displays sales credit information for orders.
- *Add Customer*: To add new Customer and Address during the Import process.
- *Pricing Attributes*: To add Pricing Attributes for Order/Line.

Processing Results

Each time you run Order Import, Order Management automatically generates an Order Import processing results summary log which identifies the total number of successful and failed imported orders.

Oracle Order Management Interface Tables and Column Descriptions

Order Import uses the following interface tables during processing:

OE_CUSTOMER_INFO_IFACE_ALL

OE_HEADERS_IFACE_ALL

OE_LINES_IFACE_ALL

OE_PRICE_ADJS_IFACE_ALL

OE_PRICE_ATTS_IFACE_ALL

OE_CREDITS_IFACE_ALL

OE_LOTSERIALS_IFACE_ALL

OE_RESERVTONS_IFACE_ALL

OE_ACTIONS_IFACE_ALL

A table listing for each interface table is provided, and additional details on Order Management database tables, see Oracle eTRM. Oracle eTRM is available hosted on Oracle Metalink.

Table 2-2 OE_CUSTOMER_INFO_IFACE_ALL

Column Name	Type	Required (C = Conditionally Required)	Conditionally Required for Booking	Derived	Optional
CUSTOMER_INFO_REF	VARCHAR2(50)	Yes			
CURRENT_CUSTOMER_NUMBER	NUMBER	Yes, if Customer_info_Type_Code is 'ADDRESS' or 'CONTACT' and the corresponding customer exists already either Current_Customer_Number or Current_Customer_Id should be populated. It must not be populated if the corresponding customer is being created simultaneously.			
CURRENT_CUSTOMER_ID	NUMBER	Yes, if Customer_info_Type_Code is 'ADDRESS' or 'CONTACT' and the corresponding customer exists already either Current_Customer_Number or Current_Customer_Id should be populated. It must not be populated if the corresponding customer is being created simultaneously.			
CUSTOMER_TYPE	VARCHAR2(30)	No (Default value of 'ORGANIZATION' will be used)			
CUSTOMER_INFO_TYPE_CODE	VARCHAR2(10)	Yes			

Table 2–2 OE_CUSTOMER_INFO_IFACE_ALL

Column Name	Type	Required (C = Conditionally Required)	Conditionally Required for Booking	Derived	Optional
PARENT_CUSTOMER_REF	VARCHAR2(50)	Yes, if the corresponding customer is being created simultaneously			
ORGANIZATION_NAME	VARCHAR2(30)	Yes, if Customer_Info_Type_Code is 'ACCOUNT' and Customer_Type is 'ORGANIZATION'			
PERSON_FIRST_NAME	VARCHAR2(150)	Yes, if Customer_Type_Code is 'ADDRESS' and Customer_Type is 'PERSON'			
PERSON_MIDDLE_NAME	VARCHAR2(60)				
PERSON_LAST_NAME	VARCHAR2(150)	Yes, if Customer_Type is 'PERSON'			
PERSON_NAME_SUFFIX	VARCHAR2(30)				
PERSON_TITLE	VARCHAR2(60)				
CUSTOMER_NUMBER	NUMBER	Populate only if Customer_info_Type_Code is 'ACCOUNT' and automatic numbering is not set to occur in Accounts Receivable.			
EMAIL_ADDRESS	VARCHAR2(2000)	Yes, if OM's profile option OM: Email Required On New Customer is set to 'Yes' and Customer_Info_Type_Code is 'ACCOUNT'			

Table 2–2 OE_CUSTOMER_INFO_IFACE_ALL

Column Name	Type	Required (C = Conditionally Required)	Conditionally Required for Booking	Derived	Optional
PARTY_NUMBER	VARCHAR2(30)	Yes, if Customer_Info_Type_Code is 'ACCOUNT' and the customer should be added to an existing party, populate this field with that party_number. Must be left null to create a new party for the new customer.			
PHONE_COUNTRY_CODE	VARCHAR2(10)				
PHONE_AREA_CODE	VARCHAR2(10)				
PHONE_NUMBER	VARCHAR2(40)				
PHONE_EXTENSION	VARCHAR2(20)				
COUNTRY	VARCHAR2(60)	Yes, if Customer_Info_Type_Code is 'ADDRESS'			
ADDRESS1...ADDRESS4	VARCHAR2(240)	Yes, if Customer_Info_Type_Code is 'ADDRESS'			
CITY	VARCHAR2(60)	Yes, if Customer_Info_Type_Code is 'ADDRESS'			
POSTAL_CODE	VARCHAR2(60)	Yes, if Customer_Info_Type_Code is 'ADDRESS'			
STATE	VARCHAR2(60)	Yes, if Customer_Info_Type_Code is 'ADDRESS'			
PROVINCE	VARCHAR2(60)				
COUNTY	VARCHAR2(60)				

Table 2–2 OE_CUSTOMER_INFO_IFACE_ALL

Column Name	Type	Required (C = Conditionally Required)	Conditionally Required for Booking	Derived	Optional
IS_SHIP_TO_ADDRESS	VARCHAR2(1)	Yes, if Customer_Info_Type_Code is 'ADDRESS' and entering Ship To Address			
IS_BILL_TO_ADDRESS	VARCHAR2(1)	Yes, if Customer_Info_Type_Code is 'ADDRESS' and entering Bill To Address			
IS_DELIVER_TO_ADDRESS	VARCHAR2(1)	Yes, if Customer_Info_Type_Code is 'ADDRESS' and entering Deliver To Address			
SITE_NUMBER	VARCHAR2(80)	Yes, if Customer_Info_Type_Code is 'ADDRESS' and the customer should be added to an existing site, populate this field with that site_number.			
LOCATION_NUMBER	VARCHAR2(40)				
NEW_PARTY_ID	NUMBER	internal use only			
NEW_PARTY_NUMBER	NUMBER	internal use only			
NEW_ACCOUNT_ID	NUMBER	internal use only			
NEW_ACCOUNT_NUMBER	NUMBER	internal use only			
NEW_CONTACT_ID	NUMBER	internal use only			
NEW_ADDRESS_ID_SHIP	NUMBER	internal use only			
NEW_ADDRESS_ID_BILL	NUMBER	internal use only			
NEW_ADDRESS_ID_DELIVER	NUMBER	internal use only			

Table 2–2 OE_CUSTOMER_INFO_IFACE_ALL

Column Name	Type	Required (C = Conditionally Required)	Conditionally Required for Booking	Derived	Optional
ATTRIBUTE_CATEGORY	VARCHAR2(30)				
ATTRIBUTE1...24	VARCHAR2(150)				
GLOBAL_ATTRIBUTE_CATEGORY	VARCHAR2(30)				
GLOBAL_ATTRIBUTE1.. GLOBAL_ATTRIBUTE20	VARCHAR2(150)				
CREATION_DATE	DATE				
CREATED_BY	NUMBER				
LAST_UPDATE_DATE	DATE				
LAST_UPDATED_BY	NUMBER				
LAST_UPDATE_LOGIN	NUMBER				
REQUEST_ID	NUMBER				
ERROR_FLAG	VARCHAR2(1				
REJECTED_FLAG	VARCHAR2(1)				
ORG_ID	NUMBER				

Table 2–3 OE_HEADERS_IFACE_ALL

Column Name	Type	Required (C = Conditionally Required)	Conditionally Required for Booking	Derived	Optional
ORDER_SOURCE_ID	NUMBER	C			
ORIG_SYS_DOCUMENT_REF	VARCHAR2(50)	REQUIRED			
CHANGE_SEQUENCE	VARCHAR2(50)				X
CHANGE_REQUEST_CODE	VARCHAR2(30)				X
ORDER_SOURCE	VARCHAR2(30)	C			
ORG_ID	NUMBER			X	

Table 2–3 OE_HEADERS_IFACE_ALL

Column Name	Type	Required (C = Conditionally Required)	Conditionally Required for Booking	Derived	Optional
HEADER_ID	NUMBER			X	
ORDER_NUMBER	NUMBER			X	
VERSION_NUMBER	NUMBER				X
ORDERED_DATE	DATE			X	
ORDER_CATEGORY	VARCHAR2(30)				
ORDER_TYPE_ID	NUMBER	C			
ORDER_TYPE	VARCHAR2(30)	C			
PRICE_LIST_ID	NUMBER		C		
PRICE_LIST	VARCHAR2(30)		C		
CONVERSION_RATE	NUMBER	C			
CONVERSION_RATE_	DATE	C			
DATE					
CONVERSION_TYPE_	VARCHAR2(30)	C			
CODE					
CONVERSION_TYPE	VARCHAR2(30)	C			
TRANSACTIONAL_	VARCHAR2(15)			X	
CURR_CODE					
TRANSACTIONAL_	VARCHAR2(30)			X	
CURR					
SALESREP_ID	NUMBER		X		
SALESREP	VARCHAR2(30)		X		
SALES_CHANNEL_	VARCHAR2(30)				X
CODE					
RETURN_REASON_	VARCHAR2(30)	C			
CODE					
TAX_POINT_CODE	VARCHAR2(30)	For future use only			
TAX_POINT	VARCHAR2(30)				X
TAX_EXEMPT_FLAG	VARCHAR2(30)				X

Table 2–3 OE_HEADERS_IFACE_ALL

Column Name	Type	Required (C = Conditionally Required)	Conditionally Required for Booking	Derived	Optional
TAX_EXEMPT_NUMBER	VARCHAR2(50)				X
TAX_EXEMPT_REASON_CODE	VARCHAR2(30)	C			
TAX_EXEMPT_REASON	VARCHAR2(30)	C			
AGREEMENT_ID	NUMBER				X
AGREEMENT	VARCHAR2(50)				X
INVOICING_RULE_ID	NUMBER		X		
INVOICING_RULE	VARCHAR2(30)		X		
ACCOUNTING_RULE_ID	NUMBER		X		
ACCOUNTING_RULE	VARCHAR2(30)		X		
PAYMENT_TERM_ID	NUMBER		X		
PAYMENT_TERM	VARCHAR2(30)		X		
DEMAND_CLASS_CODE	VARCHAR2(30)				X
DEMAND_CLASS	VARCHAR2(30)				X
SHIPMENT_PRIORITY_CODE	VARCHAR2(30)				X
SHIPMENT_PRIORITY	VARCHAR2(30)				X
SHIPPING_METHOD_CODE	VARCHAR2(30)				X
SHIPPING_METHOD	VARCHAR2(30)				X
FREIGHT_CARRIER_CODE	VARCHAR2(30)				X
FREIGHT_TERMS_CODE	VARCHAR2(30)				X
FREIGHT_TERMS	VARCHAR2(30)				X
FOB_POINT_CODE	VARCHAR2(30)				X
FOB_POINT	VARCHAR2(30)				X
PARTIAL_SHIPMENTS_ALLOWED	VARCHAR2(1)				X

Table 2–3 OE_HEADERS_IFACE_ALL

Column Name	Type	Required (C = Conditionally Required)	Conditionally Required for Booking	Derived	Optional
SHIP_TOLERANCE_ ABOVE	NUMBER				X
SHIP_TOLERANCE_ BELOW	NUMBER				X
SHIPPING_ INSTRUCTIONS	VARCHAR2(240)				X
PACKING_ INSTRUCTIONS	VARCHAR2(240)				X
ORDER_DATE_TYPE_ CODE	VARCHAR2(30)				X
EARLIEST_SCHEDULE_ LIMIT	NUMBER			X	
LATEST_SCHEDULE_ LIMIT	NUMBER			X	
CUSTOMER_PO_ NUMBER	VARCHAR2(50)				X
CUSTOMER_PAYMENT_ TERM_ID					
CUSTOMER_PAYMENT_ TERM					
PAYMENT_TYPE_CODE	VARCHAR2(30)				X
PAYMENT_AMOUNT	NUMBER				X
CHECK_NUMBER	VARCHAR2(50)				X
CREDIT_CARD_CODE	VARCHAR2(30)				X
CREDIT_CARD_ HOLDER_NAME	VARCHAR2(50)				X
CREDIT_CARD_ NUMBER	VARCHAR2(50)				X
CREDIT_CARD_ EXPIRATION_DATE	DATE				X

Table 2–3 OE_HEADERS_IFACE_ALL

Column Name	Type	Required (C = Conditionally Required)	Conditionally Required for Booking	Derived	Optional
CREDIT_CARD_APPROVAL_CODE	VARCHAR2(50)				X
SOLD_FROM_ORG_ID	NUMBER		C		
SOLD_FROM_ORG	VARCHAR2(30)		C		
SOLD_TO_ORG_ID	NUMBER		C		
SOLD_TO_ORG	VARCHAR2(30)		C		
SOLD_TO_PARTY_ID	NUMBER				
SOLD_TO_PARTY_NUMBER	VARCHAR2(30)				
SHIP_FROM_ORG_ID	NUMBER				X
SHIP_FROM_ORG	VARCHAR2(30)				X
SHIP_TO_ORG_ID	NUMBER				X
SHIP_TO_ORG	VARCHAR2(30)				X
INVOICE_TO_ORG_ID	NUMBER				X
INVOICE_TO_ORG	VARCHAR2(30)				X
DELIVER_TO_ORG_ID	NUMBER				X
DELIVER_TO_ORG	VARCHAR2(30)			X	
DELIVER_TO_CUSTOMER_NUMBER	VARCHAR2(30)			X	
DELIVER_TO_CUSTOMER	VARCHAR2(30)			X	
SOLD_TO_CONTACT_ID	NUMBER				X
SOLD_TO_CONTACT	VARCHAR2(30)				X
SHIP_TO_CONTACT_ID	NUMBER				X
SHIP_TO_CONTACT	VARCHAR2(30)				X
INVOICE_TO_CONTACT_ID	NUMBER				X
INVOICE_TO_CONTACT	VARCHAR2(30)				X

Table 2–3 OE_HEADERS_IFACE_ALL

Column Name	Type	Required (C = Conditionally Required)	Conditionally Required for Booking	Derived	Optional
DELIVER_TO_CONTACT_ID	NUMBER				X
DELIVER_TO_CONTACT	VARCHAR2(30)				X
CUSTOMER_ID	NUMBER				X
CUSTOMER_NAME	VARCHAR2(30)		C		
SHIPMENT_PRIORITY_CODE_INT	VARCHAR2(30)				X
SHIP_TO_ADDRESS1	VARCHAR2(30)		C		
SHIP_TO_ADDRESS2	VARCHAR2(30)		C		
SHIP_TO_ADDRESS3	VARCHAR2(30)		C		
SHIP_TO_ADDRESS4	VARCHAR2(30)		C		
SHIP_TO_CITY	VARCHAR2(30)		C		
SHIP_TO_CONTACT_FIRST_NAME	VARCHAR2(30)				X
SHIP_TO_CONTACT_LAST_NAME	VARCHAR2(30)				X
SHIP_TO_COUNTY	VARCHAR2(30)		C		
SHIP_TO_CUSTOMER	VARCHAR2(30)		C		
SHIP_TO_CUSTOMER_NUMBER	VARCHAR2(30)		C		
SHIP_TO_POSTAL_CODE	VARCHAR2(30)		C		
SHIP_TO_PROVINCE	VARCHAR2(30)		C		
SHIP_TO_SITE_INT	VARCHAR2(30)			X	
SHIP_TO_STATE	VARCHAR2(30)		C		
SHIP_TO_COUNTRY	VARCHAR2(30)		C		
INVOICE_ADDRESS1	VARCHAR2(35)		C		
INVOICE_ADDRESS2	VARCHAR2(35)		C		
INVOICE_ADDRESS3	VARCHAR2(35)		C		

Table 2–3 OE_HEADERS_IFACE_ALL

Column Name	Type	Required (C = Conditionally Required)	Conditionally Required for Booking	Derived	Optional
INVOICE_ADDRESS4	VARCHAR2(35)		C		
INVOICE_CITY	VARCHAR2(30)		C		
INVOICE_COUNTRY	VARCHAR2(20)		C		
INVOICE_COUNTY	VARCHAR2(25)		C		
INVOICE_CUSTOMER	VARCHAR2(60)		C		
INVOICE_CUSTOMER_ NUMBER	VARCHAR2(30)		C		
INVOICE_POSTAL_ CODE	VARCHAR2(15)		C		
INVOICE_PROVINCE_ INT	VARCHAR2(30)		C		
INVOICE_SITE	VARCHAR2(30)			X	
INVOICE_SITE_CODE	VARCHAR2(30)			X	
INVOICE_STATE	VARCHAR2(30)		C		
INVOICE_TO_ CONTACT_FIRST_NAME	VARCHAR2(30)				X
INVOICE_TO_ CONTACT_LAST_NAME	VARCHAR2(30)				X
ORDERED_BY_ CONTACT_FIRST_NAME	VARCHAR2(30)				X
ORDERED_BY_ CONTACT_LAST_NAME	VARCHAR2(30)				X
DROP_SHIP_FLAG	VARCHAR2(1)				X
BOOKED_FLAG	VARCHAR2(1)				X
CLOSED_FLAG	VARCHAR2(1)				X
CANCELLED_FLAG	VARCHAR2(1)				X
REJECTED_FLAG	VARCHAR2(1)				X
CONTEXT	VARCHAR2(30)				X

Table 2–3 OE_HEADERS_IFACE_ALL

Column Name	Type	Required (C = Conditionally Required)	Conditionally Required for Booking	Derived	Optional
ATTRIBUTE1..20	VARCHAR2(240)				X
HEADER_PO_CONTEXT	VARCHAR2(30)				X
PO_ATTRIBUTE_1..20	VARCHAR2(240)				X
PO_REVISION_DATE	DATE				X
GLOBAL_ATTRIBUTE_ CATEGORY	VARCHAR2(30)				X
GLOBAL_ ATTRIBUTE1..20	VARCHAR2(240)				X
CREATED_BY	NUMBER	REQUIRED			
CREATION_DATE	DATE	REQUIRED			
LAST_UPDATED_BY	NUMBER	REQUIRED			
LAST_UPDATE_DATE	DATE	REQUIRED			
LAST_UPDATE_LOGIN	NUMBER				X
PROGRAM_ APPLICATION_ID	NUMBER				X
PROGRAM_ID	NUMBER				X
PROGRAM_UPDATE_ DATE	DATE				X
REQUEST_ID	NUMBER			X	
REQUEST_DATE	DATE			X	
SUBMISSION_DATETIME	DATE				X
OPERATION_CODE	VARCHAR2(30)	REQUIRED			
ERROR_FLAG	VARCHAR2(1)				X
READY_FLAG	VARCHAR2(1)				X
STATUS_FLAG	VARCHAR2(1)				X
FORCE_APPLY_FLAG	VARCHAR2(1)				X
CHANGE_REASON	VARCHAR2(30)				X

Table 2–3 OE_HEADERS_IFACE_ALL

Column Name	Type	Required (C = Conditionally Required)	Conditionally Required for Booking	Derived	Optional
CHANGE_COMMENTS	VARCHAR2(200)				X
TP_CONTEXT	VARCHAR2(30)				X
TP_ATTRIBUTE1..20	VARCHAR2(240)				X
BLANKET_NUMBER	NUMBER				
ORIG_SYS_CUSTOMER_REF	Varchar2(50)	Yes, if the customer is being created as Sold To for the order being imported			
ORIG_SHIP_ADDRESS_REF	Varchar2(50)	Yes, if Ship_To Address is being added for a customer.			
ORIG_BILL_ADDRESS_REF	Varchar2(50)	Yes, if Bill_To Address is being added for a customer			
ORIG_DELIVER_ADDRESS_REF	Varchar2(50)	Yes, if Deliver_To Address is being added for a customer			
SOLD_TO_CONTACT_REF	Varchar2(50)	Yes, if Sold_To_Contact is being added for a customer			
SHIP_TO_CONTACT_REF	Varchar2(50)	Yes, if Ship_To_Contact is being added for a customer			
BILL_TO_CONTACT_REF	Varchar2(50)	Yes, if Bill_To_Contact is being added for a customer			

Table 2–3 OE_HEADERS_IFACE_ALL

Column Name	Type	Required (C = Conditionally Required)	Conditionally Required for Booking	Derived	Optional
DELIVER_TO_ CONTACT_REF	Varchar2(50)	Yes, if Deliver_To_ Contact is being added for a customer			
XML_MESSAGE_ID	Varchar2(30)				X
XML_TRANSACTION_ varchar2(30)TYPE_CODE					X
TRANSACTION_PHASE_ CODE	Varchar2(30)				X
SALES_DOCUMENT_ NAME	Varchar2(240)				X
QUOTE_NUMBER	Number				X
QUOTE_DATE	Date				X
USER_STATUS_CODE	Varchar2(30)				X
EXPIRATION_DATE	Date				X
END_CUSTOMER_ NAME	VARCHAR2(360)				X
END_CUSTOMER_ NUMBER	VARCHAR2(50)				X
END_CUSTOMER_ PARTY_NUMBER	VARCHAR2(30)				X
END_CUSTOMER_ID	NUMBER				X
END_CUSTOMER_ PARTY_ID	NUMBER				X
END_CUSTOMER_ORG_ CONTACT_ID	NUMBER				X
END_CUSTOMER_ CONTACT_ID	NUMBER				X
END_CUSTOMER_ CONTACT	VARCHAR2(360)				X

Table 2–3 OE_HEADERS_IFACE_ALL

Column Name	Type	Required (C = Conditionally Required)	Conditionally Required for Booking	Derived	Optional
END_CUSTOMER_LOCATION	VARCHAR2(240)				X
END_CUSTOMER_SITE_USE_ID	NUMBER				X
END_CUSTOMER_PARTY_SITE_ID	NUMBER				X
END_CUSTOMER_PARTY_SITE_USE_ID	NUMBER				X
END_CUSTOMER_ADDRESS1	VARCHAR2(240)				X
END_CUSTOMER_ADDRESS2	VARCHAR2(240)				X
END_CUSTOMER_ADDRESS3	VARCHAR2(240)				X
END_CUSTOMER_ADDRESS4	VARCHAR2(240)				X
END_CUSTOMER_CITY	VARCHAR2(60)				X
END_CUSTOMER_STATE	VARCHAR2(60)				X
END_CUSTOMER_POSTAL_CODE	VARCHAR2(60)				X
END_CUSTOMER_COUNTRY	VARCHAR2(60)				X
END_CUSTOMER_PROVINCE	VARCHAR2(60)				X
END_CUSTOMER_COUNTY	VARCHAR2(60)				X
IB_OWNER	VARCHAR2(60)				X
IB_CURRENT_LOCATION	VARCHAR2(60)				X
IB_INSTALLED_AT_LOCATION	VARCHAR2(60)				X

Table 2–3 OE_HEADERS_IFACE_ALL

Column Name	Type	Required (C = Conditionally Required)	Conditionally Required for Booking	Derived	Optional
SOLD_TO_PARTY_ID	NUMBER				X
SOLD_TO_ORG_ CONTACT_ID	NUMBER				X
SHIP_TO_PARTY_ID	NUMBER				X
SHIP_TO_PARTY_SITE_ ID	NUMBER				X
SHIP_TO_PARTY_SITE_ USE_ID	NUMBER				X
DELIVER_TO_PARTY_ID	NUMBER				X
DELIVER_TO_PARTY_ SITE_ID	NUMBER				X
DELIVER_TO_PARTY_ SITE_USE_ID	NUMBER				X
INVOICE_TO_PARTY_ID	NUMBER				X
INVOICE_TO_PARTY_ SITE_ID	NUMBER				X
INVOICE_TO_PARTY_ SITE_USE_ID	NUMBER				X
SHIP_TO_ORG_ CONTACT_ID	NUMBER				X
DELIVER_TO_ORG_ CONTACT_ID	NUMBER				X
INVOICE_TO_ORG_ CONTACT_ID	NUMBER				X
SOLD_TO_PARTY_ NUMBER	VARCHAR2(30)				X

Table 2–3 OE_HEADERS_IFACE_ALL

Column Name	Type	Required (C = Conditionally Required)	Conditionally Required for Booking	Derived	Optional
SHIP_TO_PARTY_NUMBER	VARCHAR2(30)				X
INVOICE_TO_PARTY_NUMBER	VARCHAR2(30)				X
DELIVER_TO_PARTY_NUMBER	VARCHAR2(30)				X

OE_HEADERS_IFACE_ALL Derived Values

- TRANSACTIONAL_CURR_CODE = FND_CURRENCIES.CURRENCY_CODE
- SOLD_FROM_ORG_ID = HR_ALL_ORGANIZATION_UNITS.ORGANIZATION_ID
- ACCOUNTING_RULE_ID = RA_RULES.RULE_ID
- INVOICING_RULE_ID = RA_RULES.RULE_ID
- SALESREP_ID = RA_SALESREPS_ALL.SALESREP_ID
- SALESREP = RA_SALESREPS_ALL.NAME
- PAYMENT_TERM_ID = RA_TERMS_B.TERM_ID
- CUSTOMER_PAYMENT_TERM_ID = RA_TERMS_B.TERM_ID
- PAYMENT_TERM = RA_TERMS_TL.NAME
- CUSTOMER_PAYMENT_TERM = RA_TERMS_TL.NAME
- AGREEMENT_ID = OE_AGREEMENTS_B.AGREEMENT_ID
- ORDER_SOURCE_ID = OE_ORDER_SOURCES.ORDER_SOURCE_ID
- HEADER_ID = OE_ORDER_HEADERS_ALL.HEADER_ID
- PRICE_LIST_ID = QP_LIST_HEADERS_TL.LIST_HEADER_ID
- PRICE_LIST = QP_LIST_HEADERS_TL.NAME

Table 2–4 OE_HEADERS_IFACE_ALL Conditional Settings

Column Name	Conditional Setting requirement
ORDER_SOURCE_ID ORDER_SOURCE	Condition is that either one of the columns should be populated
ORDER_TYPE_ID ORDER_TYPE	Condition is that either one of the columns should be populated
CONVERSION_RATE CONVERSION_RATE_DATE	Condition is that either one of the columns should be populated
CONVERSION_TYPE_CODE CONVERSION_TYPE	Condition is that either one of the columns should be populated
TAX_EXEMPT_REASON_CODE TAX_EXEMPT_REASON	Condition is that either one of the columns should be populated
RETURN_REASON_CODE	Required for returns <i>only</i>
PRICE_LIST_ID PRICE_LIST	Condition is that either one of the columns should be populated
SOLD_FROM_ORG_ID SOLD_FROM_ORG	Condition is that either one of the columns should be populated
SOLD_TO_ORG_ID SOLD_TO_ORG CUSTOMER_NAME	Condition is that Sold_To_Org_ID and Customer_Name are required.

Table 2–4 OE_HEADERS_IFACE_ALL Conditional Settings

Column Name	Conditional Setting requirement
SHIP_TO_ADDRESS1..4 SHIP_TO_CITY SHIP_TO_COUNTY SHIP_TO_CUSTOMER SHIP_TO_CUSTOMER_NUMBER SHIP_TO_POSTAL_CODE SHIP_TO_PROVINCE SHIP_TO_SITE_INT SHIP_TO_STATE SHIP_TO_COUNTRY	These columns or <i>Ship_to_Org_id</i> should be provided
INVOICE_ADDRESS1..4 INVOICE_CITY INVOICE_COUNTRY INVOICE_COUNTY INVOICE_CUSTOMER INVOICE_CUSTOMER_NUMBER INVOICE_POSTAL_CODE INVOICE_PROVINCE_INT INVOICE_STATE	These columns or <i>Invoice_To_Org_id</i> should be provided.
DELIVER_TO_ADDRESS1..4 DELIVER_TO_CITY DELIVER_TO_COUNTRY DELIVER_TO_COUNTY DELIVER_TO_CUSTOMER DELIVER_TO_CUSTOMER_NUMBER DELIVER_TO_POSTAL_CODE DELIVER_TO_PROVINCE_INT DELIVER_TO_STATE	These columns or <i>Deliver_To_Org_id</i> should be provided.

Note: Attribute columns are for customer use only and will NEVER be used by Oracle Development as a means to enable any type of system functionality.

Table 2–5 OE_LINES_IFACE_ALL

Column Name	Type	Required (C = Conditionally Required)	Conditionally Required for Booking	Derived	Optional
ORDER_SOURCE_ID	NUMBER	REQUIRED			
ORIG_SYS_DOCUMENT_REF	VARCHAR2(50)	REQUIRED			
ORIG_SYS_LINE_REF	VARCHAR2(50)	REQUIRED			
ORIG_SYS_SHIPMENT_REF	VARCHAR2(50)	REQUIRED			
CHANGE_SEQUENCE	VARCHAR2(50)				X
CHANGE_REQUEST_CODE	VARCHAR2(30)				X
ORG_ID	NUMBER			X	
LINE_NUMBER	NUMBER			X	
SHIPMENT_NUMBER	NUMBER			X	
LINE_ID	NUMBER			X	
SPLIT_FROM_LINE_ID	NUMBER				X
LINE_TYPE_ID	NUMBER			X	
LINE_TYPE	VARCHAR2(30)			X	
ITEM_TYPE_CODE	VARCHAR2(30)			X	X
INVENTORY_ITEM_ID	NUMBER	C			
INVENTORY_ITEM	VARCHAR2(30)	REQUIRED			
TOP_MODEL_LINE_REF	VARCHAR2(50)	C			
LINK_TO_LINE_REF	VARCHAR2(50)	C			
EXPLOSION_DATE	DATE			X	
ATO_LINE_ID	NUMBER			X	

Table 2-5 OE_LINES_IFACE_ALL

Column Name	Type	Required (C = Conditionally Required)	Conditionally Required for Booking	Derived	Optional
COMPONENT_SEQUENCE_ID	NUMBER			X	
COMPONENT_CODE	VARCHAR2(50)			X	X
SORT_ORDER	VARCHAR2(240)			X	
MODEL_GROUP_NUMBER	NUMBER			X	
OPTION_NUMBER	NUMBER			X	
OPTION_FLAG	VARCHAR2(1)				X
SHIP_MODEL_COMPLETE_FLAG	VARCHAR2(1)			X	
SOURCE_TYPE_CODE	VARCHAR2(30)				X
SCHEDULE_STATUS_CODE	VARCHAR2(30)		C		
SCHEDULE_SHIP_DATE	DATE		C		
SCHEDULE_ARRIVAL_DATE	DATE		C		
ACTUAL_ARRIVAL_DATE	DATE				X
REQUEST_DATE	DATE	REQUIRED			
PROMISE_DATE	DATE		C		X
SCHEDULE_DATE	DATE				X
DELIVERY_LEAD_TIME	NUMBER	REQUIRED			
DELIVERY_ID	NUMBER	REQUIRED			
ORDERED_QUANTITY	NUMBER	REQUIRED			
ORDER_QUANTITY_UOM	VARCHAR2(3)	REQUIRED			
SHIPPING_QUANTITY	NUMBER	C			
SHIPPING_QUANTITY_UOM	VARCHAR2(3)	C			
SHIPPED_QUANTITY	NUMBER	C			
CANCELLED_QUANTITY	NUMBER	C			

Table 2–5 OE_LINES_IFACE_ALL

Column Name	Type	Required (C = Conditionally Required)	Conditionally Required for Booking	Derived	Optional
FULFILLED_QUANTITY	NUMBER	C			
PRICING_QUANTITY	NUMBER	C			
PRICING_QUANTITY_ UOM	VARCHAR2(3)	C			
SOLD_FROM_ORG_ID	NUMBER		C		
SOLD_FROM_ORG	VARCHAR2(30)		C		
SOLD_TO_ORG_ID	NUMBER		C		
SOLD_TO_ORG	VARCHAR2(30)		C		
SHIP_FROM_ORG_ID	NUMBER				X
SHIP_FROM_ORG	VARCHAR2(30)				X
SHIP_TO_ORG_ID	NUMBER				X
SHIP_TO_ORG	VARCHAR2(30)				X
DELIVER_TO_ORG_ID	NUMBER			X	
DELIVER_TO_ORG	VARCHAR2(30)			X	
INVOICE_TO_ORG_ID	NUMBER		C		
INVOICE_TO_ORG	VARCHAR2(30)		C		
SHIP_TO_ADDRESS1	VARCHAR2(30)		C		
SHIP_TO_ADDRESS2	VARCHAR2(30)		C		
SHIP_TO_ADDRESS3	VARCHAR2(30)		C		
SHIP_TO_ADDRESS4	VARCHAR2(30)		C		
SHIP_TO_CITY	VARCHAR2(30)		C		
SHIP_TO_COUNTY	VARCHAR2(30)		C		
SHIP_TO_STATE	VARCHAR2(30)		C		
SHIP_TO_POSTAL_CODE	VARCHAR2(30)		C		
SHIP_TO_COUNTRY	VARCHAR2(30)		C		

Table 2-5 OE_LINES_IFACE_ALL

Column Name	Type	Required (C = Conditionally Required)	Conditionally Required for Booking	Derived	Optional
SHIP_TO_CONTACT_ FIRST_NAME	VARCHAR2(30)				X
SHIP_TO_CONTACT_ LAST NAME	VARCHAR2(30)				X
SHIP_TO_CONTACT_JOB_ TITLE	VARCHAR2(30)				X
SHIP_TO_CONTACT_ AREA_CODE1	VARCHAR2(10)				X
SHIP_TO_CONTACT_ AREA_CODE2	VARCHAR2(10)				X
SHIP_TO_CONTACT_ AREA_CODE3	VARCHAR2(10)				X
SHIP_TO_CONTACT_ID	NUMBER				X
SHIP_TO_CONTACT	VARCHAR2(30)				X
DELIVER_TO_CONTACT_ ID	NUMBER				X
DELIVER_TO_CONTACT	VARCHAR2(30)				X
INVOICE_TO_CONTACT_ ID	NUMBER				X
INVOICE_TO_CONTACT	VARCHAR2(30)				X
DROP_SHIP_FLAG	VARCHAR2(1)				X
VEH_CUS_ITEM_CUM_ KEY_ID	NUMBER				X
CUST_PRODUCTION_SEQ_ NUM	NUMBER				X
LOAD_SEQ_NUMBER	NUMBER				X
OVER_SHIP_REASON_ CODE	VARCHAR2(30)				X
OVER_SHIP_RESOLVED_ FLAG	VARCHAR2(1)				X

Table 2–5 OE_LINES_IFACE_ALL

Column Name	Type	Required (C = Conditionally Required)	Conditionally Required for Booking	Derived	Optional
AUTHORIZED_TO_SHIP_FLAG	VARCHAR2(1)				X
SHIP_TOLERANCE_ABOVE	NUMBER				X
SHIP_TOLERANCE_BELOW	NUMBER				X
SHIP_SET_ID	NUMBER				X
SHIP_SET_NAME	VARCHAR2(30)				X
ARRIVAL_SET_ID	NUMBER				X
ARRIVAL_SET_NAME	VARCHAR2(30)				X
INVOICE_SET_ID	NUMBER				X
INVOICE_SET_NAME	VARCHAR2(30)				X
FULFILLMENT_SET_ID	NUMBER				X
FULFILLMENT_SET_NAME	VARCHAR2(30)				X
PRICE_LIST_ID	NUMBER		C		
PRICE_LIST	VARCHAR2(30)		C		
PRICING_DATE	DATE		C		
UNIT_LIST_PRICE	NUMBER		C		
UNIT_LIST_PRICE_PER_PQTY	NUMBER				
UNIT_SELLING_PRICE	NUMBER		C		
UNIT_SELLING_PRICE_PQTY	NUMBER				
CALCULATE_PRICE_FLAG	VARCHAR2(1)				X
TAX_CODE	VARCHAR2(50)				X
TAX	VARCHAR2(50)				X
TAX_VALUE	NUMBER				X
TAX_DATE	DATE				X

Table 2–5 OE_LINES_IFACE_ALL

Column Name	Type	Required (C = Conditionally Required)	Conditionally Required for Booking	Derived	Optional
TAX_POINT_CODE	VARCHAR2(30)				For future use
TAX_POINT	VARCHAR2(30)				X
TAX_EXEMPT_FLAG	VARCHAR2(30)				X
TAX_EXEMPT_NUMBER	VARCHAR2(50)				X
TAX_EXEMPT_REASON_CODE	VARCHAR2(30)				X
TAX_EXEMPT_REASON	VARCHAR2(30)				X
AGREEMENT_ID	NUMBER				X
AGREEMENT	VARCHAR2(30)				X
INVOICING_RULE_ID	NUMBER		C		
INVOICING_RULE	VARCHAR2(30)		C		
ACCOUNTING_RULE_ID	NUMBER		C		
ACCOUNTING_RULE	VARCHAR2(30)		C		
PAYMENT_TERM_ID	NUMBER		C		
PAYMENT_TERM	VARCHAR2(30)		C		
DEMAND_CLASS_CODE	VARCHAR2(30)				X
DEMAND_CLASS	VARCHAR2(30)		C		X
SHIPMENT_PRIORITY_CODE	VARCHAR2(30)				X
SHIPMENT_PRIORITY	VARCHAR2(30)				X
SHIPPING_METHOD_CODE	VARCHAR2(30)				X
SHIPPING_METHOD	VARCHAR2(30)				X
SHIPPING_INSTRUCTIONS	VARCHAR2(240)				X
PACKING_INSTRUCTIONS	VARCHAR2(240)				X
FREIGHT_CARRIER_CODE	VARCHAR2(30)				X

Table 2–5 OE_LINES_IFACE_ALL

Column Name	Type	Required (C = Conditionally Required)	Conditionally Required for Booking	Derived	Optional
FREIGHT_TERMS_CODE	VARCHAR2(30)				X
FREIGHT_TERMS	VARCHAR2(30)				X
FOB_POINT_CODE	VARCHAR2(30)				X
FOB_POINT	VARCHAR2(30)				X
SALESREP_ID	NUMBER			X	
SALESREP	VARCHAR2(30)		X		
RETURN_REASON_CODE	VARCHAR2(30)	C			
REFERENCE_TYPE	VARCHAR2(30)	C			
REFERENCE_HEADER_ID	NUMBER	C			
REFERENCE_HEADER	VARCHAR2(30)	C			
REFERENCE_LINE_ID	NUMBER	C			
REFERENCE_LINE	VARCHAR2(30)	C			
CREDIT_INVOICE_LINE_ID	NUMBER				X
CUSTOMER_PO_NUMBER	VARCHAR2(50)				X
CUSTOMER_LINE_NUMBER	VARCHAR2(50)				X
CUSTOMER_SHIPMENT_NUMBER	VARCHAR2(50)				X
CUSTOMER_ITEM_ID	NUMBER				X
CUSTOMER_ITEM_ID_TYPE	VARCHAR2(30)				X
CUSTOMER_ITEM_NAME	VARCHAR2(200)				X
CUSTOMER_ITEM_REVISION	VARCHAR2(50)				X
CUSTOMER_ITEM_NET_PRICE	NUMBER				X

Table 2-5 OE_LINES_IFACE_ALL

Column Name	Type	Required (C = Conditionally Required)	Conditionally Required for Booking	Derived	Optional
CUSTOMER_PAYMENT_TERM_ID					
CUSTOMER_PAYMENT_TERM					
DEMAND_BUCKET_TYPE_CODE	VARCHAR2(30)				X
DEMAND_BUCKET_TYPE	VARCHAR2(50)				X
SCHEDULE_ITEM_DETAIL	VARCHAR2(30)				X
DEMAND_STREAM	VARCHAR2(30)				X
CUSTOMER_DOCK_CODE	VARCHAR2(30)				X
CUSTOMER_DOCK	VARCHAR2(50)				X
CUSTOMER_JOB	VARCHAR2(50)				X
CUSTOMER_PRODUCTION_LINE	VARCHAR2(50)				X
CUST_MODEL_SERIAL_NUMBER	VARCHAR2(50)				X
PROJECT_ID	NUMBER				X
PROJECT	VARCHAR2(30)				X
TASK_ID	NUMBER				X
TASK	VARCHAR2(30)				X
END_ITEM_UNIT_NUMBER	VARCHAR2(30)				X
ITEM_REVISION	VARCHAR2(3)				X
SERVICE_DURATION	NUMBER	C			
SERVICE_START_DATE	DATE	C			
SERVICE_END_DATE	DATE	C			
SERVICE_COTERMINATE_FLAG	VARCHAR2(1)	C			

Table 2–5 OE_LINES_IFACE_ALL

Column Name	Type	Required (C = Conditionally Required)	Conditionally Required for Booking	Derived	Optional
UNIT_SELLING_PERCENT	NUMBER			X	
UNIT_LIST_PERCENT	NUMBER			X	
UNIT_PERCENT_BASE_PRICE	NUMBER			X	
SERVICE_NUMBER	NUMBER			X	
SERVICED_LINE_ID	NUMBER			X	
FULFILLED_FLAG	VARCHAR2(1)				X
CLOSED_FLAG	VARCHAR2(1)				X
CANCELLED_FLAG	VARCHAR2(1)				X
REJECTED_FLAG	VARCHAR2(1)				X
CONTRACT_PO_NUMBER	VARCHAR2(150)				X
LINE_PO_CONTEXT	VARCHAR2(30)				X
ATTRIBUTE1..20	VARCHAR2(240)				X
INDUSTRY_CONTEXT	VARCHAR2(30)				X
INDUSTRY_ATTRIBUTE1..30	VARCHAR2(240)				X
PRICING_CONTEXT	VARCHAR2(150)				X
PRICING_ATTRIBUTE1..10	VARCHAR2(240)				X
PRICING_ATTRIBUTE10	VARCHAR2(240)				X
GLOBAL_ATTRIBUTE_CATEGORY	VARCHAR2(30)				X
GLOBAL_ATTRIBUTE1..20	VARCHAR2(240)				X
RETURN_ATTRIBUTE1..20	VARCHAR2(240)				X
INVENTORY_ITEM_SEGMENT_1...20	VARCHAR2(240)				X
SERVICE_CONTEXT	VARCHAR2(30)				X
SERVICE_ATTRIBUTE1..20	VARCHAR2(240)				X

Table 2-5 OE_LINES_IFACE_ALL

Column Name	Type	Required (C = Conditionally Required)	Conditionally Required for Booking	Derived	Optional
CREATED_BY	NUMBER	REQUIRED			
CREATION_DATE	DATE	REQUIRED			
LAST_UPDATED_BY	NUMBER	REQUIRED			
LAST_UPDATE_DATE	DATE				X
LAST_UPDATE_LOGIN	NUMBER				X
PROGRAM_APPLICATION_ID	NUMBER				X
PROGRAM_ID	NUMBER				X
PROGRAM_UPDATE_DATE	DATE				X
REQUEST_ID	NUMBER	REQUIRED			
OPERATION_CODE	VARCHAR2(30)				X
ERROR_FLAG	VARCHAR2(1)				X
STATUS_FLAG	VARCHAR2(1)				X
CHANGE_REASON	VARCHAR2(30)				X
CHANGE_COMMENTS	VARCHAR2(2000)				X
CONFIG_HEADER_ID	NUMBER			X	X
CONFIG_REV_NBR	NUMBER			X	X
CONFIGURATION_ID	NUMBER			X	X
SERVICE_TXN_REASON_CODE	VARCHAR2(30)				X
SERVICE_TXN_COMMENTS	VARCHAR2(2000)				X
BLANKET_NUMBER	NUMBER				
BLANKET_LINE_NUMBER	NUMBER				

Table 2–5 OE_LINES_IFACE_ALL

Column Name	Type	Required (C = Conditionally Required)	Conditionally Required for Booking	Derived	Optional
OVERRIDE_ATP_DATE_ CODE	Varchar2(30)	No Conditional Settings: Schedule_ship_ date, Schedule_ arrival_date	No	No	Y Comment: Based on the Ordered Date Type, either of the dates should be populated to override the line.
LATE_DEMAND_ PENALTY_FACTOR	Number	No	No	No	Y Values should be greater than 0.
SHIP_TO_CUSTOMER_ NAME	Varchar2(360)				
SHIP_TO_CUSTOMER_ NUMBER	Varchar2(30)				
INVOICE_TO_CUSTOMER_ NAME	Varchar2(360)				
INVOICE_TO_CUSTOMER_ NUMBER	Varchar2(30)				
DELIVER_TO_CUSTOMER_ NAME	Varchar2(360)				
DELIVER_TO_CUSTOMER_ NUMBER	Varchar2(30)				
DELIVER_TO_ADDRESS1	Varchar2(240)				
DELIVER_TO_ADDRESS2	Varchar2(240)				
DELIVER_TO_ADDRESS3	Varchar2(240)				
DELIVER_TO_ADDRESS4	Varchar2(240)				

Table 2–5 OE_LINES_IFACE_ALL

Column Name	Type	Required (C = Conditionally Required)	Conditionally Required for Booking	Derived	Optional
DELIVER_TO_CITY	Varchar2(60)				
DELIVER_TO_COUNTY	Varchar2(60)				
DELIVER_TO_COUNTRY	Varchar2(60)				
DELIVER_TO_STATE	Varchar2(60)				
DELIVER_TO_PROVINCE	Varchar2(60)				
DELIVER_TO_POSTAL_CODE	Varchar2(60)				
ORIG_SHIP_ADDRESS_REF	Varchar2(50)	Yes, if Ship_To_Address is being added for the customer at the line level			
ORIG_BILL_ADDRESS_REF	Varchar2(50)	Yes, if Bill_To_Address is being added for the customer at the line level			
ORIG_DELIVER_ADDRESS_REF	Varchar2(50)	Yes, if Deliver_To_Address is being added for the customer at the line level			
SHIP_TO_CONTACT_REF	Varchar2(50)	Yes, if Ship_To_Contact is being added for the customer at the line level			
BILL_TO_CONTACT_REF	Varchar2(50)	Yes, if Bill_To_Contact is being added for the customer at the line level			

Table 2–5 OE_LINES_IFACE_ALL

Column Name	Type	Required (C = Conditionally Required)	Conditionally Required for Booking	Derived	Optional
DELIVER_TO_CONTACT_REF	Varchar2(50)	Yes, if Deliver_To_Contact is being added for the customer at the line level			
XML_TRANSACTION_TYPE_CODE					
SHIP_TO_PARTY_ID	NUMBER				X
SHIP_TO_PARTY_SITE_ID	NUMBER				X
SHIP_TO_PARTY_SITE_USE_ID	NUMBER				X
DELIVER_TO_PARTY_ID	NUMBER				X
DELIVER_TO_PARTY_SITE_ID	NUMBER				X
DELIVER_TO_PARTY_SITE_USE_ID	NUMBER				X
INVOICE_TO_PARTY_ID	NUMBER				X
INVOICE_TO_PARTY_SITE_ID	NUMBER				X
INVOICE_TO_PARTY_SITE_USE_ID	NUMBER				X
SHIP_TO_ORG_CONTACT_ID	NUMBER				X
DELIVER_TO_ORG_CONTACT_ID	NUMBER				X
INVOICE_TO_ORG_CONTACT_ID	NUMBER				X
SHIP_TO_PARTY_NUMBER	VARCHAR2(30)				X
INVOICE_TO_PARTY_NUMBER	VARCHAR2(30)				X

Table 2-5 OE_LINES_IFACE_ALL

Column Name	Type	Required (C = Conditionally Required)	Conditionally Required for Booking	Derived	Optional
DELIVER_TO_PARTY_NUMBER	VARCHAR2(30)				X
END_CUSTOMER_NAME	VARCHAR2(360)				X
END_CUSTOMER_NUMBER	VARCHAR2(50)				X
END_CUSTOMER_PARTY_NUMBER	VARCHAR2(30)				X
END_CUSTOMER_ID	NUMBER				X
END_CUSTOMER_PARTY_ID	NUMBER				X
END_CUSTOMER_ORG_CONTACT_ID	NUMBER				X
END_CUSTOMER_CONTACT_ID	NUMBER				X
END_CUSTOMER_CONTACT	VARCHAR2(360)				X
END_CUSTOMER_LOCATION	VARCHAR2(240)				X
END_CUSTOMER_SITE_USE_ID	NUMBER				X
END_CUSTOMER_PARTY_SITE_ID	NUMBER				X
END_CUSTOMER_PARTY_SITE_USE_ID	NUMBER				X
END_CUSTOMER_ADDRESS1	VARCHAR2(240)				X
END_CUSTOMER_ADDRESS2	VARCHAR2(240)				X
END_CUSTOMER_ADDRESS3	VARCHAR2(240)				X

Table 2–5 OE_LINES_IFACE_ALL

Column Name	Type	Required (C = Conditionally Required)	Conditionally Required for Booking	Derived	Optional
END_CUSTOMER_ADDRESS4	VARCHAR2(240)				X
END_CUSTOMER_CITY	VARCHAR2(60)				X
END_CUSTOMER_STATE	VARCHAR2(60)				X
END_CUSTOMER_POSTAL_CODE	VARCHAR2(60)				X
END_CUSTOMER_COUNTRY	VARCHAR2(60)				X
END_CUSTOMER_PROVINCE	VARCHAR2(60)				X
END_CUSTOMER_COUNTY	VARCHAR2(60)				X
IB_OWNER	VARCHAR2(60)				X
IB_CURRENT_LOCATION	VARCHAR2(60)				X
IB_INSTALLED_AT_LOCATION	VARCHAR2(60)				X

OE_LINES_IFACE_ALL Derived Values

- AGREEMENT_ID = OE_AGREEMENTS_TL.AGREEMENT_ID
- SHIP_FROM_ORG_ID = HR_ALL_ORGANIZATION_UNITS.ORGANIZATION_ID
- SOLD_FROM_ORG_ID = HR_ALL_ORGANIZATION_UNITS.ORGANIZATION_ID
- ACCOUNTING_RULE_ID = RA_RULES.RULE_ID
- INVOICING_RULE_ID = RA_RULES.RULE_ID
- SALESREP_ID = RA_SALESREPS_ALL.SALESREP_ID
- PRICE_LIST_ID = QP_LIST_HEADERS_TL.LIST_HEADER_ID
- PAYMENT_TERM_ID = RA_TERMS_B.TERM_ID

- CUSTOMER_PAYMENT_TERM_ID = RA_TERMS_B.TERM_ID
- SALESREP = RA_SALESREPS_ALL.NAME
- CUSTOMER_PAYMENT_TERM = RA_TERMS_TL.NAME
- PAYMENT_TERM = RA_TERMS_TL.NAME
- PRICE_LIST = QP_LIST_HEADERS_TL.NAME
- ORDER_SOURCE_ID = OE_ORDER_SOURCES.ORDER_SOURCE_ID
- ORIG_SYS_DOCUMENT_REF =
- OE_HEADERS_IFACE_ALL.ORIG_SYS_DOCUMENT_REF
- LINK_TO_LINE_REF = OE_LINES_IFACE_ALL.ORIG_SYS_LINE_REF
- TOP_MODEL_LINE_REF = OE_LINES_IFACE_ALL.ORIG_SYS_LINE_REF

Table 2-6 OE_LINES_IFACE_ALL Conditional Settings

Column Name	Conditional Setting requirement
INVENTORY_ITEM_ID INVENTORY_ITEM	Condition is that either column should be populated
TOP_MODEL_LINE_REF LINK_TO_LINE_REF	Required for model items only
SHIPPING_QUANTITY SHIPPING_QUANTITY_UOM	Condition is that both columns should be populated, if populated since they are not required.
PRICING_QUANTITY PRICING_QUANTITY_UOM	Condition is that both columns should be populated, if populated since they are not required.
RETURN_REASON_CODE	Only required for returns.
REFERENCE_TYPE REFERENCE_HEADER_ID REFERENCE_HEADER REFERENCE_LINE_ID REFERENCE_LINE	Reference_Type should be there to populate either reference_header_id/reference_header and reference_line_id/reference_line (either id or value column should be populated; not both)

Table 2–6 OE_LINES_IFACE_ALL Conditional Settings

Column Name	Conditional Setting requirement
SERVICE_DURATION SERVICE_START_DATE SERVICE_END_DATE SERVICE_COTERMINATE_FLAG	Should be populated only for service items
SOLD_FROM_ORG_ID SOLD_FROM_ORG	Condition is that either column should be populated
SOLD_TO_ORG_ID SOLD_TO_ORG	Condition is that either column should be populated
INVOICE_TO_ORG_ID INVOICE_TO_ORG	Condition is that either column should be populated
SHIP_TO_ADDRESS1..4 SHIP_TO_CITY SHIP_TO_COUNTY SHIP_TO_STATE SHIP_TO_POSTAL_CODE SHIP_TO_COUNTRY	These columns or Ship_to_Org_id should be present.
PRICE_LIST_ID PRICE_LIST	Condition is that either column should be populated
ACCOUNTING_RULE_ID ACCOUNTING_RULE	Condition is that either column should be populated
PAYMENT_TERM_ID PAYMENT_TERM	Condition is that either column should be populated

Table 2–6 OE_LINES_IFACE_ALL Conditional Settings

Column Name	Conditional Setting requirement
DEMAND_CLASS_CODE DEMAND_CLASS	Condition is that either column should be populated, if populated since they are not required.
SHIP_TO_CUSTOMER_NAME SHIP_TO_CUSTOMER_NUMBER INVOICE_TO_CUSTOMER_NAME INVOICE_TO_CUSTOMER_NUMBER DELIVER_TO_CUSTOMER_NAME DELIVER_TO_CUSTOMER_NUMBER	
DELIVER_TO_ADDRESS1 DELIVER_TO_ADDRESS2 DELIVER_TO_ADDRESS3 DELIVER_TO_ADDRESS4 DELIVER_TO_CITY DELIVER_TO_COUNTY DELIVER_TO_COUNTRY DELIVER_TO_STATE DELIVER_TO_PROVINCE DELIVER_TO_POSTAL_CODE	

Note: Attribute columns are for customer use only and will NEVER be used by Oracle Development as a means to enable any type of system functionality.

Table 2-7 OE_PRICE_ADJS_IFACE_ALL

Column Name	Type	Required (C = Conditionally Required)	Conditionally Required for Booking	Derived	Optional
AC_CONTEXT	VARCHAR2(150)				
AC_ATTIBUTE1...15	VARCHAR2(240)				
ATTRIBUTE1...15	VARCHAR2(240)				X
ADJUSTED_AMOUNT	VARCHAR2()				X
ADJUSTED_AMOUNT_PER_PQTY	VARCHAR2()				X
APPLIED_FLAG	VARCHAR2(1)				X
ARITHMETIC_OPERATOR	VARCHAR2()				X
AUTOMATIC_FLAG	VARCHAR2(1)				X
CHANGE_REQUEST_CODE	VARCHAR2(30)				X
CHANGE_SEQUENCE	VARCHAR2(50)				X
CHANGE_REASON_CODE	VARCHAR2(30)				
CHANGE_REASON_TEXT	VARCHAR2(2000)				
CHARGE_SUBTYPE_CODE	VARCHAR2(30)				
CHARGE_TYPE_CODE	VARCHAR2(30)				
CONTEXT	VARCHAR2(30)				X
COST_ID	NUMBER				
CREATED_BY	NUMBER	REQUIRED			
CREATION_DATE	DATE	REQUIRED			
CREDIT_OR_CHARGE_FLAG	VARCHAR2(1)				
DISCOUNT_ID	NUMBER				
DISCOUNT_LINE_ID	NUMBER				
DISCOUNT_NAME	VARCHAR2(30)				
ERROR_FLAG	VARCHAR2(1)				X
ESTIMATED_FLAG	VARCHAR2(1)				

Table 2-7 OE_PRICE_ADJS_IFACE_ALL

Column Name	Type	Required (C = Conditionally Required)	Conditionally Required for Booking	Derived	Optional
INC_IN_SALES_ PERFORMANCE	VARCHAR2(1)				
INCLUDE_ON_RETURNS_ FLAG	VARCHAR2(1)				
INTERFACE_STATUS	VARCHAR2(1000)				
INVOICED_FLAG	VARCHAR2(1)				
LAST_UPDATE_DATE	DATE	REQUIRED			
LAST_UPDATE_LOGIN	NUMBER				X
LAST_UPDATED_BY	NUMBER	REQUIRED			
LIST_HEADER_ID	NUMBER				
LIST_LINE_ID	NUMBER				
LIST_LINE_NUMBER	NUMBER				
LIST_LINE_TYPE_CODE	VARCHAR2(30)				
LIST_NAME	VARCHAR2(240)				
MODIFIED_FROM	VARCHAR2(240)				
MODIFIED_TO	VARCHAR2(240)				
MODIFIER_MECHANISM_ TYPE_CODE	VARCHAR2(30)				
MODIFIER_NAME	VARCHAR2(240)				
OPERAND	NUMBER				
OPERAND_PER_PQTY	NUMBER				
OPERATION_CODE	VARCHAR2(30)	REQUIRED			
ORDER_SOURCE_ID	NUMBER				
ORG_ID	NUMBER			X	
ORIG_SYS_DISCOUNT_REF	VARCHAR2(50)	REQUIRED			
ORIG_SYS_DOCUMENT_REF	VARCHAR2(50)	REQUIRED			

Table 2-7 OE_PRICE_ADJS_IFACE_ALL

Column Name	Type	Required (C = Conditionally Required)	Conditionally Required for Booking	Derived	Optional
ORIG_SYS_LINE_REF	VARCHAR2(50)	Conditional or optional (Required only if line level adjustment/cr edit)			
ORIG_SYS_SHIPMENT_REF	VARCHAR2(50)	OPTIONAL			
PARENT_ADJUSTMENT_ID	NUMBER				
PERCENT	NUMBER				X
PRICING_PHASE_ID	NUMBER				
PROGRAM_APPLICATION_ID	NUMBER				X
PROGRAM_ID	NUMBER				X
PROGRAM_UPDATE_DATE	DATE				X
REQUEST_ID	NUMBER				X
SOLD_TO_ORG	VARCHAR2(360)				X
SOLD_TO_ORG_ID	NUMBER				X
STATUS_FLAG	VARCHAR2(1)				X
TAX_CODE	VARCHAR2(50)				
UPDATE_ALLOWED	VARCHAR2(1)				
UPDATED_FLAG	VARCHAR2(1)				
VERSION_NUMBER	VARCHAR2(30)				

OE_PRICE_ADJS_IFACE_ALL Derived Values

- ORDER_SOURCE_ID = OE_ORDER_SOURCES.ORDER_SOURCE_ID
- ORIG_SYS_DISCOUNT_REF = OE_ORDER_HEADERS_ALL.Orig_SYS_DOCUMENT_REF
- ORIG_SYS_LINE_REF = OE_ORDER_LINES_ALL.Orig_SYS_LINE_REF

- ORIG_SYS_SHIPMENT_REF = OE_ORDER_LINES_ALL.ORIG_SYS_SHIPMENT_REF
- LIST_HEADER_ID = QP_LIST_HEADERS_B.LIST_HEADER_ID
- LIST_NAME = QP_LIST_HEADERS_TL.NAME
- LIST_LINE_ID = QP_LIST_LINES.LIST_LINE_ID

Table 2-8 OE_PRICE_ATTS_IFACE_ALL

Column Name	Type	Required (C = Conditionally Required)	Conditionally Required for Booking	Derived	Optional
ATTRIBUTE1...15	VARCHAR2(240)				X
CHANGE_REQUEST_CODE	VARCHAR2(30)				X
CHANGE_SEQUENCE	VARCHAR2(50)				X
CONTEXT	VARCHAR2(30)				X
CREATED_BY	NUMBER	REQUIRED			
CREATION_DATE	DATE	REQUIRED			
CREDIT_OR_CHARGE_FLAG	VARCHAR2(1)				
ERROR_FLAG	VARCHAR2(1)				X
FLEX_TITLE					
INTERFACE_STATUS	VARCHAR2(1000)				
LAST_UPDATE_DATE	DATE	REQUIRED			
LAST_UPDATE_LOGIN	NUMBER				X
LAST_UPDATED_BY	NUMBER	REQUIRED			
OPERATION_CODE	VARCHAR2(30)	REQUIRED			
ORDER_SOURCE_ID					
ORG_ID	NUMBER			X	
ORIG_SYS_ATT_REF	VARCHAR2(50)	REQUIRED			
ORIG_SYS_DOCUMENT_REF	VARCHAR2(50)	REQUIRED			

Table 2–8 OE_PRICE_ATTS_IFACE_ALL

Column Name	Type	Required (C = Conditionally Required)	Conditionally Required for Booking	Derived	Optional
ORIG_SYS_LINE_REF	VARCHAR2(50)	Conditional or optional (Required only if line level adjustment/cr edit)			
ORIG_SYS_SHIPMENT_REF	VARCHAR2(50)	OPTIONAL			
PRICING_ATTRIBUTE1...100					
PROGRAM_APPLICATION_ID	NUMBER				X
PROGRAM_ID	NUMBER				X
PROGRAM_UPDATE_DATE	DATE				X
REQUEST_ID	NUMBER				X
SOLD_TO_ORG	VARCHAR2(360)				X
SOLD_TO_ORG_ID	NUMBER				X
STATUS_FLAG	VARCHAR2(1)				X

OE_PRICE_ATTS_IFACE_ALL Derived Values

- ORDER_SOURCE_ID = OE_ORDER_SOURCES.ORDER_SOURCE_ID
- ORIG_SYS_DOCUMENT_REF = OE_ORDER_HEADERS_ALL.ORIG_SYS_DOCUMENT_REF
- ORIG_SYS_LINE_REF = OE_ORDER_LINES_ALL.ORIG_SYS_LINE_REF
- ORIG_SYS_SHIPMENT_REF = OE_ORDER_LINES_ALL.ORIG_SYS_SHIPMENT_REF

Table 2–9 OE_CREDITS_IFACE_ALL

Column Name	Type	Required (C= Conditionally Required)	Conditionally Required for Booking	Derived	Optional
ORDER_SOURCE_ID	NUMBER	REQUIRED			
ORIG_SYS_DOCUMENT_REF	VARCHAR2(50)	REQUIRED			
ORIG_SYS_LINE_REF	VARCHAR2(50)	Conditional or optional (Required only if line level adjustment/credit)			
ORIG_SYS_SHIPMENT_REF	VARCHAR2(50)	OPTIONAL			
ORIG_SYS_CREDIT_REF	VARCHAR2(50)	REQUIRED			
CHANGE_SEQUENCE	VARCHAR2(50)				X
CHANGE_REQUEST_CODE	VARCHAR2(30)				X
ORG_ID	NUMBER			X	
SALESREP_ID	NUMBER			X	
SALESREP	VARCHAR2(30)			X	
SALES_CREDIT_TYPE_ID	NUMBER	C			
SALES_CREDIT_TYPE	VARCHAR2(30)	C			
SOLD_TO_ORG	VARCHAR2(360)				X
SOLD_TO_ORG_ID	NUMBER				X
QUOTA_FLAG	VARCHAR2(1)			X	
PERCENT	NUMBER	REQUIRED			
CONTEXT	VARCHAR2(30)				X
ATTRIBUTE1..15	VARCHAR2(240)				X
CREATED_BY	NUMBER			X	
CREATION_DATE	DATE			X	
LAST_UPDATED_BY	NUMBER			X	

Table 2–9 OE_CREDITS_IFACE_ALL

Column Name	Type	Required (C= Conditionally Required)	Conditionally Required for Booking	Derived	Optional
LAST_UPDATE_DATE	DATE			X	
LAST_UPDATE_LOGIN	NUMBER			X	
PROGRAM_APPLICATION_ID	NUMBER				X
PROGRAM_ID	NUMBER				X
PROGRAM_UPDATE_DATE	DATE				X
REQUEST_ID	NUMBER				X
OPERATION_CODE	VARCHAR2(30)	REQUIRED			
ERROR_FLAG	VARCHAR2(1)				X
STATUS_FLAG	VARCHAR2(1)				X

Table 2–10 OE_CREDITS_IFACE_ALL Conditional Settings

Column Name	Conditional Setting requirement
SALES_CREDIT_TYPE_ID & SALES_CREDIT_TYPE	Condition is that either one these columns should be populated

OE_CREDITS_IFACE_ALL Derived Values

- ORDER_SOURCE_ID = OE_ORDER_SOURCES.ORDER_SOURCE_ID
- ORIG_SYS_DOCUMENT_REF = OE_HEADERS_IFACE_ALL.ORIG_SYS_DOCUMENT_REF
- ORIG_SYS_LINE_REF = OE_LINES_IFACE_ALL.ORIG_SYS_LINE_REF
- ORIG_SYS_SHIPMENT_REF = OE_LINES_IFACE_ALL.ORIG_SYS_SHIPMENT_REF

Table 2–11 OE_LOTSERIALS_IFACE_ALL

Column Name	Type	Required (C= Conditionally Required)	Conditionally Required for Booking	Derived	Optional
ORDER_SOURCE_ID	NUMBER	REQUIRED			
ORIG_SYS_DOCUMENT_REF	VARCHAR2(50)	REQUIRED			
ORIG_SYS_LINE_REF	VARCHAR2(50)	Conditional or optional (Required only if line level adjustment/credit)			
ORIG_SYS_SHIPMENT_REF	VARCHAR2(50)	OPTIONAL			X
ORIG_SYS_LOTSERIAL_REF	VARCHAR2(50)	REQUIRED			
CHANGE_SEQUENCE	VARCHAR2(50)				X
CHANGE_REQUEST_CODE	VARCHAR2(30)				X
ORG_ID	NUMBER			X	
LOT_NUMBER	NUMBER	C			
FROM_SERIAL_NUMBER	VARCHAR2(30)	C			
TO_SERIAL_NUMBER	VARCHAR2(30)	C			
QUANTITY	NUMBER	REQUIRED			
SOLD_TO_ORG	VARCHAR2(360)				X
SOLD_TO_ORG_ID	NUMBER				X
CONTEXT	VARCHAR2(30)				X
ATTRIBUTE1..15	VARCHAR2(240)				X
CREATED_BY	NUMBER	REQUIRED			
CREATION_DATE	DATE	REQUIRED			
LAST_UPDATED_BY	NUMBER	REQUIRED			
LAST_UPDATE_DATE	DATE	REQUIRED			
LAST_UPDATE_LOGIN	NUMBER	REQUIRED			

Table 2–11 OE_LOTSERIALS_IFACE_ALL

Column Name	Type	Required (C= Conditionally Required)	Conditionally Required for Booking	Derived	Optional
PROGRAM_APPLICATION_ID	NUMBER				X
PROGRAM_ID	NUMBER				X
PROGRAM_UPDATE_DATE	DATE				X
REQUEST_ID	NUMBER				X
OPERATION_CODE	VARCHAR2(30)	REQUIRED			
ERROR_FLAG	VARCHAR2(1)				X
STATUS_FLAG	VARCHAR2(1)				X

OE_LOTSERIALS_IFACE_ALL derived Values

- ORDER_SOURCE_ID = OE_ORDER_SOURCES.ORDER_SOURCE_ID
- ORIG_SYS_DOCUMENT_REF = OE_ORDER_LINES_ALL.ORIG_SYS_DOCUMENT_REF
- ORIG_SYS_LINE_REF = OE_LINES_IFACE_ALL.ORIG_SYS_LINE_REF
- ORIG_SYS_SHIPMENT_REF = OE_LINES_IFACE_ALL.ORIG_SYS_SHIPMENT_REF

Table 2–12 OE_RESERVTSN_IFACE_ALL

Column Name	Type	Required (C= Conditionally Required)	Conditionally Required for Booking	Derived	Optional
ORDER_SOURCE_ID	NUMBER	REQUIRED			
ORIG_SYS_DOCUMENT_REF	VARCHAR2(50)	REQUIRED			
ORIG_SYS_LINE_REF	VARCHAR2(50)	REQUIRED			
ORIG_SYS_SHIPMENT_REF	VARCHAR2(50)				X

Table 2–12 OE_RESERVTONS_IFACE_ALL

Column Name	Type	Required (C= Conditionally Required)	Conditionally Required for Booking	Derived	Optional
ORIG_SYS_RESERVATION_REF	VARCHAR2(50)	REQUIRED			
CHANGE_SEQUENCE	VARCHAR2(50)				X
ORG_ID	NUMBER			X	
INVENTORY_ITEM_ID	NUMBER	REQUIRED			
REVISION	VARCHAR2(3)				X
LOT_NUMBER_ID	NUMBER				X
LOT_NUMBER	VARCHAR2(30)				X
SOLD_TO_ORG	VARCHAR2(360)				X
SOLD_TO_ORG_ID	NUMBER				X
SUBINVENTORY_ID	NUMBER				
SUBINVENTORY_CODE	VARCHAR2(10)				X
LOCATOR_ID	NUMBER				X
QUANTITY	NUMBER	REQUIRED			
ATTRIBUTE_CATEGORY	VARCHAR2(30)				X
ATTRIBUTE1..15	VARCHAR2(240)				X
OPERATION_CODE	VARCHAR2(30)	REQUIRED			
REQUEST_ID	NUMBER				X
ERROR_FLAG	VARCHAR2(1)				X

OE_RESERVTONS_IFACE_ALL Derived Values

- ORDER_SOURCE_ID = OE_ORDER_SOURCES.ORDER_SOURCE_ID
- ORIG_SYS_DOCUMENT_REF = OE_ORDER_HEADERS_ALL.ORIG_SYS_DOCUMENT_REF
- ORIG_SYS_LINE_REF = OE_ORDER_LINES_ALL.ORIG_SYS_LINE_REF

- ORIG_SYS_SHIPMENT_REF = OE_ORDER_LINES_ALL.ORIG_SYS_SHIPMENT_REF
- INVENTORY_ITEM_ID = MTL_SYSTEM_ITEMS_B.INVENTORY_ITEM_ID

Table 2–13 OE_ACTIONS_IFACE_ALL

Column Name	Type	Required (C= Conditionally Required)	Conditionally Required for Booking	Derived	Optional
ORDER_SOURCE_ID	NUMBER				X
ORIG_SYS_DOCUMENT_REF	VARCHAR2(50)	REQUIRED			
ORIG_SYS_LINE_REF	VARCHAR2(50)				X
ORIG_SYS_SHIPMENT_REF	VARCHAR2(50)				X
CHANGE_SEQUENCE	VARCHAR2(50)				X
ORG_ID	NUMBER			X	
HOLD_ID	NUMBER				X
HOLD_TYPE_CODE	VARCHAR2(1)				X
HOLD_TYPE_ID	NUMBER				X
HOLD_UNTIL_DATE	DATE				X
RELEASE_REASON_CODE	VARCHAR2(30)				X
SOLD_TO_ORG	VARCHAR2(360)				X
SOLD_TO_ORG_ID	NUMBER				X
COMMENTS	VARCHAR2(240)				X
CONTEXT	VARCHAR2(240)				X
ATTRIBUTE1..15	VARCHAR2(240)				X
REQUEST_ID	NUMBER				X

Table 2–13 OE_ACTIONS_IFACE_ALL

Column Name	Type	Required (C= Conditionally Required)	Conditionally Required for Booking	Derived	Optional
OPERATION_CODE	VARCHAR2(30)	REQUIRED			
ERROR_FLAG	VARCHAR2(1)				X
STATUS_FLAG	VARCHAR2(1)				X

OE_ACTIONS_IFACE_ALL Derived Values

- ORDER_SOURCE_ID = OE_ORDER_SOURCES.ORDER_SOURCE_ID
- ORIG_SYS_DOCUMENT_REF = OE_HEADERS_IFACE_ALL.ORIG_SYS_DOCUMENT_REF
- ORIG_SYS_LINE_REF = OE_LINES_IFACE_ALL.ORIG_SYS_LINE_REF
- ORIG_SYS_SHIPMENT_REF = OE_LINES_IFACE_ALL.ORIG_SYS_SHIPMENT_REF

Process Order Application Open Interface

The Sales Order has been modeled as a business object that Oracle Order Management owns. The Sales Order business object comprises of several entities, namely, Header, Header Sales Credits, Header Price Adjustments, Header Pricing Attributes, Header Adjustment Attributes, Header Adjustment Associations, Lines, Line Sales Credits, Line Price Adjustments, Line Pricing Attributes, Line Adjustment Attributes, Line Adjustment Associations, and Line Lot Serial Numbers.

The Process Order Application Program Interface (API) is designed as the mechanism through which all data manipulation (inserts, updates and deletes) may be performed on the Sales Order business object entities and their attributes, in a consistent manner. Besides these, certain other action requests such as applying holds, attachments, booking etc. can also be processed using the Sales Order API. Business logic in the API not only takes care of updates to the attributes but also make calls to other functions depending on the changes to attribute values.

The importance of using the Process Order API for all data manipulation to the Sales Order Business object cannot be over stressed. It must be understood that by using the Process Order API we not only avoid duplication of business logic in many functions but also move towards the distributed solution approach.

Process Order API Features

Operations on the Sales Order Object

The Process Order API can be used to create, update or delete the following entities that the sales order business object consists of.

Table 2–14 Process Orders Entities and Associated Tables

Entity	Table Name
Order Header	OE_ORDER_HEADERS_ALL
Order Price Adjustments	OE_PRICE_ADJUSTMENTS
Order Sales Credits	OE_SALES_CREDITS
Order Line	OE_ORDER_LINES_ALL
Order Pricing Attributes	OE_ORDER_PRICE_ATTRIBS
Order Adjustment Attributes	OE_PRICE_ADJ_ATTRIBS
Order Adjustment Associations	OE_PRICE_ADJ ASSOCS
Line Sales Credits	OE_SALES_CREDITS
Line Price Adjustments	OE_PRICE_ADJUSTMENTS
Line Pricing Attributes	OE_ORDER_PRICE_ATTRIBS
Line Adjustment Attributes	OE_PRICE_ADJ_ATTRIBS
Line Adjustment Associations	OE_PRICE_ADJ ASSOCS
Lot Serial Numbers	OE_LOT_SERIAL_NUMBERS

Passing Parameters By Values

Process Order API provides users the capability of passing the attributes on the order entities by their display values instead of their internal identifiers (IDs or codes).

For e.g. to specify the customer on the order, the user can either pass in the customer ID on the header record, `p_header_rec.sold_to_org_id` or send in the display name of the customer on the header value record, `p_header_val_rec.sold_to_org`.

The values are internally resolved into the identifiers for all the entity records passed to process order. For the value fields that could not be resolved, error

messages are posted to the OM message stack and none of the records are processed any further.

If both the value and the identifier fields are populated for the same attribute, then information messages are posted to the OM message stack for such attributes. The identifier field takes preference and further processing is based on this field.

Pricing an Order/Line

Pricing in process order API can be controlled using flag `calculate_price_flag` on order line record. When set to 'Y' the process order API fetches the list price and applies adjustments and charges. When set to N, the process order API would fetch a list price if the list price is not passed and no adjustments would be applied. When set to P, all the modifiers which are associated with phases having override freeze flag set to Y are applied. That mainly includes freight charges.

You may use the Process Order Interface for order repricing, provided the order data you are updating has an change that will trigger repricing. Once Process Order is invoked and program logic determines an attribute that can trigger order repricing has been updated, a call to the pricing engine is made to reprice the order.

The following order attributes (database columns) can trigger Process Order to reprice an order or order line:

- `agreement_id`
- `cust_po_number`
- `inventory_item_id`
- `invoice_to_org_id`
- `ordered_item_id`
- `ordered_item`
- `line_category_code`
- `line_type_id`
- `ordered_quantity`
- `ordered_quantity_uom`
- `preferred_grade`
- `payment_term_id`
- `price_list_id`

- pricing_date
- request_date
- ship_to_org_id
- sold_to_org_id
- service_start_date
- service_end_date
- service_duration
- service_period

You can also choose to update order pricing via the action *Price Order* from the Order Organizer or Sales Orders window.

Note: You cannot interface an order line having a price list with a currency code different from the existing or newly created order header's currency code. An error message will be displayed in the Process Messages window.

Scheduling/Reservations

Process order API can be used to perform scheduling actions on order lines. Scheduling actions include: schedule, unschedule, reserve and unreserve.

The schedule_action_code field provided on the order line record (line_rec_type) can be used to provide the action which needs to be performed on the order line.

Reservations can also be performed by passing the reserved_quantity on the order line record. You do not need to send in the schedule action if reserved_quantity is passed.

Process Order API will also automatically schedule or re-schedule the lines if the schedule_ship_date or schedule_arrival_date field is passed or updated respectively.

Alternatively, you can just set the profile option *OM: AutoSchedule* to Yes and all standard lines will be automatically scheduled as they are created. This holds true only if the lines are not part of any set.

Return Lines

Process order can be used to create and update return lines also. To create a return line, you can either pass in the line category of *RETURN* on the order line record and the line type would default to the inbound line type of the order type. Alternatively, you can also provide a line type of the type *Return* on the order line record.

Additionally, if you want to specify a reference for the return line, you can pass in the return flexfields (return_context, return_attribute1-2).

Column Return_Context can have the following values to determine the reference type:

- Sales Order
- Customer PO
- Invoice
- Serial Number

Return_Attribute1... Return_Attribute2 can have the following values depending on the reference type:

1. Sales Order
 - return_Attribute1: Header ID
 - return_Attribute2: Line ID
2. Customer PO
 - return_Attribute1: Header ID
 - return_Attribute2: Line ID
3. Invoice
 - return_Attribute1: Invoice Header ID
 - return_Attribute2: Invoice Line ID
4. Serial Number
 - return_Attribute1: Inventory Item ID
 - return_Attribute2: Serial Number

Special Considerations for using Process Order for return orders

Creation of a non-referenced RMA: If you wish to process orders for a return order for a non referenced return, you must

- populate all required attributes for creating a return order
- use an order category of RETURN or MIXED for the Order Header record.
- populate all required attributes for creating a return order lines.
 - For Order Line Record, the line category should be set to RETURN, and specify a RETURN LINE TYPE within column `line_type_id`.
 - `Line_type_id` is optional, provided a default `line_type` has been defined for the specified Order Type.)
 - Additionally, you will need to populate the column `reason_code` for all return lines. Valid values are those values defined for the Order Management quickcode `CREDIT_MEMO_REASON`.

Creation of a Referenced RMA (if you want to return an existing outbound line) If you create a referenced RMA, you should copy the Header Record for the return from the referenced Order header record, modifying the Order Type to category RETURN or MIXED (`order_type_id` from `oe_transaction_types_all`).

For the Order Line record, populate the following attributes only:

1. `Line_category_code`: RETURN
2. `return_context`: ORDER
3. `return_attribute1`: `header_id` from the referenced order.
4. `return_attribute2`: `line_id` from the referenced order line.
5. `calculate_price_flag`: Set it to P if you want to retain the original price, the flag to Y if you want to reprice the RMA line.
6. `line_type_id`: Assign a `line_type_id` from RMA line. `Line_type_id` is optional, provided a default `line_type` has been defined for the specified Order Line Type.)
7. `return_reason_code`: Populate a reason code from `lookup_type = CREDIT_MEMO_REASON`.
8. Sales credit details: populate the `header_level/line level` sales credits from the referenced order.

Holds/Releases

The existing hold sources are evaluated and if they are applicable, holds are applied and released automatically on the orders or order lines when created/updated via the process order API.

For e.g. if there is a hold source defined for the customer ABC, all orders for that customer are placed on hold as they are entered.

Attachments

If the profile option *OM: Apply Automatic Attachments* is set to *Yes* and if attachment rules are applicable, attachments are automatically applied to the order header or order lines when they are created via the process order API.

Note: Attachments are NOT automatically deleted or re-applied if the order or line is updated.

The caller can also send in an explicit action request to apply attachments. Please refer to the section Action Requests for more details.

Sets

Process order can be used to add or delete order lines from ship sets, arrival sets or fulfillment sets.

User can add a line into a new set by passing set name (ship_set/arrival_set/fulfillment_set) on the line record or user can use set id (ship_set_id/arrival_set_id/fulfillment_set_id) to add to an existing set.

Automatic Fulfillment Set Functionality Support for Order Import.

From 11.5.10 Order Import functionality will support addition, removal, and moving lines to multiple fulfillment sets. This can be achieved by populating the Actions table with corresponding action code.

User Procedures

- Populate the Order Import Header Interface tables with Header Information
- Populate the Order Import Line Interface tables with Line Information

- There are three requirements Add the line to a Specified fulfillment Set (**ADD**) and Remove the line from specified fulfillment Set (**REMOVE**) and Move the line to new fulfillment set.
- Populate the Actions Interface table with Action code and Fulfillment set Name.

Consider the following example.

There are two lines in table OE_ORDER_LINES_ALL with the lines in fulfillment sets as shown.

Table 2–15 Example 1

Line_id	Fulfillment Set
1000	F1
2000	F2, F3

For adding, removing and moving lines from multiple fulfillment the following steps can achieve sets.

1. Populate Order Import Lines Interface Table.

Table 2–16 Example 2

Orig_sys_line_Ref	Operation Code
1000	UPDATE
2000	UPDATE

2. Populate Order Import Actions Interface Table.

Table 2–17 Example 3

For Orig_sys_line_ref	Operation Code	Fulfillment Set Name
1000	ADD_FULFILLMENT_SET	F3
1000	REMOVE_FULFILLMENT_SET	F1
2000	ADD_FULFILLMENT_SET	F4
2000	REMOVE_FULFILLMENT_SET	F2
2000	ADD_FULFILLMENT_SET	F5

After Order import the lines in the Order Lines table OE_ORDER_LINES_ALL

Table 2–18 Example 4

Line_Id	Fulfillment Set
1000	F3
2000	F3, F4, F5

Splits

Process order can be used to split an existing order line into shipments. User can perform splits by calling process order with both the line records, one to reduce the quantity on the existing line record with the action of SPLIT and another to create the new line with the remaining quantity and a reference to the line that it was split from. For e.g. if user wants to split a line with original quantity 10 into 6 and 4 he must populate the table in the following manner. In the first update call to process order user cannot update any other attributes other than ordered quantity.

```

line_tbl(1).split_action_code := 'SPLIT'
line_tbl(1).split_by := 'USER'
line_tbl(1).ordered_quantity := 6
line_tbl(1).operation = oe_globals.g_opr_update
line_tbl(2).split_action_code := 'SPLIT'
line_tbl(2).split_by := 'USER'
line_tbl(2).ordered_quantity := 4
line_tbl(2).operation = oe_globals.g_opr_create
line_tbl(2).split_from_line_id := line_tbl(1).line_id

```

User can also optionally pass in the change reason and change comments on the original line record that is being updated.

Cancellation

Process order can be used to cancel orders and order lines. User should update the cancelled flag to Y on the header record if user intends to cancel the entire order. Cancellation on the line is performed by reducing the ordered quantity on the line record. User has to supply change reason and optionally change comments on the line record.

Tax Calculation

Process order will check whether the transaction type on the order line is taxable or whether user has specified that tax should be calculated (set the tax exempt flag to Required). The calculated tax is always an estimated value and is internally stored as a price adjustment.

Freight and Special Charges

Freight and special charges can be setup in pricing as modifiers. When the order or line is priced, these charges get applied on the order or line. These do not affect the unit selling price and are also stored as price adjustments.

Users can also specify the freight and special charges to be applied by setting up the price adjustment records appropriately. Such price adjustments should have the `list_line_type_code` parameter set to *CHARGE*.

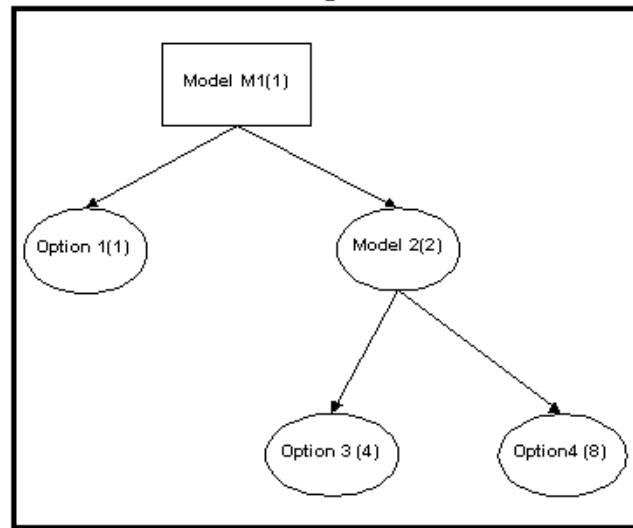
Cascading

Process Order will automatically cascade changes to other dependent entities. For e.g.

- cascade quantity changes from model lines to all options lines

- if a set identifying attribute is changed on one line of the set, the change will be automatically applied to all lines on the set.

Processing Sequence is very important when considering automatic cascading. Process Order will go by the following order when processing the line entity. First, all standard lines and model lines are processed. These are followed by processing of requests in their arrival order. The importance of declaring the processing sequence may be depicted by the following example. Consider that a model M1 has O1 and M2 as two of its options. M2 has O2 and O3 as two options defined for it. Graphically represented, it would look like this.

Figure 2–1 Simplified graphical representation of an ATO configuration

Let us assume that the following requests are received by Process Order API:

Request 1: Update the quantity of Option4 to 20.

Request 2: Update the quantity of Model2 to 4.

If requests are processed in the order that they were received, the following would occur:

Option 4 would be updated to 20 assuming that the new ratio between Model2 and Option4 (1:10) falls within a valid range. Next Model2 would be updated to 4 (assuming again that ratio of Model1 to Model2 is valid). That would then result in quantity cascading, thereby updating Option3 to 8 and Option4 to 40. So we end up with:

Model2 = 4, Option3 = 8, Option4 = 40.

If processing sequence of requests is based on item type (models first, classes next, options after that etc.), the following would be the sequence in which the requests would be processed:

Request 2 would be processed first, with quantity of Model2 to be updated to 4. Next quantity of Option4 would be updated to 20 (assuming ratios fall in the right range). So we would end up with:

Model2 = 4, Option3 = 8 and Option4 = 20.

The above example highlights the differences with processing lines (termed in the example as requests) in the sequential arrival order versus processing based on item types.

As stated earlier, process order API will process Standard Lines and top level Model lines first., then option classes and option lines and service lines in the end. For lines in the same category, the requests would be processed in the order that they are received.

Functional Overview

Public Procedures

Process Order API provides the following public procedures for operations and queries on the entities for the sales order object.

- `Process_Order`
- This is the main procedure used to create, update and delete the entities on the sales order. This procedure can also be used to perform other actions via the records or explicit action requests.

Please refer to the next section on the details of the processing.

- `Get_Order`
This procedure queries and returns all records belonging to a single sales order object.
- `Lock_Order`
Locks the entities of a sales order object. Use the `get_order` procedure to retrieve the entity records and then call `lock_order` with the records to be locked.

Processing the Sales Order business object

Process order can only process entities belonging to one sales order in one i.e. in a single call, it can accept one header record and a table of header adjustment, header sales credits, lines, line adjustments, line sales credit and line lot serial numbers records for processing. The entities are processed in the following sequence:

1. Process Header Record
2. Process Header Adjustments

3. Process Header Pricing Attributes
4. Process Header Adjustment Attributes
5. Process Header Adjustment Associations
6. Process Header Sales Credits
7. Check Entity context to make sure all lines belong to one header
8. Process Lines
9. Process Line Adjustments
10. Process Line Pricing Attributes
11. Process Line Adjustment Attributes
12. Process Line Adjustment Associations
13. Process Line Sales Credits
14. Process Line Lot Serial Numbers
15. Perform Cross Entity Logic for Sales Order business object

The procedure `process_order` performs the following actions for each entity on the order object:

Attribute Level Security Check If operation on the record is UPDATE or CREATE, then for all the attributes that have changed between the old and the new record, constraints are evaluated to check if the user is allowed to change the attribute. An error is raised if there was at least one attribute that failed security check and the record is not processed further.

Attribute Validation All the attributes that are passed in by the caller on the entity record are validated. Errors are posted to the OM message stack for all attributes that are invalid. An error is raised at the end if at least one attribute failed validation and the record is not processed further.

Clear Dependent Attributes If operation is UPDATE, then the fields dependent on the updated fields are cleared (or set to MISSING values). This is done so that the dependent fields are re-defaulted. For e.g. if the customer on the order is being updated, then the contacts, ship to and bill to on the order should be re-defaulted.

NOTE: If the user is also trying to update the dependent fields in the same call to the process order API, then the fields will not be cleared and the user-specified value will be used instead.

Defaulting Defaulting occurs for each missing attribute.

Attributes that are not explicitly passed by the user and therefore, retain the values on the initialized record are defined as MISSING attributes. For e.g. all the number fields on the entity records are initialized to the value FND_API.G_MISS_NUM and thus, all number fields with this value are MISSING attributes.

- Check security if new default value is different from old value and if there is a valid constraint, then an error is raised and abort processing of this record.
- Validate the default value if not null. If new default value is NOT valid, then set the attribute to NULL.
- If default value is valid, clear the dependent attributes (set to MISSING values).

Note: If the user has specified the values of these dependent attributes in the same call to process order, then the user-specified values will take preference and will not be cleared.

- Re-default all the missing attributes (dependent attributes) till all attributes have a value.

Record Validation

There are three main validation steps:

1. Check for Required Attributes. Validate that all the required attributes have been specified on the entity and even if one required attribute is missing, raise an error and quit. For e.g. inventory item is a required field on the order line entity.
2. Check for Conditionally Required Attributes. Validate that all attributes that are required based on the status of the entity have been specified and if at least one is not specified, raise an error and quit. For e.g. for a booked line, ship to location is required.
3. Cross-Attribute Validation. Validate all those attributes that are dependent on the value of another attribute. At the end of the validation, if at least one

attribute is not valid, then raise an error and quit. For e.g. verify that the ship to is valid for the customer.

Entity Security Check Entity level security check (constraints that are not attribute-specific) is done once again before the entity record is posted to database as defaulting may have changed the values of some attributes for e.g. ship set may have defaulted onto the line and there is a constraint against inserting a line into a ship set where all lines in the ship set have been invoiced.

Any constraints that may have been setup for DELETE operation on this entity are also evaluated here.

Database Write Write the changes to the database.

Cross Record Logic After all the records have been processed for one entity, other cross record logic like validations, automatic application of holds etc. is done. For e.g. after all records are processed for the entity *order sales credit*, validate that the total percent adds up to 100.

Cross Entity Logic After all the entities have been processed, cross entity logic is performed for the business object as a whole.

Note: In addition to the above a check is made to validate entity context for the line to ensure all lines belong to same header.

Start Processing Record

```

if operation          if operation
DELETE                CREATE or UPDATE
Attribute Security Check
error
Attribute Validation
if operation          if operation          error
CREATE                UPDATE
Clear Dependent Attributes
Set the error
status on the record
Defaulting
errorRollback all changes.
Exit with error status
Record Validation
error

```

```
Entity Security Check
Database Write
Process
Next
Record Record Processed
Successfully
error
All records processed
Cross Record Logic error
All entities processed
Process Action Requests
Exit with Success Status
```

Setting Up the Process Order Procedure

Before using the API, set up and activate the following parameters:

- Version number
- Initialize message list
- Initialize the IN record parameters to be interfaced to missing values
- Setup the parameters on the records to be interfaced

Parameter Descriptions

The following chart describes all parameters used by the public API OE_ORDER_PUB.PROCESS_ORDER. All of the inbound and outbound parameters are listed. The OUT parameters have to be assigned when calling the API and are therefore marked as Required fields. Additional information on these parameters follows.

Table 2–19 OE_ORDER_PUB.PROCESS_ORDER

Parameter	Usage	Type	Required	Optional	Description	Default
p_api_version_number	IN	NUMBER	X		Used to compare the incoming API call's version number with the current version number. An error is returned if the version numbers are incompatible	
p_init_msg_list	IN	Varchar2		X	Requests that the API initialize the message list on your behalf.	FND_API.G_FALSE
p_return_values	IN	Varchar2		X	Requests that the API send back the values on your behalf.	FND_API.G_FALSE
x_return_status	OUT	Varchar2	X		Returns the status, indicates whether the request was processed successfully or no Success: FND_API.G_RET_STS_SUCCESS Error: FND_API.G_RET_STS_ERROR Unexpected Error: FND_API.G_RET_STS_UNEXP_ERROR	
x_msg_count	OUT	NUMBER	X		Indicates number of error messages API has encountered. If the x_msg_count is greater than 1, then the list of messages must be retrieved using the call OE_MSG_PUB.GET. This api can be called with the following parameter values: p_msg_index => 1 p_encoded => F p_data => 1_message p_msg_index_out => 1_msg_index_out where 1_message and 1_msg_index_out should be local variables of types Varchar2(2000 and Number respectively	

Table 2–19 OE_ORDER_PUB.PROCESS_ORDER

Parameter	Usage	Type	Required	Optional	Description	Default
x_msg_data	OUT	Varchar2	X		Displays error message text. If the x_msg_count is equal to 1, then this contains the actual message	
p_header_rec	IN	PL/SQL Record		X	Use this parameter to send in operations on the order header entity	G_MISS_HEADER_REC
p_old_header_rec	IN	PL/SQL Record		X	Use this parameter to send in the old record for the order header entity.	G_MISS_HEADER_REC
p_header_val_rec	IN	PL/SQL Record		X	Use this parameter to send in the display values for fields on the order header entity.	G_MISS_HEADER_VAL_REC
p_old_header_val_rec	IN	PL/SQL Record		X	Use this parameter to send in display values for the fields on the old record for the order header entity.	G_MISS_HEADER_VAL_REC
p_header_adj_tbl	IN	PL/SQL Table		X	Use this parameter to send in operations on the order price adjustment entities	G_MISS_HEADER_ADJ_TBL
p_old_header_adj_tbl	IN	PL/SQL Table		X	Use this parameter to send in the old records for the order price adjustment entities.	G_MISS_HEADER_ADJ_TB
p_header_adj_val_tbl	IN	PL/SQL Table		X	Use this parameter to send in the display values for fields on the order price adjustment entities	G_MISS_HEADER_ADJ_VAL_TBL
p_old_header_adj_val_tbl	IN	PL/SQL Table		X	Use this parameter to send in display values for the fields on the old records for the order price adjustment entities.	G_MISS_HEADER_ADJ_VAL_TBL

Table 2–19 OE_ORDER_PUB.PROCESS_ORDER

Parameter	Usage	Type	Required	Optional	Description	Default
p_header_ price_att_ tbl	IN	PL/SQL Table		X	Use this parameter to send in operations on the order pricing Attributes entities	G_MISS_ HEADER_ _PRICE_ ATT_TBL
p_old_ header_ price_att_ tbl	IN	PL/SQL Table		X	Use this parameter to send in the old records for the order pricing Attributes entities	G_MISS_ HEADER_ _PRICE_ ATT_TBL
p_header_ adj_att_tbl	IN	PL/SQL Table		X	Use this parameter to send in operations on the order Adjustment Attributes entities.	G_MISS_ HEADER_ _ADJ_ ATT_TBL
p_old_ header_ adj_att_tbl	IN	PL/SQL Table		X	Use this parameter to send in the old records for the order Adjustment Attributes entities.	G_MISS_ HEADER_ _ADJ_ ATT_TBL
p_header_ adj_assoc_ tbl	IN	PL/SQL Table		X	Use this parameter to send in operations on the order Adjustment Associations entities.	G_MISS_ HEADER_ _ADJ_ ASSOC_ TBL
p_old_ header_ adj_assoc_ tbl	IN	PL/SQL Table		X	Use this parameter to send in the old records for the order Adjustment Associations entities.	G_MISS_ HEADER_ _ADJ_ ASSOC_ TBL
p_header_ scredit_tbl	IN	PL/SQL Table		X	Use this parameter to send in operations on the order sales credits.	G_MISS_ HEADER_ _SCREDIT_ _TBL
p_old_ header_ scredit_tbl	IN	PL/SQL Table		X	Use this parameter to send in the old record for the order sales credits.	G_MISS_ HEADER_ _SCREDIT_ _TBL

Table 2–19 OE_ORDER_PUB.PROCESS_ORDER

Parameter	Usage	Type	Required	Optional	Description	Default
p_header_scredit_val_tbl	IN	PL/SQL Table		X	Use this parameter to send in the display values for fields on the order sales credits.	G_MISS_HEADER_SCREREDIT_VAL_TBL
p_old_header_scredit_val_tbl	IN	PL/SQL Table		X	Use this parameter to send in display values for the fields on the old record for order sales credits.	G_MISS_HEADER_SCREREDIT_VAL_TBL
p_line_tbl	IN	PL/SQL Table		X	Use this parameter to send in operations on the order lines.	G_MISS_LINE_TBL
p_old_line_tbl	IN	PL/SQL Table		X	Use this parameter to send in the old records for the order lines.	G_MISS_LINE_TBL
p_line_val_tbl	IN	PL/SQL Table		X	Use this parameter to send in the display values for fields on the order lines.	G_MISS_LINE_VAL_TBL
p_old_line_val_tbl	IN	PL/SQL Table		X	Use this parameter to send in display values for the fields on the old records for order lines.	G_MISS_LINE_VAL_TBL
p_line_adj_tbl	IN	PL/SQL Table		X	Use this parameter to send in operations on the line price adjustment entities.	G_MISS_LINE_ADJ_TBL
p_old_line_adj_tbl	IN	PL/SQL Table		X	Use this parameter to send in the old records for the line price adjustment entities.	G_MISS_LINE_ADJ_TBL
p_line_adj_val_tbl	IN	PL/SQL Table		X	Use this parameter to send in the display values for fields on the line price adjustment entities.	G_MISS_LINE_ADJ_VAL_TBL

Table 2–19 OE_ORDER_PUB.PROCESS_ORDER

Parameter	Usage	Type	Required	Optional	Description	Default
p_old_line_adj_val_tbl	IN	PL/SQL Table		X	Use this parameter to send in display values for the fields on the old records for the line price adjustment entities.	G_MISS_LINE_ADJ_VAL_TBL
p_line_adj_tbl	IN	PL/SQL Table		X	Use this parameter to send in operations on the Line pricing Attributes entities.	G_MISS_LINE_PRICE_ATT_TBL
p_old_line_adj_tbl	IN	PL/SQL Table		X	Use this parameter to send in the old records for the Line pricing Attributes entities.	G_MISS_LINE_PRICE_ATT_TBL
p_line_adj_val_tbl	IN	PL/SQL Table		X	Use this parameter to send in operations on the Line Adjustment Attributes entities.	G_MISS_LINE_ADJ_ATT_TBL
p_old_line_adj_val_tbl	IN	PL/SQL Table		X	Use this parameter to send in the old records for the Line Adjustment Attributes entities.	G_MISS_LINE_ADJ_ATT_TBL
p_line_price_att_tbl	IN	PL/SQL Table		X		
p_old_line_price_att_tbl	IN	PL/SQL Table		X		
p_line_adj_att_tbl	IN	PL/SQL Table		X		
p_old_line_adj_att_tbl	IN	PL/SQL Table		X		
p_line_adj_assoc_tbl	IN	PL/SQL Table		X	Use this parameter to send in operations on the Line Adjustment Associations entities.	G_MISS_LINE_ADJ_ASSOC_TBL

Table 2–19 OE_ORDER_PUB.PROCESS_ORDER

Parameter	Usage	Type	Required	Optional	Description	Default
p_old_line_adj_assoc_tbl	IN	PL/SQL Table		X	Use this parameter to send in the old records for the Line Adjustment Associations entities.	G_MISS_LINE_ADJ_ASSOC_TBL
p_line_scredit_tbl	IN	PL/SQL Table		X	Use this parameter to send in operations on the line sales credits.	G_MISS_LINE_SCREDIT_TBL
p_old_line_scredit_tbl	IN	PL/SQL Table		X	Use this parameter to send in the old record for the line sales credits.	G_MISS_LINE_SCREDIT_TBL
p_line_scredit_val_tbl	IN	PL/SQL Table		X	Use this parameter to send in the display values for fields on the line sales credits	G_MISS_LINE_SCREDIT_VAL_TBL
p_old_line_scredit_val_tbl	IN	PL/SQL Table		X	Use this parameter to send in display values for the fields on the old record for line sales credits.	G_MISS_LINE_SCREDIT_VAL_TBL
p_lot_serial_tbl	IN	PL/SQL Table		X	Use this parameter to send in operations on the lot serials.	G_MISS_LOT_SERIAL_TBL
p_old_lot_serial_tbl	IN	PL/SQL Table		X	Use this parameter to send in the old record for the lot serials.	G_MISS_LOT_SERIAL_TBL
p_lot_serial_val_tbl	IN	PL/SQL Table		X	Use this parameter to send in the display values for fields on the lot serials.	G_MISS_LOT_SERIAL_VAL_TBL

Table 2–19 OE_ORDER_PUB.PROCESS_ORDER

Parameter	Usage	Type	Required	Optional	Description	Default
p_old_lot_serial_val_tbl	IN	PL/SQL Table		X	Use this parameter to send in display values for the fields on the old record for lot serials.	G_MISS_LOT_SERIAL_VAL_TBL
p_action_request_tbl	IN	PL/SQL Table		X	Use this to send in requests to process other actions on the order	G_MISS_REQUEST_TBL
x_header_rec	OUT	PL/SQL Table	X		Returns the processed order header record.	
x_header_val_rec	OUT	PL/SQL Table	X		Returns the values for the processed order header record for the sales order if p_return_values was set to FND_API.G_TRUE	
x_header_adj_tbl	OUT	PL/SQL Table	X		Returns the records for the processed price adjustments	
x_header_adj_val_tbl	OUT	PL/SQL Table	X		Returns the values for the processed header price adjustments if p_return_values was set to FND_API.G_TRUE	
x_header_price_att_tbl	OUT	PL/SQL Table	X		Returns the records for the processed header pricing attributes	
x_header_adj_att_tbl	OUT	PL/SQL Table	X		Returns the records for the processed header adjustment attributes	
x_header_adj_assoc_tbl	OUT	PL/SQL Table	X		Returns the records for the processed header adjustment associations	
x_header_scredit_tbl	OUT	PL/SQL Table	X		Returns the records for the processed header adjustment associations	
x_header_scredit_val_tbl	OUT	PL/SQL Table	X		Returns the values for the processed header sales credits if p_return_values was set to FND_API.G_TRUE	

Table 2–19 OE_ORDER_PUB.PROCESS_ORDER

Parameter	Usage	Type	Required	Optional	Description	Default
x_line_tbl	OUT	PL/SQL Table	X		Returns the processed order lines.	
x_line_val_tbl	OUT	PL/SQL Table	X		Returns the values for the processed order lines if p_return_values was set to FND_API.G_TRUE	
x_line_adj_tbl	OUT	PL/SQL Table	X		Returns the processed line price adjustments for the sales order	
x_line_adj_val_tbl	OUT	PL/SQL Table	X		Returns the values for the processed line price adjustments for the sales order if p_return_values was set to FND_API.G_TRUE	
x_line_price_att_tbl	OUT	PL/SQL Table	X		Returns the records for the processed line adjustment attributes	
x_line_adj_att_tbl	OUT	PL/SQL Table	X		Returns the records for the processed line adjustment associations	
x_line_adj_assoc_tbl	OUT	PL/SQL Table	X		Returns the records for the processed line adjustment associations	
x_line_scredit_tbl	OUT	PL/SQL Table	X		Returns the processed sales credits.	
x_line_scredit_val_tbl	OUT	PL/SQL Table	X		Returns the values for the processed sales credits for the sales order if p_return_values was set to FND_API.G_TRUE	

Table 2–19 OE_ORDER_PUB.PROCESS_ORDER

Parameter	Usage	Type	Required	Optional	Description	Default
x_lot_serial_tbl	OUT	PL/SQL Table	X		Returns the processed lot serials.	
x_lot_serial_val_tbl	OUT	PL/SQL Table	X		Returns the values for the processed lot serials if p_return_values was set to FND_API.G_TRUE	
x_action_request_tbl	OUT	PL/SQL Table	X		Returns the status for each of the action requests that were passed to process order.	

Setting Up the Get_Order Procedure

Before using the API, set up and activate the following parameters:

- Version number
- Initialize message list
- Pass the header ID of the order to be queried

Parameter Descriptions

The following chart describes all parameters used by the public API OE_ORDER_PUB.GET_ORDER. All of the inbound and outbound parameters are listed. Additional information on these parameters follows.

Table 2–20 OE_ORDER_PUB.GET_ORDER

Parameter	Usage	Type	Required	Optional	Description	Default
p_api_version_number	IN	NUMBER	X		Used to compare the incoming API call's version number with the current version number. An error is returned if the version numbers are incompatible.	
p_init_msg_list	IN	Varchar2		X	Requests that the API initialize the message list on your behalf.	FND_API.G_FALSE
p_return_values	IN	Varchar2		X	Requests that the API send back the values on your behalf.	FND_API.G_FALSE
p_commit	IN	Varchar2		X	Requests that the API update information for you after it completes its function.	FND_API.G_FALSE
x_return_status	OUT	Varchar2		X	Returns the status, indicates whether the request was processed successfully or not. Valid values include: <i>Success:</i> FND_API.G_RET_STS_SUCCESS <i>Error:</i> FND_API.G_RET_STS_ERROR <i>Unexpected Error:</i> FND_API.G_RET_STS_UNEXP_ERROR	

Table 2–20 OE_ORDER_PUB.GET_ORDER

Parameter	Usage	Type	Required	Optional	Description	Default
x_msg_count	OUT	NUMBER		X	Indicates number of error messages API has encountered. If the x_msg_count is greater than 1, then the list of messages must be retrieved using the call OE_MSG_PUB.GET. This api can be called with the following parameter values: p_msg_index => I p_encoded => F p_data => 1_message p_msg_index_out => 1_msg_index_out where 1_message and 1_msg_index_out should be local variables of types Varchar2(2000) and Number respectively.	
x_msg_data	OUT	Varchar2			Displays error message text. If the x_msg_count is equal to 1, then this contains the actual message.	
p_header_id	IN	NUMBER	X		Parameter to identify the sales order that is to be queried.	FND_API.G_MISS_NUM
p_header	IN	Varchar2		X		
x_header_rec	OUT	PL/SQL Table	X		Returns the queried order header record for the sales order.	
x_header_val_rec	OUT	PL/SQL Table	X		Returns the values for the queried order header record for the sales order if p_return_values was set to FND_API.G_TRUE	

Table 2–20 OE_ORDER_PUB.GET_ORDER

Parameter	Usage	Type	Required	Optional	Description	Default
x_header_adj_tbl	OUT	PL/SQL Table	X		Returns the queried header price adjustments for the sales order.	
x_header_adj_val_tbl	OUT	PL/SQL Table	X		Returns the values for the queried header price adjustments for the sales order if p_return_values was set to FND_API.G_TRUE	
x_header_price_att_tbl	OUT	PL/SQL Table	X			
x_header_adj_att_tbl	OUT	PL/SQL Table	X			
x_header_adj_assoc_tbl	OUT	PL/SQL Table	X			
x_header_scredit_tbl	OUT	PL/SQL Table	X		Returns the queried header sales credits for the sales order.	
x_header_scredit_val_tbl	OUT	PL/SQL Table	X		Returns the values for the queried header sales credits for the sales order if p_return_values was set to FND_API.G_TRUE	
x_line_tbl	OUT	PL/SQL Table	X		Returns the queried order lines for the sales order.	
x_line_val_tbl	OUT	PL/SQL Table	X		Returns the values for the queried order lines for the sales order if p_return_values was set to FND_API.G_TRUE	
x_line_adj_tbl	OUT	PL/SQL Table	X		Returns the queried line price adjustments for the sales order	

Table 2–20 OE_ORDER_PUB.GET_ORDER

Parameter	Usage	Type	Required	Optional	Description	Default
x_line_adj_val_tbl	OUT	PL/SQL Table	X		Returns the values for the queried line price adjustments for the sales order if p_return_values was set to FND_API.G_TRUE	
x_line_price_att_tbl	OUT	PL/SQL Table	X			
x_line_adj_att_tbl	OUT	PL/SQL Table	X			
x_line_adj_assoc_tbl	OUT	PL/SQL Table	X			
x_line_scredit_tbl	OUT	PL/SQL Table	X		Returns the queried line sales credits for the sales order.	
x_line_scredit_val_tbl	OUT	PL/SQL Table	X		Returns the values for the queried line sales credits for the sales order if p_return_values was set to FND_API.G_TRUE	
x_lot_serial_tbl	OUT	PL/SQL Table	X		Returns the queried lot serials for the sales order.	
x_lot_serial_val_tbl	OUT	PL/SQL Table	X		Returns the values for the queried lot serials for the sales order if p_return_values was set to FND_API.G_TRUE	

Setting Up the Lock_Order Procedure

Before using the API, set up and activate the following parameters:

- Version number
- Initialize message list
- Query all the entities for the sales order using the Get_Order procedure.

- Set the operation parameter to OE_GLOBALS.G_OPR_LOCK only on the entity records to be locked in this call.

Parameter Descriptions

The following chart describes all parameters used by the public API OE_ORDER_PUB.LOCK_ORDER. All of the inbound and outbound parameters are listed. The OUT parameters have to be assigned when calling the API and are therefore marked as Required fields. Additional information on these parameters follows.

Table 2-21 OE_ORDER_PUB.LOCK_ORDER

Parameter	Usage	Type	Required	Optional	Description	Default
p_api_version_number	IN	NUMBER	X		Used to compare the incoming API call's version number with the current version number. An error is returned if the version numbers are incompatible.	
p_init_msg_list	IN	Varchar2		X	Requests that the API initialize the message list on your behalf.	FND_API.G_FALSE
p_return_values	IN	Varchar2		X	Requests that the API send back the values on your behalf.	FND_API.G_FALSE
x_return_status	OUT	Varchar2	X		Returns the status, indicates whether the request was processed successfully or not. Valid values include: <i>Success:</i> FND_API.G_RET_STS_SUCCESS <i>Error:</i> FND_API.G_RET_STS_ERROR <i>Unexpected Error:</i> FND_API.G_RET_STS_UNEXP_ERROR	

Table 2-21 OE_ORDER_PUB.LOCK_ORDER

Parameter	Usage	Type	Required	Optional	Description	Default
x_msg_count	OUT	NUMBER	X		Indicates number of error messages API has encountered. If the x_msg_count is greater than 1, then the list of messages must be retrieved using the call OE_MSG_PUB.GET. This api can be called with the following parameter values: p_msg_index => I p_encoded => F p_data => 1_message p_msg_index_out => 1_msg_index_out where 1_message and 1_msg_index_out should be local variables of types Varchar2(2000) and Number respectively.	
x_msg_data	OUT	Varchar2	X		Displays error message text. If the x_msg_count is equal to 1, then this contains the actual message.	
p_header_rec	IN	PL/SQL Record		X	Use this parameter to send in operations on the order header entity.	G_MISS_HEADER_REC
p_header_val_rec	IN	PL/SQL Record		X	Use this parameter to send in the display values for fields on the order header entity.	G_MISS_HEADER_VAL_REC
p_header_adj_tbl	IN	PL/SQL Table		X	Use this parameter to send in operations on the order price adjustment entities.	G_MISS_HEADER_ADJ_TBL
p_header_adj_val_tbl	IN	PL/SQL Table		X	Use this parameter to send in the display values for fields on the order price adjustment entities.	G_MISS_HEADER_ADJ_VAL_TBL

Table 2–21 OE_ORDER_PUB.LOCK_ORDER

Parameter	Usage	Type	Required	Optional	Description	Default
p_header_price_att_tbl	IN	PL/SQL Table		X	Use this parameter to send in operations on the order pricing Attributes entities.	G_MISS_HEADER_PRICE_ATT_TBL
p_header_adj_att_tbl	IN	PL/SQL Table		X	Use this parameter to send in operations on the order Adjustment Attributes entities.	G_MISS_HEADER_ADJ_ATT_TBL
p_header_adj_assoc_tbl	IN	PL/SQL Table		X	Use this parameter to send in operations on the order Adjustment Associations entities.	G_MISS_HEADER_ADJ_ASSOC_TBL
p_header_scredit_tbl	IN	PL/SQL Table		X	Use this parameter to send in operations on the order sales credits.	G_MISS_HEADER_SCREDIT_TBL
p_header_scredit_val_tbl	IN	PL/SQL Table		X	Use this parameter to send in the display values for fields on the order sales credits.	G_MISS_HEADER_SCREDIT_VAL_TBL
p_line_tbl	IN	PL/SQL Table		X	Use this parameter to send in operations on the order lines.	G_MISS_LINE_TBL
p_line_val_tbl	IN	PL/SQL Table		X	Use this parameter to send in the display values for fields on the order lines.	G_MISS_LINE_VAL_TBL
p_line_adj_tbl	IN	PL/SQL Table		X	Use this parameter to send in operations on the line price adjustment entities.	G_MISS_LINE_ADJ_TBL

Table 2-21 OE_ORDER_PUB.LOCK_ORDER

Parameter	Usage	Type	Required	Optional	Description	Default
p_line_adj_val_tbl	IN	PL/SQL Table		X	Use this parameter to send in the display values for fields on the line price adjustment entities.	G_MISS_LINE_ADJ_VAL_TBL
p_line_adj_tbl	IN	PL/SQL Table		X	Use this parameter to send in operations on the Line pricing Attributes entities.	G_MISS_LINE_PRICE_ATT_TBL
p_line_adj_val_tbl	IN	PL/SQL Table		X	Use this parameter to send in operations on the Line Adjustment Attributes entities.	G_MISS_LINE_ADJ_ATT_TBL
p_line_adj_assoc_tbl	IN	PL/SQL Table		X	Use this parameter to send in operations on the Line Adjustment Associations entities.	G_MISS_LINE_ADJ_ASSOC_TBL
p_line_scredit_tbl	IN	PL/SQL Table		X	Use this parameter to send in operations on the line sales credits.	G_MISS_LINE_SCREDIT_TBL
p_line_scredit_val_tbl	IN	PL/SQL Table		X	Use this parameter to send in the display values for fields on the line sales credits.	G_MISS_LINE_SCREDIT_VAL_TBL
p_lot_serial_tbl	IN	PL/SQL Table		X	Use this parameter to send in operations on the lot serials.	G_MISS_LOT_SERIAL_TBL
p_lot_lot_serial_val_tbl	IN	PL/SQL Table		X	Use this parameter to send in the display values for fields on the lot serials.	G_MISS_LOT_SERIAL_VAL_TBL
x_header_rec	OUT	PL/SQL Table	X		Returns the processed order header record.	

Table 2-21 OE_ORDER_PUB.LOCK_ORDER

Parameter	Usage	Type	Required	Optional	Description	Default
x_header_val_rec	OUT	PL/SQL Table	X		Returns the values for the processed order header record for the sales order if p_return_values was set to FND_API.G_TRUE	
x_header_adj_tbl	OUT	PL/SQL Table	X		Returns the records for the processed price adjustments	
x_header_adj_val_tbl	OUT	PL/SQL Table	X		Returns the values for the processed header price adjustments if p_return_values was set to FND_API.G_TRUE	
x_header_price_att_tbl	OUT	PL/SQL Table	X		Returns the records for the processed header pricing attributes	
x_header_adj_att_tbl	OUT	PL/SQL Table	X		Returns the records for the processed header adjustment attributes	
x_header_adj_assoc_tbl	OUT	PL/SQL Table	X		Returns the records for the processed header adjustment associations	
x_header_scredit_tbl	OUT	PL/SQL Table	X		Returns the records for the processed header adjustment associations	
x_header_scredit_val_tbl	OUT	PL/SQL Table	X		Returns the values for the processed header sales credits if p_return_values was set to FND_API.G_TRUE	
x_line_tbl	OUT	PL/SQL Table	X		Returns the processed order lines.	
x_line_val_tbl	OUT	PL/SQL Table	X		Returns the values for the processed order lines if p_return_values was set to FND_API.G_TRUE	

Table 2–21 OE_ORDER_PUB.LOCK_ORDER

Parameter	Usage	Type	Required	Optional	Description	Default
x_line_adj_tbl	OUT	PL/SQL Table	X		Returns the processed line price adjustments for the sales order	
x_line_adj_val_tbl	OUT	PL/SQL Table	X		Returns the values for the processed line price adjustments for the sales order if p_return_values was set to FND_API.G_TRUE	
x_line_price_att_tbl	OUT	PL/SQL Table	X		Returns the records for the processed line adjustment attributes	
x_line_adj_att_tbl	OUT	PL/SQL Table	X			
x_line_adj_assoc_tbl	OUT	PL/SQL Table	X		Returns the records for the processed line adjustment associations	
x_line_scredit_tbl	OUT	PL/SQL Table	X		Returns the processed sales credits.	
x_line_scredit_val_tbl	OUT	PL/SQL Table	X	X	Returns the values for the processed sales credits for the sales order if p_return_values was set to FND_API.G_TRUE	
x_lot_serial_tbl	OUT	PL/SQL Table	X		Returns the processed lot serials.	
x_lot_serial_val_tbl	OUT	PL/SQL Table	X		Returns the values for the processed lot serials if p_return_values was set to FND_API.G_TRUE	

PL/SQL Record Structures

All the columns on the underlying order management tables are available on the record types that can be passed to the process order API. However, callers cannot provide values for certain columns via the public process order API. Such columns

are marked as *Derived* columns as these are derived internally and updated by the API. If user provides values for these fields, it could result in data corruption.

For each column (if the record type is not a value record type, val_rec_type), the following information has been documented:

- **Datatype:** data type for this field
- **Required:** *X* if required at entry, *C* if conditionally required
- **Required at Booking:** *X* if required at booking, *C* if required at booking based on some other conditions.
- **Derived:** For internal use, users cannot update these fields.
- **Optional:** *X* if users can either provide values for these fields or could be defaulted.

The value record structures have all fields that can be optionally passed instead of the identifier. If the identifier field is not passed when required, the value field can be passed instead and the value will be resolved into the identifier.

These record and table structures are defined in the specifications of the process order API, OE_ORDER_PUB.

Header_Rec_Type

For column descriptions, please refer to the Order Management TRM for the table OE_ORDER_HEADERS_ALL.

- Please note that columns attribute1 - attribute15 are currently defined as VARCHAR2(240), but only the first 150 characters are passed to Oracle Receivables during Invoice Interface. These columns contain Additional Header Information Descriptive Flexfield data

Table 2–22 OE_ORDER_HEADERS_ALL

Parameter	Datatype	Required	Required at Booking	Derived	Optional
accounting_rule_id	NUMBER				X
agreement_id	NUMBER		C - if order type requires		X
attribute1.... attribute20	VARCHAR2(240)				X
booked_flag	VARCHAR2(1)			X	

Table 2–22 OE_ORDER_HEADERS_ALL

Parameter	Datatype	Required	Required at Booking	Derived	Optional
cancelled_flag	VARCHAR2(1)				X
context	VARCHAR2(30)				X
conversion_rate	NUMBER		C - if conversion type is <i>User</i>		X
conversion_rate_date	DATE		C - if conversion type is <i>User</i>		X
conversion_type_code	VARCHAR2(30)		C - if order currency not same as SOB currency		X
customer_preference_set_code	VARCHAR2(30)				X
created_by	NUMBER			X	
creation_date	DATE			X	
cust_po_number	VARCHAR2(50)		C - if order type requires		X
deliver_to_contact_id	NUMBER				X
deliver_to_org_id	NUMBER				X
demand_class_code	VARCHAR2(30)				X
earliest_schedule_limit	NUMBER			X	
expiration_date	DATE			X	
fob_point_code	VARCHAR2(30)				X
freight_carrier_code	VARCHAR2(30)			X	
freight_terms_code	VARCHAR2(30)				X
global_attribute1...20	VARCHAR2(240)				X
global_attribute_category	VARCHAR2(30)				X
tp_context	VARCHAR2(30)				X

Table 2–22 OE_ORDER_HEADERS_ALL

Parameter	Datatype	Required	Required at Booking	Derived	Optional
tp_attribute1.... tp_attribute20	VARCHAR2(240)				X
header_id	NUMBER	C - for UPDATE & DELETE operations			X
invoice_to_contact_id	NUMBER				X
invoice_to_org_id	NUMBER		X		
invoicing_rule_id	NUMBER				X
last_updated_by	NUMBER			X	
last_update_date	DATE			X	
last_update_login	NUMBER			X	
latest_schedule_limit	NUMBER				X
open_flag	VARCHAR2(1)			X	
order_category_code	VARCHAR2(30)			X	
ordered_date	DATE		X		X
order_date_type_code	VARCHAR2(30)				X
order_number	NUMBER	C - for manual order types			X
order_source_id	NUMBER				X
order_type_id	NUMBER	X			
org_id	NUMBER			X	
orig_sys_document_ref	VARCHAR2(50)				X
partial_shipments_allowed	VARCHAR2(1)			X	
payment_term_id	NUMBER		C - for order lines, not return lines		X
price_list_id	NUMBER		X		X
pricing_date	DATE	X			X

Table 2–22 OE_ORDER_HEADERS_ALL

Parameter	Datatype	Required	Required at Booking	Derived	Optional
program_application_id	NUMBER				X
program_id	NUMBER				X
program_update_date	DATE				X
request_date	DATE				X
request_id	NUMBER				X
return_reason_code	VARCHAR2(30)				X
salesrep_id	NUMBER		X		X
sales_channel_code	VARCHAR2(30)				X
shipment_priority_code	VARCHAR2(30)				X
shipping_method_code	VARCHAR2(30)				X
ship_from_org_id	NUMBER		C - for return lines		
ship_tolerance_above	NUMBER				X
ship_tolerance_below	NUMBER				X
ship_to_contact_id	NUMBER				X
ship_to_org_id	NUMBER		C - for order lines, not return lines		
sold_from_org_id	NUMBER		X		
sold_to_contact_id	NUMBER				X
sold_to_org_id	NUMBER		X		X
source_document_id	NUMBER				X
source_document_type_id	NUMBER				X
tax_exempt_flag	VARCHAR2(30)		X		X
tax_exempt_number	VARCHAR2(50)				X
tax_exempt_reason_code	VARCHAR2(30)	C - if tax exempt flag is <i>Exempt</i>			X

Table 2–22 OE_ORDER_HEADERS_ALL

Parameter	Datatype	Required	Required at Booking	Derived	Optional
tax_point_code	VARCHAR2(30)				for future use
transactional_curr_code	VARCHAR2(3)	X			
version_number	NUMBER			X	
return_status	VARCHAR2(1)			X	
db_flag	VARCHAR2(1)			X	
operation	VARCHAR2(30)	X			
first_ack_code	VARCHAR2(30)			X	
first_ack_date	DATE			X	
last_ack_code	VARCHAR2(30)			X	
last_ack_date	DATE			X	
change_reason	VARCHAR2(30)	C -if constraints setup requires reason			X
change_comments	VARCHAR2(2000)				X
change_sequence	VARCHAR2(50)			X	
change_request_code	VARCHAR2(30)			X	
ready_flag	VARCHAR2(1)			X	
status_flag	VARCHAR2(1)			X	
force_apply_flag	VARCHAR2(1)			X	
drop_ship_flag	VARCHAR2(1)			X	
customer_payment_term_id	Obsolete	Obsolete	Obsolete	Obsolete	Obsolete
payment_type_code	VARCHAR2(30)				X
payment_amount	NUMBER		C - if payment type other than <i>Credit Card</i>		
check_number	VARCHAR2(50)		C - if payment type is <i>Check</i>		

Table 2–22 OE_ORDER_HEADERS_ALL

Parameter	Datatype	Required	Required at Booking	Derived	Optional
credit_card_code	VARCHAR2(80)				X
credit_card_holder_name	VARCHAR2(80)		C - if payment type is <i>Credit Card</i>		
credit_card_number	VARCHAR2(80)		C - if payment type is <i>Credit Card</i>		
credit_card_expiration_date	DATE		C - if payment type is <i>Credit Card</i>		
credit_card_approval_code	VARCHAR2(80)			X	X
credit_card_approval_date	DATE			X	
shipping_instructions	VARCHAR2(2000)				X
packing_instructions	VARCHAR2(2000)				X
flow_status_code	VARCHAR2(30)			X	
booked_date	DATE			X	
marketing_source_code_id	NUMBER				X
xml_message_id	NUMBER				X
xml_transaction_type_code	VARCHAR2(30)				X

TYPE Header_Tbl_Type IS TABLE OF Header_Rec_Type
INDEX BY BINARY_INTEGER;

Header_Rec_Type**Table 2–23 Header_Rec_Type**

Parameter	Value
ship_to_customer_id	NUMBER
invoice_to_customer_id	NUMBER
deliver_to_customer_id	NUMBER
IB_OWNER	VARCHAR2(60)
IB_INSTALLED_AT_LOCATION	VARCHAR2(60)
IB_CURRENT_LOCATION	VARCHAR2(60)
END_CUSTOMER_ID	NUMBER
END_CUSTOMER_CONTACT_ID	NUMBER
END_CUSTOMER_SITE_USE_ID	NUMBER
sold_to_party_id	NUMBER
sold_to_org_contact_id	NUMBER
ship_to_party_id	NUMBER
ship_to_party_site_id	NUMBER
ship_to_party_site_use_id	NUMBER
deliver_to_party_id	NUMBER
deliver_to_party_site_id	NUMBER
deliver_to_party_site_use_id	NUMBER
invoice_to_party_id	NUMBER
invoice_to_party_site_id	NUMBER
invoice_to_party_site_use_id	NUMBER
ship_to_customer_party_id	NUMBER
deliver_to_customer_party_id	NUMBER
invoice_to_customer_party_id	NUMBER

Table 2–23 Header_Rec_Type

Parameter	Value
ship_to_customer_id	NUMBER
ship_to_org_contact_id	NUMBER
deliver_to_org_contact_id	NUMBER
invoice_to_org_contact_id	NUMBER
sold_to_party_number	varchar2(30)
ship_to_party_number	varchar2(30)
invoice_to_party_number	varchar2(30)
deliver_to_party_number	varchar2(30)

Header_Val_Rec_Type**Table 2–24 Header_Val_Rec_Type**

Parameter	accounting_rule
agreement	
conversion_type	
deliver_to_address1	
deliver_to_address2	
deliver_to_address3	
deliver_to_address4	
deliver_to_contact	
deliver_to_location	
deliver_to_org	
demand_class	
fob_point	
freight_terms	
invoice_to_address1	
invoice_to_address2	
invoice_to_address3	

Table 2–24 Header_Val_Rec_Type

Parameter accounting_rule
invoice_to_address4
invoice_to_contact
invoice_to_location
invoice_to_org
invoicing_rule
order_source
order_type
payment_term
price_list
return_reason
salesrep
shipment_priority
ship_from_address1
ship_from_address2
ship_from_address3
ship_from_address4
ship_from_location
ship_from_org
ship_to_address1
ship_to_address2
ship_to_address3
ship_to_address4
ship_to_contact
ship_to_location
ship_to_org
sold_to_contact
sold_to_org

Table 2–24 Header_Val_Rec_Type

Parameter accounting_rule	
sold_from_org	
tax_exempt	
tax_exempt_reason	
tax_point	
customer_payment_term - <i>obsolete</i>	
payment_type	
credit_card	
status	
freight_carrier	
shipping_method	
order_date_type	
customer_number	
sales_channel	
ship_to_customer_name	VARCHAR2(360)
invoice_to_customer_name	VARCHAR2(360)
ship_to_customer_number	VARCHAR2(50)
invoice_to_customer_number	VARCHAR2(50)
deliver_to_customer_number	VARCHAR2(50)
deliver_to_customer_name	VARCHAR2(360)
end_customer_name	VARCHAR2(360)
end_customer_number	VARCHAR2(50)
end_customer_contact	VARCHAR2(360)
end_cust_contact_last_name	VARCHAR2(240)
end_cust_contact_first_name	VARCHAR2(240)
end_customer_site_address1	VARCHAR2(240)
end_customer_site_address2	VARCHAR2(240)
end_customer_site_address3	VARCHAR2(240)

Table 2–24 Header_Val_Rec_Type

Parameter accounting_rule	
end_customer_site_address4	VARCHAR2(240)
end_customer_site_state	VARCHAR2(240)
end_customer_site_country	VARCHAR2(240)
end_customer_site_location	VARCHAR2(240)
end_customer_site_zip	VARCHAR2(240)
end_customer_site_county	VARCHAR2(240)
end_customer_site_province	VARCHAR2(240)
end_customer_site_city	VARCHAR2(240)
end_customer_site_postal_code	VARCHAR2(240)
ib_owner_dsp	VARCHAR2(60)
ib_installed_at_location_dsp	VARCHAR2(60)
ib_current_location_dsp	VARCHAR2(60)

TYPE Header_Val_Tbl_Type IS TABLE OF Header_Val_Rec_Type
INDEX BY BINARY_INTEGER;

Header_Adj_Rec_Type

For column descriptions, please refer to the Order Management TRM for the table
OE_PRICE_ADJUSTMENTS

Table 2–25 OE_PRICE_ADJUSTMENTS

Parameter	Datatype	Required	Derived	Optional
attribute1 - attribute15	VARCHAR2(240)			
automatic_flag	VARCHAR2(1)	X	qp_list_lines	
context	VARCHAR2(30)			X
created_by	NUMBER		X	
creation_date	DATE		X	
discount_id	NUMBER		X	
discount_line_id	NUMBER		X	

Table 2–25 OE_PRICE_ADJUSTMENTS

Parameter	Datatype	Required	Derived	Optional
header_id	NUMBER	X		
last_updated_by	NUMBER		X	
last_update_date	DATE		X	
last_update_login	NUMBER		X	
percent	NUMBER		X	
price_adjustment_id	NUMBER	C -for update and delete		X
program_application_id	NUMBER			X
program_id	NUMBER			X
program_update_date	DATE			X
request_id	NUMBER			X
return_status	VARCHAR2(1)		X	
db_flag	VARCHAR2(1)		X	
operation	VARCHAR2(30)	X		
line_index	NUMBER			X
orig_sys_discount_ref	VARCHAR2(50)			X
change_request_code	VARCHAR2(30)		X	
status_flag	VARCHAR2(1)		X	
list_header_id	NUMBER	C - not required only for tax records		
list_line_id	NUMBER	C - not required only for tax records		
list_line_type_code	VARCHAR2(30)		X	
modifier_mechanism_type_code	VARCHAR2(30)		X	
modified_from	NUMBER		X	
modified_to	NUMBER		X	
updated_flag	VARCHAR2(1)		X	
update_allowed	VARCHAR2(1)		X	

Table 2–25 OE_PRICE_ADJUSTMENTS

Parameter	Datatype	Required	Derived	Optional
applied_flag	VARCHAR2(1)			
change_reason_code	VARCHAR2(30)			X
change_reason_text	VARCHAR2(2000)			X
operand	NUMBER			X
operand_per_pqty	NUMBER			X
arithmetic_operator	VARCHAR2(30)		X	
cost_id	NUMBER		X	
tax_code	VARCHAR2(30)			
tax_exempt_flag	VARCHAR2(1)			
tax_exempt_number	VARCHAR2(80)			
tax_exempt_reason_code	VARCHAR2(30)			
parent_adjustment_id	NUMBER	X -for update and delete		
invoiced_flag	VARCHAR2(1)		X	
estimated_flag	VARCHAR2(1)		X	
inc_in_sales_performance	VARCHAR2(1)		X	
split_action_code	VARCHAR2(30)			X
adjusted_amount	NUMBER		X	
adjusted_amount_per_pqty	NUMBER		X	
pricing_phase_id	NUMBER		X	
charge_type_code	VARCHAR2(30)		X	
charge_subtype_code	VARCHAR2(30)		X	
list_line_no	VARCHAR2(240)		X	
source_system_code	VARCHAR2(30)		X	
benefit_qty	NUMBER		X	
benefit_uom_code	VARCHAR2(3)		X	
print_on_invoice_flag	VARCHAR2(1)		X	
expiration_date	DATE		X	

Table 2-25 OE_PRICE_ADJUSTMENTS

Parameter	Datatype	Required	Derived	Optional
rebate_transaction_type_code	VARCHAR2(30)		X	
rebate_transaction_reference	VARCHAR2(80)		X	
rebate_payment_system_code	VARCHAR2(30)		X	
redeemed_date	DATE		X	
redeemed_flag	VARCHAR2(1)		X	
accrual_flag	VARCHAR2(1)		X	
range_break_quantity	NUMBER		X	
accrual_conversion_rate	NUMBER		X	
pricing_group_sequence	NUMBER		X	
modifier_level_code	VARCHAR2(30)		X	
price_break_type_code	VARCHAR2(30)		X	
substitution_attribute	VARCHAR2(30)		X	
proration_type_code	VARCHAR2(30)		X	
credit_or_charge_flag	VARCHAR2(1)		X	
include_on_returns_flag	VARCHAR2(1)		X	
ac_attribute1... ac_attribute15	VARCHAR2(240)			X
ac_context	VARCHAR2(150)			X

TYPE Header_Adj_Tbl_Type IS TABLE OF Header_Adj_Rec_Type
INDEX BY BINARY_INTEGER;

Header_Adj_Val_Rec_Type

Table 2–26 Header_Adj_Val_Rec_Type

Header_Adj_Val_Rec_Type	
Parameter	
discount	
list_name	

TYPE Header_Adj_Val_Tbl_Type IS TABLE OF Header_Adj_Val_Rec_Type
INDEX BY BINARY_INTEGER;

Header_Price_Att_Rec_Type

For column descriptions, please refer to the Order Management TRM for the table
OE_ORDER_PRICE_ATTRIBS

Table 2–27 OE_ORDER_PRICE_ATTRIBS

Parameter	Datatype	Required	Derived	Optional
order_price_attrib_id	NUMBER	X -for update and delete		X
header_id	NUMBER	X		
creation_date	DATE		X	
created_by	NUMBER		X	
last_update_date	DATE		X	
last_updated_by	NUMBER		X	
last_update_login	NUMBER		X	
program_application_id	NUMBER			X
program_id	NUMBER			X
program_update_date	DATE			X
request_id	NUMBER			X
flex_title	VARCHAR2(60)	X		
pricing_context	VARCHAR2(30)	X		

Table 2–27 OE_ORDER_PRICE_ATTRIBS

Parameter	Datatype	Required	Derived	Optional
pricing_attribute1... pricing_attribute100	VARCHAR2(240)	at least one of these columns should have a value		X
context	VARCHAR2(30)			X
attribute1... attribute15	VARCHAR2(240)			X
Override_Flag	VARCHAR2(1)			X
return_status	VARCHAR2(1)		X	
db_flag	VARCHAR2(1)		X	
operation	VARCHAR2(30)	X		

TYPE Header_Price_Att_Tbl_Type is TABLE of Header_Price_Att_rec_Type
INDEX by BINARY_INTEGER

Header_Adj_Att_Rec_Type

For column descriptions, please refer to the Order Management TRM for the table OE_PRICE_ADJ_ATTRIBS.

Table 2–28 OE_PRICE_ADJ_ATTRIBS

Parameter	Datatype	Required	Derived	Optional
price_adj_attrib_id	NUMBER	C -for update and delete		
price_adjustment_id	NUMBER	C - if adj_index not passed		
cdj_index	NUMBER	C - if price_adjustment_id not passed		
flex_title	VARCHAR2(60)	X		
pricing_context	VARCHAR2(30)	X		
pricing_attribute	VARCHAR2(30)	X		
creation_date	DATE		X	
created_by	NUMBER		X	
last_update_date	DATE		X	

Table 2–28 OE_PRICE_ADJ_ATTRIBS

Parameter	Datatype	Required	Derived	Optional
last_updated_by	NUMBER		X	
last_update_login	NUMBER		X	
program_application_id	NUMBER			X
program_id	NUMBER			X
program_update_date	DATE			X
request_id	NUMBER			X
pricing_attr_value_from	VARCHAR2(240)	X		
pricing_attr_value_to	VARCHAR2(240)			X
comparison_operator	VARCHAR2(30)			X
return_status	VARCHAR2(1)		X	
db_flag	VARCHAR2(1)		X	
operation	VARCHAR2(30)			

TYPE Header_Adj_Att_Rec_Type is TABLE of Header_Adj_Att_rec_Type
INDEX by BINARY_INTEGER;

Type Header_Adj_Assoc_Rec_Type is RECORD

For column descriptions, please refer to the Order Management TRM for the table:

OE_PRICE_ADJ ASSOCS

Table 2–29 OE_PRICE_ADJ ASSOCS

Parameter	Datatype	Required	Derived	Optional
price_adj_assoc_id	NUMBER	C - for update and delete		
line_id	NUMBER			
Line_Index	NUMBER			
price_adjustment_id	NUMBER	C - if adj_index not passed		

Table 2-29 OE_PRICE_ADJ ASSOCS

Parameter	Datatype	Required	Derived	Optional
Adj_index	NUMBER	C - if price_ adjustment_id not passed		
rltd_Price_Adj_Id	NUMBER			X
Rltd_Adj_Index	NUMBER			X
creation_date	DATE		X	
created_by	NUMBER		X	
last_update_date	DATE		X	
last_updated_by	NUMBER		X	
last_update_login	NUMBER		X	
program_application_id	NUMBER			X
program_id	NUMBER			X
program_update_date	DATE			X
request_id	NUMBER			X
return_status	VARCHAR2(1)		X	
db_flag	VARCHAR2(1)		X	
operation	VARCHAR2(3 0)			X

TYPE Header_Adj_Assoc_Tbl_Type is TABLE of Header_Adj_Assoc_rec_Type
INDEX by BINARY_INTEGER;

TYPE Header_Scredit_Rec_Type IS RECORD

For column descriptions, please refer to the Order Management TRM for the table:

OE_SALES_CREDITS**Table 2–30 OE_SALES_CREDITS**

Parameter	Datatype	Required	Derived	Optional
attribute1... attribute15	VARCHAR2(240)			X
context	VARCHAR2(30)			X
created_by	NUMBER		X	
creation_date	DATE		X	
dw_update_advice_ flag	VARCHAR2(1)			X
header_id	NUMBER			X
last_updated_by	NUMBER		X	
last_update_date	DATE		X	
last_update_login	NUMBER		X	
percent	NUMBER	X		
salesrep_id	NUMBER	X		
sales_credit_type_id	NUMBER	X		
sales_credit_id	NUMBER			X
wh_update_date	DATE			X
return_status	VARCHAR2(1)		X	
db_flag	VARCHAR2(1)		X	
operation	VARCHAR2(30)	X		
orig_sys_credit_ref	VARCHAR2(50)			X
change_request_code	VARCHAR2(30)		X	
status_flag	VARCHAR2(1)		X	

TYPE Header_Scredit_Tbl_Type IS TABLE OF Header_Scredit_Rec_Type

INDEX BY BINARY_INTEGER;

TYPE Header_Scredit_Val_Rec_Type IS RECORD

(salesrep, sales_credit_type);

TYPE Header_Scredit_Val_Tbl_Type IS TABLE OF Header_Scredit_Val_Rec_Type
INDEX BY BINARY_INTEGER;

Line_Rec_Type

For column descriptions, please refer to the Order Management TRM for the table
OE_ORDER_LINES_ALL

- Please note that columns attribute1 - attribute15 are currently defined as VARCHAR2(240), but only the first 150 characters are passed to Oracle Receivables during Invoice Interface. These columns contain Additional Order Information Descriptive Flexfield data

Table 2-31 OE_ORDER_LINES_ALL

Parameter	Datatype	Required	Required at Booking	Derived	Optional
accounting_rule_id	NUMBER				X
actual_arrival_date	DATE			X	
actual_shipment_date	DATE			X	
agreement_id	NUMBER				X
arrival_set_id	NUMBER				X
ato_line_id	NUMBER			X	
attribute1...attribute20	VARCHAR2(240)				X
authorized_to_ship_flag	VARCHAR2(1)				X
auto_selected_quantity	NUMBER			X	
booked_flag	VARCHAR2(1)			X	
cancelled_flag	VARCHAR2(1)			X	
cancelled_quantity	NUMBER			X	
cancelled_quantity2	NUMBER			X	
commitment_id	NUMBER				X
component_code	VARCHAR2(1000)				X
component_number	NUMBER			X	
component_sequence_id	NUMBER			X	

Table 2–31 OE_ORDER_LINES_ALL

Parameter	Datatype	Required	Required at Booking	Derived	Optional
config_header_id	NUMBER			X	X
config_rev_nbr	NUMBER			X	X
config_display_sequence	NUMBER			X	
configuration_id	NUMBER			X	X
context	VARCHAR2(30)				X
created_by	NUMBER			X	
creation_date	DATE			X	
credit_invoice_line_id	NUMBER			X	
customer_dock_code	VARCHAR2(50)				X
customer_job	VARCHAR2(50)				X
customer_production_line	VARCHAR2(50)				X
customer_trx_line_id	NUMBER			X	
cust_model_serial_number	VARCHAR2(50)				X
cust_po_number	VARCHAR2(50)				X
cust_production_seq_num	VARCHAR2(50)				X
delivery_lead_time	NUMBER				X
deliver_to_contact_id	NUMBER				X
deliver_to_org_id	NUMBER				X
demand_bucket_type_code	VARCHAR2(30)				X
demand_class_code	VARCHAR2(30)				X
dep_plan_required_flag	VARCHAR2(1)				X
earliest_acceptable_date	DATE			X	
end_item_unit_number	VARCHAR2(30)	C - depends on the item			X
explosion_date	DATE			X	
fob_point_code	VARCHAR2(30)				X

Table 2–31 OE_ORDER_LINES_ALL

Parameter	Datatype	Required	Required at Booking	Derived	Optional
freight_carrier_code	VARCHAR2(30)			X	
freight_terms_code	VARCHAR2(30)				X
fulfilled_quantity	NUMBER			X	
fulfilled_quantity2	NUMBER			X	
global_attribute1...20	VARCHAR2(240)				X
global_attribute_category	VARCHAR2(30)				X
header_id	NUMBER				X
industry_attribute1...30	VARCHAR2(240)				X
industry_context	VARCHAR2(30)				X
tp_context	VARCHAR2(30)				X
tp_attribute1...20	VARCHAR2(240)				X
intermed_ship_to_org_id	NUMBER				X
intermed_ship_to_contact_id	NUMBER				X
inventory_item_id	NUMBER	X			
invoice_interface_status_code	VARCHAR2(30)			X	X
invoice_to_contact_id	NUMBER				X
invoice_to_org_id	NUMBER		X		X
invoicing_rule_id	NUMBER				X
ordered_item	VARCHAR2(2000)	C: for Generic Items			
item_revision	VARCHAR2(3)				X
item_type_code	VARCHAR2(30)			X	
last_updated_by	NUMBER			X	
last_update_date	DATE			X	
last_update_login	NUMBER			X	
latest_acceptable_date	DATE				X

Table 2–31 OE_ORDER_LINES_ALL

Parameter	Datatype	Required	Required at Booking	Derived	Optional
line_category_code	VARCHAR2(30)			X	X
line_id	NUMBER	C - for UPDATE & DELETE operations			
line_number	NUMBER				X
line_type_id	NUMBER	X			X
link_to_line_ref	VARCHAR2(50)				X
link_to_line_id	NUMBER			X	
link_to_line_index	NUMBER				X
model_group_number	NUMBER			X	
mfg_component_sequence_id	NUMBER			X	
open_flag	VARCHAR2(1)			X	
option_flag	VARCHAR2(1)			X	
option_number	NUMBER			X	
ordered_quantity	NUMBER		X		X
ordered_quantity2	NUMBER				X
order_quantity_uom	VARCHAR2(3)		X		X
ordered_quantity_uom2	VARCHAR2(3)			X	
org_id	NUMBER			X	
orig_sys_document_ref	VARCHAR2(50)				X
orig_sys_line_ref	VARCHAR2(50)				X
over_ship_reason_code	VARCHAR2(30)			X	
over_ship_resolved_flag	VARCHAR2(1)				X
payment_term_id	NUMBER		C- for order lines, not return lines		X
planning_priority	NUMBER			X	
preferred_grade	VARCHAR2(4)				X

Table 2-31 OE_ORDER_LINES_ALL

Parameter	Datatype	Required	Required at Booking	Derived	Optional
price_list_id	NUMBER		X		
pricing_attribute1... pricing_attribute10	VARCHAR2(240)			X	
pricing_context	VARCHAR2(240)			X	
pricing_date	DATE				X
pricing_quantity	NUMBER			X	
pricing_quantity_uom	VARCHAR2(3)			X	
program_application_id	NUMBER				X
program_id	NUMBER				X
program_update_date	DATE				X
project_id	NUMBER				X
promise_date	DATE				X
re_source_flag	VARCHAR2(1)			X	
reference_customer_trx_ line_id	NUMBER			X	
reference_header_id	NUMBER			X	
reference_line_id	NUMBER			X	
reference_type	VARCHAR2(30)			X	
request_date	DATE				X
request_id	NUMBER				X
reserved_quantity	NUMBER				X
return_attribute1... return_attribute20	VARCHAR2(240)				X
return_context	VARCHAR2(30)				X
return_reason_code	VARCHAR2(30)	C - for return lines only			
rla_schedule_type_code	VARCHAR2(30)				X

Table 2–31 OE_ORDER_LINES_ALL

Parameter	Datatype	Required	Required at Booking	Derived	Optional
salesrep_id	NUMBER				X
schedule_arrival_date	DATE				X
schedule_ship_date	DATE				X
schedule_action_code	VARCHAR2(30)				X
schedule_status_code	VARCHAR2(30)			X	
shipment_number	NUMBER			X	
shipment_priority_code	VARCHAR2(30)				X
shipped_quantity	NUMBER			X	
shipped_quantity2	NUMBER			X	
shipping_interfaced_flag	VARCHAR2(1)			X	
shipping_method_code	VARCHAR2(30)				X
shipping_quantity	NUMBER			X	
shipping_quantity2	NUMBER			X	
shipping_quantity_uom	VARCHAR2(3)			X	
shipping_quantity_uom2	VARCHAR2(3)			X	
ship_from_org_id	NUMBER		C - for return lines		X
ship_model_complete_flag	VARCHAR2(30)			X	
ship_set_id	NUMBER				X
fulfillment_set_id	NUMBER				X
ship_tolerance_above	NUMBER				X
ship_tolerance_below	NUMBER				X
ship_to_contact_id	NUMBER				X
ship_to_org_id	NUMBER		C - for order lines, not return lines		X
sold_to_org_id	NUMBER		X		

Table 2-31 OE_ORDER_LINES_ALL

Parameter	Datatype	Required	Required at Booking	Derived	Optional
sold_from_org_id	NUMBER		X		
sort_order	VARCHAR2(240)			X	
source_document_id	NUMBER				X
source_document_line_id	NUMBER				X
source_document_type_id	NUMBER				X
source_type_code	VARCHAR2(30)				X
split_from_line_id	NUMBER				X
task_id	NUMBER	C - depends on the project			
tax_code	VARCHAR2(50)		X		X
tax_date	DATE		X		X
tax_exempt_flag	VARCHAR2(30)		X		X
tax_exempt_number	VARCHAR2(50)				X
tax_exempt_reason_code	VARCHAR2(30)	C -if tax status is <i>Exempt</i>			X
tax_point_code	VARCHAR2(30)				for future use
tax_rate	NUMBER			X	
tax_value	NUMBER			X	
top_model_line_ref	VARCHAR2(50)				X
top_model_line_id	NUMBER			X	
top_model_line_index	NUMBER				X
unit_list_price	NUMBER		X		X
unit_list_price_per_pqty	NUMBER				X
unit_selling_price	NUMBER		X		X
unit_selling_price_per_pqty	NUMBER				X

Table 2–31 OE_ORDER_LINES_ALL

Parameter	Datatype	Required	Required at Booking	Derived	Optional
veh_cus_item_cum_key_id	NUMBER				X
visible_demand_flag	VARCHAR2(1)			X	
return_status	VARCHAR2(1)			X	
db_flag	VARCHAR2(1)			X	
operation	VARCHAR2(30)	X			
first_ack_code	VARCHAR2(30)			X	
first_ack_date	DATE			X	
last_ack_code	VARCHAR2(30)			X	
last_ack_date	DATE			X	
change_reason	VARCHAR2(30)				X
change_comments	VARCHAR2(2000)				X
arrival_set	VARCHAR2(30)				X
ship_set	VARCHAR2(30)				X
fulfillment_set	VARCHAR2(30)				X
order_source_id	NUMBER				X
orig_sys_shipment_ref	VARCHAR2(50)				X
change_sequence	VARCHAR2(50)				X
change_request_code	VARCHAR2(30)			X	
status_flag	VARCHAR2(1)			X	
drop_ship_flag	VARCHAR2(1)			X	
customer_line_number	VARCHAR2(50)				X
customer_shipment_number	VARCHAR2(50)				X
customer_item_net_price	NUMBER				X
customer_payment_term_id	Obsolete	Obsolete	Obsolete	Obsolete	Obsolete

Table 2–31 OE_ORDER_LINES_ALL

Parameter	Datatype	Required	Required at Booking	Derived	Optional
ordered_item_id	NUMBER	C: for Customer Items			
item_identifier_type	VARCHAR2(25)	C: for <i>customer</i> and <i>generic</i> items			X
shipping_instructions	VARCHAR2(2000)				X
packing_instructions	VARCHAR2(2000)				X
calculate_price_flag	VARCHAR2(1)				X
invoiced_quantity	NUMBER			X	
service_txn_reason_code	VARCHAR2(30)				X
service_txn_comments	VARCHAR2(2000)				X
service_duration	NUMBER		C - for service lines		X
service_period	VARCHAR2(3)	C - for service lines			X
service_start_date	DATE	C - for service lines			X
service_end_date	DATE	C - for service lines			X
service_coterminate_flag	VARCHAR2(1)				X
unit_list_percent	NUMBER				X
unit_selling_percent	NUMBER				X
unit_percent_base_price	NUMBER			X	
service_number	NUMBER			X	
service_reference_type_code	VARCHAR2(30)	C - for service lines			X
service_reference_line_id	NUMBER	C - for service lines			X
service_reference_system_id	NUMBER				X

Table 2–31 OE_ORDER_LINES_ALL

Parameter	Datatype	Required	Required at Booking	Derived	Optional
service_ref_order_number	NUMBER	C - for service reference type of Order			X
service_ref_line_number	NUMBER	C - for service reference type of Order			X
service_ref_shipment_number	NUMBER				X
service_ref_option_number	NUMBER				X
service_line_index	NUMBER				X
Line_set_id	NUMBER			X	
split_by	VARCHAR2(240)				X
Split_Action_Code	VARCHAR2(30)				X
shippable_flag	VARCHAR2(1)			X	
model_remnant_flag	VARCHAR2(1)			X	
flow_status_code	VARCHAR2(30)			X	
fulfilled_flag	VARCHAR2(1)			X	
fulfillment_method_code	VARCHAR2(30)			X	
revenue_amount	NUMBER			X	
marketing_source_code_id	NUMBER				X
fulfillment_date	DATE			X	
semi_processed_flag	BOOLEAN			X	
override_atp_date_code	Varchar2(30)	No	No	No	Y
late_demand_penalty_factor	Number	No	No	No	Y
xml_transaction_type_code	Varchar2(30)				Y

TYPE Line_Tbl_Type IS TABLE OF Line_Rec_Type

Index BY BINARY_INTEGER;

Line_Rec_Type**Table 2–32** *Line_Rec_Type*

Parameter	Value
IB_OWNER	VARCHAR2(60)
IB_INSTALLED_AT_LOCATION	VARCHAR2(60)
IB_CURRENT_LOCATION	VARCHAR2(60)
END_CUSTOMER_ID	NUMBER
END_CUSTOMER_CONTACT_ID	NUMBER
END_CUSTOMER_SITE_USE_ID	NUMBER
ship_to_party_id	NUMBER
ship_to_party_site_id	NUMBER
ship_to_party_site_use_id	NUMBER
deliver_to_party_id	NUMBER
deliver_to_party_site_id	NUMBER
deliver_to_party_site_use_id	NUMBER
invoice_to_party_id	NUMBER
invoice_to_party_site_id	NUMBER
invoice_to_party_site_use_id	NUMBER
ship_to_customer_party_id	NUMBER
deliver_to_customer_party_id	NUMBER
invoice_to_customer_party_id	NUMBER
ship_to_org_contact_id	NUMBER
deliver_to_org_contact_id	NUMBER
invoice_to_org_contact_id	NUMBER

Table 2–32 Line_Rec_Type

Parameter	Value
IB_OWNER	VARCHAR2(60)
ship_to_party_number	varchar2(30)
invoice_to_party_number	varchar2(30)
deliver_to_party_number	varchar2(30)

Line_Val_Rec_Type**Table 2–33 Line_Val_Rec_Type**

Parameter
accounting_rule
agreement
commitment
deliver_to_address1
deliver_to_address2
deliver_to_address3
deliver_to_address4
deliver_to_contact
deliver_to_location
deliver_to_org
demand_class
demand_bucket_type
fob_point
freight_terms
inventory_item
invoice_to_address1
invoice_to_address2
invoice_to_address3
invoice_to_address4

Table 2–33 *Line_Val_Rec_Type*

Parameter
invoice_to_contact
invoice_to_location
invoice_to_org
invoicing_rule
item_type
line_type
over_ship_reason
payment_term
price_list
project
return_reason
rla_schedule_type
salesrep
shipment_priority
ship_from_address1
ship_from_address2
ship_from_address3
ship_from_address4
ship_from_location
ship_from_org
ship_to_address1
ship_to_address2
ship_to_address3
ship_to_address4
ship_to_contact
ship_to_location
ship_to_org

Table 2–33 Line_Val_Rec_Type

Parameter
source_type
intermed_ship_to_address1
intermed_ship_to_address2
intermed_ship_to_address3
intermed_ship_to_address4
intermed_ship_to_contact
intermed_ship_to_location
intermed_ship_to_org
sold_to_org
sold_from_org
task
tax_exempt
tax_exempt_reason
tax_point
veh_cus_item_cum_key
visible_demand
customer_payment_term - <i>obsolete</i>
ref_order_number
ref_line_number
ref_shipment_number
ref_option_number
ref_invoice_number
ref_invoice_line_number
credit_invoice_number
tax_group
status
freight_carrier

Table 2–33 *Line_Val_Rec_Type*

Parameter	
shipping_method	
ship_to_customer_name	VARCHAR2(360)
invoice_to_customer_name	VARCHAR2(360)
ship_to_customer_number	VARCHAR2(50)
invoice_to_customer_number	VARCHAR2(50)
deliver_to_customer_number	VARCHAR2(50)
deliver_to_customer_name	VARCHAR2(360)
end_customer_name	VARCHAR2(360)
end_customer_number	VARCHAR2(50)
end_customer_contact	VARCHAR2(360)
end_cust_contact_last_name	VARCHAR2(240)
end_cust_contact_first_name	VARCHAR2(240)
end_customer_site_address1	VARCHAR2(240)
end_customer_site_address2	VARCHAR2(240)
end_customer_site_address3	VARCHAR2(240)
end_customer_site_address4	VARCHAR2(240)
end_customer_site_location	VARCHAR2(240)
end_customer_site_state	VARCHAR2(240)
end_customer_site_country	VARCHAR2(240)
end_customer_site_zip	VARCHAR2(240)
end_customer_site_county	VARCHAR2(240)
end_customer_site_province	VARCHAR2(240)
end_customer_site_city	VARCHAR2(240)
end_customer_site_postal_code	VARCHAR2(240)
ib_owner_dsp	VARCHAR2(60)
ib_current_location_dsp	VARCHAR2(60)
ib_installed_at_location_dsp	VARCHAR2(60)

TYPE Line_Val_Tbl_Type IS TABLE OF Line_Val_Rec_Type
INDEX BY BINARY_INTEGER;

TYPE Line_Adj_Rec_Type IS RECORD

For column descriptions, please refer to the Order Management TRM for the table
OE_PRICE_ADJUSTMENTS

Table 2–34 OE_PRICE_ADJUSTMENTS

Parameter	Datatype	Required	Derived	Optional
attribute1.... attribute15	VARCHAR2(240)			
automatic_flag	VARCHAR2(1)	X		X
context	VARCHAR2(30)			X
created_by	NUMBER		X	
creation_date	DATE		X	
discount_id	NUMBER		X	
discount_line_id	NUMBER		X	
header_id	NUMBER	X		
last_updated_by	NUMBER		X	
last_update_date	DATE		X	
last_update_login	NUMBER		X	
line_id				
percent	NUMBER		X	
price_adjustment_id	NUMBER	C -for update and delete		X
program_application_id	NUMBER			X
program_id	NUMBER			X
program_update_date	DATE			X
request_id	NUMBER			X
return_status	VARCHAR2(1)		X	
db_flag	VARCHAR2(1)		X	
operation	VARCHAR2(30)	X		

Table 2–34 OE_PRICE_ADJUSTMENTS

Parameter	Datatype	Required	Derived	Optional
line_index	NUMBER			X
orig_sys_discount_ref	VARCHAR2(50)			X
change_request_code	VARCHAR2(30)			X
status_flag	VARCHAR2(1)			X
list_header_id	NUMBER	C - not required only for tax records		
list_line_id	NUMBER	C - not required only for tax records		
list_line_type_code	VARCHAR2(30)		X	
modifier_mechanism_type_code	VARCHAR2(30)		X	
modified_from	NUMBER		X	
modified_to	NUMBER		X	
updated_flag	VARCHAR2(1)			
update_allowed	VARCHAR2(1)		X	
applied_flag	VARCHAR2(1)		X	
change_reason_code	VARCHAR2(30)			X
change_reason_text	VARCHAR2(2000)			X
operand	NUMBER			X
operand_per_pqty	NUMBER			
arithmetic_operator	VARCHAR2(30)		X	
cost_id	NUMBER		X	
tax_code	VARCHAR2(30)			
tax_exempt_flag	VARCHAR2(1)			
tax_exempt_number	VARCHAR2(80)			
tax_exempt_reason_code	VARCHAR2(30)			
parent_adjustment_id	NUMBER	X -for update and delete		

Table 2–34 OE_PRICE_ADJUSTMENTS

Parameter	Datatype	Required	Derived	Optional
invoiced_flag	VARCHAR2(1)		X	
estimated_flag	VARCHAR2(1)		X	
inc_in_sales_performance	VARCHAR2(1)		X	
split_action_code	VARCHAR2(30)			X
adjusted_amount	NUMBER		X	
adjusted_amount_per_pqty	NUMBER			
pricing_phase_id	NUMBER		X	
charge_type_code	VARCHAR2(30)		X	
charge_subtype_code	VARCHAR2(30)		X	
list_line_no	VARCHAR2(240)		X	
source_system_code	VARCHAR2(30)		X	
benefit_qty	NUMBER		X	
benefit_uom_code	VARCHAR2(3)		X	
print_on_invoice_flag	VARCHAR2(1)		X	
expiration_date	DATE		X	
rebate_transaction_type_code	VARCHAR2(30)		X	
rebate_transaction_reference	VARCHAR2(80)		X	
rebate_payment_system_code	VARCHAR2(30)		X	
redeemed_date	DATE		X	
redeemed_flag	VARCHAR2(1)		X	
accrual_flag	VARCHAR2(1)		X	
range_break_quantity	NUMBER		X	
accrual_conversion_rate	NUMBER		X	
pricing_group_sequence	NUMBER		X	
modifier_level_code	VARCHAR2(30)		X	
price_break_type_code	VARCHAR2(30)		X	
substitution_attribute	VARCHAR2(30)		X	

Table 2–34 OE_PRICE_ADJUSTMENTS

Parameter	Datatype	Required	Derived	Optional
proration_type_code	VARCHAR2(30)		X	
credit_or_charge_flag	VARCHAR2(1)		X	
include_on_returns_flag	VARCHAR2(1)		X	
ac_attribute1...15	VARCHAR2(240)			X
ac_context	VARCHAR2(150)			X

TYPE Line_Adj_Tbl_Type IS TABLE OF Line_Adj_Rec_Type

INDEX BY BINARY_INTEGER;

TYPE Line_Adj_Val_Rec_Type IS RECORD

(discount List_name)

TYPE Line_Adj_Val_Tbl_Type IS TABLE OF Line_Adj_Val_Rec_Type

INDEX BY BINARY_INTEGER;

Line_Price_Att_Rec_Type

For column descriptions, please refer to the Order Management TRM for the table
OE_ORDER_PRICE_ATTRIBS

Table 2–35 OE_ORDER_PRICE_ATTRIBS

Parameter	Datatype	Required	Derived	Optional
order_price_attrib_id	NUMBER			X
header_id	NUMBER			
line_id	NUMBER			
line_index	NUMBER			
creation_date	DATE		X	
created_by	NUMBER		X	
last_update_date	DATE		X	
last_updated_by	NUMBER		X	
last_update_login	NUMBER		X	

Table 2–35 OE_ORDER_PRICE_ATTRIBS

Parameter	Datatype	Required	Derived	Optional
program_application_id	NUMBER			X
program_id	NUMBER			X
program_update_date	DATE			X
request_id	NUMBER			X
flex_title	VARCHAR2(60)			
pricing_context	VARCHAR2(30)			
pricing_attribute1... pricing_attribute100	VARCHAR2(240)			
context	VARCHAR2(30)			
attribute1..15	VARCHAR2(240)			
Override_Flag	VARCHAR2(1)			
return_status	VARCHAR2(1)		X	
db_flag	VARCHAR2(1)		X	
operation	X			

TYPE Line_Price_Att_Tbl_Type is TABLE of Line_Price_Att_rec_Type
INDEX by BINARY_INTEGER

Line_Adj_Att_Rec_Type

For column descriptions, please refer to the Order Management TRM for the table
OE_PRICE_ADJ_ATTRIBS

Table 2–36 OE_PRICE_ADJ_ATTRIBS

Parameter	Datatype	Required	Derived	Optional
price_adj_attrib_id	NUMBER	C -for update and delete		
price_adjustment_id	NUMBER	C - if adj_index not passed		
Adj_index	NUMBER	C - if price_adjustment_ id not passed		
flex_title	VARCHAR2(60)	X		

Table 2–36 OE_PRICE_ADJ_ATTRIBS

Parameter	Datatype	Required	Derived	Optional
pricing_context	VARCHAR2(30)	X		
pricing_attribute	VARCHAR2(30)	X		
creation_date	DATE		X	
created_by	NUMBER		X	
last_update_date	DATE		X	
last_updated_by	NUMBER		X	
last_update_login	NUMBER		X	
program_application_id	NUMBER			X
program_id	NUMBER			X
program_update_date	DATE			X
request_id	NUMBER			X
pricing_attr_value_from	VARCHAR2(240)	X		
pricing_attr_value_to	VARCHAR2(240)			X
comparison_operator	VARCHAR2(30)			X
return_status	VARCHAR2(1)		X	
db_flag	VARCHAR2(1)		X	
operation	VARCHAR2(30)	X		

TYPE Line_Adj_Att_Tbl_Type is TABLE of Line_Adj_Att_rec_Type

INDEX by BINARY_INTEGER

Line_Adj_Assoc_Rec_Type

For column descriptions, please refer to the Order Management TRM for the table OE_PRICE_ADJ ASSOCS

Table 2–37 OE_PRICE_ADJ ASSOCS

Parameter	Datatype	Required	Derived	Optional
price_adj_assoc_id	NUMBER	C - for update and delete		
line_id	NUMBER	C - if line_index not passed		
Line_Index	NUMBER	C - if line_id not passed		
price_adjustment_id	NUMBER	C - if adj_index not passed		
Adj_index	NUMBER	C - if price_adjustment_id not passed		
rltd_Price_Adj_Id	NUMBER			X
Rltd_Adj_Index	NUMBER			X
creation_date	DATE		X	
created_by	NUMBER		X	
last_update_date	DATE		X	
last_updated_by	NUMBER		X	
last_update_login	NUMBER		X	
program_application_id	NUMBER			X
program_id	NUMBER			X
program_update_date	DATE			X
request_id	NUMBER			X
return_status	VARCHAR2(1)		X	
db_flag	VARCHAR2(1)		X	
operation	VARCHAR2(30)	X		

TYPE Line_Adj_Assoc_Tbl_Type is TABLE of Line_Adj_Assoc_rec_Type
INDEX by BINARY_INTEGER;

TYPE Line_Scredit_Rec_Type IS RECORD

For column descriptions, please refer to the Order Management TRM for the table
OE_SALES_CREDITS

Table 2–38 OE_SALES_CREDITS

Parameter	Datatype	Required	Derived	Optional
attribute1....attribute15	VARCHAR2(240)			X
context	VARCHAR2(30)			X
created_by	NUMBER		X	
creation_date	DATE		X	
dw_update_advice_flag	VARCHAR2(1)			X
header_id	NUMBER			X
last_updated_by	NUMBER		X	
last_update_date	DATE		X	
last_update_login	NUMBER		X	
line_id	NUMBER			X
percent	NUMBER	X		
salesrep_id	NUMBER	X		
sales_credit_type_id	NUMBER	X		
sales_credit_id	NUMBER			X
wh_update_date	DATE			X
return_status	VARCHAR2(1)		X	
db_flag	VARCHAR2(1)		X	
operation	VARCHAR2(30)	X		
line_index	NUMBER			X
orig_sys_credit_ref	VARCHAR2(50)			X
change_request_code	VARCHAR2(30)		X	
status_flag	VARCHAR2(1)		X	

TYPE Line_Scredit_Tbl_Type IS TABLE OF Line_Scredit_Rec_Type
INDEX BY BINARY_INTEGER;

```
-- Line_Scredit value record type
TYPE Line_Scredit_Val_Rec_Type IS RECORD
(   salesrep
```



```

        , sales_credit_type
    );
TYPE Line_Scredit_Val_Tbl_Type IS TABLE OF Line_Scredit_Val_Rec_Type
INDEX BY BINARY_INTEGER;

```

Lot_Serial_Rec_Type

For column descriptions, please refer to the Order Management TRM for the table OE_LOT_SERIAL_NUMBERS

Table 2–39 OE_LOT_SERIAL_NUMBERS

Parameter	Datatype	Required	Derived	Optional
attribute1....attribute15	VARCHAR2(240)			X
context	VARCHAR2(30)			X
created_by	NUMBER		X	
creation_date	DATE		X	
from_serial_number	VARCHAR2(30)	C - if lot number not passed		X
last_updated_by	NUMBER		X	
last_update_date	DATE		X	
last_update_login	NUMBER		X	
line_id	NUMBER	C - if line index not passed		X
lot_number	VARCHAR2(30)	C - if lot serial information not available		X
lot_serial_id	NUMBER	C - for update and delete operation		X
quantity	NUMBER	X		X
to_serial_number	VARCHAR2(30)	C - if lot number not passed		X
return_status	VARCHAR2(1)		X	
db_flag	VARCHAR2(1)		X	
operation	VARCHAR2(30)	X		

Table 2–39 OE_LOT_SERIAL_NUMBERS

Parameter	Datatype	Required	Derived	Optional
line_index	NUMBER	C - if line id not passed		
orig_sys_lotserial_ref	VARCHAR2(50)			X
change_request_code	VARCHAR2(30)		X	
status_flag	VARCHAR2(1)		X	
line_set_id	NUMBER		X	

TYPE Lot_Serial_Tbl_Type IS TABLE OF Lot_Serial_Rec_Type
INDEX BY BINARY_INTEGER;
TYPE Lot_Serial_Val_Rec_Type IS RECORD
(line lot_serial);
TYPE Lot_Serial_Val_Tbl_Type IS TABLE OF Lot_Serial_Val_Rec_Type
INDEX BY BINARY_INTEGER;

Process Order Usage

Process Order is a complex API and a number of operations on a single order object and its entities can be performed via this API. This section is intended to help a user of the process order API in identifying the required parameters for some common operations and to give an understanding of the business flow behind each of these operations.

CREATE operation

For creating new order entities, the entity records and entity tables passed to Process_Order should have the operation OE_GLOBALS.G_OPR_CREATE. Pass in the known attributes on these records and Process Order will default the other *missing* attributes.

Note: If you pass in NULL values for some attributes, process order will interpret it as insertion of NULL into those attributes column and will NOT get default values for those attributes.

Line adjustments, line sales credits and line lot serial numbers belong to an order line and the ID of this parent line is stored on these records. When inserting the parent line and the child entities of this line together, the ID of the parent line is unknown. In this case, the `line_index` field on the child entity records indicates the index for the parent entity record in the lines table. After the line is processed, the line ID is retrieved and based on the `line_index`, the correct `line_ID` is populated on the child entity record.

Similarly for config items, the model line's ID is stored on the option lines. Again, when inserting models and options in a configuration together, the line ID for the model line will not be available. The `top_model_line_index` field on the option lines can be used to indicate the index for this model line in the lines table. Since models and standard lines are processed before the options, this line ID can then be populated before the option lines are processed.

The following steps are executed for each entity within process order:

- Check security
- Attribute level validation
- Entity level validation
- Posted to the database
- Cross Record Logic

Create Operation Example 1

Creating a new order with 2 lines and 1 line adjustment and the adjustment belongs to the second line.

```
-- SETTING UP THE HEADER RECORD
-- Initialize record to missing
l_header_rec := OE_ORDER_PUB.G_MISS_HEADER_REC;
-- Required attributes (e.g. Order Type and Customer)
l_header_rec.order_type_id := 1000;
l_header_rec.sold_to_org_id := 100;
-- Other attributes
l_header_rec.price_list_id := 10;
.....
-- Null attribute: no defaulting for freight terms
l_header_rec.freight_term_code = NULL;
-- Indicates to process order that a new header is being created
l_header_rec.operation := OE_GLOBALS.G_OPR_CREATE;

-- FIRST LINE RECORD
```

```

-- Initialize record to missing
l_line_tbl(1) := OE_ORDER_PUB.G_MISS_LINE_REC;
-- Line attributes
l_line_tbl(1).inventory_item_id := 311;
l_line_tbl(1).ordered_quantity := 1;
l_line_tbl(1).operation := OE_GLOBALS.G_OPR_CREATE;

-- SECOND LINE RECORD
-- Initialize record to missing
l_line_tbl(2) := OE_ORDER_PUB.G_MISS_LINE_REC;
-- Line attributes
l_line_tbl(2).inventory_item_id := 312;
l_line_tbl(2).ordered_quantity := 2;
l_line_tbl(2).operation := OE_GLOBALS.G_OPR_CREATE;

-- LINE ADJUSTMENT RECORD
-- Initialize record to missing
l_line_adj_tbl(1) := OE_ORDER_PUB.G_MISS_LINE_ADJ_REC;
-- Attributes for the line adjustment
l_line_adj_tbl(1).discount_id := 1;
l_line_adj_tbl(1).percent := 5;
l_line_adj_tbl(1).operation := OE_GLOBALS.G_OPR_CREATE;
-- Indicator that this adjustment belongs to the second line
l_line_adj_tbl(1).line_index := 2;

-- CALL TO PROCESS ORDER
OE_Order_PUB.Process_Order(
.....
.....
-- Passing just the entity records that are a part of this order
p_header_rec => l_header_rec
p_line_tbl=> l_line_tbl
p_line_adj_tbl=> l_line_adj_tbl
-- OUT variables
x_header_rec=> l_header_rec
x_header_scredit_tbl=> l_header_scredit_tbl
x_header_adj_tbl=> l_header_adj_tbl
x_line_tbl=> l_line_tbl
x_line_scredit_tbl=> l_line_scredit_tbl
x_line_adj_tbl=> l_line_adj_tbl
.....
x_return_status=> l_return_status
x_msg_count=> l_msg_count
x_msg_data=> l_msg_data);

```

```

-- Retrieve messages
if l_msg_count > 0 then
    for l_index in 1..l_msg_count loop
        l_msg_data := oe_msg_pub.get(p_msg_index => l_index, p_encoded => 'F');
    end loop;
end if;

-- Check the return status
if x_return_status = FND_API.G_RET_STS_SUCCESS then
    success;
else
    failure;
end if;

```

Create Operation Example 2

Inserting a new line into an existing order.

Warning: You cannot insert new lines (or, process any other entity) belonging to different orders in one process order call

```

-- NEW LINE RECORD
l_line_tbl(1) := OE_ORDER_PUB.G_MISS_LINE_REC;
-- Primary key of the parent entity (If not passed, this will be retrieved
from the parent record, p_header_rec, if it was also passed to process order
else there will be an error in the processing of this line).
l_line_tbl(1).header_id := 1000;
-- Attributes for the new line
l_line_tbl(1).inventory_item_id := 311;
l_line_tbl(1).ordered_quantity := 1;
-- Indicator that a new line is being created
l_line_tbl(1).operation := OE_GLOBALS.G_OPR_CREATE;

-- CALL TO PROCESS ORDER
OE_ORDER_PUB.Process_Order(.....
-- Passing just the entity records that are being created
p_line_tbl=> l_line_tbl
-- OUT variables
.....);

```

UPDATE operation

Update of any attributes on the entities of the order object should always go through the process order API. For updates, the operations on the entity record

being updated should be OE_GLOBALS.G_OPR_UPDATE. Only the attributes that are being updated, the primary key of this entity and the primary key of the parent entity, if any, need to be passed and others can be set to missing. Process Order will query these missing attributes from the database.

Note: If you pass in NULL values for some attributes, process order will update existing value of those attributes, if any, with NULL.

Process Order can process entities belonging to one order object in one call. In a call where one entity record is being updated, another entity record can also be created as long as they belong to the same order object. For e.g. a line can be updated (G_OPR_UPDATE) and another line inserted (G_OPR_CREATE) in the same process order call if both lines have the same header_id.

The following processing steps are executed for each entity record that is updated:

- Check security for update of the changed attributes
- Validation of updated attributes
- Clearing of (or set to missing) attributes dependent on the updated attributes
- Defaulting of missing attributes
- Entity level validation
- Entity security check
- Posted to the database
- Cross Record Logic

Update Operation Example

Updating the bill to organization and order quantity on an order line.

```
-- LINE RECORD WITH THE CHANGES
-- Changed attributes
l_line_tbl(1) := OE_ORDER_PUB.G_MISS_LINE_REC.
l_line_tbl(1).invoice_to_org_id := 322;
l_line_tbl(1).ordered_quantity := 2;
-- Primary key of the entity i.e. the order line
l_line_tbl(1).line_id := 1000;
-- Indicates to process order that this is an update operation
```

```

l_line_tbl(1).operation := OE_GLOBALS.G_OPR_UPDATE;

-- CALL TO PROCESS ORDER
OE_ORDER_PUB.Process_Order(.....
-- Entity records
p_line_tbl=> l_line_tbl
-- OUT variables
.....);

```

DELETE operation

For deletes, the operations on the entity records being deleted should be OE_GLOBALS.G_OPR_DELETE. Only the primary key of the entity being deleted needs to be passed.

Deletes are cascaded down to the child entities for e.g. to delete an order, only the header record with the header ID needs to be passed to process order and all the child entities i.e. lines, header sales credits, header adjustments are deleted. Deleting the lines results in the deletion of line adjustments, line sales credits and line lot serial numbers as well. Also, it would delete delayed requests logged by or against this entity and its child entities.

Deletes would also result in the deletion of any holds and attachments associated with the deleted entity and its child entities. The workflow status information for this entity is also purged.

The following processing steps are executed for each entity record that is deleted:

- Entity security check.

Note: Child entities will always be deleted if parent entity passes security check. There will not be a separate security check for child entities

- Entity level validation for deletes.
- Delete posted to the database. This will also delete child entities, holds and attachments.
- Cross Record Logic

Delete Operation Example

```

-- Only the record for the parent entity needs to be passed. Since the whole
order is being deleted, only the header record needs to be passed.

```

```
-- Primary key of this order i.e. the header
l_header_rec := OE_ORDER_PUB.G_MISS_LINE_REC;
-- Indicate to process order that the order is to be deleted
l_header_rec.operation := OE_GLOBALS.G_OPR_DELETE;

-- CALL TO PROCESS ORDER
OE_ORDER_PUB.Process_Order(.....
-- Entity records
p_header_rec=> l_header_rec
-- OUT variables
.....);
```

Process Order and Action Requests

Process order API can also process some other actions on the order object. For e.g. booking the order, applying/releasing holds.

For each action request, a request record should be passed in the action request table (p_action_request_tbl) to process order and the parameters of the request record can provide additional information needed to carry out that action. The validation_level parameter determines whether validation of the parameters of the request record needs to be performed.

If the ID of the order or line for which this request needs to get executed is not known as it is being created in the same call, the ENTITY_INDEX field on the request record indicates the index for the entity record in the entity table passed to

Following are the action requests currently supported and the parameters on the request record that are used in processing the request.

Book the order

This request is used to book the order.

Table 2–40 Book Order

Parameter	Description
request_type	OE_GLOBALS.G_BOOK_ORDER
entity_code	This should be always set to OE_GLOBALS.G_ENTITY_HEADER as booking is an order level action.
entity_id	Header ID of the order to be booked. If the order is also being created in the same call to process order, then the user does not need to provide this value.

Apply Automatic Attachments

The attachment rules will be evaluated for the entity and if applicable, documents will be attached to the entity.

Table 2–41 Apply Automatic Attachments

Parameter	Description
request_type	OE_GLOBALS.G_APPLY_AUTOMATIC_ATCHMT
entity_code	OE_GLOBALS.G_ENTITY_HEADER if the rules are to be evaluated for the order header OE_GLOBALS.G_ENTITY_LINE if rules are to be evaluated for the order line.
entity_id	Header ID if the entity is G_ENTITY_HEADER, Line ID if the entity is G_ENTITY_LINE
entity_index	If entity is G_ENTITY_LINE and the line is being created in the same call to process order, then send in the index position for this line in the table, p_line_tbl.

Apply Hold

Holds will be applied based on the parameters sent in on the request record.

Table 2–42 Apply Hold

Parameter	Description
request_type	OE_GLOBALS.G_APPLY_HOLD
entity_code	OE_GLOBALS.G_ENTITY_ORDER for order or OE_GLOBALS.G_ENTITY_LINE for line.
entity_id	ID of the order or line to be held
param1	Hold ID to identify the type of hold that should be applied. (HOLD_ID from OE_HOLD_DEFINITIONS)
param2	Hold entity code for the hold source to be created. C: Customer hold source S: Bill To or Ship To hold source I: Item hold source O: Order hold source W: Warehouse Hold Source

Table 2–42 Apply Hold

Parameter	Description
param3	Hold entity ID <i>C, B, or S</i> : for Org ID <i>O</i> : Header ID <i>I</i> : Inventory Item ID
param4	Hold comment
date_param1	Hold Until Date
param6-param20	Attribute1-15 of the descriptive flexfield associated with the hold source record.

Release Hold

Releases the holds associated with an order or line.

Table 2–43 Release Hold

Parameter	Description
request_type	OE_GLOBALS.G_RELEASE_HOLD
entity_code	OE_GLOBALS.G_ENTITY_ORDER for order or OE_GLOBALS.G_ENTITY_LINE for line.
entity_id	ID of the order or line to be released from hold
param1	Hold ID to specify the type of hold that is to be removed.
param2	Hold entity code on the hold source associated with the hold to be released <i>C</i> : Customer hold source <i>S</i> : Bill To or Ship To hold source <i>I</i> : Item hold source <i>O</i> : Order hold source <i>W</i> : Warehouse Hold Source
param3	Hold entity ID: <i>C or S</i> : for Org ID <i>O</i> : Header ID <i>I</i> : Inventory Item ID

Table 2–43 Release Hold

Parameter	Description
param4	Release Reason Code
param5	Release Comment

Delink Config

This request deletes the configuration item.

Table 2–44 Delink Config

Parameter	Description
request_type	OE_GLOBALS.G_DELINK_CONFIG
entity_code	OE_GLOBALS.G_ENTITY_LINE
entity_id	Line ID of the top model line of the ATO configuration

Match and Reserve

This request checks if an existing configuration item matches the configuration created by the user and if configuration item is available, it reserves it.

Table 2–45 Match and Reserve

Parameter	Description
request_type	OE_GLOBALS.G_MATCH_AND_RESERVE
entity_code	OE_GLOBALS.G_ENTITY_LINE
entity_id	Line ID of the top model line of the ATO configuration

Get Ship Method

This action is supported only OM pack I onwards. The actions can be performed only if OM system parameter 'Enable Ship Method' is enabled. OM will use the routing guides defined in product transportation to get the most appropriate carrier and ship method, dependent on defined business rules. The action always works on the entire order at a time. The action ignores following type of lines while deriving the ship method,

- a. Dropship lines
- b. Return lines
- c. Service lines

- d. CONFIG lines (The line created after progressing ATO model once it is booked and scheduled)
- e. Lines that are already shipped
- f. Lines that are already fulfilled
- g. Lines that are already closed
- h. Lines which do not have all required parameters. e.g. ship from, ship to.

Table 2–46 Get Ship Method

Parameter	Description
request_type	OE_GLOBALS.G_GET_SHIP_METHOD
entity_code	OE_GLOBALS.G_ENTITY_HEADER or OE_GLOBALS.G_ENTITY_HEADER
entity_id	Header ID if entity is LINE Line ID if entity is LINE

Action Request Example

Process order can also process other action requests on the order or line as has been documented already.

Apply a hold to an order line due to a defective item.

```
-- ACTION REQUEST RECORD

-- Indicates that it is a line level action
l_request_rec.entity := OE_GLOBALS.G_ENTITY_LINE;
-- primary key of the line on which the hold is to be applied
l_request_rec.entity_id := 100;
-- name of the action request
l_request_rec.request_name := OE_GLOBALS.G_APPLY_HOLD;

-- request record parameters
-- defective product hold (hold_id)
l_request_rec.param1 := 4;
-- indicator that it is an item hold (hold_entity_code)
l_request_rec.param2 = 'I';
-- Id of the item (hold_entity_id)
l_request_rec.param3 := 3214;

-- inserting request record into the action request table
```

```
l_action_request_tbl := l_request_rec;
```

See:

Oracle Applications Message Reference Manual

Oracle Order Management User's Guide

Integrating Oracle Order Management with Oracle Receivables and Invoicing

Oracle Order Management provides functionality to integrate with Oracle Receivables. Using AutoInvoice, you can create invoices, create credit memos and credits on account, recognize revenue, and manage sales credits.

Basic Needs

Oracle Order Management and Oracle Receivables provide features you need to satisfy the following integration needs:

- Create accurate and timely invoices, credit memos, and credits on account from Order Management transactions
- Control when order transactions are invoiced

Major Features

Invoicing Activity

Order Management provides a program that automatically collects order and return information and populates the Oracle Receivables AutoInvoice interface tables. Using order management transaction types and profile options, you control certain accounting aspects of the information being transferred to Oracle Receivables. Invoicing module is modeled using workflow and can happen after shipping for shippable flows and any time after booking for non-shippable flows. Invoicing can be placed at both order line level workflows as well as order header level workflows. If placed at the order header level, all lines in an order are populated in AutoInvoice interface tables at the same time.

Invoice Source

You must setup invoice sources in Oracle Receivables prior to using Oracle Order Management. Order Management uses the invoice source setup at the transaction type level or the profile option OM: Invoice Source when populating order line information into AutoInvoice interface tables. When defining invoice sources in Oracle Receivables, you must create at least one invoice source for Order Management's use if you want to interface orders and returns for processing by AutoInvoice.

The following table shows the necessary field values for the Transaction Source window in Oracle Receivables.

Table 2–47 Transaction Sources Window: Required Settings

Field in Transaction Sources Window	Necessary Value
Batch Source region	
Type	Imported
Status	Active
Automatic Transaction Numbering	Yes if OM: Invoice Numbering Method is set to Automatic, No if set to Delivery Name
Customer Information region	Customer Information region
Sold-to Customer	Id
Bill-to Customer	Id
Bill-to Address	Id
Bill-to Contact	Id
Ship-to Customer	Id
Ship-to Address	Id
Ship-to Contact	Id
Payment Method Rule	(any)
Customer Bank Account	(any)
Accounting Information region	Accounting Information region
Invoicing Rule	Id
Accounting Rule	Id
Accounting Flexfield	(any)

Table 2–47 Transaction Sources Window: Required Settings

Field in Transaction Sources Window	Necessary Value
Derive Date	(any)
Payment Terms	Id
Revenue Account allocation	(any)
Other Information region	Other Information region
Transaction Type	Id
Memo Reason	Id
Agreement	Id
Memo Line Rule	Id
Sales Territory	(any)
Inventory Item	Id
Unit of Measure	Id
FOB Point	Code
Freight Carrier	Code
Related Document	Id
Sales Credits Validation region	Sales Credits Validation region
Salesperson	Id
Sales Credit Type	Id
Sales Credit	Percent

Note: In the Batch Source region, you must define at least one transaction source with automatic invoice numbering, regardless of your setting for the *OM: Invoice Numbering Method* profile.

Automatic Tax Calculation

As orders from Order Management are processed, AutoInvoice automatically calculates sales tax based on the Sales Tax Location flexfield combination. If you have designated a customer as tax-exempt, AutoInvoice will not tax any items billed for the customer. Similarly, if you have designated an item as non-taxable, AutoInvoice will not tax the item.

Automatic Account Code Creation

Oracle Receivables uses AutoAccounting to determine the revenue account for all transactions from Order Management. AutoAccounting lets you define what information is used to define the various segments of your Accounting Flexfield.

Accounting and Invoicing Rules

Order Management uses accounting and invoicing rules. This information is transferred to Oracle Receivables and used to determine the invoice date (invoicing rule) and general ledger distribution records (accounting rule). Order Management passes an invoicing rule and accounting rule for each order transaction interfaced to Oracle Receivables, except for when the accounting rule is Immediate, in which case Order Management does not pass any value (inserts null).

Accounting Rules

Order Management determines the accounting rule for sales order lines based on the following hierarchy.

Table 2–48 Accounting Rule Hierarchy for Sales Order Lines

-	Accounting Rule Hierarchy for Sales Order Lines
1	If you referenced an agreement on the order that does not allow override of the accounting rule, Order Management inserts the accounting rule from the associated agreement (OE_AGREEMENTS.ACCOUNTING_RULE_ID); if not, then...
2	If you referenced a commitment on the order line that is associated with an agreement that does not allow override of the accounting rule, Order Management inserts the accounting rule from the agreement (OE_AGREEMENTS.ACCOUNTING_RULE_ID); if not, then...

Table 2–48 Accounting Rule Hierarchy for Sales Order Lines

-	Accounting Rule Hierarchy for Sales Order Lines
3	Line transaction types associated with the order line (OE_TRANSACTION_TYPES.ACCOUNTING_RULE_ID).
4	If you defined an accounting rule for the item, Order Management will use the accounting rule for the item (MTL_SYSTEM_ITEMS.ACCOUNTING_RULE_ID); if not, then...
5	If you referenced a commitment on the order that is associated with an agreement that does allow override of the accounting rule, Order Management inserts the accounting rule from the associated agreement (OE_AGREEMENTS.ACCOUNTING_RULE_ID); if not, then...
6	If you referenced an agreement on the order line that does allow override of the accounting rule, Order Management inserts the accounting rule from the agreement (OE_AGREEMENTS.ACCOUNTING_RULE_ID); if not, then...
7	In all other cases, Order Management inserts the accounting rule on the order.

Invoicing Rules

Order Management determines the invoicing rule for a sales order line based on the following hierarchy:

Table 2–49 Invoicing Rule Hierarchy for Sales Order Lines

-	Invoicing Rule Hierarchy for Sales Order Lines
1	If you referenced an agreement on the order that does not allow override of the invoicing rule, Order Management inserts the invoicing rule from the agreement (OE_AGREEMENTS.INVOICING_RULE_ID); if not, then...
2	If you referenced a commitment on the order line that is associated with an agreement that does not allow override of the invoicing rule, Order Management inserts the invoicing rule from the agreement (OE_AGREEMENTS.INVOICING_RULE_ID); if not, then...

Table 2–49 Invoicing Rule Hierarchy for Sales Order Lines

-	Invoicing Rule Hierarchy for Sales Order Lines
3	Line transaction types associated with the order line (OE_TRANSACTION_TYPES.ACCOUNTING_RULE_ID).
4	If you defined an invoicing rule for the item, Order Management will use the invoicing rule for the item (MTL_SYSTEM_ITEMS.INVOICING_RULE_ID); if not, then...
5	If you referenced a commitment on the order line that is associated with an agreement that does allow override of the invoicing rule, Order Management inserts the invoicing rule from that agreement (OE_AGREEMENTS.INVOICING_RULE_ID); if not, then...
6	If you referenced an agreement on the order that does allow override of the invoicing rule, Order Management inserts the invoicing rule from the agreement. (OE_AGREEMENTS.INVOICING_RULE_ID); if not, then...
7	In all other cases, Order Management inserts the invoicing rule on the order.

Credit Method for Accounting Rule

Order Management transfers a Credit Method for Accounting Rule for each return line. This credit method is recognized only by invoices that use duration accounting rules. You can assign a Credit Method for Accounting Rule to the order type of the return. If the Credit Method for Accounting Rule field for the order type is null, then Order Management transfers LIFO (Last In First Out).

Credit Method for Installments

Order Management transfers a Credit Method for Installments for each return line. This credit method is used for crediting an invoice that uses split payment terms. You can assign a Credit Method for Installments to the order type of the return. If the Credit Method for Installments field for the order type is null, then Order Management transfers LIFO (Last In First Out).

Internal Sales Orders

The Invoicing Activity does not process internal sales order lines, even if it is placed in the internal sales order's workflow.

Internal sales orders are orders that originate in Oracle Purchasing as internal requisitions, and are imported to Order Management as internal sales orders using Order Import.

See:

Using AutoAccounting, *Oracle Receivables User's Guide*

Accounting Rules, *Oracle Receivables User's Guide*

Entering Commitments, *Oracle Receivables User's Guide*

Defining Items, *Oracle Inventory User's Guide*

Accounting for Credit Memos, *Oracle Receivables User's Guide*

Invoicing of ATO Configurations

Invoicing Attributes

For ATO configurations, Order Management considers the base model's item attribute of a configuration to see if it should consider passing invoice information to Oracle Receivables for each order line in the configuration. If you have the item attributes Invoiceable Item and Invoice Enabled set to Yes for the base model item, Order Management then considers these item attributes for each component in the bill of material for the model to see if they should be invoiced in Oracle Receivables. If the item attributes Invoiceable Item or Invoice Enabled are set to No for the base model item, Order Management does not pass invoicing information to Oracle Receivables for any order lines for the components within the configuration, regardless of the item attribute settings.

Required for Revenue Attribute

The bill of material attribute Required for Revenue allows you to define specific items in a bill that must be shipped before their parent can be invoiced. In all cases the control applies to only one level, the immediate parent. Except for classes, the control relationship is the child affecting the parent.

For example: Included Item A has the Required for Revenue attribute set to Yes. Option A is not eligible to interface to Oracle Receivables until Included Item A is shipped, even if Option A is also shippable and has shipped. All other components, including Model A and Model 1, are eligible to interface regardless of Included Item A's shipment status.

Example #2: Included Item B has the Required for Revenue attribute set to Yes. Model 1 is not eligible to interface to Oracle Receivables until Included Item B is shipped. And again, Option A is not eligible to interface until Included Item A is shipped.

Example #3: If any item below a class in a bill has the Required for Revenue attribute set to Yes, then that item must be shipped before the parent item and the other items in the class are eligible to interface. For example, in the figure above, Included Item C has the Required for Revenue attribute set to Yes. Therefore, both Option C and Class C are not eligible to interface until Included Item C is shipped.

See:
Item Attributes Used by Order Management, *Oracle Order Management User's Guide*
Overview of Bills of Material, *Oracle Bills of Material User's Guide*

Understanding the Receivables Interface Tables

Oracle Order Management inserts information into two of the three AutoInvoice interface tables (RA_INTERFACE_LINES and RA_INTERFACE_SALES_CREDITS). RA_INTERFACE_DISTRIBUTIONS is not described in this essay because all account code creation is done by AutoInvoice based on the AutoAccounting rules you have defined. The following describes what information Order Management interfaces for each order and order line, each sales credit, and each freight charge.

Note: Order Management line numbers are populated in the following manner within RA_INTERFACE_LINES::

- INTERFACE_LINE_ATTRIBUTE6 (line_id)
- INTERFACE_LINE_ATTRIBUTE12 (shipment_number)
- INTERFACE_LINE_ATTRIBUTE13 (option_number)
- INTERFACE_LINE_ATTRIBUTE14 (service_number)

Table 2–50 RA_INTERFACE_LINES

Column Name	Null?	Type	Description
INTERFACE_LINE_ID		NUMBER(15)	Order Management does not insert a value into this column.
INTERFACE_LINE_CONTEXT		Varchar2(30)	Order Management inserts your value for the OM: Source Code profile option
INTERFACE_LINE_ATTRIBUTE1		Varchar2(30)	Order Management inserts OE_HEADERS.ORDER_NUMBER.

Table 2–50 RA_INTERFACE_LINES

Column Name	Null?	Type	Description
INTERFACE_LINE_ATTRIBUTE2		Varchar2(30)	Order Management inserts OE_ORDER_TYPES_V.NAME in the base language.
INTERFACE_LINE_ATTRIBUTE3		Varchar2(30)	<i>For a Shipped order line and for Freight Charges:</i> Order Management inserts substr(wsh_new_deliveries.name1..30 <i>For a Non-shipped order line or a Return Line:</i> Order Management inserts 0 (zero).
INTERFACE_LINE_ATTRIBUTE4		Varchar2(30)	<i>For a Shipped order line and for Freight Charges:</i> Order Management inserts: substr(wsh_new_deliveries.waybill1..30 <i>For a Non-shipped order line or a Return Line:</i> Order Management inserts 0 (zero).
INTERFACE_LINE_ATTRIBUTE5		Varchar2(30)	<i>For a Sales order or return line:</i> Order Management inserts the number of times the order or return line has been interfaced for invoice or credit. <i>For Freight charges:</i> Order Management inserts a value of 1(one).
INTERFACE_LINE_ATTRIBUTE6		Varchar2(30)	<i>For a Sales order or return line:</i> Order Management inserts OE_ORDER_LINES.LINE_ID. <i>For Freight charges:</i> OE_CHARGE_LINES_V.CHARGE_ID.
INTERFACE_LINE_ATTRIBUTE7		Varchar2(30)	<i>For a shipped order line, a return line, or for Freight Charges:</i> Order Management inserts zero (0).
INTERFACE_LINE_ATTRIBUTE8		Varchar2(30)	<i>For a shipped order line, a non-shipped order line, a return line, or for Freight Charges:</i> Order Management inserts zero (0).
INTERFACE_LINE_ATTRIBUTE9		Varchar2(30)	Order Management inserts the customer item number, if one is defined. Otherwise, it inserts 0 (zero).
INTERFACE_LINE_ATTRIBUTE10		Varchar2(30)	Order Management inserts OE_ORDER_LINES.SHIP_FROM_ORD_ID.
INTERFACE_LINE_ATTRIBUTE11		Varchar2(30)	Order Management inserts oe_price_adjustments.price_adjustment_id for discount lines: Note: profile (OM: Show Discount Details on Invoice must be Yes.). For all other lines the value 0 is passed.

Table 2–50 RA_INTERFACE_LINES

Column Name	Null?	Type	Description
INTERFACE_LINE_ATTRIBUTE12		Varchar2(30)	Order Management inserts oe_order_lines_all.shipment_number
INTERFACE_LINE_ATTRIBUTE13		Varchar2(30)	Order Management inserts oe_order_lines_all.option_number
INTERFACE_LINE_ATTRIBUTE14		Varchar2(30)	Order Management inserts oe_order_lines_all.service_number
INTERFACE_LINE_ATTRIBUTE15		Varchar2(30)	Order Management does not insert a value into this column
BATCH_SOURCE_NAME	Not Null	Varchar2(50)	Order Management enters the invoice source name using the following rules: If OM: <i>Invoice Numbering Method</i> is set to <i>Delivery Name</i> , but line is non-shippable or a return line, Order Management passes the value of the profile option OM: <i>Non-Delivery Invoice Source</i> . For all other lines, Order Management uses the following sequence to determine the invoice source: 1. Order Transaction Type associated with the order line being interfaced to AutoInvoice. 2. Order Line Transaction Type associated with the order line being interfaced to AutoInvoice. 3. Value of the profile option OM: <i>Invoice Source</i> .
SET_OF_BOOKS_ID	Not Null	NUMBER (15)	Order Management inserts the ID from the OE: Set of Books system parameter for the operating unit of the order line.
LINE_TYPE	Not Null	Varchar2(20)	<i>For a Sales order or return line:</i> Order Management inserts LINE. <i>For Freight charges:</i> Order Management inserts FREIGHT for shipment freight charges if the profile option TAX: <i>Invoice Freight as Revenue</i> is set to No, otherwise the system inserts LINE.

Table 2–50 RA_INTERFACE_LINES

Column Name	Null?	Type	Description
DESCRIPTION	Not Null	Varchar2(240)	<p><i>For a Sales order or return line:</i></p> <p>Order Management inserts MTL_SYSTEM_ITEMS.DESCRPTION for the item <i>or</i> if order line uses item identifiers (item/generic item):</p> <ul style="list-style-type: none"> ■ if a description exists, customer item/generic item description is passed. ■ if the description is null then mtl_system_items_vl.description is passed <p><i>For Freight charges:</i> Order Management inserts description of the change (OE_CHARGE_TYPES.description) for shipment freight charges.</p>
CURRENCY_CODE	Not Null	Varchar2(30)	oe_order_headers.transactiona_curr_code for all shipped/non shipped order/return lines.
AMOUNT		NUMBER	<p><i>For a Sales order or return line:</i> Order Management inserts a calculated amount (OE_ORDER_LINES.UNIT_SELLING_PRICE multiplied by OE_ORDER_LINES.ORDERED_QUANTITY) based on the calculated quantity. Order Management rounds the amount based on the minimum accounting unit and precision associated with the currency of the order.</p> <p>The amount sign will match the sign on the quantity based on the value of RA_CUST_TRX_TYPES.CREATION_SIGN.</p> <p><i>For Freight charges:</i> Order Management inserts: oe_charge_lines_v.charge_amount</p>
CUST_TRX_TYPE_NAME		Varchar2(20)	Order Management does not insert a value into this column
CUST_TRX_TYPE_ID		NUMBER (15)	Please refer to the heading CUST_TRX_TYPE_ID at the end of this table for more information
TERM_NAME		Varchar2(15)	Order Management does not insert a value into this column.

Table 2–50 RA_INTERFACE_LINES

Column Name	Null?	Type	Description
TERM_ID		NUMBER (15)	<p>For a Sales order line: Order Management inserts OE_ORDER_LINES.PAYMENT_TERM_ID.</p> <p>For a Return line: Order Management does not insert a value into this column.</p> <p>For Freight charges: Order Management inserts: oe_order_lines_all.payment_term_id.</p>
ORIG_SYSTEM_BILL_CUSTOMER_REF		Varchar2(240)	Order Management does not insert a value into this column.
ORIG_SYSTEM_BILL_CUSTOMER_ID		NUMBER(15)	<p>Order Management inserts:</p> <pre>SELECT customer_id FROM oe_invoice_to_orgs_v WHERE organization_id = oe_order_lines.invoice_to_org_id</pre>
ORIG_SYSTEM_BILL_ADDRESS_REF		Varchar2(240)	Order Management does not insert a value into this column.
ORIG_SYSTEM_BILL_ADDRESS_ID		NUMBER (15)	<p>Order Management inserts:</p> <pre>SELECT address_id FROM oe_invoice_to_orgs_v WHERE organization_id = oe_order_lines.invoice_to_org_id</pre>
ORIG_SYSTEM_BILL_CONTACT_REF		Varchar2(240)	Order Management does not insert a value into this column.
ORIG_SYSTEM_BILL_CONTACT_ID		NUMBER (15)	Order Management inserts: oe_order_lines.invoice_to_contact_id
ORIG_SYSTEM_SHIP_CUSTOMER_REF		Varchar2(240)	Order Management does not insert a value into this column
ORIG_SYSTEM_SHIP_CUSTOMER_ID		NUMBER (15)	<p>Order Management inserts</p> <pre>SELECT customer_id FROM oe_ship_to_orgs_v WHERE organization_id = oe_order_lines.ship_to_org_id</pre>
ORIG_SYSTEM_SHIP_ADDRESS_REF		Varchar2(240)	Order Management does not insert a value into this column.

Table 2–50 RA_INTERFACE_LINES

Column Name	Null?	Type	Description
ORIG_SYSTEM_SHIP_ADDRESS_ID		NUMBER(15)	<p>For a Sales order or return line: Order Management inserts RA_SITE_USES.ADDRESS_ID FROM RA_ADDRESSES WHERE RA_SITE_USES.SITE_USE_ID = NVL(OE_ORDER_LINES.SHIP_TO_SITE_USE_ID, OE_ORDER_HEADERS.SHIP_TO_SITE_USE_ID))).</p> <p>For Freight charges: Order Management inserts RA_SITE_USES.ADDRESS_ID FROM RA_ADDRESSES WHERE RA_SITE_USES.SITE_USE_ID = OE_ORDER_HEADERS.SHIP_TO_SITE_USE_ID).</p>
ORIG_SYSTEM_SHIP_CONTACT_REF		Vaqrchar2(24)	Order Management does not insert a value into this column.
ORIG_SYSTEM_SHIP_CONTACT_ID		NUMBER(15)	Order Management inserts: oe_order_lines.ship_to_contact_id
ORIG_SYSTEM_SOLD_CUSTOMER_REF		Varchar2(240)	Order Management does not insert a value into this column.
ORIG_SYSTEM_SOLD_CUSTOMER_ID		NUMBER(15)	Order Management inserts OE_ORDER_HEADERS.SOLD_TO_ORG_ID.
LINK_TO_LINE_ID		NUMBER(15)	Order Management does not insert a value into this column.
LINK_TO_LINE_CONTEXT		Vharchar2(30)	Order Management does not insert a value into this column.
LINK_TO_LINE_ATTRIBUTE1 - 15		Vharchar2(30)	Order Management does not insert <i>any</i> values into <i>any</i> of these columns.
RECEIPT_METHOD_NAME		Vharchar2(30)	Order Management does not insert a value into this column.
RECEIPT_METHOD_ID		NUMBER(15)	Order Management does not insert a value into this column.
CONVERSION_TYPE	Not NULL	Varchar2(30)	Order Management inserts NVL(OE_ORDER_HEADERS.CONVERSION_TYPE, <i>User</i>).
CONVERSION_DATE		Date	Order Management inserts OE_ORDER_HEADERS.CONVERSION_DATE.
CONVERSION_RATE		NUMBER	Order Management inserts OE_ORDER_HEADERS.CONVERSION_RATE.
CUSTOMER_TRX_ID		NUMBER(15)	Order Management does not insert a value into this column.

Table 2–50 RA_INTERFACE_LINES

Column Name	Null?	Type	Description
TRX_DATE		Date	Order Management does not insert a value into this column.
GL_DATE		Date	Order Management does not insert a value into this column.
DOCUMENT_NUMBER		NUMBER(15)	Order Management does not insert a value into this column.
TRX_NUMBER		Varchar2(240)	If the OM: Invoice Numbering Method profile is set to Automatic, AutoInvoice determines a unique number for this transaction. If the profile is set to <i>Delivery Name</i> , Order Management inserts a delivery name. An index is appended if the delivery has more than one invoice. For example, Order Management might insert <i>delivery</i> for the first invoice, <i>delivery-1</i> for the second, <i>delivery-2</i> for the third, and so on.
LINE_NUMBER		NUMBER(15)	Order Management does not insert a value into this column.
QUANTITY		NUMBER	Order Management inserts a calculated quantity based on the type of line being interfaced. This calculation is based on rules for fulfillment, over and under shipments, required for revenue and what was previously invoiced. <i>For a Sales order line:</i> The quantity will be either negative or positive, depending on the value of RA_CUST_TRX_TYPES.CREATION_SIGN associated with the invoice type for the sales order. If RA_CUST_TRX_TYPES.CREATION_SIGN is N, then quantity passed is -1 multiplied by the quantity calculated.
QUANTITY_ORDERED		NUMBER	<i>For a Sales order line or return line:</i> Order Management inserts OE_ORDER_LINES.ORDERED_QUANTITY. <i>For Freight charges:</i> Order Management does not insert a value into this column
UNIT_SELLING_PRICE		NUMBER	<i>For a Sales order line or return lines:</i> Order Management inserts OE_ORDER_LINES.UNIT_SELLING_PRICE. <i>For Freight charges:</i> Order Management does not insert a value into this column.

Table 2–50 RA_INTERFACE_LINES

Column Name	Null?	Type	Description
UNIT_SELLING_PRICE_PER_PQTY		NUMBER	Unit Selling Price Per Pricing Quantity.
UNIT_STANDARD_PRICE		NUMBER	<i>For a Sales order line or return line:</i> Order Management inserts OE_ORDER_LINES.UNIT_LIST_PRICE. <i>For Freight charges:</i> Order Management does not insert a value into this column.
PRINTING_OPTION		Varchar2(20)	Order Management does not insert a value into this column.
INTERFACE_STATUS		Varchar2(1)	Order Management does not insert a value into this column.
REQUEST_ID		NUMBER(15)	Order Management does not insert a value into this column.
RELATED_BATCH_SOURCE_NAME		Varchar2(50)	Order Management does not insert a value into this column.
RELATED_TRX_NUMBER		Varchar2(20)	Order Management does not insert a value into this column.
RELATED_CUSTOMER_TRX_ID		NUMBER(15)	Order Management does not insert a value into this column.
PREVIOUS_CUSTOMER_TRX_ID		NUMBER(15)	Order Management does not insert a value into this column.
CREDIT_METHOD_FOR_ACCT_RULE		Varchar2(30)	<i>For a Sales order line:</i> Order Management does not insert a value into this column. <i>For a Return line:</i> Order Management inserts from based upon the following hierarchy: 1. Line transaction type 2. Order transaction type 3. else insert constant value <i>LIFO</i>
CREDIT_METHOD_FOR_INSTALLMENTS		Varchar2(30)	<i>For a Sales order line:</i> Order Management does not insert a value into this column. <i>For a Return line:</i> Order Management inserts from based upon the following hierarchy: 1. Line transaction type 2. Order transaction type

Table 2–50 RA_INTERFACE_LINES

Column Name	Null?	Type	Description
REASON_CODE		Varchar2(30)	<i>For a Sales order line:</i> Order Management does not insert a value into this column. <i>For a Return line:</i> Order Management inserts OE_ORDER_LINES.RETURN_REASON_CODE
TAX_RATE		NUMBER	Order Management does not insert a value into this column.
TAX_CODE		Varchar2(50)	If OM: Tax: Allow Override of Tax Code = Yes, then pass oe_order_lines.tax_code
TAX_PRECEDENCE		NUMBER(15)	Order Management does not insert a value into this column.
EXCEPTION_ID		NUMBER(15)	Order Management does not insert a value into this column.
EXEMPTION_ID		NUMBER(15)	Order Management does not insert a value into this column.
SHIP_DATE_ACTUAL		Date	<i>For a Shipped order lines and for Freight charge lines:</i> Order Management inserts: wsh_new_deliveries.initial_pickup_date. <i>For a Non-shipped order line:</i> Order Management does not insert a value into this column. <i>For Freight charges:</i> Order Management inserts OE_ORDER_LINE_ACTUAL_SHIPMENT_DATE.
FOB_POINT		Varchar2(20)	Order Management only populates this column if the order line being invoiced has been shipped. <i>For a Shipped order line:</i> Order Management inserts OE_ORDER_LINES.FOB_POINT_CODE. <i>For a Non-shipped order line:</i> Order Management does not insert a value into this column. <i>For Freight charges:</i> Order Management inserts OE_ORDER_LINES.FOB_POINT_CODE.
SHIP_VIA		Varchar2(20)	Either nvl(wsh_carrier_ship_methods_v.freight_code or oe_order_lines.freight_carrier_code)
WAYBILL_NUMBER		Varchar2(50)	<i>For a Shipped order line and For Freight charges:</i> Order Management inserts: substr(wsh_new_deliveries.waybill,1,30) <i>For a Non-shipped order line:</i> Order Management does not insert a value into this column.

Table 2–50 RA_INTERFACE_LINES

Column Name	Null?	Type	Description
INVOICING_RULE_NAME		Varchar2(30)	Order Management does not insert a value into this column.
INVOICING_RULE_ID		NUMBER(15)	<i>For a Sales order line and for Freight charge lines:</i> Order Management inserts DECODE(ACCOUNTING_RULE_ID,1, NULL,INVOICING_RULE_ID). <i>For a Non-shipped order line:</i> Order Management does not insert a value into this column.
ACCOUNTING_RULE_NAME		Varchar2(30)	Order Management does not insert a value into this column.
ACCOUNTING_RULE_ID		NUMBER(15)	<i>For a Sales order line and for Freight charge lines:</i> Order Management inserts DECODE(ACCOUNTING_RULE_ID,1, NULL,ACCOUNTING_RULE_ID). <i>For a Return line:</i> Order Management does not insert a value into this column.
ACCOUNTING_RULE_DURATION		NUMBER(15)	If the accounting rule is of type <i>Variable Duration</i> then insert <i>Service Duration</i> for Service lines.
RULE_START_DATE		Date	Order Management does not insert a value into this column.
PRIMARY_SALESREP_NUMBER		Varchar2(30)	Order Management does not insert a value into this column.
PRIMARY_SALESREP_ID		NUMBER(15)	Order Management inserts NVL (OE_ORDER_LINES.SALESREP_ID, OE_ORDER_HEADERS.SALESREP_ID).
SALES_ORDER		Varchar2(50)	Order Management inserts OE_ORDER_HEADERS.ORDER_NUMBER.
SALES_ORDER_LINE		Varchar2(30)	<i>For a Sales order or return line:</i> Order Management inserts OE_ORDER_LINES.LINE_NUMBER. <i>For Freight charges:</i> Order Management does not insert a value into this column.
SALES_ORDER_DATE		Date	Order Management inserts OE_ORDER_HEADERS.ORDERED_DATE.
SALES_ORDER_SOURCE		Varchar2(50)	Order Management inserts your value for the OM: <i>Source Code</i> profile option.
SALES_ORDER_REVISION		NUMBER	Order Management does not insert a value into this column.

Table 2–50 RA_INTERFACE_LINES

Column Name	Null?	Type	Description
PURCHASE_ORDER		Varchar2(50)	Order Management inserts OE_ORDER_HEADERS.LINES.CUST_PO_NUMBER.
PURCHASE_ORDER_REVISION		Varchar2(50)	Order Management does not insert a value into this column.
PURCHASE_ORDER_DATE		Date	Order Management does not insert a value into this column.
AGREEMENT_NAME		Varchar2(30)	Order Management does not insert a value into this column.
AGREEMENT_ID		NUMBER(15)	<p>For a Sales order line: Order Management inserts OE_ORDER_LINES.AGREEMENT_ID.</p> <p>For a Return line with a purchase order or sales order reference: Order Management inserts the AGREEMENT_ID from the referenced order line.</p> <p>For return lines with an invoice reference: Order Management does not insert a value into this column.</p> <p>For Freight charges: Order Management does not insert a value into this column.</p>
MEMO_LINE_NAME		Varchar2(50)	Order Management does not insert a value into this column.
MEMO_LINE_ID		NUMBER(15)	Order Management does not insert a value into this column.
INVENTORY_ITEM_ID		NUMBER(15)	<p>For a Sales order or return line: Order Management inserts OE_ORDER_LINES.INVENTORY_ITEM_ID.</p> <p>For Freight charges: Order Management does not insert a value into this column. If the profile option TAX: Invoice Freight As Revenue is set to Yes, the value of the profile option TAX: Inventory Item For Freight is used.</p>
MTL_SYSTEM_ITEMS_SEG1.....20		Varchar2(30)	Order Management does not insert values into any of these columns.

Table 2–50 RA_INTERFACE_LINES

Column Name	Null?	Type	Description
REFERENCE_LINE_ID		NUMBER(15)	<p>For a Sales order line: Order Management inserts OE_ORDER_LINES.COMMITMENT_ID.</p> <p>For a Return line: Order Management inserts OE_ORDER_LINES.CREDIT_INVOICE_LINE_ID.</p> <p>For Freight charges: Order Management does not insert a value into this column.</p>
REFERENCE_LINE_CONTEXT		Varchar2(30)	Order Management does not insert a value into this column.
REFERENCE_LINE_ATTRIBUTE1....15		Varchar2(30)	Order Management does not insert a value into this column.
TERRITORY_ID		NUMBER(15)	Order Management does not insert a value into this column.
TERRITORY_SEGMENT1....20		Varchar2(25)	Order Management does not insert a value into this column.
ATTRIBUTE_CATEGORY		Varchar2(30)	<p>For a Sales order or return line: Order Management inserts OE_ORDER_LINES.CONTEXT.</p> <p>For Freight charges: Order Management inserts OE_CHARGE_LINES_V.CONTEXT.</p>
ATTRIBUTE1...15		Varchar2(150)	<p>For a Sales order or return line: Order Management inserts OE_ORDER_LINES.ATTRIBUTE1-15.</p> <p>For Freight charges: Order Management inserts OE_CHARGE_LINES_V.ATTRIBUTE1-15.</p>
HEADER_ATTRIBUTE_CATEGORY		Varchar2(30)	<p>For Sales order or return line: Order Management inserts OE_ORDER_HEADERS.CONTEXT.</p> <p>For Freight charges: Order Management does not insert a value into this column.</p>
HEADER_ATTRIBUTE1...15		Varchar2(150)	<p>For a Sales order or return line: Order Management inserts OE_ORDER_LINES.ATTRIBUTE1-15.</p> <p>For Freight charges: Order Management inserts oe_order_headers.attribute1-15.</p>
COMMENTS		Varchar2(240)	Order Management does not insert a value into this column.
INTERNAL_NOTES		Varchar2(240)	Order Management does not insert a value into this column.

Table 2–50 RA_INTERFACE_LINES

Column Name	Null?	Type	Description
INITIAL_CUSTOMER_TRX_ID		NUMBER(15)	Order Management does not insert a value into this column.
USSGL_TRANSACTION_CODE_CONTEXT		Varchar2(30)	Order Management does not insert a value into this column.
USSGL_TRANSACTION_CODE		Varchar2(30)	Order Management does not insert a value into this column.
ACCTD_AMOUNT		NUMBER	Order Management does not insert a value into this column.
CUSTOMER_BANK_ACCOUNT_ID		NUMBER(15)	Order Management does not insert a value into this column.
CUSTOMER_BANK_ACCOUNT_NAME		Varchar2(25)	Order Management does not insert a value into this column.
UOM_CODE		Varchar2(3)	<i>For a Sales order or return line:</i> Order Management inserts OE_ORDER_LINES.SHIPPING_QUANTITY_UOM. <i>For Freight charges:</i> Order Management does not insert a value into this column.
UOM_NAME		Varchar2(25)	Order Management does not insert a value into this column.
DOCUMENT_NUMBER_SEQUENCE_ID		NUMBER(15)	Order Management does not insert a value into this column.
REASON_CODE_NAME		Varchar2(30)	Order Management does not insert a value into this column.
VAT_TAX_ID		NUMBER(15)	Order Management does not insert a value into this column.
LOCATION_RATE_ID		NUMBER(15)	Order Management does not insert a value into this column.
REASON_CODE_MEANING		Varchar2(80)	Order Management does not insert a value into this column.
LAST_PERIOD_TO_CREDIT		NUMBER	Order Management does not insert a value into this column.
PAYING_CUSTOMER_ID		NUMBER(15)	Order Management does not insert a value into this column.
PAYING_SITE_USE_ID		NUMBER(15)	Order Management does not insert a value into this column.

Table 2–50 RA_INTERFACE_LINES

Column Name	Null?	Type	Description
TAX_EXEMPT_FLAG		Varchar2(10)	Order Management inserts OE_HEADERS.TAX_EXEMPT_FLAG for order lines.
SALES_TAX_ID		NUMBER(15)	Order Management does not insert a value into this column.
CREATED_BY		NUMBER(15)	Order Management enters an identification number to identify the user who created the record. No Validation occurs
CREATION_DATE		Date	Order Management enters the current system date when a record is created.
LAST_UPDATED_BY		NUMBER(15)	Order Management enters an identification number to identify the user who created or who most recently modified the record. No Validation occurs
LAST_UPDATE_DATE		Date	Order Management enters the current date when a record is updated. Standard Validation occurs
LOCATION_SEGMENT_ID		NUMBER(15)	Order Management does not insert a value into this column.
TAX_EXEMPT_REASON_CODE		Varchar2(30)	Order Management inserts OE_ORDER_LINE.TAX_EXEMPT_REASON_CODE for order lines.
TAX_EXEMPT_NUMBER		Varchar2(80)	Order Management inserts OE_ORDER_LINE.TAX_EXEMPT_NUMBER for order lines.
TAX_EXEMPT_REASON_CODE_MEANING		Varchar2(80)	Order Management does not insert a value into this column.

Derived Values**CUST_TRX_TYPE_ID**

Value is determined based upon the following hierarchy:

For a Sales order line:

- i. Line transaction type
- j. Order Transaction type

k. Profile *OM: Invoice Transaction type*

For referenced return line: Retrieve receivables transaction type from:

- a. Line transaction type
- b. order transaction type only if order_category is RETURN.
- c. Derive credit memo type based on return context and reference information.
- d. profile *OM: Credit memo Transaction Type*

For non-referenced return line:

- a. Line transaction type
- b. order transaction type only if order_category is RETURN.
- c. Derive credit memo type based on return context and reference information.
- d. profile *OM: Credit memo Transaction Type*

For freight charges:

Freight charges are value is the same as the order/return line that is interfacing the freight charge.

RA_INTERFACE_SALESCREDITS

Order Management inserts one row for each sales credit row according to the following hierarchy:

1. Insert sales credits associated with the line; if none exists but the line is part of a configuration (ITEM_TYPE_CODE is CLASS, KIT or STANDARD and OPTION_FLAG is Y), then;
2. Insert sales credits associated with the model *parent* line; if none exists, then;
 - Insert sales credits associated with the order header.

Table 2–51 RA_INTERFACE_SALESCREDITS

Column	Null?	Type	Description
INTERFACE_SALESCREDIT_ID		NUMBER(15)	Order Management does not insert a value into this column.
INTERFACE_LINE_ID		NUMBER(15)	Order Management does not insert a value into this column.
INTERFACE_LINE_CONTEXT		Varchar2(30)	Order Management inserts your value for the OM: Source Code profile option.
INTERFACE_LINE_ATTRIBUTE1		Varchar2(30)	Order Management inserts OE_ORDER_LINE.ORDER_NUMBER.
INTERFACE_LINE_ATTRIBUTE2		Varchar2(30)	Order Management inserts OE_TRANSACTION_TYPES.NAME.
INTERFACE_LINE_ATTRIBUTE3		Varchar2(30)	For a Shipped order line and for Freight Charges: Order Management inserts substr(wsh_new_deliveries.name1..30 For a Non-shipped order line or a Return Line: Order Management inserts 0 (zero).
INTERFACE_LINE_ATTRIBUTE4		Varchar2(30)	For a Shipped order line and for Freight Charges: Order Management inserts: substr(wsh_new_deliveries.waybill1..30 For a Non-shipped order line or a Return Line: Order Management inserts 0 (zero). Order Management inserts SUBSTR (WSH_DELIVERY_LINE_STATUS_V.WAYBILL, 1, 30).
INTERFACE_LINE_ATTRIBUTE5		Varchar2(30)	Order Management inserts the number of times the order, freight charge, or return line has been interfaced for invoice or credit.
INTERFACE_LINE_ATTRIBUTE6		Varchar2(30)	For a Sales order or return line: Order Management inserts OE_ORDER_LINES.LINE_ID. For Freight charges: OE_CHARGE_LINES_V.CHARGE_ID.
INTERFACE_LINE_ATTRIBUTE7		Varchar2(30)	For a Shipped order line, a Return line, or Freight charges: Order Management inserts 0 (zero).

Table 2–51 RA_INTERFACE_SALESCREDITS

Column	Null?	Type	Description
INTERFACE_LINE_ATTRIBUTE8		Varchar2(30)	<i>For a Shipped order line:</i> Order Management inserts WSH_DEPARTURES. BILL_OF_LADING. <i>For a Return line, or Freight charges:</i> Order Management inserts 0 (zero)
INTERFACE_LINE_ATTRIBUTE9		Varchar2(30)	Order Management inserts the customer item number if one is defined. Otherwise, it inserts 0 (zero).
INTERFACE_LINE_ATTRIBUTE10		Varchar2(30)	Order Management inserts OE_ORDER_LINES. SHIP_FROM_ORG_ID.
INTERFACE_LINE_ATTRIBUTE11		Varchar2(30)	oe_price_adjustments.price_adjustment_id for discount lines: profile (OM: Show Discount Details on Invoice must be Yes.). For all other lines '0' is passed.
INTERFACE_LINE_ATTRIBUTE12....15		Varchar2(30)	Order Management does not insert a value into this column.
SALESREP_NUMBER		Varchar2(30)	Order Management does not insert a value into this column.
SALESREP_ID		NUMBER(15)	Order Management inserts OE_SALES_CREDIT.SALESREP_ID.
SALES_CREDIT_TYPE_NAME			Order Management does not insert a value into this column.
SALES_CREDIT_TYPE_ID		NUMBER(15)	Order Management inserts OE_SALES_CREDIT.SALES_CREDIT_TYPE_ID.
SALES_CREDIT_AMOUNT_SPLIT		NUMBER	Order Management does not insert a value into this column.
SALES_CREDIT_PERCENT_SPLIT		NUMBER	Order Management inserts OE_SALES_CREDIT.PERCENT.
INTERFACE_STATUS		Varchar(1)	Order Management does not insert a value into this column.

Table 2–51 RA_INTERFACE_SALESCREDITS

Column	Null?	Type	Description
REQUEST_ID		NUMBER(15)	Order Management does not insert a value into this column.
ATTRIBUTE_CATEGORY		Varchar(30)	Order Management inserts OE_SALES_CREDIT.CONTEXT.
ATTRIBUTE1...15		Varchar(150)	Order Management inserts OE_SALES_CREDIT.ATTRIBUTE1....15.

Release Management Open Interface

Topics covered in the Oracle Release Management Open Interface are:

- [Understanding the Interface Tables](#) on page 3-2
- [RLM_INTERFACE_HEADERS_ALL](#) on page 3-2
- [RLM_INTERFACE_LINES_ALL](#) on page 3-22

Understanding the Interface Tables

Oracle Release Management uses two Oracle tables in which the Demand Processor receives data that you import from other systems.

- RLM_INTERFACE_HEADERS_ALL
- RLM_INTERFACE_LINES_ALL

When the Demand Processor receives data, it validates and converts your import data into customer demand schedules within Oracle Release Management and order lines within Oracle Order Management.

RLM_INTERFACE_HEADERS_ALL

This table stores the header level details of the Release Management schedules. It has details pertaining to the specific schedule and customer level information. The Release Management Demand Processor uses this table to maintain a current picture of demand schedule headers.

Table 3–1 RLM_INTERFACE_HEADERS_ALL

Column Name	Type	Required	Derived	Optional
HEADER_ID	NUMBER	X	-	-
CUST_ADDRESS_1_EXT	VARCHAR2(35)	-	-	X
CUST_ADDRESS_2_EXT	VARCHAR2(35)	-	-	X
CUST_ADDRESS_3_EXT	VARCHAR2(35)	-	-	X
CUST_ADDRESS_4_EXT	VARCHAR2(35)	-	-	X
CUST_ADDRESS_5_EXT	VARCHAR2(35)	-	-	X
CUST_ADDRESS_6_EXT	VARCHAR2(35)	-	-	X
CUST_ADDRESS_7_EXT	VARCHAR2(35)	-	-	X
CUST_ADDRESS_8_EXT	VARCHAR2(35)	-	-	X
CUST_ADDRESS_9_EXT	VARCHAR2(35)	-	-	X
CUST_CITY_EXT	VARCHAR2(30)	-	-	X
CUST_COUNTRY_EXT	VARCHAR2(3)	-	-	X
CUST_COUNTY_EXT	VARCHAR2(25)	-	-	X
CUSTOMER_EXT	VARCHAR2(30)	-	-	X

Table 3–1 RLM_INTERFACE_HEADERS_ALL

Column Name	Type	Required	Derived	Optional
CUST_NAME_EXT	VARCHAR2(20)	-	-	X
ONE_TIME_CUST_FLAG_EXT	VARCHAR2(35)	-	-	X
CURRENCY_EXT	VARCHAR2(35)	-	-	X
TAX_EXEMPT_FLAG_EXT	VARCHAR2(35)	-	-	X
TAX_ID_EXT	VARCHAR2(35)	-	-	X
CUST_POSTAL_CD_EXT	VARCHAR2(15)	-	-	X
CUST_PROVINCE_EXT	VARCHAR2(10)	-	-	X
CUST_STATE_EXT	VARCHAR2(10)	-	-	X
CUSTOMER_ID	NUMBER	-	-	X
ECE_PRIMARY_ADDRESS_ID	NUMBER	-	-	X
ECE_TP_LOCATION_CODE_EXT	VARCHAR2(35)	-	-	X
ECE_TP_TRANSLATOR_CODE	VARCHAR2(35)	-	-	X
EDI_CONTROL_NUM_1	VARCHAR2(15)	-	-	X
EDI_CONTROL_NUM_2	VARCHAR2(15)	-	-	X
EDI_CONTROL_NUM_3	VARCHAR2(15)	-	-	X
EDI_TEST_INDICATOR	VARCHAR2(1)	-	-	X
HEADER_CONTACT_CODE_1	VARCHAR2(3)	-	-	X
HEADER_CONTACT_CODE_2	VARCHAR2(3)	-	-	X
HEADER_CONTACT_VALUE_1	VARCHAR2(80)	-	-	X
HEADER_CONTACT_VALUE_2	VARCHAR2(80)	-	-	X
HEADER_CONTACT_EMAIL_1_EXT	VARCHAR2(50)	-	-	X
HEADER_CONTACT_EMAIL_2_EXT	VARCHAR2(50)	-	-	X
HEADER_CONTACT_FAX_1_EXT	VARCHAR2(35)	-	-	X
HEADER_CONTACT_FAX_2_EXT	VARCHAR2(35)	-	-	X
HEADER_CONTACT_TEL_1_EXT	VARCHAR2(50)	-	-	X
HEADER_CONTACT_TEL_2_EXT	VARCHAR2(80)	-	-	X

Table 3–1 RLM_INTERFACE_HEADERS_ALL

Column Name	Type	Required	Derived	Optional
HEADER_NOTE_TEXT	VARCHAR2(4000)	-	-	X
HEADER_REF_CODE_1	VARCHAR2(3)	-	-	X
HEADER_REF_CODE_2	VARCHAR2(3)	-	-	X
HEADER_REF_CODE_3	VARCHAR2(3)	-	-	X
HEADER_REF_VALUE_1	VARCHAR2(35)	-	-	X
HEADER_REF_VALUE_2	VARCHAR2(35)	-	-	X
HEADER_REF_VALUE_3	VARCHAR2(35)	-	-	X
ORG_ID	NUMBER	-	-	X
PROCESS_STATUS	NUMBER	-	-	X
SCHEDULE_HEADER_ID	NUMBER	-	-	X
SCHEDULE_TYPE	VARCHAR2(30)	-	X	-
SCHEDULE_TYPE_EXT	VARCHAR2(30)	-	-	X
SCHED_GENERATION_DATE	DATE	-	-	X
SCHED_HORIZON_END_DATE	DATE	-	-	X
SCHED_HORIZON_START_DATE	DATE	-	-	X
SCHEDULE_PURPOSE	VARCHAR2(30)	-	X	-
SCHEDULE_PURPOSE_EXT	VARCHAR2(30)	-	-	X
SCHEDULE_REFERENCE_NUM	VARCHAR2(35)	-	-	X
SCHEDULE_SOURCE	VARCHAR2(30)	-	-	X
LAST_UPDATE_DATE	DATE	-	X	-
LAST_UPDATED_BY	NUMBER	-	X	-
CREATION_DATE	DATE	-	X	-
CREATED_BY	NUMBER	-	X	-
ATTRIBUTE_CATEGORY	VARCHAR2(30)	-	-	X
ATTRIBUTE1	VARCHAR2(150)	-	-	X
ATTRIBUTE2	VARCHAR2(150)	-	-	X
ATTRIBUTE3	VARCHAR2(150)	-	-	X

Table 3–1 RLM_INTERFACE_HEADERS_ALL

Column Name	Type	Required	Derived	Optional
ATTRIBUTE4	VARCHAR2(150)	-	-	X
ATTRIBUTE5	VARCHAR2(150)	-	-	X
ATTRIBUTE6	VARCHAR2(150)	-	-	X
ATTRIBUTE7	VARCHAR2(150)	-	-	X
ATTRIBUTE8	VARCHAR2(150)	-	-	X
ATTRIBUTE9	VARCHAR2(150)	-	-	X
ATTRIBUTE10	VARCHAR2(150)	-	-	X
ATTRIBUTE11	VARCHAR2(150)	-	-	X
ATTRIBUTE12	VARCHAR2(150)	-	-	X
ATTRIBUTE13	VARCHAR2(150)	-	-	X
ATTRIBUTE14	VARCHAR2(150)	-	-	X
ATTRIBUTE15	VARCHAR2(150)	-	-	X
LAST_UPDATE_LOGIN	NUMBER	-	X	-
REQUEST_ID	NUMBER	-	X	-
PROGRAM_APPLICATION_ID	NUMBER	-	X	-
PROGRAM_ID	NUMBER	-	X	-
PROGRAM_UPDATE_DATE	DATE	-	X	-
TP_ATTRIBUTE1	VARCHAR2(150)	-	-	X
TP_ATTRIBUTE2	VARCHAR2(150)	-	-	X
TP_ATTRIBUTE3	VARCHAR2(150)	-	-	X
TP_ATTRIBUTE4	VARCHAR2(150)	-	-	X
TP_ATTRIBUTE5	VARCHAR2(150)	-	-	X
TP_ATTRIBUTE6	VARCHAR2(150)	-	-	X
TP_ATTRIBUTE7	VARCHAR2(150)	-	-	X
TP_ATTRIBUTE8	VARCHAR2(150)	-	-	X
TP_ATTRIBUTE9	VARCHAR2(150)	-	-	X
TP_ATTRIBUTE10	VARCHAR2(150)	-	-	X

Table 3–1 RLM_INTERFACE_HEADERS_ALL

Column Name	Type	Required	Derived	Optional
TP_ATTRIBUTE11	VARCHAR2(150)	-	-	X
TP_ATTRIBUTE12	VARCHAR2(150)	-	-	X
TP_ATTRIBUTE13	VARCHAR2(150)	-	-	X
TP_ATTRIBUTE14	VARCHAR2(150)	-	-	X
TP_ATTRIBUTE15	VARCHAR2(150)	-	-	X
TP_ATTRIBUTE_CATEGORY	VARCHAR2(30)	-	-	X
CUST_FAX_EXT	VARCHAR2(35)	-	-	X
CUST_REGION_EXT	VARCHAR2(35)	-	-	X
CUST_TAX_JURISDICTION_EXT	VARCHAR2(35)	-	-	X
CUST_TEL_EXT	VARCHAR2(35)	-	-	X
CUST_URL_EXT	VARCHAR2(100)	-	-	X
ORIGINAL_DOCUMENT_DATE	DATE	-	-	X
RESP_DOCUMENT_DATE	DATE	-	-	X
DOCUMENT_DESCRIPTION	VARCHAR2(50)	-	-	X
DOCUMENT_REV_NUM	VARCHAR2(35)	-	-	X
RESP_SCHEDULE_ID	VARCHAR2(35)	-	-	X
RESP_DOCUMENT_REV_NUM	VARCHAR2(35)	-	-	X
BUCKET_END_DATE	DATE	-	-	X
BUCKET_START_DATE	DATE	-	-	X
FLEX_BUCKET_CODE	VARCHAR2(30)	-	-	X
SUPPLIER_NAME_EXT	VARCHAR2(60)	-	-	X
ONE_TIME_SUPPLIER_FLAG_EXT	VARCHAR2(35)	-	-	X
SUPPLIER_ID	NUMBER	-	-	X
SUPPLIER_CURRENCY_EXT	VARCHAR2(35)	-	-	X
SUPPLIER_DESCRIPTION_EXT	VARCHAR2(35)	-	-	X
CUST_DESCRIPTION_EXT	VARCHAR2(35)	-	-	X
SUPPLIER_TAX_EXEMPT_FLAG_EXT	VARCHAR2(35)	-	-	X

Table 3–1 RLM_INTERFACE_HEADERS_ALL

Column Name	Type	Required	Derived	Optional
SUPPLIER_TAX_ID_EXT	VARCHAR2(35)	-	-	X
SUPPLIER_ADDRESS_1_EXT	VARCHAR2(35)	-	-	X
SUPPLIER_ADDRESS_2_EXT	VARCHAR2(35)	-	-	X
SUPPLIER_ADDRESS_3_EXT	VARCHAR2(35)	-	-	X
SUPPLIER_ADDRESS_4_EXT	VARCHAR2(35)	-	-	X
SUPPLIER_ADDRESS_5_EXT	VARCHAR2(35)	-	-	X
SUPPLIER_ADDRESS_6_EXT	VARCHAR2(240)	-	-	X
SUPPLIER_ADDRESS_7_EXT	VARCHAR2(35)	-	-	X
SUPPLIER_ADDRESS_8_EXT	VARCHAR2(35)	-	-	X
SUPPLIER_ADDRESS_9_EXT	VARCHAR2(35)	-	-	X
SUPPLIER_CITY_EXT	VARCHAR2(30)	-	-	X
SUPPLIER_COUNTY_EXT	VARCHAR2(35)	-	-	X
SUPPLIER_FAX_EXT	VARCHAR2(35)	-	-	X
SUPPLIER_POSTAL_CD_EXT	VARCHAR2(15)	-	-	X
SUPPLIER_REGION_EXT	VARCHAR2(35)	-	-	X
SUPPLIER_STATE_EXT	VARCHAR2(10)	-	-	X
SUPPLIER_PROVINCE_EXT	VARCHAR2(10)	-	-	X
SUPPLIER_TAX_JURISDICTION_EXT	VARCHAR2(35)	-	-	X
SUPPLIER_TEL_EXT	VARCHAR2(35)	-	-	X
SUPPLIER_URL_EXT	VARCHAR2(100)	-	-	X
SUP_HEADER_CONTACT_CODE_1	VARCHAR2(3)	-	-	X
SUP_HEADER_CONTACT_CODE_2	VARCHAR2(3)	-	-	X
SUP_HEADER_CONTACT_VALUE_1	VARCHAR2(80)	-	-	X
SUP_HEADER_CONTACT_VALUE_2	VARCHAR2(80)	-	-	X
SUP_HEADER_CONTACT_EMAIL_1_EXT	VARCHAR2(50)	-	-	X

Table 3–1 RLM_INTERFACE_HEADERS_ALL

Column Name	Type	Required	Derived	Optional
SUP_HEADER_CONTACT_EMAIL_2_EXT	VARCHAR2(50)	-	-	X
SUP_HEADER_CONTACT_FAX_1_EXT	VARCHAR2(35)	-	-	X
SUP_HEADER_CONTACT_FAX_2_EXT	VARCHAR2(35)	-	-	X
SUP_HEADER_CONTACT_TEL_1_EXT	VARCHAR2(50)	-	-	X
SUP_HEADER_CONTACT_TEL_2_EXT	VARCHAR2(50)	-	-	X
SUPPLIER_COUNTRY_EXT	VARCHAR2(3)	-	-	X

Column Descriptions

HEADER_ID	Number
Sequence generated unique identifier.	
CUST_ADDRESS_1_EXT	VARCHAR2(35)
Customer address line 1, as sent by the customer on the N3 segment.	
CUST_ADDRESS_2_EXT	VARCHAR2(35)
Customer address line 2, as sent by the customer on the N3 segment.	
CUST_ADDRESS_3_EXT	VARCHAR2(35)
Customer address line 3, as sent by the customer on the N3 segment.	
CUST_ADDRESS_4_EXT	VARCHAR2(35)
Customer address line 4, as sent by the customer on the N3 segment.	
CUST_ADDRESS_5_EXT	VARCHAR2(35)
Customer address line 5, as sent by the customer on the N3 segment.	

CUST_ADDRESS_6_EXT	VARCHAR2(35)
Customer address line 6, as sent by the customer on the N3 segment.	
CUST_ADDRESS_7_EXT	VARCHAR2(35)
Customer address line 7, as sent by the customer on the N3 segment.	
CUST_ADDRESS_8_EXT	VARCHAR2(35)
Customer address line 8, As sent by the customer on the N3 segment.	
CUST_ADDRESS_9_EXT	VARCHAR2(35)
Customer address line 9, as sent by the customer on the N3 segment.	
CUST_CITY_EXT	VARCHAR2(30)
Customer address city, as sent by the customer on the N4 segment.	
CUST_COUNTRY_EXT	VARCHAR2(3)
Customer country, as sent by the customer on the N4 segment.	
CUST_COUNTY_EXT	VARCHAR2(25)
Customer county, as sent by the customer on the N4 segment.	
CUSTOMER_EXT	VARCHAR2(30)
Trading Partner designator cross reference.	
CUST_NAME_EXT	VARCHAR2(60)
Customer name, as sent by the customer in the N1 or N2 segment.	
ONE_TIME_CUST_FLAG_EXT	VARCHAR2(35)
For future use.	
CURRENCY_EXT	VARCHAR2(35)
For future use.	
TAX_EXEMPT_FLAG_EXT	VARCHAR2(35)
For future use.	

TAX_ID_EXT **VARCHAR2(35)**

For future use.

CUST_POSTAL_CD_EXT **VARCHAR2(15)**

Customer address postal code, as sent by the customer on the N4 segment.

CUST_PROVINCE_EXT **VARCHAR2(10)**

Customer province, as sent by the customer on the N4 segment.

CUST_STATE_EXT **VARCHAR2(10)**

Customer state, as sent by the customer on the N4 segment.

CUSTOMER_ID **NUMBER**

Customer identifier. FK to RA_CUSTOMERS.

ECE_PRIMARY_ADDRESS_ID **NUMBER**

Customer primary address ID; populated by e-Commerce Gateway code conversion of external trading partner location code. FK to RA_ADDRESSES.

ECE_TP_LOCATION_CODE_EXT **VARCHAR2(35)**

For EDI transactions only, external trading partner location code assigned to the location in the e-Commerce Gateway common control record.

ECE_TP_TRANSLATOR_CODE **VARCHAR2(35)**

For EDI transactions only, e-Commerce Gateway trading partner code assigned to the transaction in the e-Commerce Gateway common control record.

EDI_CONTROL_NUM_1 **VARCHAR2(15)**

EDI control number 1 (ISA) assigned by the customer's EDI translator for audit.

EDI_CONTROL_NUM_2 **VARCHAR2(15)**

EDI control number 2 (GS) assigned by the customer's EDI translator for audit.

EDI_CONTROL_NUM_3 **VARCHAR2(15)**

EDI control number 3 (ST) assigned by the customer's EDI translator for audit.

EDI_TEST_INDICATOR **VARCHAR2(1)**

EDI Test/Production transaction indicator (T or P), assigned by the customer's EDI translator. E-Commerce Gateway rules determine the handling of test transactions within the Demand Processor.

HEADER_CONTACT_CODE_1 **VARCHAR2(3)**

Contact code 1 included in customer's EDI transaction (not validated).

HEADER_CONTACT_CODE_2 **VARCHAR2(3)**

Contact code 2 included in customer's EDI transaction (not validated).

HEADER_CONTACT_VALUE_1 **VARCHAR2(80)**

Values associated with contact code 1: concatenation of name, communication code, and communication number.

HEADER_CONTACT_VALUE_2 **VARCHAR2(80)**

Values associated with contact code 2: concatenation of name, communication code, and communication number.

HEADER_CONTACT_EMAIL_1_EXT **VARCHAR2(50)**

For future use.

HEADER_CONTACT_EMAIL_2_EXT **VARCHAR2(50)**

For future use.

HEADER_CONTACT_FAX_1_EXT **VARCHAR2(35)**

For future use.

HEADER_CONTACT_FAX_2_EXT **VARCHAR2(35)**

For future use.

HEADER_CONTACT_TEL_1_EXT **VARCHAR2(50)**

For future use.

HEADER_CONTACT_TEL_2_EXT **VARCHAR2(50)**

For future use.

HEADER_NOTE_TEXT **VARCHAR2(240)**

Free form note text included in the schedule at the header level.

HEADER_REF_CODE_1 **VARCHAR2(3)**

Reference code 1 included in customer's EDI transaction.

HEADER_REF_CODE_2 **VARCHAR2(3)**

Reference code 2 included in customer's EDI transaction.

HEADER_REF_CODE_3 **VARCHAR2(3)**

Reference code 3 included in customer's EDI transaction.

HEADER_REF_VALUE_1 **VARCHAR2(35)**

Value associated with reference code 1.

HEADER_REF_VALUE_2 **VARCHAR2(35)**

Value associated with reference code 2.

HEADER_REF_VALUE_3 **VARCHAR2(35)**

Value associated with reference code 3.

ORG_ID **NUMBER**

Operating unit unique ID. FK to HR_ORGANIZATION_UNITS.

PROCESS_STATUS **NUMBER**

Indicates the current processing status of a record for headers with lines in error:

- 1=do not process
- 2=available to be processed
- 3=in process
- 4=error
- 5=processed successfully

- 6='line is frozen firm'
- 7='processed with error(s)'

SCHEDULE_HEADER_ID **NUMBER**

Schedule header unique identifier. FK to RLM_SCHEDULE_HEADERS.

SCHEDULE_TYPE **VARCHAR2(30)**

Schedule Type: planning, shipping or sequenced. Validated against FND_LOOKUPS.LOOKUP_TYPE = RLM_SCHEDULE_TYPE.

SCHEDULE_TYPE_EXT **VARCHAR2(30)**

External value from which RLM_SCHEDULE_TYPE was derived in EDI Gateway code conversion.

SCHED_GENERATION_DATE **DATE**

Date customer generated either the schedule or the planning information on the schedule.

SCHED_HORIZON_END_DATE **DATE**

Schedule horizon end date, to be derived based on demand contained within the schedule if not specified.

SCHED_HORIZON_START_DATE **DATE**

Schedule horizon start date, to be derived based on demand contained within the schedule if not specified.

SCHEDULE_PURPOSE **VARCHAR2(30)**

Schedule purpose code, i.e. original, replace, cancellation, etc. Validated against FND_LOOKUPS.LOOKUP_TYPE =RLM_SCHEDULE_PURPOSE

SCHEDULE_PURPOSE_EXT **VARCHAR2(30)**

External value from which SCHEDULE_PURPOSE was derived in e-Commerce Gateway code conversion.

SCHEDULE_REFERENCE_NUM **VARCHAR2(35)**

Customer assigned schedule reference or release number.

SCHEDULE_SOURCE EDI or manual transaction source for this schedule, i.e. 830, 862, 866, DELFOR, MANUAL, etc.	VARCHAR2(30)
LAST_UPDATE_DATE Standard Who column.	DATE
LAST_UPDATED_BY Standard Who column.	NUMBER
CREATION_DATE Standard Who column.	DATE
CREATED_BY Standard Who column.	NUMBER
ATTRIBUTE_CATEGORY Descriptive flexfield context column.	VARCHAR2(30)
ATTRIBUTE1 Descriptive flexfield segment column.	VARCHAR2(150)
ATTRIBUTE2 Descriptive flexfield segment column.	VARCHAR2(150)
ATTRIBUTE3 Descriptive flexfield segment column.	VARCHAR2(150)
ATTRIBUTE4 Descriptive flexfield segment column.	VARCHAR2(150)
ATTRIBUTE5 Descriptive flexfield segment column.	VARCHAR2(150)

ATTRIBUTE6 Descriptive flexfield segment column.	VARCHAR2(150)
ATTRIBUTE7 Descriptive flexfield segment column.	VARCHAR2(150)
ATTRIBUTE8 Descriptive flexfield segment column.	VARCHAR2(150)
ATTRIBUTE9 Descriptive flexfield segment column.	VARCHAR2(150)
ATTRIBUTE10 Descriptive flexfield segment column.	VARCHAR2(150)
ATTRIBUTE11 Descriptive flexfield segment column.	VARCHAR2(150)
ATTRIBUTE12 Descriptive flexfield segment column.	VARCHAR2(150)
ATTRIBUTE13 Descriptive flexfield segment column.	VARCHAR2(150)
ATTRIBUTE14 Descriptive flexfield segment column.	VARCHAR2(150)
ATTRIBUTE15 Descriptive flexfield segment column.	VARCHAR2(150)
LAST_UPDATE_LOGIN Standard Who column.	NUMBER
REQUEST_ID Standard Who column.	NUMBER

PROGRAM_APPLICATION_ID Standard Who column.	NUMBER
PROGRAM_ID Standard Who column.	NUMBER
PROGRAM_UPDATE_DATE Standard Who column.	DATE
TP_ATTRIBUTE1 Trading Partner flexfield column.	VARCHAR2(150)
TP_ATTRIBUTE2 Trading Partner flexfield column.	VARCHAR2(150)
TP_ATTRIBUTE3 Trading Partner flexfield column.	VARCHAR2(150)
TP_ATTRIBUTE4 Trading Partner flexfield column.	VARCHAR2(150)
TP_ATTRIBUTE5 Trading Partner flexfield column.	VARCHAR2(150)
TP_ATTRIBUTE6 Trading Partner flexfield column.	VARCHAR2(150)
TP_ATTRIBUTE7 Trading Partner flexfield column.	VARCHAR2(150)
TP_ATTRIBUTE8 Trading Partner flexfield column.	VARCHAR2(150)
TP_ATTRIBUTE9 Trading Partner flexfield column.	VARCHAR2(150)

TP_ATTRIBUTE10 Trading Partner flexfield column.	VARCHAR2(150)
TP_ATTRIBUTE11 Trading Partner flexfield column.	VARCHAR2(150)
TP_ATTRIBUTE12 Trading Partner flexfield column.	VARCHAR2(150)
TP_ATTRIBUTE13 Trading Partner flexfield column.	VARCHAR2(150)
TP_ATTRIBUTE14 Trading Partner flexfield column.	VARCHAR2(150)
TP_ATTRIBUTE15 Trading Partner flexfield column.	VARCHAR2(150)
TP_ATTRIBUTE_CATEGORY Trading Partner flexfield context column.	VARCHAR2(30)
CUST_FAX_EXT For future use.	VARCHAR2(35)
CUST_REGION_EXT For future use.	VARCHAR2(35)
CUST_TAX_JURISDICTION_EXT For future use.	VARCHAR2(35)
CUST_TEL_EXT For future use.	VARCHAR2(35)
CUST_URL_EXT For future use.	VARCHAR2(100)

ORIGINAL_DOCUMENT_DATE	DATE
-------------------------------	-------------

For future use.

RESP_DOCUMENT_DATE	DATE
---------------------------	-------------

For future use.

DOCUMENT_DESCRIPTION	VARCHAR2(50)
-----------------------------	---------------------

For future use.

DOCUMENT_REV_NUM	VARCHAR2(35)
-------------------------	---------------------

For future use.

RESP_SCHEDULE_ID	VARCHAR2(35)
-------------------------	---------------------

For future use.

RESP_DOCUMENT_REV_NUM	VARCHAR2(35)
------------------------------	---------------------

For future use.

BUCKET_END_DATE	DATE
------------------------	-------------

For future use.

BUCKET_START_DATE	DATE
--------------------------	-------------

For future use.

FLEX_BUCKET_CODE	VARCHAR2(30)
-------------------------	---------------------

For future use.

SUPPLIER_NAME_EXT	VARCHAR2(60)
--------------------------	---------------------

For future use.

ONE_TIME_SUPPLIER_FLAG_EXT	VARCHAR2(35)
-----------------------------------	---------------------

For future use.

SUPPLIER_ID	NUMBER
--------------------	---------------

For future use.

SUPPLIER_CURRENCY_EXT For future use.	VARCHAR2(35)
SUPPLIER_DESCRIPTION_EXT For future use.	VARCHAR2(35)
CUST_DESCRIPTION_EXT For future use.	VARCHAR2(35)
SUPPLIER_TAX_EXEMPT_FLAG_EXT For future use.	VARCHAR2(35)
SUPPLIER_TAX_ID_EXT For future use.	VARCHAR2(35)
SUPPLIER_ADDRESS_1_EXT For future use.	VARCHAR2(35)
SUPPLIER_ADDRESS_2_EXT For future use.	VARCHAR2(35)
SUPPLIER_ADDRESS_3_EXT For future use.	VARCHAR2(35)
SUPPLIER_ADDRESS_4_EXT For future use.	VARCHAR2(35)
SUPPLIER_ADDRESS_5_EXT For future use.	VARCHAR2(35)
SUPPLIER_ADDRESS_6_EXT For future use.	VARCHAR2(240)
SUPPLIER_ADDRESS_7_EXT For future use.	VARCHAR2(35)

SUPPLIER_ADDRESS_8_EXT For future use.	VARCHAR2(35)
SUPPLIER_ADDRESS_9_EXT For future use.	VARCHAR2(35)
SUPPLIER_CITY_EXT For future use.	VARCHAR2(30)
SUPPLIER_COUNTY_EXT For future use.	VARCHAR2(35)
SUPPLIER_FAX_EXT For future use.	VARCHAR2(35)
SUPPLIER_POSTAL_CD_EXT For future use.	VARCHAR2(15)
SUPPLIER_REGION_EXT For future use.	VARCHAR2(35)
SUPPLIER_STATE_EXT For future use.	VARCHAR2(10)
SUPPLIER_PROVINCE_EXT For future use.	VARCHAR2(10)
SUPPLIER_TAX_JURISDICTION_EXT For future use.	VARCHAR2(35)
SUPPLIER_TEL_EXT For future use.	VARCHAR2(35)
SUPPLIER_URL_EXT For future use.	VARCHAR2(100)

SUP_HEADER_CONTACT_CODE_1 For future use.	VARCHAR2(3)
SUP_HEADER_CONTACT_CODE_2 For future use.	VARCHAR2(3)
SUP_HEADER_CONTACT_VALUE_1 For future use.	VARCHAR2(80)
SUP_HEADER_CONTACT_VALUE_2 For future use.	VARCHAR2(80)
SUP_HEADER_CONTACT_EMAIL_1_EXT For future use.	VARCHAR2(50)
SUP_HEADER_CONTACT_EMAIL_2_EXT For future use.	VARCHAR2(50)
SUP_HEADER_CONTACT_FAX_1_EXT For future use.	VARCHAR2(35)
SUP_HEADER_CONTACT_FAX_2_EXT For future use.	VARCHAR2(35)
SUP_HEADER_CONTACT_TEL_1_EXT For future use.	VARCHAR2(50)
SUP_HEADER_CONTACT_TEL_2_EXT For future use.	VARCHAR2(50)
SUPPLIER_COUNTRY_EXT For future use.	VARCHAR2(3)

RLM_INTERFACE_LINES_ALL

This table stores the item and item detail level information associated with customer planning, shipping, or production sequenced schedules. For production sequence schedules, the item represents the feature, not the configuration.

This table stores all details for scheduled items: dated demand, authorizations, and customer shipment/receipt information. It has a child relationship to RLM_INTERFACE_HEADERS.

Table 3–2 RLM_INTERFACE_LINES_ALL

Column Name	Type	Required	Derived	Optional
LINE_ID	NUMBER	X	-	-
HEADER_ID	NUMBER	X	-	-
AGREEMENT_ID	NUMBER	-	-	X
ATO_DATA_TYPE	VARCHAR2(30)	-	-	X
BILL_TO_ADDRESS_1_EXT	VARCHAR2(35)	-	-	X
BILL_TO_ADDRESS_2_EXT	VARCHAR2(35)	-	-	X
BILL_TO_ADDRESS_3_EXT	VARCHAR2(35)	-	-	X
BILL_TO_ADDRESS_4_EXT	VARCHAR2(35)	-	-	X
BILL_TO_ADDRESS_ID	NUMBER	-	-	X
INVOICE_TO_ORG_ID	NUMBER	-	-	X
BILL_TO_CITY_EXT	VARCHAR2(30)	-	-	X
BILL_TO_COUNTRY_EXT	VARCHAR2(3)	-	-	X
BILL_TO_COUNTY_EXT	VARCHAR2(25)	-	-	X
BILL_TO_NAME_EXT	VARCHAR2(60)	-	-	X
BILL_TO_POSTAL_CD_EXT	VARCHAR2(15)	-	-	X
BILL_TO_PROVINCE_EXT	VARCHAR2(10)	-	-	X
BILL_TO_SITE_USE_ID	NUMBER	-	-	X
BILL_TO_STATE_EXT	VARCHAR2(10)	-	-	X
CARRIER_ID_CODE_EXT	VARCHAR2(35)	-	-	X
CARRIER_QUALIFIER_EXT	VARCHAR2(3)	-	-	X
COMMODITY_EXT	VARCHAR2(35)	-	-	X

Table 3–2 RLM_INTERFACE_LINES_ALL

Column Name	Type	Required	Derived	Optional
COUNTRY_OF_ORIGIN_EXT	VARCHAR2(35)	-	-	X
CUST_ASSEMBLY_EXT	VARCHAR2(30)	-	-	X
CUST_ASSIGNED_ID_EXT	VARCHAR2(20)	-	-	X
CUST_BILL_TO_EXT	VARCHAR2(35)	-	-	X
CUST_CONTRACT_NUM_EXT	VARCHAR2(35)	-	-	X
CUSTOMER_DOCK_CODE	VARCHAR2(50)	-	-	X
CUST_INTRMD_SHIP_TO_EXT	VARCHAR2(35)	-	-	X
CUST_ITEM_PRICE_EXT	NUMBER	-	-	X
CUST_ITEM_PRICE_UOM_EXT	VARCHAR2(3)	-	-	X
CUSTOMER_ITEM_REVISION	VARCHAR2(3)	-	-	X
CUSTOMER_JOB	VARCHAR2(50)	-	-	X
CUST_MANUFACTURER_EXT	VARCHAR2(35)	-	-	X
CUST_MODEL_NUMBER_EXT	VARCHAR2(35)	-	-	X
CUST_MODEL_SERIAL_NUMBER	VARCHAR2(35)	-	-	X
CUST_ORDER_NUM_EXT	VARCHAR2(35)	-	-	X
CUST_PROCESS_NUM_EXT	VARCHAR2(35)	-	-	X
CUST_PRODUCTION_LINE	VARCHAR2(50)	-	-	X
CUST_PRODUCTION_SEQ_NUM	VARCHAR2(35)	-	-	X
CUST_SET_NUM_EXT	VARCHAR2(35)	-	-	X
CUST_SHIP_FROM_ORG_EXT	VARCHAR2(80)	-	-	X
CUST_SHIP_TO_EXT	VARCHAR2(35)	-	-	X
CUST_UOM_EXT	VARCHAR2(10)	-	-	X
CUSTOMER_ITEM_EXT	VARCHAR2(50)	-	-	X
CUSTOMER_ITEM_ID	NUMBER	-	-	X
REQUEST_DATE	DATE	-	-	X
SCHEDULE_DATE	DATE	-	-	X
DATE_TYPE_CODE	VARCHAR2(30)	-	X	-

Table 3–2 RLM_INTERFACE_LINES_ALL

Column Name	Type	Required	Derived	Optional
DATE_TYPE_CODE_EXT	VARCHAR2(30)	-	-	X
DELIVERY_LEAD_TIME	NUMBER	-	-	X
END_DATE_TIME	DATE	-	-	X
EQUIPMENT_CODE_EXT	VARCHAR2(3)	-	-	X
EQUIPMENT_NUMBER_EXT	VARCHAR2(35)	-	-	X
HANDLING_CODE_EXT	VARCHAR2(3)	-	-	X
HAZARD_CODE_EXT	VARCHAR2(10)	-	-	X
HAZARD_CODE_QUAL_EXT	VARCHAR2(3)	-	-	X
HAZARD_DESCRIPTION_EXT	VARCHAR2(80)	-	-	X
IMPORT_LICENSE_DATE_EXT	DATE	-	-	X
IMPORT_LICENSE_EXT	VARCHAR2(35)	-	-	X
INDUSTRY_ATTRIBUTE1	VARCHAR2(150)	-	-	X
INDUSTRY_ATTRIBUTE10	VARCHAR2(150)	-	-	X
INDUSTRY_ATTRIBUTE11	VARCHAR2(150)	-	-	X
INDUSTRY_ATTRIBUTE12	VARCHAR2(150)	-	-	X
INDUSTRY_ATTRIBUTE13	VARCHAR2(150)	-	-	X
INDUSTRY_ATTRIBUTE14	VARCHAR2(150)	-	-	X
INDUSTRY_ATTRIBUTE15	VARCHAR2(150)	-	-	X
INDUSTRY_ATTRIBUTE2	VARCHAR2(150)	-	-	X
INDUSTRY_ATTRIBUTE3	VARCHAR2(150)	-	-	X
INDUSTRY_ATTRIBUTE4	VARCHAR2(150)	-	-	X
INDUSTRY_ATTRIBUTE5	VARCHAR2(150)	-	-	X
INDUSTRY_ATTRIBUTE6	VARCHAR2(150)	-	-	X
INDUSTRY_ATTRIBUTE7	VARCHAR2(150)	-	-	X
INDUSTRY_ATTRIBUTE8	VARCHAR2(150)	-	-	X
INDUSTRY_ATTRIBUTE9	VARCHAR2(150)	-	-	X
INDUSTRY_CONTEXT	VARCHAR2(30)	-	-	X

Table 3–2 RLM_INTERFACE_LINES_ALL

Column Name	Type	Required	Derived	Optional
INTRMD_SHIP_TO_ID	NUMBER	-	-	X
SHIP_TO_ORG_ID	NUMBER	-	-	X
INTRMD_ST_ADDRESS_1_EXT	VARCHAR2(35)	-	-	X
INTRMD_ST_ADDRESS_2_EXT	VARCHAR2(35)	-	-	X
INTRMD_ST_ADDRESS_3_EXT	VARCHAR2(35)	-	-	X
INTRMD_ST_ADDRESS_4_EXT	VARCHAR2(35)	-	-	X
INTRMD_ST_CITY_EXT	VARCHAR2(30)	-	-	X
INTRMD_ST_COUNTRY_EXT	VARCHAR2(3)	-	-	X
INTRMD_ST_COUNTY_EXT	VARCHAR2(25)	-	-	X
INTRMD_ST_NAME_EXT	VARCHAR2(60)	-	-	X
INTRMD_ST_POSTAL_CD_EXT	VARCHAR2(15)	-	-	X
INTRMD_ST_PROVINCE_EXT	VARCHAR2(10)	-	-	X
INTRMD_ST_STATE_EXT	VARCHAR2(10)	-	-	X
INTRMD_ST_SITE_USE_ID	NUMBER	-	-	X
INVENTORY_ITEM_ID	NUMBER	-	-	X
INVENTORY_ITEM_SEGMENT1	VARCHAR2(40)	-	-	X
INVENTORY_ITEM_SEGMENT10	VARCHAR2(40)	-	-	X
INVENTORY_ITEM_SEGMENT11	VARCHAR2(40)	-	-	X
INVENTORY_ITEM_SEGMENT12	VARCHAR2(40)	-	-	X
INVENTORY_ITEM_SEGMENT13	VARCHAR2(40)	-	-	X
INVENTORY_ITEM_SEGMENT14	VARCHAR2(40)	-	-	X
INVENTORY_ITEM_SEGMENT15	VARCHAR2(40)	-	-	X
INVENTORY_ITEM_SEGMENT16	VARCHAR2(40)	-	-	X
INVENTORY_ITEM_SEGMENT17	VARCHAR2(40)	-	-	X
INVENTORY_ITEM_SEGMENT18	VARCHAR2(40)	-	-	X
INVENTORY_ITEM_SEGMENT19	VARCHAR2(40)	-	-	X
INVENTORY_ITEM_SEGMENT2	VARCHAR2(40)	-	-	X

Table 3–2 RLM_INTERFACE_LINES_ALL

Column Name	Type	Required	Derived	Optional
INVENTORY_ITEM_SEGMENT20	VARCHAR2(40)	-	-	X
INVENTORY_ITEM_SEGMENT3	VARCHAR2(40)	-	-	X
INVENTORY_ITEM_SEGMENT4	VARCHAR2(40)	-	-	X
INVENTORY_ITEM_SEGMENT5	VARCHAR2(40)	-	-	X
INVENTORY_ITEM_SEGMENT6	VARCHAR2(40)	-	-	X
INVENTORY_ITEM_SEGMENT7	VARCHAR2(40)	-	-	X
INVENTORY_ITEM_SEGMENT8	VARCHAR2(40)	-	-	X
INVENTORY_ITEM_SEGMENT9	VARCHAR2(40)	-	-	X
ITEM_CONTACT_CODE_1	VARCHAR2(3)	-	-	X
ITEM_CONTACT_CODE_2	VARCHAR2(3)	-	-	X
ITEM_CONTACT_VALUE_1	VARCHAR2(80)	-	-	X
ITEM_CONTACT_VALUE_2	VARCHAR2(80)	-	-	X
ITEM_DESCRIPTION_EXT	VARCHAR2(80)	-	-	X
ITEM_DETAIL_QUANTITY	NUMBER	-	-	X
ITEM_DETAIL_REF_CODE_1	VARCHAR2(3)	-	-	X
ITEM_DETAIL_REF_CODE_2	VARCHAR2(3)	-	-	X
ITEM_DETAIL_REF_CODE_3	VARCHAR2(3)	-	-	X
ITEM_DETAIL_REF_VALUE_1	VARCHAR2(35)	-	-	X
ITEM_DETAIL_REF_VALUE_2	VARCHAR2(35)	-	-	X
ITEM_DETAIL_REF_VALUE_3	VARCHAR2(35)	-	-	X
ITEM_DETAIL_SUBTYPE	VARCHAR2(30)	-	X	-
ITEM_DETAIL_SUBTYPE_EXT	VARCHAR2(30)	-	-	X
ITEM_DETAIL_TYPE	VARCHAR2(30)	-	X	-
ITEM_DETAIL_TYPE_EXT	VARCHAR2(30)	-	-	X
ITEM_ENG_CNG_LVL_EXT	VARCHAR2(35)	-	-	X
ITEM_MEASUREMENTS_EXT	VARCHAR2(240)	-	-	X
ITEM_NOTE_TEXT	VARCHAR2(240)	-	-	X

Table 3–2 RLM_INTERFACE_LINES_ALL

Column Name	Type	Required	Derived	Optional
ITEM_REF_CODE_1	VARCHAR2(3)	-	-	X
ITEM_REF_CODE_2	VARCHAR2(3)	-	-	X
ITEM_REF_CODE_3	VARCHAR2(3)	-	-	X
ITEM_REF_VALUE_1	VARCHAR2(35)	-	-	X
ITEM_REF_VALUE_2	VARCHAR2(35)	-	-	X
ITEM_REF_VALUE_3	VARCHAR2(35)	-	-	X
ITEM_RELEASE_STATUS_EXT	VARCHAR2(3)	-	-	X
LADING_QUANTITY_EXT	NUMBER	-	-	X
LETTER_CREDIT_EXPDT_EXT	DATE	-	-	X
LETTER_CREDIT_EXT	VARCHAR2(35)	-	-	X
LINE_REFERENCE	VARCHAR2(50)	-	-	X
LINK_TO_LINE_REF	VARCHAR2(50)	-	-	X
ORDER_HEADER_ID	NUMBER	-	-	X
ORG_ID	NUMBER	-	-	X
OTHER_NAME_CODE_1	VARCHAR2(3)	-	-	X
OTHER_NAME_CODE_2	VARCHAR2(3)	-	-	X
OTHER_NAME_VALUE_1	VARCHAR2(80)	-	-	X
OTHER_NAME_VALUE_2	VARCHAR2(80)	-	-	X
PACK_SIZE_EXT	NUMBER	-	-	X
PACK_UNITS_PER_PACK_EXT	NUMBER	-	-	X
PACK_UOM_CODE_EXT	VARCHAR2(3)	-	-	X
PACKAGING_CODE_EXT	VARCHAR2(10)	-	-	X
PARENT_LINK_LINE_REF	VARCHAR2(50)	-	-	X
PRICE_LIST_ID	NUMBER	-	-	X
PRIMARY_QUANTITY	NUMBER	-	-	X
PRIMARY_UOM_CODE	VARCHAR2(3)	-	-	X
PRIME_CONTRCTR_PART_EXT	VARCHAR2(35)	-	-	X

Table 3–2 RLM_INTERFACE_LINES_ALL

Column Name	Type	Required	Derived	Optional
PROCESS_STATUS	NUMBER	-	-	X
CUST_PO_RELEASE_NUM	VARCHAR2(35)	-	-	X
CUST_PO_DATE	DATE	-	-	X
CUST_PO_LINE_NUM	VARCHAR2(35)	-	-	X
CUST_PO_NUMBER	VARCHAR2(50)	-	-	X
QTY_TYPE_CODE	VARCHAR2(30)	-	X	-
QTY_TYPE_CODE_EXT	VARCHAR2(30)	-	-	X
RETURN_CONTAINER_EXT	VARCHAR2(35)	-	-	X
SCHEDULE_LINE_ID	NUMBER	-	-	X
ROUTING_DESC_EXT	VARCHAR2(35)	-	-	X
ROUTING_SEQ_CODE_EXT	VARCHAR2(3)	-	-	X
SCHEDULE_ITEM_NUM	NUMBER	-	-	X
SHIP_DEL_PATTERN_EXT	VARCHAR2(3)	-	-	X
SHIP_DEL_TIME_CODE_EXT	VARCHAR2(3)	-	-	X
SHIP_DEL_RULE_NAME	VARCHAR2(30)	-	-	X
SHIP_FROM_ADDRESS_1_EXT	VARCHAR2(35)	-	-	X
SHIP_FROM_ADDRESS_2_EXT	VARCHAR2(35)	-	-	X
SHIP_FROM_ADDRESS_3_EXT	VARCHAR2(35)	-	-	X
SHIP_FROM_ADDRESS_4_EXT	VARCHAR2(35)	-	-	X
SHIP_FROM_CITY_EXT	VARCHAR2(30)	-	-	X
SHIP_FROM_COUNTRY_EXT	VARCHAR2(3)	-	-	X
SHIP_FROM_COUNTY_EXT	VARCHAR2(25)	-	-	X
SHIP_FROM_NAME_EXT	VARCHAR2(60)	-	-	X
SHIP_FROM_ORG_ID	NUMBER	-	-	X
SHIP_FROM_POSTAL_CD_EXT	VARCHAR2(15)	-	-	X
SHIP_FROM_PROVINCE_EXT	VARCHAR2(10)	-	-	X
SHIP_FROM_STATE_EXT	VARCHAR2(10)	-	-	X

Table 3–2 RLM_INTERFACE_LINES_ALL

Column Name	Type	Required	Derived	Optional
SHIP_LABEL_INFO_LINE_1	VARCHAR2(80)	-	-	X
SHIP_LABEL_INFO_LINE_10	VARCHAR2(80)	-	-	X
SHIP_LABEL_INFO_LINE_2	VARCHAR2(80)	-	-	X
SHIP_LABEL_INFO_LINE_3	VARCHAR2(80)	-	-	X
SHIP_LABEL_INFO_LINE_4	VARCHAR2(80)	-	-	X
SHIP_LABEL_INFO_LINE_5	VARCHAR2(80)	-	-	X
SHIP_LABEL_INFO_LINE_6	VARCHAR2(80)	-	-	X
SHIP_LABEL_INFO_LINE_7	VARCHAR2(80)	-	-	X
SHIP_LABEL_INFO_LINE_8	VARCHAR2(80)	-	-	X
SHIP_LABEL_INFO_LINE_9	VARCHAR2(80)	-	-	X
SHIP_TO_ADDRESS_1_EXT	VARCHAR2(35)	-	-	X
SHIP_TO_ADDRESS_2_EXT	VARCHAR2(35)	-	-	X
SHIP_TO_ADDRESS_3_EXT	VARCHAR2(35)	-	-	X
SHIP_TO_ADDRESS_4_EXT	VARCHAR2(35)	-	-	X
SHIP_TO_ADDRESS_ID	NUMBER	-	-	X
DELIVER_TO_ORG_ID	NUMBER	-	-	X
SHIP_TO_CITY_EXT	VARCHAR2(30)	-	-	X
SHIP_TO_COUNTRY_EXT	VARCHAR2(3)	-	-	X
SHIP_TO_COUNTY_EXT	VARCHAR2(25)	-	-	X
SHIP_TO_NAME_EXT	VARCHAR2(60)	-	-	X
SHIP_TO_POSTAL_CD_EXT	VARCHAR2(15)	-	-	X
SHIP_TO_PROVINCE_EXT	VARCHAR2(10)	-	-	X
SHIP_TO_SITE_USE_ID	NUMBER	-	-	X
SHIP_TO_STATE_EXT	VARCHAR2(10)	-	-	X
START_DATE_TIME	DATE	-	-	X
SUBLINE_ASSIGNED_ID_EXT	VARCHAR2(20)	-	-	X
SUBLINE_CONFIG_CODE_EXT	VARCHAR2(3)	-	-	X

Table 3–2 RLM_INTERFACE_LINES_ALL

Column Name	Type	Required	Derived	Optional
SUBLINE_CUST_ITEM_EXT	VARCHAR2(50)	-	-	X
SUBLINE_CUST_ITEM_ID	NUMBER	-	-	X
SUBLINE_MODEL_NUM_EXT	VARCHAR2(35)	-	-	X
SUBLINE_QUANTITY	NUMBER	-	-	X
SUBLINE_UOM_CODE	VARCHAR2(3)	-	-	X
SUPPLIER_ITEM_EXT	VARCHAR2(35)	-	-	X
TRANSIT_TIME_EXT	VARCHAR2(22)	-	-	X
TRANSIT_TIME_QUAL_EXT	VARCHAR2(3)	-	-	X
TRANSPORT_LOC_QUAL_EXT	VARCHAR2(3)	-	-	X
TRANSPORT_LOCATION_EXT	VARCHAR2(35)	-	-	X
TRANSPORT_METHOD_EXT	VARCHAR2(3)	-	-	X
UOM_CODE	VARCHAR2(3)	-	-	X
WEIGHT_EXT	NUMBER	-	-	X
WEIGHT_QUALIFIER_EXT	VARCHAR2(3)	-	-	X
WEIGHT_UOM_EXT	VARCHAR2(3)	-	-	X
FBO_CONFIGURATION_KEY_1	VARCHAR2(35)	-	-	X
FBO_CONFIGURATION_KEY_2	VARCHAR2(35)	-	-	X
FBO_CONFIGURATION_KEY_3	VARCHAR2(35)	-	-	X
FBO_CONFIGURATION_KEY_4	VARCHAR2(35)	-	-	X
FBO_CONFIGURATION_KEY_5	VARCHAR2(35)	-	-	X
MATCH_KEY_ACROSS	VARCHAR2(150)	-	-	X
MATCH_KEY_WITHIN	VARCHAR2(150)	-	-	X
CRITICAL_KEY_ATTRIBUTES	VARCHAR2(150)	-	-	X
LAST_UPDATE_DATE	DATE	-	X	-
LAST_UPDATED_BY	NUMBER	-	X	-
CREATION_DATE	DATE	-	X	-
CREATED_BY	NUMBER	-	X	-

Table 3–2 RLM_INTERFACE_LINES_ALL

Column Name	Type	Required	Derived	Optional
ATTRIBUTE_CATEGORY	VARCHAR2(30)	-	-	X
ATTRIBUTE1	VARCHAR2(150)	-	-	X
ATTRIBUTE2	VARCHAR2(150)	-	-	X
ATTRIBUTE3	VARCHAR2(150)	-	-	X
ATTRIBUTE4	VARCHAR2(150)	-	-	X
ATTRIBUTE5	VARCHAR2(150)	-	-	X
ATTRIBUTE6	VARCHAR2(150)	-	-	X
ATTRIBUTE7	VARCHAR2(150)	-	-	X
ATTRIBUTE8	VARCHAR2(150)	-	-	X
ATTRIBUTE9	VARCHAR2(150)	-	-	X
ATTRIBUTE10	VARCHAR2(150)	-	-	X
ATTRIBUTE11	VARCHAR2(150)	-	-	X
ATTRIBUTE12	VARCHAR2(150)	-	-	X
ATTRIBUTE13	VARCHAR2(150)	-	-	X
ATTRIBUTE14	VARCHAR2(150)	-	-	X
ATTRIBUTE15	VARCHAR2(150)	-	-	X
LAST_UPDATE_LOGIN	NUMBER	-	X	-
REQUEST_ID	NUMBER	-	X	-
PROGRAM_APPLICATION_ID	NUMBER	-	X	-
PROGRAM_ID	NUMBER	-	X	-
PROGRAM_UPDATE_DATE	DATE	-	X	-
TP_ATTRIBUTE1	VARCHAR2(150)	-	-	X
TP_ATTRIBUTE2	VARCHAR2(150)	-	-	X
TP_ATTRIBUTE3	VARCHAR2(150)	-	-	X
TP_ATTRIBUTE4	VARCHAR2(150)	-	-	X
TP_ATTRIBUTE5	VARCHAR2(150)	-	-	X
TP_ATTRIBUTE6	VARCHAR2(150)	-	-	X

Table 3–2 RLM_INTERFACE_LINES_ALL

Column Name	Type	Required	Derived	Optional
TP_ATTRIBUTE7	VARCHAR2(150)	-	-	X
TP_ATTRIBUTE8	VARCHAR2(150)	-	-	X
TP_ATTRIBUTE9	VARCHAR2(150)	-	-	X
TP_ATTRIBUTE10	VARCHAR2(150)	-	-	X
TP_ATTRIBUTE11	VARCHAR2(150)	-	-	X
TP_ATTRIBUTE12	VARCHAR2(150)	-	-	X
TP_ATTRIBUTE13	VARCHAR2(150)	-	-	X
TP_ATTRIBUTE14	VARCHAR2(150)	-	-	X
TP_ATTRIBUTE15	VARCHAR2(150)	-	-	X
TP_ATTRIBUTE_CATEGORY	VARCHAR2(30)	-	-	X
LINE_NUMBER	NUMBER	-	-	X
INTMED_SHIP_TO_ORG_ID	NUMBER	-	-	X
LINE_SOURCE	VARCHAR2(30)	-	-	X
PREFERRED_GRADE	VARCHAR2(4)	-	-	X
CUST_PRODUCTION_SEQ_NUM_BEG	VARCHAR2(35)	-	-	X
CUST_PRODUCTION_SEQ_NUM_END	VARCHAR2(35)	-	-	X
ITEM_DETAIL_QUANTITY_MIN_EXT	NUMBER	-	-	X
ITEM_DETAIL_QUANTITY_MAX_EXT	NUMBER	-	-	X
ITEM_DETAIL_QUANTITY_PRIOR_EXT	NUMBER	-	-	X
REQUIREMENT_PRIORITY_EXT	NUMBER	-	-	X
PROJECT_NUMBER_EXT	VARCHAR2(50)	-	-	X
RESP_LINE_NUMBER	VARCHAR2(35)	-	-	X
TASK_NUMBER_EXT	VARCHAR2(50)	-	-	X
UPC_EXT	VARCHAR2(50)	-	-	X

Table 3–2 RLM_INTERFACE_LINES_ALL

Column Name	Type	Required	Derived	Optional
ITEM_ENG_CNG_LVL_DATE_EXT	DATE	-	-	X
AUTHORIZATION_CODE_EXT	VARCHAR2(50)	-	-	X
PACKAGING_DESC_EXT	VARCHAR2(80)	-	-	X
BILL_TO_ADDRESS_5_EXT	VARCHAR2(35)	-	-	X
BILL_TO_ADDRESS_6_EXT	VARCHAR2(35)	-	-	X
BILL_TO_ADDRESS_7_EXT	VARCHAR2(35)	-	-	X
BILL_TO_ADDRESS_8_EXT	VARCHAR2(35)	-	-	X
BILL_TO_ADDRESS_9_EXT	VARCHAR2(35)	-	-	X
ONE_TIME_BILL_TO_FLAG_EXT	VARCHAR2(35)	-	-	X
BILL_TO_DESCRIPTION_EXT	VARCHAR2(50)	-	-	X
BILL_TO_FAX_EXT	VARCHAR2(35)	-	-	X
BILL_TO_REGION_EXT	VARCHAR2(35)	-	-	X
BILL_TO_JURISDICTION_EXT	VARCHAR2(35)	-	-	X
BILL_TO_TEL_EXT	VARCHAR2(35)	-	-	X
INTRMD_ST_ADDRESS_5_EXT	VARCHAR2(35)	-	-	X
INTRMD_ST_ADDRESS_6_EXT	VARCHAR2(35)	-	-	X
INTRMD_ST_ADDRESS_7_EXT	VARCHAR2(35)	-	-	X
INTRMD_ST_ADDRESS_8_EXT	VARCHAR2(35)	-	-	X
INTRMD_ST_ADDRESS_9_EXT	VARCHAR2(35)	-	-	X
ONE_TIME_INTRMD_ST_FLAG_EXT	VARCHAR2(35)	-	-	X
INTRMD_ST_DESCRIPTION_EXT	VARCHAR2(50)	-	-	X
INTRMD_ST_FAX_EXT	VARCHAR2(35)	-	-	X
INTRMD_ST_REGION_EXT	VARCHAR2(35)	-	-	X
INTRMD_ST_JURISDICTION_EXT	VARCHAR2(35)	-	-	X
INTRMD_ST_TEL_EXT	VARCHAR2(35)	-	-	X
SHIP_TO_ADDRESS_5_EXT	VARCHAR2(35)	-	-	X

Table 3–2 RLM_INTERFACE_LINES_ALL

Column Name	Type	Required	Derived	Optional
SHIP_TO_ADDRESS_6_EXT	VARCHAR2(35)	-	-	X
SHIP_TO_ADDRESS_7_EXT	VARCHAR2(35)	-	-	X
SHIP_TO_ADDRESS_8_EXT	VARCHAR2(35)	-	-	X
SHIP_TO_ADDRESS_9_EXT	VARCHAR2(35)	-	-	X
ONE_TIME_SHIP_TO_FLAG_EXT	VARCHAR2(35)	-	-	X
SHIP_TO_DESCRIPTION_EXT	VARCHAR2(50)	-	-	X
SHIP_TO_FAX_EXT	VARCHAR2(35)	-	-	X
SHIP_TO_REGION_EXT	VARCHAR2(35)	-	-	X
SHIP_TO_JURISDICTION_EXT	VARCHAR2(35)	-	-	X
SHIP_TO_TEL_EXT	VARCHAR2(35)	-	-	X
SHIP_FROM_ADDRESS_5_EXT	VARCHAR2(35)	-	-	X
SHIP_FROM_ADDRESS_6_EXT	VARCHAR2(35)	-	-	X
SHIP_FROM_ADDRESS_7_EXT	VARCHAR2(35)	-	-	X
SHIP_FROM_ADDRESS_8_EXT	VARCHAR2(35)	-	-	X
SHIP_FROM_ADDRESS_9_EXT	VARCHAR2(35)	-	-	X
ONE_TIME_SHIP_FROM_FLAG_EXT	VARCHAR2(35)	-	-	X
SHIP_FROM_DESCRIPTION_EXT	VARCHAR2(50)	-	-	X
SHIP_FROM_FAX_EXT	VARCHAR2(35)	-	-	X
SHIP_FROM_REGION_EXT	VARCHAR2(35)	-	-	X
SHIP_FROM_JURISDICTION_EXT	VARCHAR2(35)	-	-	X
SHIP_FROM_TEL_EXT	VARCHAR2(35)	-	-	X

Column Descriptions

LINE_ID

Sequence generated unique identifier.

NUMBER

HEADER_ID**NUMBER**

Sequence generated unique identifier foreign key to the RLM_INTERFACE_HEADERS_ALL.

AGREEMENT_ID**NUMBER**

Unique identifier for agreement on which customer purchase order is associated. FK to RA_AGREEMENTS

ATO_DATA_TYPE**VARCHAR2(30)**

Code to describe what type of data is included for the ATO item:

- 1=Model and Options
- 2=Model
- 3=Options

Validated against FND_LOOKUPS.LOOKUP_TYPE =RLM_ITEM_METHOD_TYPE

BILL_TO_ADDRESS_1_EXT**VARCHAR2(35)**

Bill to address line 1, as sent by the customer on the N3 segment.

BILL_TO_ADDRESS_2_EXT**VARCHAR2(35)**

Bill to address line 2, as sent by the customer on the N3 segment.

BILL_TO_ADDRESS_3_EXT**VARCHAR2(35)**

Bill to address line 3, as sent by the customer on the N3 segment.

BILL_TO_ADDRESS_4_EXT**VARCHAR2(35)**

Bill to address line 4, as sent by the customer on the N3 segment.

BILL_TO_ADDRESS_ID**NUMBER**

Bill to address identifier. FK to RA_ADDRESSES.

INVOICE_TO_ORG_ID**NUMBER**

Unique identifier for invoice-to organization which relates to BILL_TO_ADDRESS_ID. FK in R12 customer-org data model to HR_ORGANIZATIONS.

BILL_TO_CITY_EXT Bill to address city, as sent by the customer on the N4 segment.	VARCHAR2(30)
BILL_TO_COUNTRY_EXT Bill to country, as sent by the customer on the N4 segment.	VARCHAR2(3)
BILL_TO_COUNTY_EXT Bill to county, as sent by the customer on the N4 segment.	VARCHAR2(25)
BILL_TO_NAME_EXT Bill to name, as sent by the customer in the N1 and/or N2 segment.	VARCHAR2(60)
BILL_TO_POSTAL_CD_EXT Bill to address postal code, as sent by the customer on the N4 segment.	VARCHAR2(15)
BILL_TO_PROVINCE_EXT Bill to province, as sent by the customer on the N4 segment.	VARCHAR2(10)
BILL_TO_SITE_USE_ID Bill to site use identifier. FK to RA_SITE_USES.	NUMBER
BILL_TO_STATE_EXT Bill to state, as sent by the customer on the N4 segment.	VARCHAR2(10)
CARRIER_ID_CODE_EXT The carrier id, as sent by the customer on the TD5 segment.	VARCHAR2(35)
CARRIER_QUALIFIER_EXT The carrier qualifier, as sent by the customer on the TD5 segment.	VARCHAR2(3)
COMMODITY_EXT Customer specified commodity code.	VARCHAR2(35)
COUNTRY_OF_ORIGIN_EXT Customer-specified country of origin.	VARCHAR2(35)

CUST_ASSEMBLY_EXT **VARCHAR2(30)**

The customer's assembly identification, as sent on the LIN segment of the 866 transaction.

CUST_ASSIGNED_ID_EXT **VARCHAR2(20)**

Customer assigned identification for differentiation within a transaction set, from LIN01.

CUST_BILL_TO_EXT **VARCHAR2(35)**

External customer bill to cross reference.

CUST_CONTRACT_NUM_EXT **VARCHAR2(35)**

The customer's contract number, as sent on the LIN segment of the 830 transaction.

CUSTOMER_DOCK_CODE **VARCHAR2(50)**

Customer dock code.

CUST_INTERMD_SHIP_TO_EXT **VARCHAR2(35)**

External intermediate ship to cross reference.

CUST_ITEM_PRICE_EXT **NUMBER**

Price included on the customer's EDI transaction.

CUST_ITEM_PRICE_UOM_EXT **VARCHAR2(3)**

UOM corresponding to the price included on the customer's EDI transaction.

CUSTOMER_ITEM_REVISION **VARCHAR2(35)**

Customer part revision included on schedule.

CUSTOMER_JOB **VARCHAR2(50)**

Customer job number.

CUST_MANUFACTURER_EXT **VARCHAR2(35)**

The manufacturer, as sent by the customer on the LIN segment of the 866 transaction.

CUST_MODEL_NUMBER_EXT **VARCHAR2(35)**

Customer's model number for this sequenced detail.

CUST_MODEL_SERIAL_NUMBER **VARCHAR2(35)**

Customer's vehicle identification number for this sequenced detail, e.g. VIN or Chassis ID.

CUST_ORDER_NUM_EXT **VARCHAR2(35)**

The order number, as sent by the customer on the LIN segment of the 866 and 830 transaction.

CUST_PROCESS_NUM_EXT **VARCHAR2(35)**

The process number, as sent by the customer on the LIN segment of the 866 and 862 transaction.

CUST_PRODUCTION_LINE **VARCHAR2(50)**

Customer production line.

CUSTOMER_PROD_SEQ_NUM **VARCHAR2(350)**

Customer production sequence number, or delivery number.

CUST_SET_NUM_EXT **VARCHAR2(35)**

The customer's set number, as sent on the LIN segment of the 862 and 866 transactions.

CUST_SHIP_FROM_ORG_EXT **VARCHAR2(80)**

External inventory organization cross reference.

CUST_SHIP_TO_EXT **VARCHAR2(35)**

External ship to address cross reference.

CUST_UOM_EXT **VARCHAR2(10)**

External customer unit of measure cross reference.

CUSTOMER_ITEM_EXT **VARCHAR2(50)**

External customer part number cross reference.

CUSTOMER_ITEM_ID **NUMBER**

Customer item identifier. FK to MTL_CUSTOMER_ITEMS.

REQUEST_DATE**DATE**

The date and time the customer wants the material delivered or shipped, based on whether the requirements are delivery based or ship based.

SCHEDULE_DATE**DATE**

Planned shipment date and time (request_date - delivery lead time if delivery based).

DATE_TYPE_CODE**VARCHAR2(30)**

Type of start/end date, e.g. ship, deliver, Pull signal, Cumulative. Validated against FND_LOOKUPS.LOOKUP_TYPE =RLM_DATE_TYPE_CODE

DATE_TYPE_CODE_EXT**VARCHAR2(30)**

External value from which DATE_TYPE_CODE was derived in EDI Gateway code conversion.

DELIVERY_LEAD_TIME**NUMBER**

For demand lines, the difference between arrival and shipment dates, based on default shipping method between the ship-from and ship-to locations. This is determined by the calculate ship date routine and passed into Order Import.

END_DATE_TIME**DATE**

Customer-specified optional end date/time, applicable for flexible bucketed requirements, and cumulative information such as authorizations and shipped/received.

EQUIPMENT_CODE_EXT**VARCHAR2(3)**

The equipment code, as sent by the customer on the TD3 segment.

EQUIPMENT_NUMBER_EXT**VARCHAR2(35)**

The equipment number, as sent by the customer on the TD3 segment.

HANDLING_CODE_EXT**VARCHAR2(3)**

Special handling code as sent by the customer in the TD4 segment.

HAZARD_CODE_EXT**VARCHAR2(10)**

Hazardous material code corresponding to the hazardous material code qualifier as sent by the customer in the TD4 segment.

HAZARD_CODE_QUAL_EXT **VARCHAR2(3)**

Hazardous material code qualifier as sent by the customer in the TD4 segment.

HAZARD_DESCRIPTION_EXT **VARCHAR2(80)**

Hazardous material description as sent by the customer in the TD4 segment.

IMPORT_LICENSE_DATE_EXT **DATE**

Customer's import license date.

IMPORT_LICENSE_EXT **VARCHAR2(35)**

Customer's import license for shipment destination country.

INDUSTRY_ATTR IBUTE1 **VARCHAR2(150)**

Record keeping or model year.

INDUSTRY_ATTRIBUTE10 **VARCHAR2(150)**

Industry descriptive flexfield.

INDUSTRY_ATTRIBUTE11 **VARCHAR2(150)**

Industry descriptive flexfield.

INDUSTRY_ATTRIBUTE12 **VARCHAR2(150)**

Industry descriptive flexfield.

INDUSTRY_ATTRIBUTE13 **VARCHAR2(150)**

Industry descriptive flexfield.

INDUSTRY_ATTRIBUTE14 **VARCHAR2(150)**

Industry descriptive flexfield.

INDUSTRY_ATTRIBUTE15 **VARCHAR2(150)**

Industry descriptive flexfield.

INDUSTRY_ATTRIBUTE2 **VARCHAR2(150)**

Industry descriptive flexfield.

INDUSTRY_ATTRIBUTE3 Industry descriptive flexfield.	VARCHAR2(150)
INDUSTRY_ATTRIBUTE4 Industry descriptive flexfield.	VARCHAR2(150)
INDUSTRY_ATTRIBUTE5 Industry descriptive flexfield.	VARCHAR2(150)
INDUSTRY_ATTRIBUTE6 Industry descriptive flexfield.	VARCHAR2(150)
INDUSTRY_ATTRIBUTE7 Industry descriptive flexfield.	VARCHAR2(150)
INDUSTRY_ATTRIBUTE8 Industry descriptive flexfield.	VARCHAR2(150)
INDUSTRY_ATTRIBUTE9 Industry descriptive flexfield.	VARCHAR2(150)
INDUSTRY_CONTEXT Industry descriptive context flexfield.	VARCHAR2(30)
INTERMEDIATE_SHIP_TO_ID Customer intermediate ship-to destination unique identifier. FK to RA_ADDRESSES	NUMBER
SHIP_TO_ORG_ID Unique identifier for ship-to organization which relates to SHIP_TO_ADDRESS_ID if there is no intermediate ship-to address, or to INTERMEDIATE_SHIP_TO_ID if intermediate ship-to address is specified. FK in R12 customer-org data model to HR_ORGANIZATIONS.	NUMBER
INTRMD_ST_ADDRESS_1_EXT Intermediate ship to address line 1, as sent by the customer on the N3 segment.	VARCHAR2(35)

INTRMD_ST_ADDRESS_2_EXT **VARCHAR2(35)**

Intermediate ship to address line 2, as sent by the customer on the N3 segment.

INTRMD_ST_ADDRESS_3_EXT **VARCHAR2(35)**

Intermediate ship to address line 3, as sent by the customer on the N3 segment.

INTRMD_ST_ADDRESS_4_EXT **VARCHAR2(35)**

Intermediate ship to address line 4, as sent by the customer on the N3 segment.

INTRMD_ST_CITY_EXT **VARCHAR2(30)**

Intermediate ship to address city, as sent by the customer on the N4 segment.

INTRMD_ST_COUNTRY_EXT **VARCHAR2(3)**

Intermediate ship to country, as sent by the customer on the N4 segment.

INTRMD_ST_COUNTY_EXT **VARCHAR2(25)**

Intermediate ship to county, as sent by the customer on the N4 segment.

INTRMD_ST_NAME_EXT **VARCHAR2(60)**

Intermediate ship to name, as sent by the customer in the N1 and/or N2 segment.

INTRMD_ST_POSTAL_CD_EXT **VARCHAR2(15)**

Intermediate ship to address postal code, as sent by the customer on the N4 segment.

INTRMD_ST_PROVINCE_EXT **VARCHAR2(10)**

Intermediate ship to province, as sent by the customer on the N4 segment.

INTRMD_ST_STATE_EXT **VARCHAR2(10)**

Intermediate ship to state, as sent by the customer on the N4 segment.

INTRMD_ST_SITE_USE_ID **NUMBER**

Intermediate ship to site use identifier, FK to RA_SITE_USES.

INVENTORY_ITEM_ID **NUMBER**

Inventory item identifier. FK to MTL_SYSTEM_ITEMS.

INVENTORY_ITEM_SEGMENT1 Accounting flexfield.	VARCHAR2(40)
INVENTORY_ITEM_SEGMENT10 Accounting flexfield.	VARCHAR2(40)
INVENTORY_ITEM_SEGMENT11 Accounting flexfield.	VARCHAR2(40)
INVENTORY_ITEM_SEGMENT12 Accounting flexfield.	VARCHAR2(40)
INVENTORY_ITEM_SEGMENT13 Accounting flexfield.	VARCHAR2(40)
INVENTORY_ITEM_SEGMENT14 Accounting flexfield.	VARCHAR2(40)
INVENTORY_ITEM_SEGMENT15 Accounting flexfield.	VARCHAR2(40)
INVENTORY_ITEM_SEGMENT16 Accounting flexfield.	VARCHAR2(40)
INVENTORY_ITEM_SEGMENT17 Accounting flexfield.	VARCHAR2(40)
INVENTORY_ITEM_SEGMENT18 Accounting flexfield.	VARCHAR2(40)
INVENTORY_ITEM_SEGMENT19 Accounting flexfield.	VARCHAR2(40)
INVENTORY_ITEM_SEGMENT2 Accounting flexfield.	VARCHAR2(40)

INVENTORY_ITEM_SEGMENT20 Accounting flexfield.	VARCHAR2(40)
INVENTORY_ITEM_SEGMENT3 Accounting flexfield.	VARCHAR2(40)
INVENTORY_ITEM_SEGMENT4 Accounting flexfield.	VARCHAR2(40)
INVENTORY_ITEM_SEGMENT5 Accounting flexfield.	VARCHAR2(40)
INVENTORY_ITEM_SEGMENT6 Accounting flexfield.	VARCHAR2(40)
INVENTORY_ITEM_SEGMENT7 Accounting flexfield.	VARCHAR2(40)
INVENTORY_ITEM_SEGMENT8 Accounting flexfield.	VARCHAR2(40)
INVENTORY_ITEM_SEGMENT9 Accounting flexfield.	VARCHAR2(40)
ITEM_CONTACT_CODE_1 Contact code 1 included in customer's EDI transaction.	VARCHAR2(3)
ITEM_CONTACT_CODE_2 Contact code 2 included in customer's EDI transaction.	VARCHAR2(3)
ITEM_CONTACT_VALUE_1 Values associated with contact code 1: concatenation of name, communication code, and communication number.	VARCHAR2(80)

ITEM_CONTACT_VALUE_2 **VARCHAR2(80)**

Values associated with contact code 2 : concatenation of name, communication code, and communication number.

ITEM_DESCRIPTION_EXT **VARCHAR2(80)**

Item description included on schedule.

ITEM_DETAIL_QUANTITY **NUMBER**

Requested quantity.

- If Item_Detail_Type = 0,1,2, this is the demand quantity.
- If Item_Detail_Type = 3, this is the authorization quantity.
- If Item_Detail_Type = 4, this is the shipped, received or cum quantity.
- If Item_Detail_Type = 5, this is a miscellaneous quantity, such as ahead/behind, inventory balance, etc.

ITEM_DETAIL_REF_CODE_1 **VARCHAR2(3)**

Reference code 1 included in customer's EDI transaction.

ITEM_DETAIL_REF_CODE_2 **VARCHAR2(3)**

Reference code 2 included in customer's EDI transaction.

ITEM_DETAIL_REF_CODE_3 **VARCHAR2(3)**

Reference code 3 included in customer's EDI transaction.

ITEM_DETAIL_REF_VALUE_1 **VARCHAR2(35)**

Value associated with reference code 1.

ITEM_DETAIL_REF_VALUE_2 **VARCHAR2(35)**

Value associated with reference code 2.

ITEM_DETAIL_REF_VALUE_3 **VARCHAR2(35)**

Value associated with reference code 3.

ITEM_DETAIL_SUBTYPE **VARCHAR2(30)**

Schedule item detail row sub-type:

- for form or forecast demand, Bucket type associated with the demand date/quantity;
- for authorizations, the type of authorization;
- for shipment/receipt, cumulative or last.

Validated against FND_LOOKUPS.

ITEM_DETAIL_SUBTYPE_EXT **VARCHAR2(30)**

External value from which ITEM_DETAIL_SUBTYPE was derived in EDI Gateway Code Conversion.

ITEM_DETAIL_TYPE **VARCHAR2(30)**

Schedule item detail row type: 0 = Past Due Firm1 = Firm Demand, 2 = Forecast Demand, 3 = Authorization, 4 = Shipment/Receipt Info, 5 = OtherValidated against FND_LOOKUPS.LOOKUP_TYPE =RLM_DETAIL_TYPE_CODE

ITEM_DETAIL_TYPE_EXT **VARCHAR2(30)**

External value from which ITEM_DETAIL_TYPE was derived in EDI Gateway Code Conversion.

ITEM_ENG_CNG_LVL_EXT **VARCHAR2(35)**

Customer part engineering change level included on schedule

ITEM_MEASUREMENTS_EXT **VARCHAR2(240)**

Item measurement information as sent by the customer on the MEA segments.

ITEM_NOTE_TEXT **VARCHAR2(240)**

Free form item note text included in the schedule.

ITEM_REF_CODE_1 **VARCHAR2(3)**

Reference code 1 included in customer's EDI transaction (not validated).

ITEM_REF_CODE_2 **VARCHAR2(3)**

Reference code 2 included in customer's EDI transaction (not validated).

ITEM_REF_CODE_3	VARCHAR2(3)
Reference code 3 included in customer's EDI transaction (not validated).	
ITEM_REF_VALUE_1	VARCHAR2(35)
Value associated with reference code 1.	
ITEM_REF_VALUE_2	VARCHAR2(35)
Value associated with reference code 2.	
ITEM_REF_VALUE_3	VARCHAR2(35)
Value associated with reference code 3.	
ITEM_RELEASE_STATUS_EXT	VARCHAR2(3)
Customer part release status included on schedule.	
LADING_QUANTITY_EXT	NUMBER
The lading quantity as sent by the customer on the TD1 segment.	
LETTER_CREDIT_EXPDT_EXT	DATE
Customer's letter of credit expiration date.	
LETTER_CREDIT_EXT	VARCHAR2(35)
Customer's letter of credit with international bank guaranteeing payment for international shipments.	
LINE_REFERENCE	VARCHAR2(50)
Unique line identifier within an order for a transmission identifier.	
LINK_TO_LINE_REF	VARCHAR2(50)
Link to Immediate Parent_line_reference within an order for a transmission identifier.	
ORDER_HEADER_ID	NUMBER
Order header identifier. FK to OE_ORDER_HEADERS.	

ORG_ID	NUMBER
Operating unit unique id. FK to HR_ORGANIZATION_UNITS.	
OTHER_NAME_CODE_1	VARCHAR2(3)
Other name code 1 included in customer's EDI transaction (not validated).	
OTHER_NAME_CODE_2	VARCHAR2(3)
Other name code 2 included in customer's EDI transaction (not validated).	
OTHER_NAME_VALUE_1	VARCHAR2(80)
Values associated with other name code 1: concatenation of name, code, and number.	
OTHER_NAME_VALUE_2	VARCHAR2(80)
Values associated with other name code 2: concatenation of name, code, and number.	
PACK_SIZE_EXT	NUMBER
Size of supplier units in pack, as sent by the customer on the PO4 segment.	
PACK_UNITS_PER_PACK_EXT	NUMBER
Number of inner pack units per out pack unit, as sent by the customer on the PO4 segment.	
PACK_UOM_CODE_EXT	VARCHAR2(3)
Unit of measure of supplier units in the pack, as sent by the customer in the PO4 segment.	
PACKAGING_CODE_EXT	VARCHAR2(10)
The packaging code as sent by the customer on TD1 segment.	
PARENT_LINK_LINE_REF	VARCHAR2(50)
Line reference of top model for a transmission identifier for an order.	
PRICE_LIST_ID	NUMBER
Unique identifier for price list associated with customer item or agreement on which customer purchase order is associated. FK to RA_PRICE_LISTS	

PRIMARY_QUANTITY **NUMBER**

Quantity in the primary UOM.

PRIMARY_UOM_CODE **VARCHAR2(3)**

Primary unit of measure. FK to MTL_UNITS_OF_MEASURE

PRIME_CONTRCTR_PART_EXT **VARCHAR2(35)**

The prime contractor part number, as sent by the customer on the LIN segment of the 862 segment.

PROCESS_STATUS **NUMBER**

Indicates the current processing status of a record

- 1=do not process
- 2=waiting to be processed
- 3=in process
- 4=error
- 5=processed
- 6="processed with error(s)" for headers with lines in-error

CUST_PO_RELEASE_NUM **VARCHAR2(35)**

Customer purchase order release number included on schedule.

CUST_PO_DATE **DATE**

Customer purchase order line number included on schedule.

CUST_PO_LINE_NUM **VARCHAR2(35)**

Customer-specified effectivity date of purchase order number included on schedule.

CUST_PO_NUMBER **VARCHAR2(50)**

Customer purchase order number.

QTY_TYPE_CODE **VARCHAR2(30)**

Actual or Cumulative. Validated against FND_LOOKUPS.LOOKUP_TYPE =RLM_QTY_TYPE_CODE

QTY_TYPE_CODE_EXT **VARCHAR2(30)**

External value from which QTY_TYPE_CODE was derived in EDI Gateway Code Conversion.

RETURN_CONTAINER_EXT **VARCHAR2(35)**

Returnable container specified by customer for item shipment.

RLM_SCHEDULE_LINE_ID **NUMBER**

Schedule line unique identifier. FK to RLM_SCHEDULE_LINES, except for aggregated schedule lines.

ROUTING_DESC_EXT **VARCHAR2(35)**

The routing description, as sent by the customer on the TD5 segment.

ROUTING_SEQ_CODE_EXT **VARCHAR2(3)**

Code describing the relationship of a carrier to a specific shipment movement, as sent by the customer on the TD5 segment.

SCHEDULE_ITEM_NUM **NUMBER**

Schedule Item Number, the means to identify how item demand and information is grouped by the customer within the schedule. For sequenced schedules, it is equal to CUSTOMER_PROD_SEQ_NUM. For EDI planning and shipping schedules, it is incremented in the EDI Gateway when each 2000 record is encountered on an inbound SPSI or SSSI transaction. This number is assigned for manually entered schedules. All interface lines with the same schedule item number are validated together, and pass or fail validation as a group.

SHIP_DEL_PATTERN_EXT **VARCHAR2(3)**

Customer ship delivery pattern code for this item; not integrated with delivery rules for date/quantity calculation.

SHIP_DEL_TIME_CODE_EXT **VARCHAR2(3)**

Customer ship delivery time code for this item; not integrated with delivery rules for date/quantity calculation.

SHIP_DEL_RULE_NAME **VARCHAR2(30)**

Shipment/delivery rule name for this schedule item. Initially populated only if a successful code conversion in the EDI Gateway has occurred. FK to RLM_SHIP_DELIVERY_CODES

SHIP_FROM_ADDRESS_1_EXT **VARCHAR2(35)**

Ship from address line 1, as sent by the customer on the N3 segment.

SHIP_FROM_ADDRESS_2_EXT **VARCHAR2(35)**

Ship from address line 2, as sent by the customer on the N3 segment.

SHIP_FROM_ADDRESS_3_EXT **VARCHAR2(35)**

Ship from address line 3, as sent by the customer on the N3 segment.

SHIP_FROM_ADDRESS_4_EXT **VARCHAR2(35)**

Ship from address line 4, as sent by the customer on the N3 segment.

SHIP_FROM_CITY_EXT **VARCHAR2(30)**

Ship from address city, as sent by the customer on the N4 segment.

SHIP_FROM_COUNTRY_EXT **VARCHAR2(3)**

Ship from country, as sent by the customer on the N4 segment.

SHIP_FROM_COUNTY_EXT **VARCHAR2(25)**

Ship from county, as sent by the customer on the N4 segment.

SHIP_FROM_NAME_EXT **VARCHAR2(60)**

Ship from name, as sent by the customer in the N1 or N2 segment.

SHIP_FROM_ORG_ID **NUMBER**

Ship from organization identifier. FK to MTL_PARAMETERS and HR_ORGANIZATIONS.

SHIP_FROM_POSTAL_CD_EXT **VARCHAR2(15)**

Ship from address postal code, as sent by the customer on the N4 segment.

SHIP_FROM_PROVINCE_EXT Ship from province, as sent by the customer on the N4 segment.	VARCHAR2(10)
SHIP_FROM_STATE_EXT Ship from state, as sent by the customer on the N4 segment.	VARCHAR2(10)
SHIP_LABEL_INFO_LINE_1 Pull signal bar-code label routing information - line 1.	VARCHAR2(80)
SHIP_LABEL_INFO_LINE_10 Pull signal bar-code label routing information - line 10.	VARCHAR2(80)
SHIP_LABEL_INFO_LINE_2 Pull signal bar-code label routing information - line 2.	VARCHAR2(80)
SHIP_LABEL_INFO_LINE_3 Pull signal bar-code label routing information - line 3.	VARCHAR2(80)
SHIP_LABEL_INFO_LINE_4 Pull signal bar-code label routing information - line 4.	VARCHAR2(80)
SHIP_LABEL_INFO_LINE_5 Pull signal bar-code label routing information - line 5.	VARCHAR2(80)
SHIP_LABEL_INFO_LINE_6 Pull signal bar-code label routing information - line 6.	VARCHAR2(80)
SHIP_LABEL_INFO_LINE_7 Pull signal bar-code label routing information - line 7.	VARCHAR2(80)
SHIP_LABEL_INFO_LINE_8 Pull signal bar-code label routing information - line 8.	VARCHAR2(80)
SHIP_LABEL_INFO_LINE_9 Pull signal bar-code label routing information - line 9.	VARCHAR2(80)

SHIP_TO_ADDRESS_1_EXT	VARCHAR2(35)
Ship to address line 1, as sent by the customer on the N3 segment.	
SHIP_TO_ADDRESS_2_EXT	VARCHAR2(35)
Ship to address line 2, as sent by the customer on the N3 segment.	
SHIP_TO_ADDRESS_3_EXT	VARCHAR2(35)
Ship to address line 3, as sent by the customer on the N3 segment.	
SHIP_TO_ADDRESS_4_EXT	VARCHAR2(35)
Ship to address line 4, as sent by the customer on the N3 segment.	
SHIP_TO_ADDRESS_ID	NUMBER
Ship to address identifier. FK to RA_ADDRESSES.	
DELIVER_TO_ORG_ID	NUMBER
Unique identifier for deliver-to organization which relates to SHIP_TO_ADDRESS_ID. FK in R12 customer-org data model to HR_ORGANIZATIONS	
SHIP_TO_CITY_EXT	VARCHAR2(30)
Ship to address city, as sent by the customer on the N4 segment.	
SHIP_TO_COUNTRY_EXT	VARCHAR2(3)
Ship to country, as sent by the customer on the N4 segment.	
SHIP_TO_COUNTY_EXT	VARCHAR2(25)
Ship to county, as sent by the customer on the N4 segment.	
SHIP_TO_NAME_EXT	VARCHAR2(60)
Ship to name, as sent by the customer on the N1 or N2 segment.	
SHIP_TO_POSTAL_CD_EXT	VARCHAR2(15)
Ship to address postal code, as sent by the customer on the N4 segment.	

SHIP_TO_PROVINCE_EXT **VARCHAR2(10)**

Ship to province, as sent by the customer on the N4 segment.

SHIP_TO_SITE_USE_ID **NUMBER**

Ship to site use identifier. FK to RA_SITE_USES.

SHIP_TO_STATE_EXT **VARCHAR2(10)**

Ship to state, as sent by the customer on the N4 segment.

START_DATE_TIME **DATE**

Customer-specified date and time, as transmitted by the customer on the EDI transaction.

SUBLINE_ASSIGNED_ID_EXT **VARCHAR2(20)**

Subline customer assigned identification from the SLN01. Related to but not necessarily equivalent to the baseline number, assigned identification from LIN01. For example, 1.1 or 1A might be used as a subline number to relate to baseline number 1.

SUBLINE_CONFIG_CODE_EXT **VARCHAR2(3)**

The Subline Configuration Code, as sent by the customer on the SLN segment, indicating the relationship of the subline item to the baseline item.

SUBLINE_CUST_ITEM_EXT **VARCHAR2(50)**

Subline customer item number, as sent by the customer on the SLN segment.

SUBLINE_CUST_ITEM_ID **NUMBER**

Customer item unique identifier. FK to RLM_SHIP_FROM_CUST_ITEM.

SUBLINE_MODEL_NUM_EXT **VARCHAR2(35)**

Subline customer model number, as sent by the customer on the SLN segment.

SUBLINE_QUANTITY **NUMBER**

The subline quantity, as sent by the customer on the SLN segment.

SUBLINE_UOM_CODE **VARCHAR2(3)**

The subline unit of measure (internal) cross-referenced from the UOM sent by the customer on the SLN segment. FK to MTL_UNITS_OF_MEASURE.

SUPPLIER_ITEM_EXT **VARCHAR2(35)**

Supplier item number specified by customer.

TRANSIT_TIME_EXT **VARCHAR2(22)**

The transit time, as sent by the customer on the TD5 segment.

TRANSIT_TIME_QUAL_EXT **VARCHAR2(3)**

The transit time qualifier, as sent by the customer on the TD5 segment.

TRANSPORT_LOC_QUAL_EXT **VARCHAR2(3)**

The shipping location qualifier, as sent by the customer on the TD5 segment. This identifies the type of location which specified in the corresponding shipping location.

TRANSPORT_LOCATION_EXT **VARCHAR2(35)**

The specific shipping location (such as pool point or airport) corresponding to the shipping location qualifier, as sent by the customer on the TD5 segment. This corresponds to the shipping location qualifier.

TRANSPORT_METHOD_EXT **VARCHAR2(3)**

The transportation method, as sent by the customer on the TD5 segment.

UOM_CODE **VARCHAR2(3)**

Abbreviated unit of measure code. FK to MTL_UNITS_OF_MEASURE.

WEIGHT_EXT **NUMBER**

The weight, as sent by the customer on the TD1 segment.

WEIGHT_QUALIFIER_EXT **VARCHAR2(3)**

The weight qualifier, as sent by the customer on the TD1 segment.

WEIGHT_UOM_EXT **VARCHAR2(3)**

The unit of measure corresponding to shipment weight, as sent by the customer on the TD1 segment.

FBO_CONFIGURATION_KEY_1 **VARCHAR2(35)**

For FBO Production Sequence schedules, the 1st sort key for identifying configurations indicated by the customer.

FBO_CONFIGURATION_KEY_2 **VARCHAR2(35)**

For FBO Production Sequence schedules, the 2nd sort key for identifying configurations indicated by the customer.

FBO_CONFIGURATION_KEY_3 **VARCHAR2(35)**

For FBO Production Sequence schedules, the 3rd sort key for identifying configurations indicated by the customer.

FBO_CONFIGURATION_KEY_4 **VARCHAR2(35)**

For FBO Production Sequence schedules, the 4th sort key for identifying configurations indicated by the customer.

FBO_CONFIGURATION_KEY_5 **VARCHAR2(35)**

For FBO Production Sequence schedules, the 5th sort key for identifying configurations indicated by the customer.

MATCH_ACROSS_KEY **VARCHAR2(150)**

Audit trail of match across key used when this schedule was processed.

MATCH_WITHIN_KEY **VARCHAR2(150)**

Audit trail of match within key used when this schedule was processed.

CRITICAL_KEY_ATTRIBUTES **VARCHAR2(150)**

Audit trail of critical attributes key used when this schedule was processed. An exception (warning) is generated if one of these is missing on Firm demand.

LAST_UPDATE_DATE **DATE**

Standard Who column.

LAST_UPDATED_BY

Standard Who column.

NUMBER**CREATION_DATE**

Standard Who column.

DATE**CREATED_BY**

Standard Who column.

NUMBER**ATTRIBUTE_CATEGORY**

Descriptive flexfield context column.

VARCHAR2(30)**ATTRIBUTE1**

Descriptive flexfield segment column.

VARCHAR2(150)**ATTRIBUTE2**

Descriptive flexfield segment column.

VARCHAR2(150)**ATTRIBUTE3**

Descriptive flexfield segment column.

VARCHAR2(150)**ATTRIBUTE4**

Descriptive flexfield segment column.

VARCHAR2(150)**ATTRIBUTE5**

Descriptive flexfield segment column.

VARCHAR2(150)**ATTRIBUTE6**

Descriptive flexfield segment column

VARCHAR2(150)**ATTRIBUTE7**

Descriptive flexfield segment column.

VARCHAR2(150)**ATTRIBUTE8**

Descriptive flexfield segment column.

VARCHAR2(150)

ATTRIBUTE9 Descriptive flexfield segment column.	VARCHAR2(150)
ATTRIBUTE10 Descriptive flexfield segment column.	VARCHAR2(150)
ATTRIBUTE11 Descriptive flexfield segment column.	VARCHAR2(150)
ATTRIBUTE12 Descriptive flexfield segment column.	VARCHAR2(150)
ATTRIBUTE13 Descriptive flexfield segment column.	VARCHAR2(150)
ATTRIBUTE14 Descriptive flexfield segment column	VARCHAR2(150)
ATTRIBUTE15 Descriptive flexfield segment column.	VARCHAR2(150)
LAST_UPDATE_LOGIN Standard Who column.	NUMBER
REQUEST_ID Standard Who column.	NUMBER
PROGRAM_APPLICATION_ID Standard Who column	NUMBER
PROGRAM_ID Standard Who column.	NUMBER
PROGRAM_UPDATE_DATE Standard Who column.	DATE

TP_ATTRIBUTE1 Trading partner flexfield segment column.	VARCHAR2(150)
TP_ATTRIBUTE2 Trading partner flexfield segment column.	VARCHAR2(150)
TP_ATTRIBUTE3 Trading partner flexfield segment column.	VARCHAR2(150)
TP_ATTRIBUTE4 Trading partner flexfield segment column.	VARCHAR2(150)
TP_ATTRIBUTE5 Trading partner flexfield segment column.	VARCHAR2(150)
TP_ATTRIBUTE6 Trading partner flexfield segment column.	VARCHAR2(150)
TP_ATTRIBUTE7 Trading partner flexfield segment column.	VARCHAR2(150)
TP_ATTRIBUTE8 Trading partner flexfield segment column.	VARCHAR2(150)
TP_ATTRIBUTE9 Trading partner flexfield segment column.	VARCHAR2(150)
TP_ATTRIBUTE10 Trading partner flexfield segment column.	VARCHAR2(150)
TP_ATTRIBUTE11 Trading partner flexfield segment column.	VARCHAR2(150)
TP_ATTRIBUTE12 Trading partner flexfield segment column.	VARCHAR2(150)

TP_ATTRIBUTE13 Trading partner flexfield segment column.	VARCHAR2(150)
TP_ATTRIBUTE14 Trading partner flexfield segment column.	VARCHAR2(150)
TP_ATTRIBUTE15 Trading partner flexfield segment column.	VARCHAR2(150)
TP_ATTRIBUTE_CATEGORY Trading partner flexfield context column.	VARCHAR2(30)
LINE_NUMBER Unique identifier of a line within a schedule.	NUMBER
INTMED_SHIP_TO_ORG_ID Unique identifier for intermediate ship-to organization which relates to INTERMEDIATE_SHIP_TO_ID if intermediate ship-to address is specified. FK to HR_ORGANIZATIONS	NUMBER
LINE_SOURCE Schedule Source at line level since lines can also be manually entered.	VARCHAR2(30)
PREFERRED_GRADE Preferred grade.	VARCHAR2(4)
CUST_PRODUCTION_SEQ_NBR_BEG For future use.	VARCHAR2(35)
CUST_PRODUCTION_SEQ_NBR_END For future use.	VARCHAR2(35)
ITEM_DETAIL_QUANTITY_MIN_EXT For future use.	NUMBER

ITEM_DETAIL_QUANTITY_MAX_EXT For future use.	NUMBER
ITEM_DETAIL_QUANTITY_PRIOR_EXT For future use.	NUMBER
REUIREMENT_PRIORITY_EXT For future use.	NUMBER
PROJECT_NUMBER_EXT For future use.	VARCHAR2(50)
RESP_LINE_NUMBER For future use.	VARCHAR2(35)
TASK_NUMBER_EXT For future use.	VARCHAR2(50)
UPC_EXT For future use.	VARCHAR2(50)
ITEM_ENG_CNG_LVL_DATE_EXT For future use.	DATE
AUTHORIZATION_CODE_EXT For future use.	VARCHAR2(50)
PACKAGING_DESC_EXT For future use.	VARCHAR2(80)
BILL_TO_ADDRESS_5_EXT For future use.	VARCHAR2(35)
BILL_TO_ADDRESS_6_EXT For future use.	VARCHAR2(35)

BILL_TO_ADDRESS_7_EXT For future use.	VARCHAR2(35)
BILL_TO_ADDRESS_8_EXT For future use.	VARCHAR2(35)
BILL_TO_ADDRESS_9_EXT For future use.	VARCHAR2(35)
ONE_TIME_BILL_TO_FLAG For future use.	VARCHAR2(35)
BILL_TO_DESCRIPTION_EXT For future use.	VARCHAR2(50)
BILL_TO_FAX_EXT For future use. ³⁵	VARCHAR2(35)
BILL_TO_REGION_EXT For future use.	VARCHAR2(35)
BILL_TO_JURISDICTION_EXT For future use.	VARCHAR2(35)
BILL_TO_TEL_EXT For future use.	VARCHAR2(35)
INTRMD_ST_ADDRESS_5_EXT For future use.	VARCHAR2(35)
INTRMD_ST_ADDRESS_6_EXT For future use.	VARCHAR2(35)
INTRMD_ST_ADDRESS_7_EXT For future use.	VARCHAR2(35)

INTRMD_ST_ADDRESS_8_EXT For future use.	VARCHAR2(35)
INTRMD_ST_ADDRESS_9_EXT For future use.	VARCHAR2(35)
ONE_TIME_INTRMD_ST_FLAG_EXT For future use.	VARCHAR2(35)
INTRMD_ST_DESCRIPTION_EXT For future use.	VARCHAR2(50)
INTRMD_ST_FAX_EXT For future use.	VARCHAR2(35)
INTRMD_ST_REGION_EXT For future use.	VARCHAR2(35)
INTRMD_ST_JURISDICTION_EXT For future use.	VARCHAR2(35)
INTRMD_ST_TEL_EXT For future use.	VARCHAR2(35)
SHIP_TO_ADDRESS_5_EXT For future use.	VARCHAR2(35)
SHIP_TO_ADDRESS_6_EXT For future use.	VARCHAR2(35)
SHIP_TO_ADDRESS_7_EXT For future use.	VARCHAR2(35)
SHIP_TO_ADDRESS_8_EXT For future use.	VARCHAR2(35)

SHIP_TO_ADDRESS_9_EXT For future use.	VARCHAR2(35)
ONE_TIME_SHIP_TO_FLAG_EXT For future use.	VARCHAR2(35)
SHIP_TO_DESCRIPTION_EXT For future use.	VARCHAR2(50)
SHIP_TO_FAX_EXT For future use.	VARCHAR2(35)
SHIP_TO_REGION_EXT For future use.	VARCHAR2(35)
SHIP_TO_JURISDICTION_EXT For future use.	VARCHAR2(35)
SHIP_TO_TEL_EXT For future use.	VARCHAR2(35)
SHIP_FROM_ADDRESS_5_EXT For future use.	VARCHAR2(35)
SHIP_FROM_ADDRESS_6_EXT For future use.	VARCHAR2(35)
SHIP_FROM_ADDRESS_7_EXT For future use.	VARCHAR2(35)
SHIP_FROM_ADDRESS_8_EXT For future use.	VARCHAR2(35)
SHIP_FROM_ADDRESS_9_EXT For future use.	VARCHAR2(35)

ONE_TIME_SHIP_FROM_FLAG_EXT	VARCHAR2(35)
------------------------------------	---------------------

For future use.

SHIP_FROM_DESCRIPTION_EXT	VARCHAR2(50)
----------------------------------	---------------------

For future use.

SHIP_FROM_FAX_EXT	VARCHAR2(35)
--------------------------	---------------------

For future use.

SHIP_FROM_REGION_EXT	VARCHAR2(35)
-----------------------------	---------------------

For future use.

SHIP_FROM_JURISDICTION_EXT	VARCHAR2(35)
-----------------------------------	---------------------

For future use.

SHIP_FROM_TEL_TEXT	VARCHAR2(35)
---------------------------	---------------------

For future use

Oracle Shipping Execution Public APIs

This chapter contains the following information about Oracle Shipping Execution public application program interfaces (APIs):

- [Overview of API Information](#) on page 4-2
- [Shipment Processing Using APIs](#) on page 4-3
- [API Package and Procedures Example](#) on page 4-15
- [Actions, APIs, and Parameters](#) on page 4-26
- [Application Parameter Initialization](#) on page 4-39
- [Trip Public Application Program Interface](#) on page 4-40
- [Stop Public Application Program Interface](#) on page 4-49
- [Deliveries Public Application Program Interface](#) on page 4-59
- [Exceptions Application Program Interface](#) on page 4-79
- [Delivery Details Public Application Program Interface](#) on page 4-90
- [Container Public Application Program Interface](#) on page 4-116
- [Freight Costs Public Application Program Interface](#) on page 4-134
- [Pick Release Application Program Interface](#) on page 4-148
- [Migration from Open Interfaces to Public APIs](#) on page 4-163

Overview of API Information

This chapter contains the following information about Oracle Shipping Execution Public APIs for Release 11i:

- **Shipment Processing Using APIs:** Correlation between common Shipping Transaction Form tasks and Public APIs and some sample scenarios that you can use to process shipments through ship confirmation using APIs.
- **API Package and Procedures Example:** An example of two simple procedures in the same package that use the APIs.
- **Public Application Program Interfaces:** Descriptions of the APIs.
 - **Actions, APIs, and Parameters:** Descriptions of the APIs used for various functions and the API parameters.
 - **Application Parameter Initialization:** Description of the application parameter initialization call.
 - **Trip API:** Create and update trip records and perform actions on trips.
 - **Stop API:** Create and update stop records and perform actions on stops.
 - **Deliveries API:** Create and update trip stop records and perform actions on trip stops.
 - **Delivery Details API:** Assign and unassign delivery details to and from deliveries, split a delivery detail, update a delivery detail with new information, and create trips and deliveries for multiple delivery lines.
 - **Container API:** Create container records, update container records, autopack containers, perform actions on containers.
 - **Freight Cost APIs:** Create freight cost records, update freight cost records, validate freight cost types, delete freight cost records.
- **Migration from Open Interfaces to Public APIs:** Information about migrating your shipping process from the Releases 10.7 and 11 Delivery-based Ship Confirm Open Interface to the Release 11i public APIs.

Shipment Processing Using APIs

This section contains:

- **Shipping Transaction Form/Public API Correlation Table:** This table specifies the Public APIs that correspond to some common actions that you can perform on the Shipping Transaction Form.
- **Sample Flow Scenarios:** The scenarios show different paths to ship confirmation using APIs. Each step in the flows refers to a reference number (Ref) in the API Table.

Note: As a general rule, if an attribute is not updatable from Shipping Transactions Form, then it is not updateable from the public API. If you pass a value for a non-updateable attribute, the code ignores the value and the existing value in the database is retained.

Shipping Transaction Form/Public API Correlation

The shipping flow scenarios refer to the reference numbers (Ref) in this table.

Table 4–1 Shipping Transaction Form/Public API Correlation

Ref	Function	STF Equivalent	Package. Procedure	API Action Codes	Data Requirements
1	Select delivery lines for processing	Find window > Search for Lines > Enter search criteria	This is not a specific API. To do this, populate API input parameter tables before submitting the API.	-	-
2.1	Create trips	Data Entry, Trip Data Entry > Enter data	WSH_TRIPS_PUB. CREATE_UPDATE_TRIP	CREATE	If known, provide information such as trip name, ship method, carrier ID, vehicle number.

Table 4–1 Shipping Transaction Form/Public API Correlation

Ref	Function	STF Equivalent	Package. Procedure	API Action Codes	Data Requirements
2.2	Update trips	Find window > Search for Trips > Enter trip name > Add/change/delete data	WSH_TRIPS_PUB.CREATE_UPDATE_TRIP	UPDATE	If known, provide vehicle type and vehicle number
2.3	Perform actions on trips	Find window > Search for Trips > Select action	WSH_TRIPS_PUB.TRIP_ACTION	PLAN, UNPLAN, DELETE, WT-VOL, PICK-RELEASE	-
3.1	Create stops	Data Entry, Stop Data Entry > Enter data	WSH_TRIP_STOPS_PUB.CREATE_UPDATE_STOP	CREATE	Provide trip ID or trip name, stop location ID or stop location code, planned departure date, and planned arrival date.
3.2	Update stops	Find window > Search for Stops > Add/change/delete data	WSH_TRIP_STOPS_PUB.CREATE_UPDATE_STOP	UPDATE	Provide change information, for example, planned dates.
3.3	Perform actions on stops	Find window > Search for Stops > Select action	WSH_TRIP_STOPS_PUB.STOP_ACTION	PLAN, UNPLAN, ARRIVE, CLOSE, DELETE, PICK-RELEASE	-
4.1	Create deliveries	Data Entry, Delivery Data Entry > Enter data	WSH_DELIVERIES_PUB.CREATE_UPDATE_DELIVERIES	CREATE	Provide organization, initial ship-from, and ultimate ship-to.

Table 4–1 Shipping Transaction Form/Public API Correlation

Ref	Function	STF Equivalent	Package. Procedure	API Action Codes	Data Requirements
4.2	Update deliveries	Find window > Search for Deliveries > Add/change / delete data	WSH_DELIVERIES_PUB.CREATE_UPDATE_DELIVERIES	UPDATE	-
4.3	Perform actions on deliveries	Find window > Search for Deliveries > Select action	WSH_DELIVERIES_PUB.DELIVERY_ACTION	PLAN, UNPLAN, DELETE, WT-VOL, PICK-RELEASE, RE-OPEN, CLOSE	-
4.4	Assign and unassign deliveries to and from trips and stops	Find window > Search for Deliveries > Select action	WSH_DELIVERIES_PUB.DELIVERY_ACTION	ASSIGN-TRIP, UNASSIGN-TRIP, AUTOCREATE-TRIP	For ASSIGN-TRIP, provide trip name.
4.5	Ship confirm deliveries	Find window > Search for Deliveries > Action: Ship Conform	WSH_DELIVERIES_PUB.DELIVERY_ACTION	CONFIRM	Provide actual ship date.
5.1	Assign and unassign delivery lines to and from deliveries	Find window > Search for Lines > Enter delivery line number > Select action	WSH_DELIVERY_DETAILS_PUB.DETAIL_TO_DELIVERY	ASSIGN, UNASSIGN	For ASSIGN, provide delivery ID or delivery name.

Table 4–1 Shipping Transaction Form/Public API Correlation

Ref	Function	STF Equivalent	Package. Procedure	API Action Codes	Data Requirements
5.2	Split delivery lines	Find window > Search for Lines > Enter delivery line number > Action: Split Line	WSH_DELIVERY_DETAILS_PUB.SPLIT_LINE	-	Provide split quantity.
5.3	Confirm actual shipped quantities	Find window > Search for Lines > Enter Shipped Quantity and add/change/delete data	WSH_DELIVERY_DETAILS_PUB.UPDATE_SHIPPING_ATTRIBUTES	UPDATE	-
5.4	Auto-create deliveries	Find window > Search for Lines > Enter criteria > Action: Auto-create Delivery	WSH_DELIVERY_DETAILS_PUB.AUTOCREATE_DELIVERIES	-	-
5.5	Auto-create deliveries and trips	Find window > Search for Lines > Enter criteria > Action: Auto-create Trip	WSH_DELIVERY_DETAILS_PUB.AUTOCREATE_DEL_TRIP	-	-
6.1	Create containers	Data Entry, LPN Data Entry > Enter data	WSH_CONTAINER_PUB.CREATE_CONTAINERS	-	Provide container item ID (prefix, base, and suffix)

Table 4–1 Shipping Transaction Form/Public API Correlation

Ref	Function	STF Equivalent	Package. Procedure	API Action Codes	Data Requirements
6.2	Update containers	Find window > Search for Lines/LPN > Add/change/delete data	WSH_CONTAINER_PUB.UPDATE_CONTAINER	-	-
6.3	Pack and unpack delivery lines into and from containers	Find window > Search for Lines > Enter delivery line number > Select action > Select target LPN	WSH_CONTAINER_PUB.CONTAINER_ACTIONS	PACK, UNPACK	For PACK, provide container ID or container name and delivery line ID.
6.4	Assign and unassign containers to and from deliveries	Find window > Search for LPN > Enter LPN > Select action	WSH_CONTAINER_PUB.CONTAINER_ACTIONS	ASSIGN, UNASSIGN	-
6.5	Create containers and auto-pack	Find window > Search for Lines > Enter delivery line number > Action: Auto-pack	WSH_CONTAINER_PUB.AUTO_PACK	-	Provide delivery line ID

Table 4–1 Shipping Transaction Form/Public API Correlation

Ref	Function	STF Equivalent	Package. Procedure	API Action Codes	Data Requirements
7.1	Create and update freight costs	Find window > Search for entity (Trip, Stop, Delivery, or Delivery Line) > Enter entity name> Action: Freight Cost > Enter or add/change/delete data	WSH_FREIGHT_COSTS_PUB.CREATE_UPDATE_FREIGHT_COST	CREATE, UPDATE	For trip, stop, delivery, and delivery line, provide the entity name or entity ID.
7.2	Validate freight costs	Find window > Search for entity (Trip, Stop, Delivery, or Delivery Line) > Enter entity name> Action: Freight Cost > Open Freight Name LOV and select one	WSH_FREIGHT_COSTS_PUB.VALIDATE_FREIGHT_COST_TYPE	-	-

Table 4–1 Shipping Transaction Form/Public API Correlation

Ref	Function	STF Equivalent	Package. Procedure	API Action Codes	Data Requirements
7.3	Delete freight costs	Find window > Search for entity (Trip, Stop, Delivery, or Delivery Line) > Enter entity name> Action: Freight Cost > Delete row	WSH_FREIGHT_COSTS_PUB.DELETE_FREIGHT_COSTS	-	-

Sample Flow Scenarios

The scenarios show different paths to ship confirmation using APIs. Each step in the flows refers to a reference number (Ref) in the API Table:

- Scenario 1: Pick Release, Ship Confirm
- Scenario 2: Pick Release, Ship Confirm with Serial Numbers
- Scenario 3: Create Delivery, Auto-create Trip, Pick Release Trip
- Scenario 4: Create Trip, Stop, and Delivery; Assign Entities; Pack Items into Containers
- Scenario 5: Auto-create Delivery and Trip, Auto-pack Lines, Add Freight Costs

Figure 4–1 Shipping API Flow Scenario 1

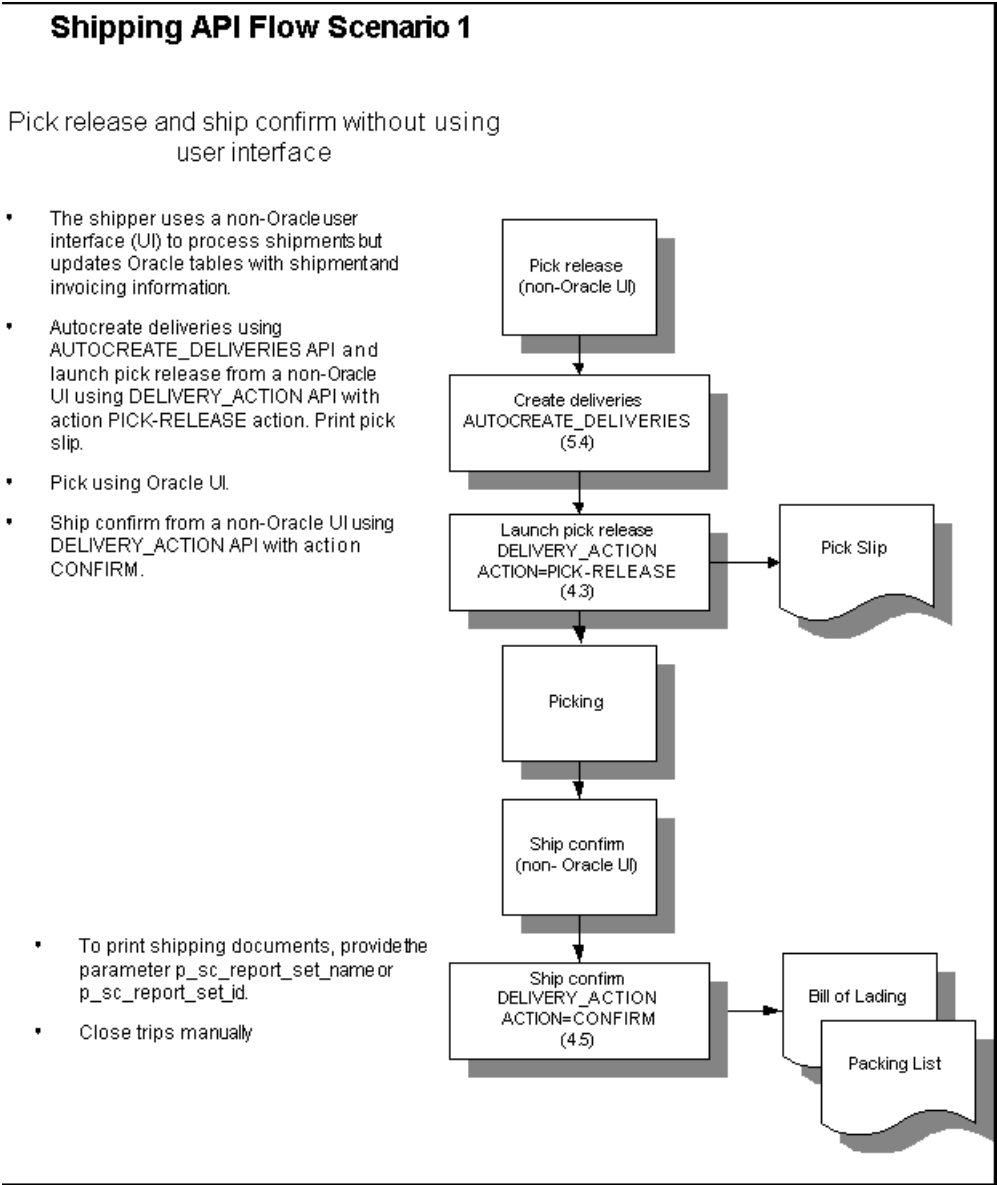


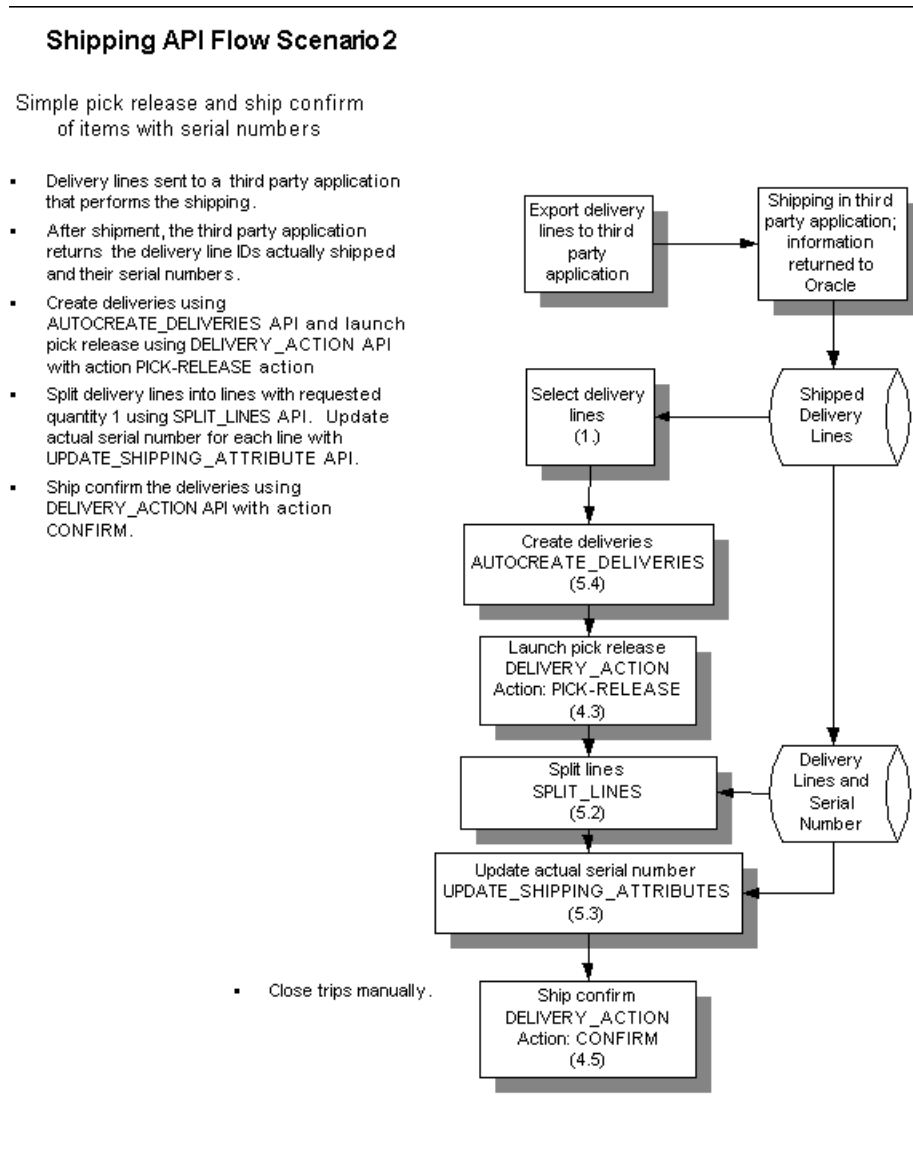
Figure 4–2 Shipping API Flow Scenario 2

Figure 4–3 Shipping API Flow Scenario 3

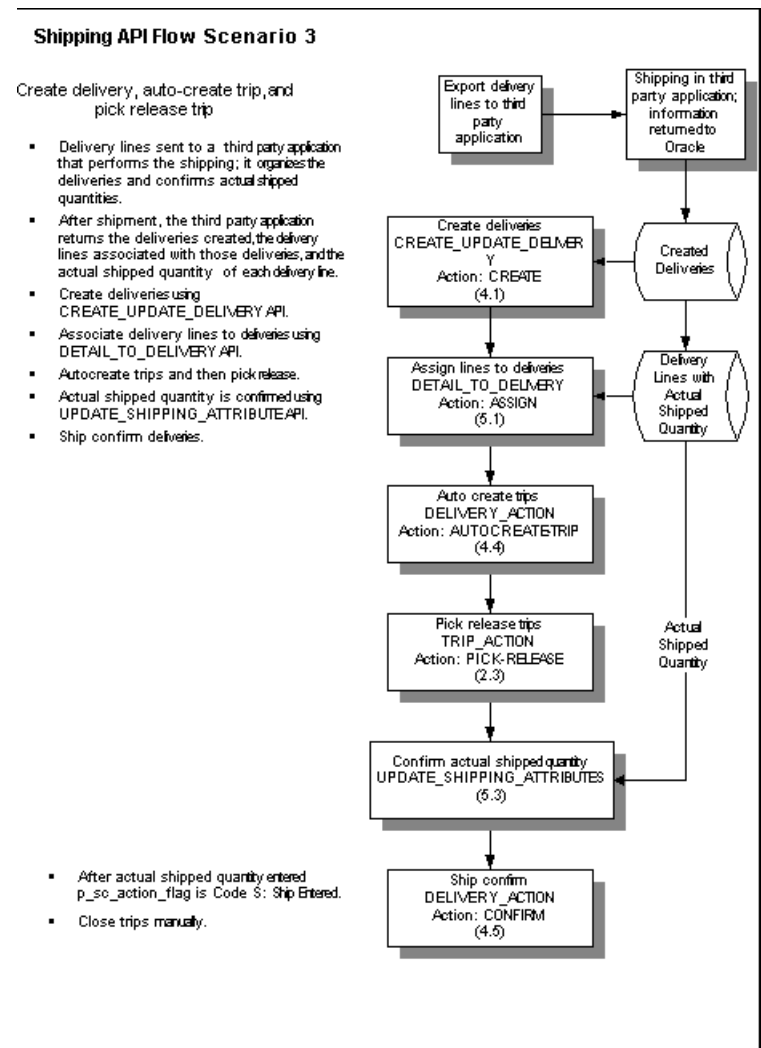


Figure 4–4 Shipping API Flow Scenario 4

Shipping API Flow Scenario 4

Create trip, stop, and delivery, assign lines to deliveries; pack items into containers

- Delivery lines sent to a third party application that performs the shipping; it organizes trips, stops, deliveries, and containers.
- After shipment the third party application returns information.
- Create trips, stops, and deliveries using the CREATE_UPDATE_<entity> APIs.
- Assign deliveries to trips using DELIVERY_ACTION API.
- Create containers, split delivery lines according to their distribution into the containers, and pack delivery lines into containers.
- Assign containers to deliveries.
- Pick release trips and ship confirm deliveries

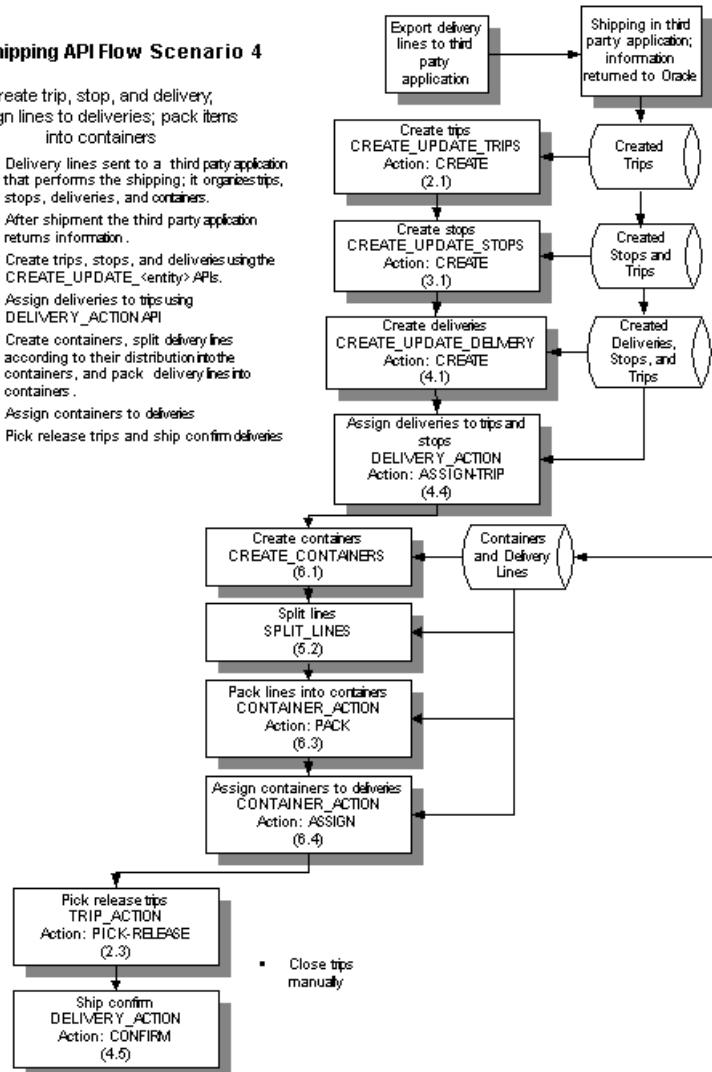
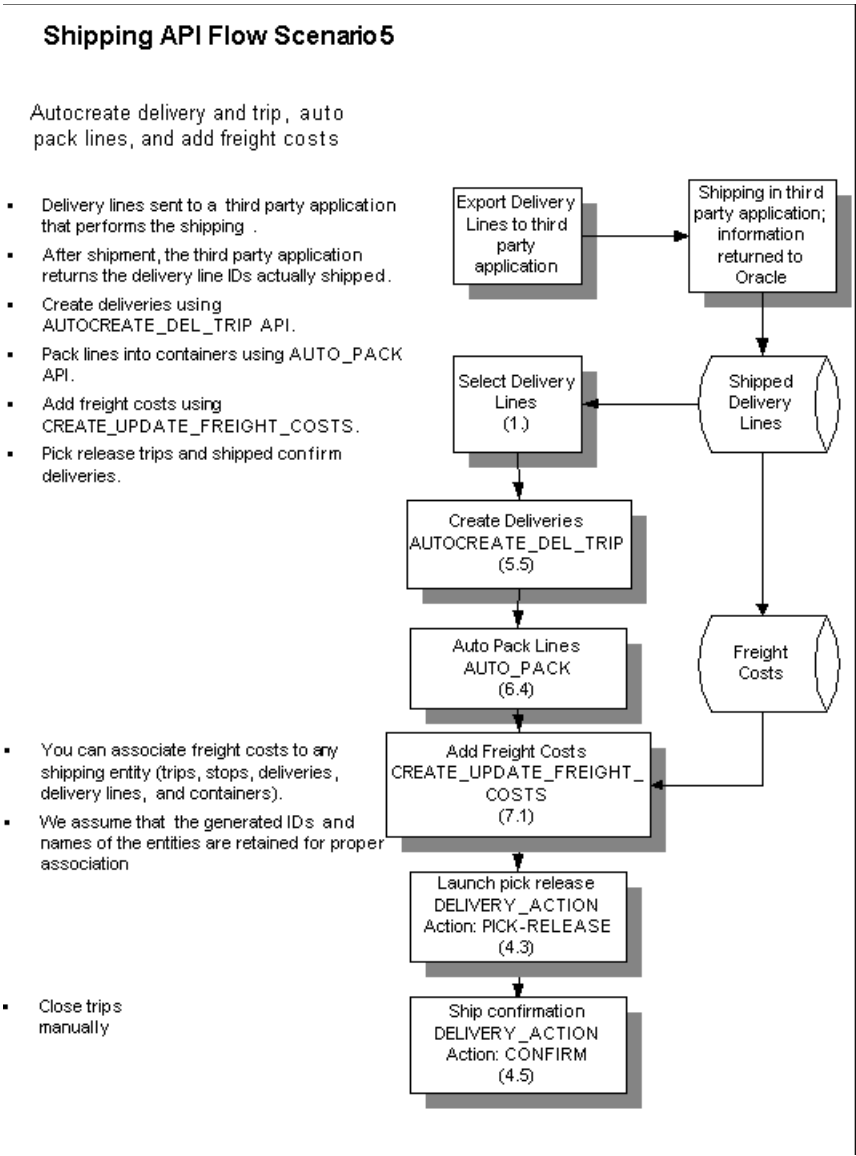


Figure 4–5 Shipping API Flow Scenario 5



API Package and Procedures Example

The following examples demonstrate two simple procedures from the same package that use the shipping public APIs. For more information on using the package, see the Script section that follows the Procedures section.

Package

```
Create or replace package WSH_SHIP_API as

procedure SHIP_CONFIRM_EXAMPLE1 (
  x_return_status      OUT          VARCHAR2,
  x_msg_count          OUT          NUMBER,
  x_msg_data           OUT          VARCHAR2);

procedure SHIP_CONFIRM_EXAMPLE2 (
  x_return_status      OUT          VARCHAR2,
  x_msg_count          OUT          NUMBER,
  x_msg_data           OUT          VARCHAR2);
END WSH_SHIP_API;
/
-- show errors package WSH_SHIP_API;
```

Procedures

```
Create or replace package body WSH_SHIP_API as
/*
```

Example 1:

This procedure can be used as an example on how to ship confirm a delivery consisting of delivery details that have already been pick released and assigned to a delivery.

Call WSH_DELIVERY_DETAILS_PUB.Update_Shipping_Attributes API to update the corresponding delivery details to ship all quantities in the first delivery detail; to back order all in the second; to stage all in the third delivery detail.

Call WSH_FREIGHT_COSTS_PUB.Create_Update_Freight_Costs API to create freight costs for the delivery that the details have been assigned to.

Call WSH_CONTAINER_PUB.Auto_Pack to pack the delivery lines into a container.

Call WSH_DELIVERIES_PUB.Delivery_Action API to ship confirm the delivery.

```
*/
```

```
procedure SHIP_CONFIRM_EXAMPLE1 (
  x_return_status      OUT          VARCHAR2,
  x_msg_count          OUT          NUMBER,
  x_msg_data           OUT          VARCHAR2) IS
```

```
/*
--Standard Parameters.
    p_api_version_number      NUMBER;
    init_msg_list             VARCHAR2(30);
    x_msg_details             VARCHAR2(3000);
    x_msg_summary             VARCHAR2(3000);
    p_validation_level        NUMBER;
    p_commit                  VARCHAR2(30);

--Parameters for WSH_DELIVERY_DETAILS_PUB.Update_Shipping_Attributes.
    source_code               VARCHAR2(15);
    changed_attributes        WSH_DELIVERY_DETAILS_PUB.ChangedAttributeTabType;

--Parameters for WSH_CONTAINER_PUB.Auto_Pack.
    p_entity_tab              WSH_UTIL_CORE.id_tab_type;
    p_entity_type             VARCHAR2(30);
    p_group_id_tab            WSH_UTIL_CORE.id_tab_type;
    p_pack_cont_flag          varchar2(30);
    x_cont_inst_tab           WSH_UTIL_CORE.id_tab_type;

--Parameters for WSH_FREIGHT_COSTS_PUB.Create_Update_Freight_Costs.
    action_code               VARCHAR2(15);
    pub_freight_costs         WSH_FREIGHT_COSTS_PUB.PubFreightCostRecType;
    freight_cost_id           NUMBER;

--Parameters for WSH_DELIVERIES_PUB.Delivery_Action.
    p_action_code             VARCHAR2(15);
    p_delivery_id             NUMBER;
    p_delivery_name           VARCHAR2(30);
    p_asg_trip_id            NUMBER;
    p_asg_trip_name          VARCHAR2(30);
    p_asg_pickup_stop_id     NUMBER;
    p_asg_pickup_loc_id      NUMBER;
    p_asg_pickup_loc_code    VARCHAR2(30);
    p_asg_pickup_arr_date    DATE;
    p_asg_pickup_dep_date    DATE;
    p_asg_dropoff_stop_id    NUMBER;
    p_asg_dropoff_loc_id     NUMBER;
    p_asg_dropoff_loc_code   VARCHAR2(30);
    p_asg_dropoff_arr_date   DATE;
    p_asg_dropoff_dep_date   DATE;
    p_sc_action_flag         VARCHAR2(10);
    p_sc_close_trip_flag     VARCHAR2(10);
    p_sc_create_bol_flag     VARCHAR2(10);
    p_sc_stage_del_flag      VARCHAR2(10);
```



```

    p_sc_trip_ship_method      VARCHAR2(30);
    p_sc_actual_dep_date       VARCHAR2(30);
    p_sc_report_set_id         NUMBER;
    p_sc_report_set_name       VARCHAR2(60);
    p_wv_override_flag         VARCHAR2(10);
    x_trip_id                  VARCHAR2(30);
    x_trip_name                 VARCHAR2(30);

/*Handle exceptions*/
    fail_api                   EXCEPTION;

BEGIN
/* Initialize return status*/
    x_return_status := WSH_UTIL_CORE.G_RET_STS_SUCCESS;

/* Call this procedure to initialize applications parameters. To determine
parameter values, refer to the Application Paramater Initialization section of
this chapter. */
FND_GLOBAL.APPS_INITIALIZE(user_id => 1001594
                           ,resp_id =>52892
                           ,resp_appl_id =>660);

/* Values for updating delivery details to ship all quantities in the first
line, stage everything in the second line, and back order all in the third. It
is assumed that the user knows the quantities in each line.
*/

    source_code := 'OE'; -- The only source code that should be used by the API
    changed_attributes(1).delivery_detail_id := 13431; -- Ship All quantities in
    this detail.
    changed_attributes(1).shipped_quantity := 1;
    changed_attributes(2).source_line_id := 13432; -- Back Order All in this
    -- detail.
    changed_attributes(2).shipped_quantity := 0;
    changed_attributes(2).cycle_count_quantity := 2;
    changed_attributes(3).source_line_id := 13433; -- Stage All in this detail.
    changed_attributes(3).shipped_quantity := 0;
    changed_attributes(3).cycle_count_quantity := 0;

--Call to WSH_DELIVERY_DETAILS_PUB.Update_Shipping_Attributes.
WSH_DELIVERY_DETAILS_PUB.Update_Shipping_Attributes(
    p_api_version_number      => 1.0,
    p_init_msg_list           => init_msg_list,
    p_commit                  => p_commit,
    x_return_status           => x_return_status,

```

```
x_msg_count          => x_msg_count,
x_msg_data           => x_msg_data,
p_changed_attributes => changed_attributes,
p_source_code        => source_code);
if (x_return_status <> WSH_UTIL_CORE.G_RET_STS_SUCCESS) then
    raise fail_api;
end if;

/* Values for creating freight costs for the delivery created for the above
delivery details. The delivery can be queried for the respective delivery detail
through wsh_delivery_assignments.
*/

pub_freight_costs.freight_cost_type_id := 1;
pub_freight_costs.unit_amount         := 20;
pub_freight_costs.currency_code       := 'USD';
pub_freight_costs.delivery_id         := 5341;

--Call to WSH_FREIGHT_COSTS_PUB.Create_Update_Freight_Costs.
WSH_FREIGHT_COSTS_PUB.Create_Update_Freight_Costs(
    p_api_version_number    => 1.0,
    p_init_msg_list         => init_msg_list,
    p_commit                => p_commit,
    x_return_status         => x_return_status,
    x_msg_count             => x_msg_count,
    x_msg_data              => x_msg_data,
    p_pub_freight_costs     => pub_freight_costs,
    p_action_code           => 'CREATE',
    x_freight_cost_id       => freight_cost_id);
if (x_return_status <> WSH_UTIL_CORE.G_RET_STS_SUCCESS) then
    raise fail_api;
end if;
/* Values for autopacking the delivery details to a container.
*/
p_entity_tab(1) := 13431;
p_entity_tab(2) := 13432;
p_entity_tab(3) := 13434;

--Call to WSH_CONTAINER_PUB.Auto_Pack
WSH_CONTAINER_PUB.Auto_Pack(
    p_api_version    => 1.0,
    p_init_msg_list  => init_msg_list,
    p_commit         => p_commit,
    p_validation_level => p_validation_level,
    x_return_status  => x_return_status,
```

```

x_msg_count      => x_msg_count,
x_msg_data       => x_msg_data,
p_entity_tab     => p_entity_tab,
p_entity_type    => 'L',
p_group_id_tab   => p_group_id_tab,
p_pack_cont_flag => p_pack_cont_flag,
x_cont_inst_tab  => x_cont_inst_tab);
if (x_return_status <> WSH_UTIL_CORE.G_RET_STS_SUCCESS) then
    raise fail_api;
end if;

/* Values for Ship Confirming the delivery.
*/
p_action_code      := 'CONFIRM'; -- The action code for ship confirm
p_delivery_id      := 5341; -- The delivery that needs to be confirmed
p_delivery_name    := '5341'; -- The delivery name,
p_sc_action_flag   := 'S'; -- Ship entered quantity.
p_sc_close_trip_flag := 'Y'; -- Close the trip after ship confirm
p_sc_trip_ship_method := 'GROUND'; -- The ship method code

-- Call to WSH_DELIVERIES_PUB.Delivery_Action.
WSH_DELIVERIES_PUB.Delivery_Action(
    p_api_version_number      => 1.0,
    p_init_msg_list           => init_msg_list,
    x_return_status           => x_return_status,
    x_msg_count               => x_msg_count,
    x_msg_data                => x_msg_data,
    p_action_code             => p_action_code,
    p_delivery_id             => p_delivery_id,
    p_delivery_name           => p_delivery_name,
    p_asg_trip_id             => p_asg_trip_id,
    p_asg_trip_name           => p_asg_trip_name,
    p_asg_pickup_stop_id      => p_asg_pickup_stop_id,
    p_asg_pickup_loc_id       => p_asg_pickup_loc_id,
    p_asg_pickup_loc_code     => p_asg_pickup_loc_code,
    p_asg_pickup_arr_date     => p_asg_pickup_arr_date,
    p_asg_pickup_dep_date     => p_asg_pickup_dep_date,
    p_asg_dropoff_stop_id     => p_asg_dropoff_stop_id,
    p_asg_dropoff_loc_id      => p_asg_dropoff_loc_id,
    p_asg_dropoff_loc_code    => p_asg_dropoff_loc_code,
    p_asg_dropoff_arr_date    => p_asg_dropoff_arr_date,
    p_asg_dropoff_dep_date    => p_asg_dropoff_dep_date,
    p_sc_action_flag          => p_sc_action_flag,
    p_sc_close_trip_flag      => p_sc_close_trip_flag,
    p_sc_create_bol_flag      => p_sc_create_bol_flag,

```

```
p_sc_stage_del_flag      => p_sc_stage_del_flag,
p_sc_trip_ship_method    => p_sc_trip_ship_method,
p_sc_actual_dep_date     => p_sc_actual_dep_date,
p_sc_report_set_id       => p_sc_report_set_id,
p_sc_report_set_name     => p_sc_report_set_name,
p_wv_override_flag       => p_wv_override_flag,
x_trip_id               => x_trip_id,
x_trip_name              => x_trip_name);

if (x_return_status <> WSH_UTIL_CORE.G_RET_STS_SUCCESS) then
    raise fail_api;
end if;

exception

when fail_api then
    WSH_UTIL_CORE.get_messages('Y', x_msg_summary, x_msg_details, x_msg_count);
    if x_msg_count > 1 then
        x_msg_data := x_msg_summary || x_msg_details;
    else
        x_msg_data := x_msg_summary;
    end if;
END SHIP_CONFIRM_EXAMPLE1;

/** Example 2:
This procedure can be used as an example on how to ship confirm delivery details
that have originally not been assigned to a delivery.

Call WSH_DELIVERIES_PUB.CREATE_UPDATE_DELIVERY to create a new delivery.
Call WSH_DELIVERY_DETAILS_PUB.Detail_to_Delivery to assign the delivery details
to the new delivery.
Call WSH_DELIVERIES_PUB.Delivery_Action to ship confirm.
**/
procedure SHIP_CONFIRM_EXAMPLE2(
    x_return_status          OUT          VARCHAR2,
    x_msg_count              OUT          NUMBER,
    x_msg_data               OUT          VARCHAR2) IS

-- Standard Parameters.
    p_api_version_number     NUMBER;
    init_msg_list            VARCHAR2(30);
    x_msg_details            VARCHAR2(3000);
    x_msg_summary            VARCHAR2(3000);
    p_validation_level       NUMBER;
    commit                  VARCHAR2(30);
```

```

-- Parameters for WSH_DELIVERIES_PUB.CREATE_UPDATE_DELIVERY
    action_code          VARCHAR2(15);
    delivery_id          NUMBER;
    delivery_info        WSH_DELIVERIES_PUB.Delivery_Pub_Rec_Type;
    name                 VARCHAR2(30);

-- Parameters for WSH_DELIVERY_DETAILS_PUB.Detail_to_Delivery
    p_delivery_id        NUMBER;
    delivery_name        VARCHAR2(30);
    p_TabOfDelDets       WSH_DELIVERY_DETAILS_PUB.id_tab_type;
    p_action             VARCHAR2(30);

-- Parameters for WSH_DELIVERIES_PUB.Delivery_Action.
    p_action_code        VARCHAR2(15);
    p_delivery_id        NUMBER;
    p_delivery_name      VARCHAR2(30);
    p_asg_trip_id        NUMBER;
    p_asg_trip_name      VARCHAR2(30);
    p_asg_pickup_stop_id NUMBER;
    p_asg_pickup_loc_id  NUMBER;
    p_asg_pickup_loc_code VARCHAR2(30);
    p_asg_pickup_arr_date DATE;
    p_asg_pickup_dep_date DATE;
    p_asg_dropoff_stop_id NUMBER;
    p_asg_dropoff_loc_id NUMBER;
    p_asg_dropoff_loc_code VARCHAR2(30);
    p_asg_dropoff_arr_date DATE;
    p_asg_dropoff_dep_date DATE;
    p_sc_action_flag     VARCHAR2(10);
    p_sc_close_trip_flag VARCHAR2(10);
    p_sc_create_bol_flag VARCHAR2(10);
    p_sc_stage_del_flag  VARCHAR2(10);
    p_sc_trip_ship_method VARCHAR2(30);
    p_sc_actual_dep_date VARCHAR2(30);
    p_sc_report_set_id   NUMBER;
    p_sc_report_set_name VARCHAR2(60);
    p_wv_override_flag   VARCHAR2(10);
    x_trip_id            VARCHAR2(30);
    x_trip_name          VARCHAR2(30);
/*Handle exceptions*/
    fail_api             EXCEPTION;

BEGIN
    -- Initialize return status

```

```
x_return_status := WSH_UTIL_CORE.G_RET_STS_SUCCESS;

-- Values for WSH_DELIVERIES_PUB.CREATE_UPDATE_DELIVERY
-- Create a new delivery for the following
  delivery_info.initial_pickup_location_id      := 204;
  delivery_info.ultimate_dropoff_location_id    := 840;
  delivery_info.gross_weight                    := 10;
  delivery_info.ship_method_code                := 'UPS';
  p_action_code                                := 'CREATE';

-- Call to WSH_DELIVERIES_PUB.CREATE_UPDATE_DELIVERY
WSH_DELIVERIES_PUB.CREATE_UPDATE_DELIVERY(
  p_api_version_number      => 1.0,
  p_init_msg_list           => init_msg_list,
  x_return_status           => return_status,
  x_msg_count               => msg_count,
  x_msg_data               => msg_data,
  p_action_code             => p_action_code,
  p_delivery_info           => delivery_info,
  p_delivery_name           => delivery_name,
  x_delivery_id             => delivery_id,
  x_name                    => name );

if (x_return_status <> WSH_UTIL_CORE.G_RET_STS_SUCCESS) then
  raise fail_api;
end if;

p_delivery_id              := delivery_id;
pub_freight_costs.delivery_id := delivery_id;

-- Values for WSH_DELIVERY_DETAILS_PUB.Detail_to_Delivery
-- Call Detail_to_Delivery with an action code of ASSIGN to assign details to a
-- delivery.
  p_TabOfDelDets(1)        := 13463;
  p_TabOfDelDets(2)        := 13464;
  p_action                  := 'ASSIGN';

-- Call to WSH_DELIVERY_DETAILS_PUB.Detail_to_Delivery.
WSH_DELIVERY_DETAILS_PUB.Detail_to_Delivery(
  p_api_version            => 1.0,
  p_init_msg_list          => init_msg_list,
  p_commit                 => commit,
  p_validation_level       => p_validation_level,
  x_return_status          => return_status,
  x_msg_count              => msg_count,
```

```

x_msg_data           => msg_data,
p_TabOfDelDets       => p_TabOfDelDets,
p_action             => p_action,
p_delivery_id        => p_delivery_id,
p_delivery_name      => delivery_name);

if (x_return_status <> WSH_UTIL_CORE.G_RET_STS_SUCCESS) then
    raise fail_api;
end if;

-- Values for Ship Confirming the delivery.
p_action_code        := 'CONFIRM'; -- The action code for ship confirm
p_delivery_id        := 5341; -- The delivery that needs to be confirmed
p_delivery_name       := '5341'; -- The delivery name,
p_sc_action_flag      := 'S'; -- Ship entered quantity.
p_sc_close_trip_flag  := 'Y'; -- Close the trip after ship confirm
p_sc_trip_ship_method := 'UPS'; -- The ship method code

-- Call to WSH_DELIVERIES_PUB.Delivery_Action.
WSH_DELIVERIES_PUB.Delivery_Action(
    p_api_version_number    => 1.0,
    p_init_msg_list         => init_msg_list,
    x_return_status         => x_return_status,
    x_msg_count             => x_msg_count,
    p_action_code           => p_action_code,
    p_delivery_id           => p_delivery_id,
    p_delivery_name         => p_delivery_name,
    p_asg_trip_id           => p_asg_trip_id,
    p_asg_trip_name         => p_asg_trip_name,
    p_asg_pickup_stop_id    => p_asg_pickup_stop_id,
    p_asg_pickup_loc_id     => p_asg_pickup_loc_id,
    p_asg_pickup_loc_code   => p_asg_pickup_loc_code,
    p_asg_pickup_arr_date   => p_asg_pickup_arr_date,
    p_asg_pickup_dep_date   => p_asg_pickup_dep_date,
    p_asg_dropoff_stop_id   => p_asg_dropoff_stop_id,
    p_asg_dropoff_loc_id    => p_asg_dropoff_loc_id,
    p_asg_dropoff_loc_code  => p_asg_dropoff_loc_code,
    p_asg_dropoff_arr_date  => p_asg_dropoff_arr_date,
    p_asg_dropoff_dep_date  => p_asg_dropoff_dep_date,
    p_sc_action_flag        => p_sc_action_flag,
    p_sc_close_trip_flag    => p_sc_close_trip_flag,
    p_sc_create_bol_flag    => p_sc_create_bol_flag,
    p_sc_stage_del_flag     => p_sc_stage_del_flag,
    p_sc_trip_ship_method   => p_sc_trip_ship_method,
    p_sc_actual_dep_date    => p_sc_actual_dep_date,

```

```
p_sc_report_set_id      => p_sc_report_set_id,
p_sc_report_set_name    => p_sc_report_set_name,
p_wv_override_flag      => p_wv_override_flag,
x_trip_id               => x_trip_id,
x_trip_name             => x_trip_name);

if (x_return_status <> WSH_UTIL_CORE.G_RET_STS_SUCCESS) then
    raise fail_api;
end if;

exception

when fail_api then
    WSH_UTIL_CORE.get_messages('Y', x_msg_summary, x_msg_details, x_msg_count);
    if x_msg_count > 1 then
        x_msg_data := x_msg_summary || x_msg_details;
    else
        x_msg_data := x_msg_summary;
    end if;

END SHIP_CONFIRM_EXAMPLE2;

END WSH_SHIP_API;
/
--show errors package body WSH_SHIP_API;
--COMMIT;
```

Script

```
set serveroutput on

Declare

x_return_status          VARCHAR2(15);
x_msg_count              NUMBER;
x_msg_data                VARCHAR2(3000);

Begin

    WSH_SHIP_API.SHIP_CONFIRM_EXAMPLE1(
        x_return_status => x_return_status;
        x_msg_count      => x_msg_count;
        x_msg_data        => x_msg_data);

    dbms_output.put_line('The return status: ' || x_return_status);
```



```
dlms_output.put_line(x_msg_data);
```

```
End;
```

Actions, APIs, and Parameters

This section specifies the APIs to use to perform actions on shipping entities and lists the parameters that the APIs use.

Each time you call an API, include the parameters listed in the column Required Parameters and, in addition, include the standard parameters `p_api_version_number`, `p_init_msg_list`, `p_commit`, `p_validation_level`, `x_return_status`, `x_msg_count`, and `x_msg_data`.

Some of the parameters may have default values in their API signatures. Refer to the Record Parameter Description section for each API to see if a parameter has a default value.

Some of the parameters are of type Table or Record. Refer to the Record Parameter Description section for each API to see table and record definitions.

Trips

Table 4–2 Trip Actions, APIs, and Parameters

Action	API	Required Parameters	Optional Parameters	Comments
Plan	WSH_TRIPS_PUB.Trip_Action	<code>p_action_code = PLAN</code> <code>p_trip_id</code> or <code>p_trip_name</code>	-	-
Unplan	WSH_TRIPS_PUB.Trip_Action	<code>p_action_code = UNPLAN</code> <code>p_trip_id</code> or <code>p_trip_name</code>	-	-
Launch pick release	WSH_TRIPS_PUB.Trip_Action	<code>p_action_code = PICK-RELEASE</code> <code>p_trip_id</code> or <code>p_trip_name</code>	-	-
Calculate weight and volume	WSH_TRIPS_PUB.Trip_Action	<code>p_action_code = WT-VOL</code> <code>p_trip_id</code> or <code>p_trip_name</code>	<code>p_wv_override_flag</code>	-

Table 4–2 Trip Actions, APIs, and Parameters

Action	API	Required Parameters	Optional Parameters	Comments
Delete	WSH_TRIPS_PUB.Trip_Action	p_action_code = DELETE p_trip_id or p_trip_name	-	-
Create	WSH_TRIPS_PUB.Create_Update_Trip	p_action_code = CREATE p_trip_info x_trip_id x_trip_name	p_trip_name	-
Update	WSH_TRIPS_PUB.Create_Update_Trip	p_action_code = UPDATE p_trip_info x_trip_id x_trip_name	p_trip_name	For p_trip_info, pass only the attributes that need update.

Stops

Table 4–3 Stop Actions, APIs, and Parameters

Action	API	Required Parameters	Optional Parameters	Comments
Plan	WSH_TRIP_STOPS_PUB.Stop_Action	p_action_code = PLAN p_stop_id	-	You can substitute all of the following for p_stop_id: p_trip_name or p_trip_id p_stop_location_code or p_stop_location_id p_planned_dep_date

Table 4–3 Stop Actions, APIs, and Parameters

Action	API	Required Parameters	Optional Parameters	Comments
Unplan	WSH_TRIP_STOPS_PUB. Stop_Action	p_action_code = UNPLAN p_stop_id	-	You can substitute all of the following for p_stop_id: p_trip_name or p_trip_id p_stop_location_code or p_stop_location_id p_planned_dep_date
Update Status - Arrive	WSH_TRIP_STOPS_PUB. Stop_Action	p_action_code = ARRIVE p_stop_id	p_actual_date p_defer_interface_flag	You can substitute all of the following for p_stop_id: p_trip_name or p_trip_id p_stop_location_code or p_stop_location_id p_planned_dep_date
Update Status - Close	WSH_TRIP_STOPS_PUB. Stop_Action	p_action_code = CLOSE p_stop_id	p_actual_date p_defer_interface_flag	You can substitute all of the following for p_stop_id: p_trip_name or p_trip_id p_stop_location_code or p_stop_location_id p_planned_dep_date

Table 4–3 Stop Actions, APIs, and Parameters

Action	API	Required Parameters	Optional Parameters	Comments
Launch pick release	WSH_TRIP_STOPS_PUB. Stop_Action	p_action_code = PICK-RELEASE p_stop_id	-	You can substitute all of the following for p_stop_id: p_trip_name or p_trip_id p_stop_location_code or p_stop_location_id p_planned_dep_date
Delete	WSH_TRIP_STOPS_PUB. Stop_Action	p_action_code = DELETE p_stop_id	-	You can substitute all of the following for p_stop_id: p_trip_name or p_trip_id p_stop_location_code or p_stop_location_id p_planned_dep_date
Create	WSH_TRIP_STOPS_PUB. Create_Update_Stop	p_action_code = CREATE p_stop_info x_stop_id	-	-
Update	WSH_TRIP_STOPS_PUB. Create_Update_Stop	p_action_code = UPDATE p_stop_info x_stop_id	p_trip_id p_trip_name p_stop_location_id p_stop_location_code p_planned_dep_date	For p_stop_info, pass only the attributes that need update.

Deliveries

Table 4–4 Delivery Actions, APIs, and Parameters

Action	API	Required Parameters	Optional Parameters	Comments
Plan	wsh_deliveries_pub.delivery_action	p_action_code = PLAN p_delivery_id or p_delivery_name x_trip_id x_trip_name	-	-
Unplan	wsh_deliveries_pub.delivery_action	p_action_code = UNPLAN p_delivery_id or p_delivery_name x_trip_id x_trip_name	-	-
Ship confirm	wsh_deliveries_pub.delivery_action	p_action_code = CONFIRM p_delivery_id or p_delivery_name x_trip_id x_trip_name	p_sc_action_flag p_sc_close_trip_flag p_sc_create_bol p_sc_stage_del_flag p_sc_trip_ship_method p_sc_actual_dep_date p_sc_report_set_id p_sc_report_set_name p_sc_defer_interface	-

Table 4–4 Delivery Actions, APIs, and Parameters

Action	API	Required Parameters	Optional Parameters	Comments
Reopen	wsh_deliveries_ pub. delivery_ action	p_action_code = RE-OPEN p_delivery_id or p_delivery_ name x_trip_id x_trip_name	-	-
Close	wsh_deliveries_ pub. delivery_ action	p_action_code = CLOSE p_delivery_id or p_delivery_ name x_trip_id x_trip_name	-	-

Table 4–4 Delivery Actions, APIs, and Parameters

Action	API	Required Parameters	Optional Parameters	Comments
Assign to trip	wsh_deliveries_ pub. delivery_ action	p_action_code = ASSIGN-TRIP p_delivery_id or p_delivery_ name p_asg_trip_id or p_asg_trip_ name x_trip_id x_trip_name	p_asg_pickup_ stop_id p_asg_pickup_ loc_id p_asg_pickup_ stop_seq p_asg_pickup_ loc_code p_asg_pickup_ arr_date p_asg_pickup_ dep_date p_asg_dropoff_ stop_id p_asg_dropoff_ loc_id p_asg_dropoff_ stop_seq p_asg_dropoff_ loc_code p_asg_dropoff_ arr_date p_asg_dropoff_ dep_date	-
Unassign from trip	wsh_deliveries_ pub. delivery_ action	p_action_code = UNASSIGN- TRIP p_delivery_id or p_delivery_ name x_trip_id x_trip_name	-	-

Table 4–4 Delivery Actions, APIs, and Parameters

Action	API	Required Parameters	Optional Parameters	Comments
Auto-create trip	wsh_deliveries_ pub. delivery_ action	p_action_code = AUTOCREATE - TRIP p_delivery_id or p_delivery_ name x_trip_id x_trip_name	-	-
Calculate weight and volume	wsh_deliveries_ pub. delivery_ action	p_action_code = WT-VOL p_delivery_id or p_delivery_ name x_trip_id x_trip_name	p_wv_ override_flag	-
Launch pick release	wsh_deliveries_ pub. delivery_ action	p_action_code = PICK- RELEASE p_delivery_id or p_delivery_ name x_trip_id x_trip_name	-	-
Delete	wsh_deliveries_ pub. delivery_ action	p_action_code = DELETE p_delivery_id or p_delivery_ name x_trip_id x_trip_name	-	-

Table 4–4 Delivery Actions, APIs, and Parameters

Action	API	Required Parameters	Optional Parameters	Comments
Create	wsh_deliveries_pub.create_delivery	p_action_code = CREATE p_delivery_info x_delivery_id x_name	p_delivery_name	-
Update	wsh_deliveries_pub.update_delivery	p_action_code = UPDATE p_delivery_info x_delivery_id x_name	p_delivery_name	For p_delivery_info, pass only the attributes that need update.
Auto-pack	WSH_CONTAINER_PUB.Auto_Pack	p_entity_tab p_entity_type p_group_id_tab p_pack_cont_flag x_cont_inst_tab	-	-

Delivery Details

Table 4–5 Delivery Detail Actions, APIs, and Parameters

Action	API	Required Parameters	Optional Parameters	Comments
Assign to delivery	wsh_delivery_details_pub.detail_to_delivery	p_TabOfDelDets p_action = ASSIGN, p_delivery_id p_delivery_name	-	-

Table 4–5 Delivery Detail Actions, APIs, and Parameters

Action	API	Required Parameters	Optional Parameters	Comments
Unassign from delivery	wsh_delivery_details_pub. detail_to_delivery	p_TabOfDelDets p_action = UNASSIGN p_delivery_id p_delivery_name	-	-
Split line	wsh_delivery_details_pub. split_line	p_from_detail_id x_split_quantity x_new_detail_id x_split_quantity2	-	-
Update	wsh_delivery_details_pub. Update_Shipping_Attributes	p_changed_attributes p_source_code	p_container_flag	For p_changed_attributes, pass only the attributes that need update.
Autocreate delivery	wsh_delivery_details_pub. Autocreate_Deliveries	p_line_rows x_del_rows	-	-
Autocreate trip	wsh_delivery_details_pub. Autocreate_del_trip	p_line_rows x_del_rows x_trip_id x_trip_name	-	-

Table 4–5 Delivery Detail Actions, APIs, and Parameters

Action	API	Required Parameters	Optional Parameters	Comments
Pack into a container	WSH_CONTAINER_PUB.Container_Actions	p_detail_tab p_action_code=PACK p_container_name or p_cont_instance_id	p_container_flag p_delivery_flag	-
Unpack from a container	WSH_CONTAINER_PUB.Container_Actions	p_detail_tab p_action_code=UNPACK p_container_name or p_cont_instance_id	p_container_flag p_delivery_flag	-
Auto-pack	WSH_CONTAINER_PUB.Auto_Pack	p_entity_tab p_entity_type p_group_id_tab p_pack_cont_flag x_cont_inst_tab	-	-

Table 4–5 Delivery Detail Actions, APIs, and Parameters

Action	API	Required Parameters	Optional Parameters	Comments
Create container	WSH_CONTAINER_PUB.Create_Containers	p_container_item_id p_container_item_name p_container_item_seg p_organization_id p_organization_code p_name_prefix p_name_suffix p_base_number p_num_digits p_quantity p_container_name x_container_ids	-	-
Update container	WSH_CONTAINER_PUB.Update_Container	p_container_rec	-	For p_container_rec, pass only the attributes that need update.
Assign container to a delivery	WSH_CONTAINER_PUB.Container_Actions	p_detail_tab p_action_code = ASSIGN p_delivery_id or p_delivery_name	p_container_flag p_delivery_flag	-

Table 4–5 Delivery Detail Actions, APIs, and Parameters

Action	API	Required Parameters	Optional Parameters	Comments
Unassign container from a delivery	WSH_CONTAINER_PUB.Container_Actions	p_detail_tab p_action_code = UNASSIGN p_delivery_id or p_delivery_name	p_container_flag p_delivery_flag	-

Application Parameter Initialization

In your scripts that call Public APIs, include a call to `fnd_global.apps_initialize` immediately after the `BEGIN`. The call syntax is

```
fnd_global.apps_initialize(user_id => l_user_id,  
                           resp_id => l_resp_id,  
                           resp_appl_id => l_resp_appl_id,  
                           security_group_id => l_security_group_id);
```

Finding Variable Values

To find the values for the variables, perform the following process:

- Navigate to the Transaction form.
- On the menu bar, navigate `Help > Diagnostics > Examine`; a window pops-up.
- In Block, enter `$PROFILES$`
- In Field, enter `USER_ID` and press Tab. Use the value in Value for `l_user_id`.
- In Field, enter `RESP_ID` and press Tab. Use the value in Value for `l_resp_id`.
- In Field, enter `RESP_APPL_ID` and press Tab. Use the value in Value for `l_resp_appl_id`.
- In Field, enter `SECURITY_GROUP_ID` and press Tab. Use the value in Value for `l_security_group_id`.

Trip Public Application Program Interface

The Trip Public Application Program Interface (API) is a public API that consists of the following two procedures in package **WSH_TRIPS_PUB**:

Create_Update_Trip: Enables you to create a new trip record and update an existing trip record.

Trip_Action: Enables you to perform certain actions on a trip.

This section describes how to use the Trip Public API and how it functions in Oracle Shipping Execution.

Create_Update_Trip API Features

The Create_Update_Trip API has the following features.

The Create_Update_Trip procedure enables you to create a new trip record or update an existing trip record in the WSH_TRIPS table. The TRIP_ID, NAME and return status of a new trip are passed as OUT parameters, while the trip NAME of an existing trip for update is passed as an IN parameter.

Functional Overview

This API creates a new trip record in WSH_TRIPS as specified by IN parameter p_action_code value CREATE. It inserts the trip information into WSH_TRIPS and returns the TRIP_ID and NAME of the new trip. It also updates an existing trip record in WSH_TRIPS as specified by IN parameter p_action_code value UPDATE. The NAME of the trip being updated is passed through IN parameter p_trip_name.

The API validates trip information such as Trip Name, Arrive After Trip Name, and Ship Method before performing the actions of creating or updating a trip record. It also checks that the insert or update statements were successful, and if not returns an error.

Procedure Parameter Descriptions

WSH_TRIPS_PUB.CREATE_UPDATE_TRIP

The following chart describes all parameters used by the public procedure WSH_TRIPS_PUB.CREATE_UPDATE_TRIP. All of the inbound and outbound parameters are listed. Additional information on these parameters follows.

Table 4–6 WSH_TRIPS_PUB.CREATE_UPDATE_TRIP Parameters

Parameter	Usage	Type	Required
p_api_version_number	IN	Number	x
p_init_msg_list	IN	Varchar2	-
x_return_status	OUT	Varchar2	-
x_msg_count	OUT	Number	-
x_msg_data	OUT	Varchar2	-
p_action_code	IN	Varchar2	x
p_trip_info	IN OUT	Record	x
p_trip_name	IN	Varchar2	-
x_trip_id	OUT	Number	-
x_trip_name	OUT	Varchar2	-

p_api_version_number

Compares the incoming API call's version number with the current version number. An error is returned if the version numbers are incompatible.

p_init_msg_list

Requests that the API initialize the message list on your behalf. If the x_msg_count is greater than 1, then the list of messages must be retrieved using the call FND_MSG_PUB.GET. The values are:

- p_msg_index => I
- p_encoded => F
- p_data => 1_message
- p_msg_index_out => 1_msg_index_out

where 1_message and 1_msg_index_out are local variables of types Varchar2(2000) and Number respectively.

Default Value: FND_API.G_FALSE

x_return_status

Requests that the API return the status of the data for you after it completes its function. Valid values include:

- Success: FND_API.G_RET_STS_SUCCESS
- Error: FND_API.G_RET_STS_ERROR
- Unexpected Error: FND_API.G_RET_STS_UNEXP_ERROR

x_msg_count

Indicates number of error messages API has encountered.

x_msg_data

Returns error message text. If the x_msg_count is equal to 1, then this contains the actual message.

p_action_code

Specifies whether API should create a new trip or update existing trip information based on its values CREATE or UPDATE.

p_trip_info

Attributes of the trip entity of type Trip_Pub_Rec_Type. These attributes are inserted/updated in WSH_TRIPS. Definition of Trip_Pub_Rec_Type follows.

p_trip_name

IN parameter to specify the Name of trip to be updated.

Default Value: FND_API.G_MISS_CHAR

x_trip_id

Trip ID of new trip being created.

x_trip_name

Name of new trip being created.

Record Parameter Descriptions

TRIP_PUB_REC_TYPE RECORD DEFINITION

To encapsulate WSH_TRIPS table definition and Value column equivalents for ID columns in a PL/SQL record, define TRIP_PUB_REC_TYPE to pass trip information to the Create_Update_Trip routine.

Table 4–7 TRIP_PUB_REC_TYPE Record Definition

Attribute	Type	Default Value
trip_id	Number	fnd_api.g_miss_num
name	Varchar2(30)	fnd_api.g_miss_char
arrive_after_trip_id	Number	fnd_api.g_miss_num
arrive_after_trip_name	Varchar2(30)	fnd_api.g_miss_char
vehicle_item_id	Number	fnd_api.g_miss_num
vehicle_item_desc	Varchar2(240)	fnd_api.g_miss_char
vehicle_organization_id	Number	fnd_api.g_miss_num
vehicle_organization_code	Varchar2(3)	fnd_api.g_miss_char
vehicle_number	Varchar2(30)	fnd_api.g_miss_char
vehicle_num_prefix	Varchar2(10)	fnd_api.g_miss_char
carrier_id	Number	fnd_api.g_miss_num
ship_method_code	Varchar2(30)	fnd_api.g_miss_char
ship_method_name	Varchar2(80)	fnd_api.g_miss_char
route_id	Number	fnd_api.g_miss_num
routing_instructions	Varchar2(2000)	fnd_api.g_miss_char
attribute_category	Varchar2(150)	fnd_api.g_miss_char
attribute1	Varchar2(150)	fnd_api.g_miss_char
attribute2	Varchar2(150)	fnd_api.g_miss_char
attribute3	Varchar2(150)	fnd_api.g_miss_char
attribute4	Varchar2(150)	fnd_api.g_miss_char
attribute5	Varchar2(150)	fnd_api.g_miss_char
attribute6	Varchar2(150)	fnd_api.g_miss_char
attribute7	Varchar2(150)	fnd_api.g_miss_char
attribute8	Varchar2(150)	fnd_api.g_miss_char
attribute9	Varchar2(150)	fnd_api.g_miss_char
attribute10	Varchar2(150)	fnd_api.g_miss_char

Table 4–7 TRIP_PUB_REC_TYPE Record Definition

Attribute	Type	Default Value
attribute11	Varchar2(150)	fnd_api.g_miss_char
attribute12	Varchar2(150)	fnd_api.g_miss_char
attribute13	Varchar2(150)	fnd_api.g_miss_char
attribute14	Varchar2(150)	fnd_api.g_miss_char
attribute15	Varchar2(150)	fnd_api.g_miss_char
creation_date	Date	fnd_api.g_miss_char
created_by	Number	fnd_api.g_miss_num
last_update_date	Date	fnd_api.g_miss_date
last_updated_by	Number	fnd_api.g_miss_num
last_update_login	Date	fnd_api.g_miss_date
program_application_id	Number	fnd_api.g_miss_num
program_id	Number	fnd_api.g_miss_num
program_update_date	Date	fnd_api.g_miss_date
request_id	Number	fnd_api.g_miss_num
service_level	Varchar2(30)	fnd_api.g_miss_char
mode_of_transport	Varchar2(30)	fnd_api.g_miss_char
operator	Varchar2(150)	fnd_api.g_mis_char

Record Parameter Attribute Validations

trip_id

Should be a valid unique element of wsh_trips.trip_id.

name

Should be a valid unique element of wsh_trips.name

arrive_after_trip_id

Should be a valid element of wsh_trips.trip_id.

arrive_after_trip_name

Should be a valid element of wsh_trips.name

vehicle_item_id

Should be a valid vehicle item in mtl_system_items.inventory_item_id as designated by mtl_system_items.vehicle_tem_flag.

vehicle_item_desc

Key flex field description for vehicle item.

vehicle_organization_id

Should be a valid element of org_organization_definitions.organization_id and be assigned to the specific item in mtl_system_items.organization_id.

vehicle_organization_code

Should be a valid element of org_organization_definitions.organization_code for that specific organization_id.

ship_method_code

Should be a valid element of fnd_lookup_values_vl.lookup_code for lookup_type SHIP_METHOD.

ship_method_name

Should be a valid element of fnd_lookup_values_vl.meaning for lookup_type SHIP_METHOD.

Error Handling

Refer to parameters p_init_msg_list, x_msg_count, and x_msg_data on retrieving error messages.

Trip_Action API Features

The Trip_Action API has the following features:

The Trip_Action procedure enables you to carry out actions on a trip. It accepts as IN parameters the trip identifiers, an action code and any additional parameters needed for specific actions, and returns a completion status.

Functional Overview

This API validates the trip identifiers and performs the following actions specified in p_action_code by calling the corresponding private procedures.

- PLAN/UNPLAN: Is called to plan or unplan trips.
- WT-VOL: Is called to calculate weight and volume for the trip.
- PICK-RELEASE: Is called to launch pick release to release lines related to the trip.
- PRINT-DOC-SETS: Is called to print document sets.
- DELETE: Is called to delete trips.

Procedure Parameter Descriptions

WSH_TRIPS_PUB.TRIP_ACTION

The following chart describes all parameters used by the public procedure WSH_TRIPS_PUB.TRIP_ACTION. All of the inbound and outbound parameters are listed. Additional information on these parameters follows.

Table 4–8 WSH_TRIPS_PUB.TRIP_ACTION Parameters

Parameter	Usage	Type	Required
p_api_version_number	IN	Number	x
p_init_msg_list	IN	Varchar2	-
x_return_status	OUT	Varchar2	-
x_msg_count	OUT	Number	-
x_msg_data	OUT	Varchar2	-
p_action_code	IN	Varchar2	x
p_trip_id	IN OUT	Record	-

Table 4–8 WSH_TRIPS_PUB.TRIP_ACTION Parameters

Parameter	Usage	Type	Required
p_trip_name	IN	Varchar2	-
p_wv_override_flag	IN	Varchar2	-

p_api_version_number

Compares the incoming API call's version number with the current version number. An error is returned if the version numbers are incompatible.

p_init_msg_list

Requests that the API initialize the message list on your behalf. If the x_msg_count is greater than 1, then the list of messages must be retrieved using the call FND_MSG_PUB.GET. The values are:

- p_msg_index => I
- p_encoded => F
- p_data => 1_message
- p_msg_index_out => 1_msg_index_out

where 1_message and 1_msg_index_out are local variables of types Varchar2(2000) and Number respectively.

Default Value: FND_API.G_FALSE

x_return_status

Requests that the API return the status of the data for you after it completes its function. Valid values include:

- Success: FND_API.G_RET_STS_SUCCESS
- Error: FND_API.G_RET_STS_ERROR
- Unexpected Error: FND_API.G_RET_STS_UNEXP_ERROR

x_msg_count

Indicates number of error messages API has encountered.

x_msg_data

Returns error message text. If the x_msg_count is equal to 1, then this contains the actual message.

p_action_code

Specifies which of the actions of PLAN, UNPLAN, WT-VOL, PICK RELEASE, DELETE the API should perform.

p_trip_id

Trip ID of trip on which actions are being performed.

p_trip_name

Name of trip on which actions are being performed.

p_wv_override_flag

Additional parameter for WT-VOL. Regardless if this is set to Y or N, weight and volume re-calculations are done for all deliveries on the trip. Default value is N.

Error Handling

Refer to parameters p_init_msg_list, x_msg_count, and x_msg_data on retrieving error messages.

Stop Public Application Program Interface

The Stop Public Application Program Interface (API) is a public API that consists of the following two procedures in package **WSH_TRIP_STOPS_PUB**:

Create_Update_Stop: Enables you to create a new stop record and update an existing stop record.

Stop_Action: Enables you to perform certain actions on a stop.

This section describes how to use the Stop Public API and how it functions in Oracle Shipping Execution.

Create_Update_Stop API Features

The Create_Update_Stop API has the following features.

The Create_Update_Stop procedure enables you to create a new stop record or update an existing stop record in WSH_TRIP_STOPS table. The STOP_ID and return status of a new stop are passed as OUT parameters, while the TRIP_ID of an existing stop for update is passed as an IN parameter.

Functional Overview

This API creates a new stop record in WSH_TRIP_STOPS as specified by IN parameter p_action_code value CREATE. It inserts stop information into WSH_TRIP_STOPS and returns the STOP_ID of the new stop. It also updates an existing stop record in WSH_TRIP_STOPS as specified by IN parameter p_action_code value UPDATE. The stop record being updated is identified by Stop Id or Trip Id/Name, Stop Location Id/Code and Planned Departure Date. Additional IN parameters are provided to update Trip Id/Name, Stop Location Id/Code and Planned Departure Date.

The API validates information such as Trip Name, Location Code, and UOMs before performing the actions of creating or updating a stop record. It also checks that the insert or update statements were successful, and if not returns an error.

Procedure Parameter Descriptions

WSH_TRIP_STOPS_PUB.CREATE_UPDATE_STOP

The following chart describes all parameters used by the public procedure WSH_TRIP_STOPS_PUB.CREATE_UPDATE_STOP. All of the inbound and outbound parameters are listed. Additional information on these parameters follows.

Table 4–9 WSH_TRIP_STOPS_PUB.CREATE_UPDATE_STOP Parameters

Parameter	Usage	Type	Required
p_api_version_number	IN	Number	x
p_init_msg_list	IN	Varchar2	-
x_return_status	OUT	Varchar2	-
x_msg_count	OUT	Number	-
x_msg_data	OUT	Varchar2	-
p_action_code	IN	Varchar2	x
p_stop_info	IN OUT	Record	x
p_trip_id	IN	Number	-
p_trip_name	IN	Varchar2	-
p_stop_location_id	IN	Number	-
p_location_code	IN	Varchar2	-
p_planned_dep_date	IN	Varchar2	-
x_stop_id	OUT	Number	-

p_api_version_number

Compares the incoming API call's version number with the current version number. An error is returned if the version numbers are incompatible.

p_init_msg_list

Requests that the API initialize the message list on your behalf. If the x_msg_count is greater than 1, then the list of messages must be retrieved using the call FND_MSG_PUB.GET. The values are:

- p_msg_index => I
- p_encoded => F
- p_data => 1_message
- p_msg_index_out => 1_msg_index_out

where 1_message and 1_msg_index_out are local variables of types Varchar2(2000) and Number respectively.

Default Value: FND_API.G_FALSE

x_return_status

Requests that the API return the status of the data for you after it completes its function. Valid values include:

- Success: FND_API.G_RET_STS_SUCCESS
- Error: FND_API.G_RET_STS_ERROR
- Unexpected Error: FND_API.G_RET_STS_UNEXP_ERROR

x_msg_count

Indicates number of error messages API has encountered.

x_msg_data

Displays error message text. If the x_msg_count is equal to 1, then this contains the actual message.

p_action_code

Specifies whether API should create a new stop or update existing stop information depending on its values CREATE and UPDATE.

p_stop_info

Attributes of the stop entity of type Trip_Stop_Pub_Rec_Type. These attributes are inserted/updated in WSH_TRIP_STOPS. Definition of Trip_Stop_Pub_Rec_Type follows.

p_trip_id

IN parameter to specify trip ID of trip stop record being updated.

Default Value: FND_API.G_MISS_NUM

p_trip_name

IN parameter to specify trip name of trip stop record that is needed to be updated.

Default Value: FND_API.G_MISS_CHAR

p_stop_location_id

IN parameter to specify stop location ID of trip stop record being updated.

Default Value: FND_API.G_MISS_NUM

p_location_code

IN parameter to specify stop location code of stop record that is needed to be updated.

Default Value: FND_API.G_MISS_CHAR

p_planned_dep_date

IN parameter to specify planned date of departure.

Default Value: FND_API.G_MISS_DATE

x_stop_id

OUT parameter to specify new stop ID of new stop record being created.

Record Parameter Descriptions

TRIP_STOP_PUB_REC_TYPE RECORD DEFINITION

To encapsulate WSH_TRIP_STOPS table definition and Value column equivalents for ID columns in a PL/SQL record, define TRIP_STOP_PUB_REC_TYPE to pass trip information to the Create_Update_Stop routine.

Table 4–10 TRIP_STOP_PUB_REC_TYPE RECORD DEFINITION

Attribute	Type	Default Value
stop_id	Number	fnd_api.g_miss_num
trip_id	Number	fnd_api.g_miss_num
trip_name	Varchar2(30)	fnd_api.g_miss_char
stop_location_id	Number	fnd_api.g_miss_num
stop_location_code	Varchar2(20)	fnd_api.g_miss_char
planned_arrival_date	Date	fnd_api.g_miss_date
planned_departure_date	Date	fnd_api.g_miss_date
actual_arrival_date	Date	fnd_api.g_miss_date
actual_departure_date	Date	fnd_api.g_miss_date
departure_gross_weight	Number	fnd_api.g_miss_num
departure_net_weight	Number	fnd_api.g_miss_num
weight_uom_code	Varchar2(3)	fnd_api.g_miss_char

Table 4–10 TRIP_STOP_PUB_REC_TYPE RECORD DEFINITION

Attribute	Type	Default Value
weight_uom_desc	Varchar2(25)	fnd_api.g_miss_char
departure_volume	Number	fnd_api.g_miss_num
volume_uom_code	Varchar2(3)	fnd_api.g_miss_char
volume_uom_desc	Varchar2(25)	fnd_api.g_miss_char
departure_seal_code	Varchar2(30)	fnd_api.g_miss_char
departure_fill_percent	Number	fnd_api.g_miss_char
tp_attribute_category	Varchar2(150)	fnd_api.g_miss_char
tp_attribute1	Varchar2(150)	fnd_api.g_miss_char
tp_attribute2	Varchar2(150)	fnd_api.g_miss_char
tp_attribute3	Varchar2(150)	fnd_api.g_miss_char
tp_attribute4	Varchar2(150)	fnd_api.g_miss_char
tp_attribute5	Varchar2(150)	fnd_api.g_miss_char
tp_attribute6	Varchar2(150)	fnd_api.g_miss_char
tp_attribute7	Varchar2(150)	fnd_api.g_miss_char
tp_attribute8	Varchar2(150)	fnd_api.g_miss_char
tp_attribute9	Varchar2(150)	fnd_api.g_miss_char
tp_attribute10	Varchar2(150)	fnd_api.g_miss_char
tp_attribute11	Varchar2(150)	fnd_api.g_miss_char
tp_attribute12	Varchar2(150)	fnd_api.g_miss_char
tp_attribute13	Varchar2(150)	fnd_api.g_miss_char
tp_attribute14	Varchar2(150)	fnd_api.g_miss_char
tp_attribute15	Varchar2(150)	fnd_api.g_miss_char
attribute_category	Varchar2(150)	fnd_api.g_miss_char
attribute1	Varchar2(150)	fnd_api.g_miss_char
attribute2	Varchar2(150)	fnd_api.g_miss_char
attribute3	Varchar2(150)	fnd_api.g_miss_char
attribute4	Varchar2(150)	fnd_api.g_miss_char

Table 4–10 TRIP_STOP_PUB_REC_TYPE RECORD DEFINITION

Attribute	Type	Default Value
attribute5	Varchar2(150)	fnd_api.g_miss_char
attribute6	Varchar2(150)	fnd_api.g_miss_char
attribute7	Varchar2(150)	fnd_api.g_miss_char
attribute8	Varchar2(150)	fnd_api.g_miss_char
attribute9	Varchar2(150)	fnd_api.g_miss_char
attribute10	Varchar2(150)	fnd_api.g_miss_char
attribute11	Varchar2(150)	fnd_api.g_miss_char
attribute12	Varchar2(150)	fnd_api.g_miss_char
attribute13	Varchar2(150)	fnd_api.g_miss_char
attribute14	Varchar2(150)	fnd_api.g_miss_char
attribute15	Varchar2(150)	fnd_api.g_miss_char
creation_date	Date	fnd_api.g_miss_date
created_by	Number	fnd_api.g_miss_num
last_update_date	Date	fnd_api.g_miss_date
last_updated_by	Number	fnd_api.g_miss_num
last_update_login	Date	fnd_api.g_miss_date
program_application_id	Number	fnd_api.g_miss_num
program_id	Number	fnd_api.g_miss_num
program_update_date	Date	fnd_api.g_miss_date
request_id	Number	fnd_api.g_miss_num

Record Parameter Attribute Validations

stop_id

Should be a valid unique element of wsh_trip_stops.stop_id.

trip_id

Should be a valid unique element of wsh_trips.trip_id.

trip_name

Should be a valid unique element of wsh_trips.name

stop_location_id

Should be a valid element of hr_locations.location_id

stop_location_code

Should be a valid element of hr_locations.location_code.

weight_uom_code

Should be a valid element of mtl_units_of_measure.uom_code for a weight uom class.

weight_uom_desc

Should be a valid element of mtl_units_of_measure.unit_of_measure for a weight uom class.

volume_uom_code

Should be a valid element of mtl_units_of_measure.uom_code for a volume uom class.

volume_uom_desc

Should be a valid element of mtl_units_of_measure.unit_of_measure for a volume uom class.

Error Handling

Refer to parameters p_init_msg_list, x_msg_count, and x_msg_data on retrieving error messages.

Stop_Action API Features

The Stop_Action API has the following features:

The Trip_Action procedure enables you to carry out actions on a stop. It accepts as IN parameters the trip identifiers, an action code and any additional parameters needed for specific actions, and returns a completion status.

Functional Overview

This API validates the stop identifiers and performs the following actions specified in p_action_code by calling the corresponding private procedures.

- PLAN/UNPLAN: Is called to plan or unplan stops.
- ARRIVE/CLOSE: Is called to update departure or arrival dates and to close a stop. This action needs an additional parameter P_ACTUAL_DATE for actual departure or arrival updates.
- PICK-RELEASE: Launches pick release to release lines related to a stop.
- DELETE: Is called to delete stops.

Procedure Parameter Descriptions

WSH_TRIP_STOP_PUB.STOP_ACTION

The following chart describes all parameters used by the public procedure WSH_TRIP_STOP_PUB.STOP_ACTION. All of the inbound and outbound parameters are listed. Additional information on these parameters follows.

Table 4–11 WSH_TRIP_STOP_PUB.STOP_ACTION Parameters

Parameter	Usage	Type	Required
p_api_version_number	IN	Number	x
p_init_msg_list	IN	Varchar2	-
x_return_status	OUT	Varchar2	-
x_msg_count	OUT	Number	-
x_msg_data	OUT	Varchar2	-
p_action_code	IN	Varchar2	x
p_stop_id	in	Number	-

Table 4–11 WSH_TRIP_STOP_PUB.STOP_ACTION Parameters

Parameter	Usage	Type	Required
p_trip_id	IN	Number	-
p_trip_name	IN	Varchar2	-
p_stop_location_id	IN	Varchar2	-
p_stop_location_code	IN	Varchar2	-
p_planned_dep_date	IN	Date	-
p_actual_date	IN	Date	-
p_defer_interface_flag	IN	Varchar2	-

p_api_version_number

Compares the incoming API call's version number with the current version number. An error is returned if the version numbers are incompatible.

p_init_msg_list

Requests that the API initialize the message list on your behalf. If the x_msg_count is greater than 1, then the list of messages must be retrieved using the call FND_MSG_PUB.GET. The values are:

- p_msg_index => I
- p_encoded => F
- p_data => 1_message
- p_msg_index_out => 1_msg_index_out

where 1_message and 1_msg_index_out are local variables of types Varchar2(2000) and Number respectively.

Default Value: FND_API.G_FALSE

x_return_status

Requests that the API return the status of the data for you after it completes its function. Valid values include:

- Success: FND_API.G_RET_STS_SUCCESS
- Error: FND_API.G_RET_STS_ERROR
- Unexpected Error: FND_API.G_RET_STS_UNEXP_ERROR

x_msg_count

Indicates number of error messages API has encountered.

x_msg_data

Returns error message text. If the x_msg_count is equal to 1, then this contains the actual message.

p_action_code

Specifies which of the actions of PLAN, UNPLAN, ARRIVE, CLOSE, PICK RELEASE, DELETE the API should perform.

p_stop_id

Stop ID of trip stop on which actions need to be performed.

p_stop_id

Stop ID of trip stop on which actions need to be performed.

p_trip_id

Trip ID of trip stop on which actions need to be performed.

p_trip_name

Trip name of trip stop on which actions need to be performed.

p_stop_location_id

Stop location ID of trip stop on which actions need to be performed.

p_stop_location_code

Stop location code of trip stop on which actions need to be performed.

p_planned_dep_date

Planned departure date of trip stop on which actions need to be performed.

p_actual_date

Actual date of arrival/departure of trip stop.

p_defer_interface_flag

Flag to submit/defer concurrent requests to INV and OE interfaces.

Deliveries Public Application Program Interface

The Deliveries Public Application Program Interface (API) is a public API that consists of the following procedures in **WSH_DELIVERIES_PUB**:

- **Create_Update_Delivery:** Enables you to create a new trip stop record and update an existing trip stop record.
- **Delivery_Action:** Enables you to perform certain actions on a trip stop.
- **Generate_Documents:** Generates the document set for a delivery.

This section describes how to use the Deliveries Public API and how it functions in Oracle Shipping Execution.

Create_Update_Delivery API Features

The Create_Update_Delivery API has the following features.

The Create_Update_Delivery procedure enables you to create a new delivery record or update an existing delivery record in WSH_NEW_DELIVERIES table. The DELIVERY_ID, NAME and return status of a new delivery are passed as OUT parameters, while the DELIVERY_ID or NAME of an existing delivery for update is passed as an IN parameter.

Functional Overview

This API creates a new delivery record in WSH_NEW_DELIVERIES as specified by IN parameter p_action_code value CREATE. Inserts delivery information into WSH_NEW_DELIVERIES and returns the DELIVERY_ID of the new delivery. It also updates an existing delivery record in WSH_NEW_DELIVERIES as specified by IN parameter p_action_code value UPDATE. Delivery record being updated is identified by Delivery Name. An additional IN parameter is provided to identify delivery by Name for update.

The API validates delivery information such as Name, Customer Number, and Ship Method before performing the actions CREATE or UPDATE. It also checks that the insert or update statements were successful, and if not, returns an error.

Procedure Parameter Descriptions

WSH_DELIVERIES_PUB.CREATE_UPDATE_DELIVERY

The following chart describes all parameters used by the public procedure WSH_DELIVERIES_PUB.CREATE_UPDATE_DELIVERY. All of the inbound and outbound parameters are listed. Additional information on these parameters follows.

Table 4–12 WSH_DELIVERIES_PUB.CREATE_UPDATE_DELIVERY Parameters

Parameter	Usage	Type	Required
p_api_version_number	IN	Number	x
p_init_msg_list	IN	Varchar2	-
x_return_status	OUT	Varchar2	-
x_msg_count	OUT	Number	-
x_msg_data	OUT	Varchar2	-
p_action_code	IN	Varchar2	x
p_delivery_info	IN OUT	Record	x
p_delivery_name	IN	Varchar2	-
x_delivery_id	OUT	Number	-
x_name	OUT	Varchar2	-

p_api_version_number

Compares the incoming API call's version number with the current version number. An error is returned if the version numbers are incompatible.

p_init_msg_list

Requests that the API initialize the message list on your behalf. If the x_msg_count is greater than 1, then the list of messages must be retrieved using the call FND_MSG_PUB.GET. The values are:

- p_msg_index => I
- p_encoded => F
- p_data => 1_message
- p_msg_index_out => 1_msg_index_out

where 1_message and 1_msg_index_out are local variables of types Varchar2(2000) and Number respectively.

Default Value: FND_API.G_FALSE

x_return_status

Requests that the API return the status of the data for you after it completes its function. Valid values include:

- Success: FND_API.G_RET_STS_SUCCESS
- Error: FND_API.G_RET_STS_ERROR
- Unexpected Error: FND_API.G_RET_STS_UNEXP_ERROR

x_msg_count

Indicates number of error messages API has encountered.

x_msg_data

Displays error message text. If the x_msg_count is equal to 1, then this contains the actual message.

p_action_code

Specifies whether API should create a new delivery or update existing delivery information depending on its values CREATE and UPDATE.

p_delivery_info

Attributes of the delivery entity of type Delivery_Pub_Rec_Type. These attributes are inserted/updated in WSH_NEW_DELIVERIES. Definition of Delivery_Pub_Rec_Type follows.

p_delivery_name

IN parameter to specify name of delivery record being updated.

Default Value: FND_API.G_MISS_CHAR

x_delivery_id

OUT parameter to specify delivery ID of new delivery record being created.

x_name

OUT parameter to specify name of new delivery record being created.

Record Parameter Descriptions

DELIVERY_PUB_REC_TYPE RECORD DEFINITION

To encapsulate WSH_NEW_DELIVERIES table definition and Value column equivalents for ID columns in a PL/SQL record, define DELIVERIES_PUB_REC_TYPE to pass delivery information to the Create_Update_Delivery routine.

Table 4–13 DELIVERY_PUB_REC_TYPE RECORD DEFINITION

Attribute	Type	Default
delivery_id	Number	fnd_api.g_miss_num
name	Varchar2(30)	fnd_api.g_miss_char
delivery_type	Varchar2(30)	fnd_api.g_miss_char
loading_sequence	Number	fnd_api.g_miss_num
loading_order_flag	Varchar2(2)	fnd_api.g_miss_char
loading_order_desc	Varchar2(20)	fnd_api.g_miss_char
initial_pickup_date	Date	fnd_api.g_miss_date
initial_pickup_location_id	Number	fnd_api.g_miss_num
initial_pickup_location_code	Varchar2(20)	fnd_api.g_miss_char
organization_id	Number	fnd_api.g_miss_num
organization_code	Varchar2(3)	fnd_api.g_miss_char
ultimate_dropoff_location_id	Number	fnd_api.g_miss_num
ultimate_dropoff_location_code	Varchar2(20)	fnd_api.g_miss_char
ultimate_dropoff_date	Date	fnd_api.g_miss_date
customer_id	Number	fnd_api.g_miss_num
customer_number	Varchar2(30)	fnd_api.g_miss_char
intmed_ship_to_location_id	Number	fnd_api.g_miss_num
intmed_ship_to_location_code	Varchar2(20)	fnd_api.g_miss_char
pooled_ship_to_location_id	Number	fnd_api.g_miss_num
pooled_ship_to_location_code	Varchar2(20)	fnd_api.g_miss_char
carrier_id	Number	fnd_api.g_miss_num
carrier_code	Varchar2(25)	fnd_api.g_miss_char

Table 4–13 DELIVERY_PUB_REC_TYPE RECORD DEFINITION

Attribute	Type	Default
ship_method_code	Varchar2(30)	fnd_api.g_miss_char
ship_method_name	Varchar2(80)	fnd_api.g_miss_char
freight_terms_code	Varchar2(30)	fnd_api.g_miss_char
freight_terms_name	Varchar2(80)	fnd_api.g_miss_char
fob_code	Varchar2(30)	fnd_api.g_miss_char
fob_name	Varchar2(80)	fnd_api.g_miss_char
fob_location_id	Number	fnd_api.g_miss_num
fob_location_code	Varchar2(20)	fnd_api.g_miss_char
waybill	Varchar2(30)	fnd_api.g_miss_char
dock_code	Varchar2(30)	fnd_api.g_miss_char
acceptance_flag	Varchar2(1)	fnd_api.g_miss_char
accepted_by	Varchar2(150)	fnd_api.g_miss_char
accepted_date	Date	fnd_api.g_miss_date
acknowledged_by	Varchar2(150)	fnd_api.g_miss_char
confirmed_by	Varchar2(150)	fnd_api.g_miss_char
confirm_date	Date	fnd_api.g_miss_date
asn_date_sent	Date	fnd_api.g_miss_date
asn_status_code	Varchar2(15)	fnd_api.g_miss_char
asn_seq_number	Number	fnd_api.g_miss_num
gross_weight	Number	fnd_api.g_miss_num
net_weight	Number	fnd_api.g_miss_num
weight_uom_code	Varchar2(3)	fnd_api.g_miss_char
weight_uom_desc	Varchar2(25)	fnd_api.g_miss_char
volume	Number	fnd_api.g_miss_num
volume_uom_code	Varchar2(3)	fnd_api.g_miss_char
volume_uom_desc	Varchar2(25)	fnd_api.g_miss_char
additional_shipment_info	Varchar2(500)	fnd_api.g_miss_char

Table 4–13 DELIVERY_PUB_REC_TYPE RECORD DEFINITION

Attribute	Type	Default
currency_code	Varchar2(15)	fnd_api.g_miss_char
currency_name	Varchar2(80)	fnd_api.g_miss_char
attribute_category	Varchar2(150)	fnd_api.g_miss_char
attribute1	Varchar2(150)	fnd_api.g_miss_char
attribute2	Varchar2(150)	fnd_api.g_miss_char
attribute3	Varchar2(150)	fnd_api.g_miss_char
attribute4	Varchar2(150)	fnd_api.g_miss_char
attribute5	Varchar2(150)	fnd_api.g_miss_char
attribute6	Varchar2(150)	fnd_api.g_miss_char
attribute7	Varchar2(150)	fnd_api.g_miss_char
attribute8	Varchar2(150)	fnd_api.g_miss_char
attribute9	Varchar2(150)	fnd_api.g_miss_char
attribute10	Varchar2(150)	fnd_api.g_miss_char
attribute11	Varchar2(150)	fnd_api.g_miss_char
attribute12	Varchar2(150)	fnd_api.g_miss_char
attribute13	Varchar2(150)	fnd_api.g_miss_char
attribute14	Varchar2(150)	fnd_api.g_miss_char
attribute15	Varchar2(150)	fnd_api.g_miss_char
tp_attribute_category	Varchar2(150)	fnd_api.g_miss_char
tp_attribute1	Varchar2(150)	fnd_api.g_miss_char
tp_attribute2	Varchar2(150)	fnd_api.g_miss_char
tp_attribute3	Varchar2(150)	fnd_api.g_miss_char
tp_attribute4	Varchar2(150)	fnd_api.g_miss_char
tp_attribute5	Varchar2(150)	fnd_api.g_miss_char
tp_attribute6	Varchar2(150)	fnd_api.g_miss_char
tp_attribute7	Varchar2(150)	fnd_api.g_miss_char
tp_attribute8	Varchar2(150)	fnd_api.g_miss_char

Table 4–13 DELIVERY_PUB_REC_TYPE RECORD DEFINITION

Attribute	Type	Default
tp_attribute9	Varchar2(150)	fnd_api.g_miss_char
tp_attribute10	Varchar2(150)	fnd_api.g_miss_char
tp_attribute11	Varchar2(150)	fnd_api.g_miss_char
tp_attribute12	Varchar2(150)	fnd_api.g_miss_char
tp_attribute13	Varchar2(150)	fnd_api.g_miss_char
tp_attribute14	Varchar2(150)	fnd_api.g_miss_char
tp_attribute15	Varchar2(150)	fnd_api.g_miss_char
global_attribute_category	Varchar2(30)	fnd_api.g_miss_char
global_attribute1	Varchar2(150)	fnd_api.g_miss_char
global_attribute2	Varchar2(150)	fnd_api.g_miss_char
global_attribute3	Varchar2(150)	fnd_api.g_miss_char
global_attribute4	Varchar2(150)	fnd_api.g_miss_char
global_attribute5	Varchar2(150)	fnd_api.g_miss_char
global_attribute6	Varchar2(150)	fnd_api.g_miss_char
global_attribute7	Varchar2(150)	fnd_api.g_miss_char
global_attribute8	Varchar2(150)	fnd_api.g_miss_char
global_attribute9	Varchar2(150)	fnd_api.g_miss_char
global_attribute10	Varchar2(150)	fnd_api.g_miss_char
global_attribute11	Varchar2(150)	fnd_api.g_miss_char
global_attribute12	Varchar2(150)	fnd_api.g_miss_char
global_attribute13	Varchar2(150)	fnd_api.g_miss_char
global_attribute14	Varchar2(150)	fnd_api.g_miss_char
global_attribute15	Varchar2(150)	fnd_api.g_miss_char
global_attribute16	Varchar2(150)	fnd_api.g_miss_char
global_attribute17	Varchar2(150)	fnd_api.g_miss_char
global_attribute18	Varchar2(150)	fnd_api.g_miss_char
global_attribute19	Varchar2(150)	fnd_api.g_miss_char

Table 4–13 DELIVERY_PUB_REC_TYPE RECORD DEFINITION

Attribute	Type	Default
global_attribute20	Varchar2(150)	fnd_api.g_miss_char
creation_date	Date	fnd_api.g_miss_date
created_by	Number	fnd_api.g_miss_num
last_update_date	Date	fnd_api.g_miss_date
last_updated_by	Number	fnd_api.g_miss_num
last_update_login	Number	fnd_api.g_miss_num
program_application_id	Number	fnd_api.g_miss_num
program_id	Number	fnd_api.g_miss_num
program_update_date	Date	fnd_api.g_miss_date
request_id	Number	fnd_api.g_miss_num

Record Parameter Attribute Validations

delivery_id

Should be a valid unique element of wsh_new_deliveries.delivery_id.

delivery_name

Should be a valid unique element of wsh_new_deliveries.name.

delivery_type

Should be either STANDARD.

loading_order_flag

Should be a valid element of fnd_lookup_values_vl.lookup_code for lookup_type LOADING_ORDER.

loading_order_desc

Should be a valid element of fnd_lookup_values_vl.meaning for lookup_type LOADING_ORDER.

initial_pickup_location_id

Should be a valid element of hr_locations.location_id.

initial_pickup_location_code

Should be a valid element of hr_locations.location_code.

organization_id

Should be a valid element of org_organization_definitions.organization_id.

organization_code

Should be a valid element of org_organization_definitions.organization_code.

ultimate_dropoff_location_id

Should be a valid element of hr_locations.location_id.

ultimate_dropoff_location_code

Should be a valid element of hr_locations.location_code.

customer_id

Should be a valid element of ra_customers.customer_id.

customer_code

Should be a valid element of ra_customers.customer_code.

intmed_ship_to_location_id

Should be a valid element of hr_locations.location_id.

intmed_ship_to_location_code

Should be a valid element of hr_locations.location_code.

pooled_ship_to_location_id

Should be a valid element of hr_locations.location_id.

pooled_ship_to_location_code

Should be a valid element of hr_locations.location_code.

ship_method_code

Should be a valid element of fnd_lookup_values_vl.lookup_code for lookup_type SHIP_METHOD.

ship_method_name

Should be a valid element of `fnd_lookup_values_vl.meaning` for `lookup_type` SHIP_METHOD.

freight_terms_code

Should be a valid element of `fnd_lookup_values_vl.lookup_code` for `lookup_type` FREIGHT_TERMS.

freight_terms_name

Should be a valid element of `fnd_lookup_values_vl.meaning` for `lookup_type` FREIGHT_TERMS.

fob_code

Should be a valid element of `fnd_lookup_values_vl.lookup_code` for `lookup_type` FOB.

fob_terms_name

Should be a valid element of `fnd_lookup_values_vl.meaning` for `lookup_type` FOB.

fob_location_id

Should be a valid element of `hr_locations.location_id`.

fob_location_code

Should be a valid element of `hr_locations.location_code`.

weight_uom_code

Should be a valid element of `mtl_units_of_measure.uom_code` for a weight uom class

weight_uom_desc

Should be a valid element of `mtl_units_of_measure.unit_of_measure` for a weight uom class.

volume_uom_code

Should be a valid element of `mtl_units_of_measure.uom_code` for a volume uom class.

volume_uom_desc

Should be a valid element of mtl_units_of_measure.unit_of_measure for a volume uom class.

currency_code

Should be a valid element of fnd_currencies_vl.currency_code.

currency_name

Should be a valid element of fnd_currencies_vl.currency_name.

Error Handling

Refer to parameters p_init_msg_list, x_msg_count, and x_msg_data on retrieving error messages.

Delivery_Action API Features

The Delivery_Action API has the following features:

The Delivery_Action procedure enables you to carry out actions on a delivery. It accepts as IN parameters the delivery identifiers, an action code and any additional parameters needed for specific actions, and returns a completion status.

Note: Before you call this public API, initialize your environment (using `fn_d_global.apps_initialize`) in the script or package that you use to call it.

Functional Overview

This API validates the delivery identifiers and performs the following actions specified in `p_action_code` by calling the corresponding private procedures:

- **PLAN/UNPLAN:** Is called to plan or unplan deliveries.
- **WT-VOL:** Is called to calculate weight and volume for the delivery. This action needs an additional parameter `P_WV_OVERRIDE_FLAG`. Regardless if this is set to Y or N, weight and volume re-calculations are done for all deliveries on the trip.
- **CONFIRM:** Is called to confirm delivery. Requires additional parameters starting with `P_SC`
- **RE-OPEN, CLOSE:** Is called to update status of delivery. This action needs an additional parameter `P_ACTUAL_DATE` for actual departure or arrival updates.
- **ASSIGN-TRIP:** Is called to assign a trip to delivery. It requires additional parameters starting with `P_ASG`.
- **UNASSIGN-TRIP:** Is called to unassign trip from delivery.
- **AUTOCREATE-TRIP:** Is called to autcreate trip for the delivery.
- **PICK-RELEASE:** Launches pick release to release lines related to a delivery.
- **DELETE:** Is called to delete a delivery.

Procedure Parameter Descriptions

WSH_DELIVERIES_PUB.DELIVERY_ACTION

The following chart describes all parameters used by the public procedure WSH_DELIVERIES_PUB.DELIVERY_ACTION. All of the inbound and outbound parameters are listed. Additional information on these parameters follows.

Table 4–14 WSH_DELIVERIES_PUB.DELIVERY_ACTION Parameters

Parameter	Usage	Type	Required
p_api_version_number	IN	Number	x
p_init_msg_list	IN	Varchar2	-
x_return_status	OUT	Varchar2	-
x_msg_count	OUT	Number	-
x_msg_data	OUT	Varchar2	-
p_action_code	IN	Varchar2	x
p_delivery_id	IN	Number	-
p_delivery_name	IN	Varchar2	-
p_asg_trip_id	IN	Number	x
p_asg_trip_name	IN	Varchar2	-
p_asg_pickup_stop_id	IN	Number	-
p_asg_pickup_loc_id	IN	Number	-
p_asg_pickup_loc_code	IN	Varchar2	-
p_asg_pickup_arr_date	IN	Date	-
p_asg_pickup_dep_date	IN	Number	-
p_asg_dropoff_stop_id	IN	Number	-
p_asg_dropoff_loc_id	IN	Number	-
p_asg_dropoff_loc_code	IN	Varchar2	-
p_asg_dropoff_arr_date	IN	Date	-
p_asg_dropoff_dep_date	IN	Date	-
p_sc_action_flag	IN	Varchar2	-
p_sc_close_trip_flag	IN	Varchar2	-
p_sc_create_bol_flag	IN	Varchar2	-
p_sc_stage_del_flag	IN	Varchar2	-

Table 4–14 WSH_DELIVERIES_PUB.DELIVERY_ACTION Parameters

Parameter	Usage	Type	Required
p_sc_trip_ship_method	IN	Varchar2	-
p_sc_actual_dep_date	IN	Varchar2	-
p_sc_report_set_id	IN	Number	-
p_sc_report_set_name	IN	Varchar2	-
p_sc_rule_id	IN	Number	-
p_sc_rule_name	IN	Varchar2	-
p_sc_defer_interface_flag	IN	Varchar2	-
p_wv_override_flag	IN	Varchar2	-
x_trip_id	OUT	Varchar2	-
x_trip_name	OUT	Varchar2	-
p_actual_date	IN	Date	-

p_api_version_number

Compares the incoming API call's version number with the current version number. An error is returned if the version numbers are incompatible.

p_init_msg_list

Requests that the API initialize the message list on your behalf. If the x_msg_count is greater than 1, then the list of messages must be retrieved using the call FND_MSG_PUB.GET. The values are:

- p_msg_index => I
- p_encoded => F
- p_data => 1_message
- p_msg_index_out => 1_msg_index_out

where 1_message and 1_msg_index_out are local variables of types Varchar2(2000) and Number respectively.

Default Value: FND_API.G_FALSE

x_return_status

Requests that the API return the status of the data for you after it completes its function. Valid values include:

- Success: FND_API.G_RET_STS_SUCCESS
- Error: FND_API.G_RET_STS_ERROR
- Unexpected Error: FND_API.G_RET_STS_UNEXP_ERROR

x_msg_count

Indicates number of error messages API has encountered.

x_msg_data

Returns error message text. If the x_msg_count is equal to 1, then this contains the actual message.

p_action_code

Specifies which of the actions of PLAN, UNPLAN, CONFIRM, RE-OPEN, CLOSE, ASSIGN-TRIP, UNASSIGN-TRIP, AUTOCREATE-TRIP, WT-VOL, PICK RELEASE, DELETE the API should perform.

p_delivery_name

Name of delivery on which actions need to be performed.

p_delivery_id

Delivery ID on which actions need to be performed.

Default Value: NULL.

p_asg_trip_id

Trip identifier for assignment of trip to delivery. Used when assigning and un-assigning a delivery to a trip.

Default Value: NULL.

p_trip_name

Name of trip for assignment to delivery. Used when assigning a trip.

Default Value: NULL.

p_asg_pickup_stop_id

Stop ID for pickup assignment. Used when assigning a trip.

Default Value: NULL.

p_asg_pickup_loc_id

Stop location for pickup assignment. Used when assigning a trip.

Default Value: NULL.

p_asg_pickup_loc_code

Stop location code for pickup assignment. Used when assigning a trip.

Default Value: NULL.

p_asg_pickup_arr_date

Stop location arrival date for pickup assignment. Used when assigning a trip.

Default Value: NULL.

p_asg_pickup_dep_date

Stop location departure date for pickup assignment. Used when assigning a trip.

Default Value: NULL.

p_asg_dropoff_stop_id

Stop ID for dropoff assignment. Used when assigning a trip.

Default Value: NULL.

p_asg_dropoff_loc_id

Stop location for dropoff assignment. Used when assigning a trip.

Default Value: NULL.

p_asg_dropoff_loc_code

Stop location code for dropoff assignment. Used when assigning a trip.

Default Value: NULL.

p_asg_dropoff_arr_date

Stop location arrival date for dropoff assignment. Used when assigning a trip.

Default Value: NULL.

p_asg_dropoff_dep_date

Stop location departure date for dropoff assignment. Used when assigning a trip.

Default Value: NULL.

p_sc_action_flag

Ship Confirm option - S, B, T, A, C. Used when ship confirming a delivery.

Default Value: S

p_sc_close_trip_flag

Ship Confirm close trip flag. Used when ship confirming a delivery.

Default Value: N

p_sc_create BOL flag

Ship Confirm create Bill of Lading flag. Used when ship confirming a delivery.

Default Value: N.

p_sc_rule_id

Used when action is CONFIRM.

Default Value: Null.

p_sc_rule_name

Used when action is CONFIRM.

Default Value: Null.

p_sc_stage_del_flag

Ship Confirm create delivery for stage quantity flag. Used when ship confirming a delivery.

Default Value: Y.

p_sc_trip_ship_method

Ship Confirm trip ship method. Used when ship confirming a delivery.

Default Value: NULL.

p_defer_interface_flag

Ship Confirm flag to submit/defer concurrent requests to INV and OE interfaces.

Default Value: Y

p_wv_override_flag

Override flag for weight volume calculations. Regardless if this is set to Y or N, weight and volume re-calculations are done for all deliveries on the trip

Default Value: N

x_trip_id

ID of autocreated trip.

x_trip_name

Name of autocreated trip.

Error Handling

Refer to parameters p_init_msg_list, x_msg_count, and x_msg_data on retrieving error messages.

Generate_Documents API Features

The Generate_Documents API generates the document set for a delivery. It takes the report_set_name, organization_code, and a table of delivery_names as the input parameter and passes the return_status, message_data and message_count as the output parameters.

Functional Overview

This API is for backward compatibility. A call to this API generates the document set specified in the IN parameter report_set_name for all the deliveries in the IN parameter delivery_names.

Setting Up the Generate_Documents API

Parameter Descriptions

The following chart describes all parameters used by the public Generate_Documents API. All of the inbound and outbound parameters are listed. Additional information on these parameters follows.

Table 4–15 Generate_Documents Parameters

Parameter	Usage	Type	Required
p_report_set_name	IN	VARCHAR2	-
p_organization_code	IN	VARCHAR2	-
p_delivery_name	IN	WSH_UTIL_ CORE. Column_Tab_ Type	-
x_msg_count	OUT	NUMBER	-
x_msg_data	OUT	VARCHAR2	-
x_return_status	OUT	VARCHAR2	-

p_report_set_name

This is the report_Set that needs to be generated for all the deliveries specified in p_delivery_name parameter.

p_organization_code

This is the organization code for which the document _set needs to be generated.

p_delivery_name

This is the list of delivery_names for which the document set needs to be generated.

x_msg_count

Indicates number of error messages API has encountered.

x_msg_data

Contains error message text. If X_msg_count is 1, then this contains the complete message.

x_return_status

Requests that the API return the status of the data for you after it completes its function. Valid values include the following:

- Success: FND_API.G_RET_STS_SUCCESS
- Error: FND_API.G_RET_STS_ERROR
- Unexpected Error: FND_API.G_RET_STS_UNEXP_ERROR

Exceptions Application Program Interface

Exceptions Application Program Interface (API) is a public API that consists of the following procedure in the package WSH_EXCEPTIONS_PUB.

Exception_Action: Enables you to carry out actions on an exception. It accepts as IN parameters the exception identifiers, an action code and returns a completion status.

Get_Exceptions: Enables you to access detailed information about an exception and stores it in an exception table.

Exception_Action API Features

The **Exception_Action** procedure enables you to carry out actions on an exception. It accepts as IN parameters the exceptions identifiers in a table, an action, and returns a completion status.

Note: Before you call this public API, initialize your environment (using `fnf_global.apps_initialize`) in the script or package that you use to call it.

Functional Overview

This API validates its input parameters `p_action` and performs the following actions depending on its value:

- **PURGE:** It is used to purge existing exceptions
- **LOG:** It is used to log an exception based on the values in the input exceptions record (`p_exception_rec`)
- **CHANGE_STATUS:** Its is used to change the status of an exception

Setting Up the Exception_Actions API

Parameter Descriptions

The following chart describes all parameters used by the public **Exception_Actions** API. All of the inbound and outbound parameters are listed. Additional information on these parameters follows.

Table 4–16 Exception_Actions Parameters

Parameter	Usage	Type	Required	Derived	Optional
p_api_version_number	IN	NUMBER	x	-	-
p_init_msg_list	IN	VARCHAR2	-	-	x
p_commit	IN	VARCHAR2	-	-	x
p_action	IN	VARCHAR2	x	-	-
p_validation_level	IN	NUMBER	-	-	x
p_exceptions_rec	IN OUT	XC_ACTION_REC_TYPE	x	-	-
x_msg_count	OUT	NUMBER	-	x	-
x_msg_data	OUT	VARCHAR2	-	x	-
x_return_status	OUT	VARCHAR2	-	x	-

p_api_version_number

Used to compare the incoming API call's version number with the current version number. An error is returned if the version numbers are incompatible.

p_init_msg_list

Requests that the API initialize the message list on your behalf. If the x_msg_count is greater than 1, then the list of messages must be retrieved using the call FND_MSG_PUB.GET. The values are:

- p_msg_index => I
- p_encoded => F
- p_data => 1_message
- p_msg_index_out => 1_msg_index_out
where 1_message and 1_msg_index_out are local variables of types Varchar2(2000 and Number respectively.
- Default Value: FND_API.G_FALSE

p_commit

Requests that the API update information for you after it completes its function.
Default Value: FND_API.G_FALSE

x_return_status

Requests that the API return the status of the data for you after it completes its function. Valid values include:

- Success: FND_API.G_RET_STS_SUCCESS
- Error: FND_API.G_RET_STS_ERROR
- Unexpected Error: FND_API.G_RET_STS_UNEXP_ERROR

x_msg_count

Indicates number of error messages API has encountered.

x_msg_data

Displays error message text. If the x_msg_count is equal to 1, then this contains the actual message.

p_validation_level

This parameter is not currently used.

p_action

This parameter specifies the actions that needs to be performed on the exception. Valid values for this parameter are 'LOG', 'PURGE' and 'CHANGE_STATUS'.

p_exception_rec

This record structure has all the exception specific entries.

The table below describes all parameters that are used by the XC_ACTION_REC_TYPE TYPE record. Additional information on these parameters follows.

XC_ACTION_REC_TYPE

Table 4-17 XC_ACTION_REC_TYPE Parameters

Parameter	Type	Default Value
request_id	NUMBER	
batch_id	NUMBER	
exception_id	NUMBER	
exception_name	VARCHAR2	

Table 4–17 XC_ACTION_REC_TYPE Parameters

Parameter	Type	Default Value
logging_entity	VARCHAR2	
logging_entity_id	NUMBER	
manually_logged	VARCHAR2	
message	VARCHAR2	
logged_at_location_code	VARCHAR2	
exception_location_code	VARCHAR2	
severity	VARCHAR2	
delivery_name	VARCHAR2	
trip_name	VARCHAR2	
stop_location_id	NUMBER	
delivery_detail_id	NUMBER	
container_name	VARCHAR2	
ord_id	NUMBER	
inventory_item_id	NUMBER	
lot_number	VARCHAR2	
sublot_number	VARCHAR2	
revision	VARCHAR2	
serial_number	VARCHAR2	
unit_of_measure	VARCHAR2	
quantity	NUMBER	
unit_of_measure2	VARCHAR2	
quantity2	NUMBER	
subinventory	VARCHAR2	
locator_id	NUMBER	
error_message	VARCHAR2	
attribute_category	VARCHAR2	
attribute1	VARCHAR2	

Table 4-17 XC_ACTION_REC_TYPE Parameters

Parameter	Type	Default Value
attribute2	VARCHAR2	
attribute3	VARCHAR2	
attribute4	VARCHAR2	
attribute5	VARCHAR2	
attribute6	VARCHAR2	
attribute7	VARCHAR2	
attribute8	VARCHAR2	
attribute9	VARCHAR2	
attribute10	VARCHAR2	
attribute11	VARCHAR2	
attribute12	VARCHAR2	
attribute13	VARCHAR2	
attribute14	VARCHAR2	
attribute15	VARCHAR2	
departure_date	DATE	
arrival_date	DATE	
exception_type	VARCHAR2	
status	VARCHAR2	
departure_date_to	DATE	
arrival_date_to	DATE	
creation_date	DATE	
creation_date_to	DATE	
data_older_no_of_days	NUMBER	
new_status	VARCHAR2	
caller	VARCHAR2	
phase	NUMBER	

The following fields are used for the purge action:

- `exception_type`: Specifies which type of exception needs to be purged.
- `status`: Specifies the status of exceptions that should be selected to be purged
- `departure_date_to`: Specifies that exceptions with `departure_dates` before this date should be purged
- `arrival_date_to`: Specifies that exceptions with `arrival_dates` before this date should be purged
- `creation_date`: Specifies that exceptions with `creation_date` same as this date should be purged
- `creation_date_to`: Specifies that exceptions with `creation_dates` before this date should be purged
- `data_older_no_of_days`: Specifies the maximum age of exceptions that should be purged
- `departure_date`: Specifies the `departure_date` of exceptions that should be purged
- `arrival_date`: Specifies the arrival date of exceptions that should be purged
- `logged_at_location_code`: Specifies the `location_code` where the exceptions have been logged that should be purged
- `exception_location_code`: Specifies the `location_code` of the exceptions that should be purged
- `severity`: Specifies the severity of the exceptions that should be purged
- `delivery_name`: Specifies the delivery name of the exception that should be purged
- `exception_name`: Specifies the name of the exception that should be purged
- `logging_entity`: Specifies the entity whose exceptions need to be purged
- `request_id`: Specifies the `request_id` whose exceptions needs to be purged

The following fields are used for the `change_status` action:

- `exception_name`: Specifies the `exception_name` whose status needs to be changed
- `logging_entity`: Specifies the logging entity whose exceptions need to change status

- `logging_entity_id`: Specifies the logging entity id whose exceptions need to change status
- `new_status`: Specifies the new status of the exceptions that satisfy the fields above

The following fields are used for Logging exceptions. The value of these fields are directly assigned to the corresponding columns in the table `WSH_EXCEPTION` when the new exception record is created.

- `request_id`
- `batch_id`
- `exception_id`
- `exception_name`
- `logging_entity`
- `logging_entity_id`
- `manually_logged`
- `message`
- `logged_at_location_code`
- `exception_location_code`
- `severity`
- `delivery_name`
- `trip_name`
- `stop_location_id`
- `delivery_detail_id`
- `container_name`
- `org_id`
- `inventory_item_id`
- `lot_number`
- `sublot_number`
- `revision`
- `serial_number`

- unit_of_measure
- quantity
- unit_of_measure2
- quantity2
- subinventory
- locator_id
- error_message
- attribute_category
- attribute1
- attribute2
- attribute3
- attribute4
- attribute5
- attribute6
- attribute7
- attribute8
- attribute9
- attribute10
- attribute11
- attribute12
- attribute13
- attribute14
- attribute15
- departure_date
- arrival_date

Get_Exceptions API Features

The Get_Exceptions API gets the exceptions based on the entity_name and the entity_id passed to it. The fetched details are stored in the table x_exceptions_tab which is the output parameter of the API.

Functional Overview

The Get_Exceptions API takes in p_logging_entity_id and p_logging_entity_name as IN parameters and passes out the exceptions that have been logged for this entity. The valid values of p_logging_entity_name are 'DELIVERY', 'TRIP', 'STOP', 'CONTAINER' and 'DETAIL'. Thus, if p_logging_entity is 'DELIVERY', then p_logging_entity_id should be a valid delivery_id.

Setting Up the Get_Exceptions API

Parameter Descriptions

The following chart describes all parameters used by the public Get_Exceptions API. All of the inbound and outbound parameters are listed. Additional information on these parameters follows.

Table 4–18 Get_Exceptions Parameters

Parameter	Usage	Type	Required	Derived	Optional
p_api_version_number	IN	NUMBER	x	-	-
p_init_msg_list	IN	VARCHAR2	-	-	x
x_return_status	OUT	VARCHAR2	-	-	x
x_msg_count	OUT	NUMBER	-	-	x
x_msg_data	OUT	VARCHAR2	-	x	-
p_logging_entity_id	IN	NUMBER	x	-	-
p_logging_entity	IN	VARCHAR2	x	-	-
x_excptions_tab	OUT	XC_TAB_TYPE	x	-	-

p_api_version_number

Used to compare the incoming API call's version number with the current version number. An error is returned if the version numbers are incompatible.

p_init_msg_list

Requests that the API initialize the message list on your behalf. If the x_msg_count is greater than 1, then the list of messages must be retrieved using the call FND_MSG_PUB.GET. The values are:

- p_msg_index => I
- p_encoded => F
- p_data => 1_message
- p_msg_index_out => 1_msg_index_out

where 1_message and 1_msg_index_out are local variables of types Varchar2(2000) and Number respectively.

Default Value: FND_API.G_FALSE

x_return_status

Requests that the API return the status of the data for you after it completes its function. Valid values include:

- Success: FND_API.G_RET_STS_SUCCESS
- Error: FND_API.G_RET_STS_ERROR
- Unexpected Error: FND_API.G_RET_STS_UNEXP_ERROR

x_msg_count

Indicates number of error messages API has encountered.

x_msg_data

Displays error message text. If the x_msg_count is equal to 1, then this contains the actual message.

p_logging_entity_name

Specifies the entity for which we are querying the exception. Valid values are 'DELIVERY', 'TRIP', 'STOP', 'CONTAINER' and 'DETAIL'.

P_logging_entity_id

This is the entity Id for which the exceptions details are being fetched.

x_exceptions_tab

This is the out parameter where the details of the exception are stored.

The x_exceptions_tab fields are as follows:

Table 4–19 x_exceptions_tab Fields

Parameter	Type	Default Value
exception_id	NUMBER	-
exception_name	VARCHAR2	-
status	VARCHAR2	-

Delivery Details Public Application Program Interface

The Delivery Details Public Application Program Interface (API) is a public API that consists of the following procedures in package **WSH_DELIVERY_DETAILS_PUB**:

Detail_To_Delivery: Enables you to assign/unassign a delivery detail to/from a delivery

Split_Line: Splits a delivery detail into two delivery details with the requested quantity summed up to the original delivery detail.

Update_Shipping_Attributes: Updates a delivery detail with new information, and the **Autocreate_Deliveries** and **Autocreate_Del_Trip** procedures.

Note: The creation of a delivery detail is not provided in the open API since it should be part of the order entry process.

This section describes how to use the Delivery Details Public API and how it functions in Oracle Shipping Execution.

Detail_To_Delivery API Features

The Detail_To_Delivery API has the following features.

The Detail_To_Delivery procedure performs two actions: assign Delivery Details to a Delivery, or unassign Delivery Details from a Delivery.

Functional Overview

This procedure assigns Delivery Details to a Delivery as specified by IN parameter p_action value ASSIGN. If the action is ASSIGN, either the Delivery ID or the Delivery Name need to be specified. It also unassigns Delivery Details from a Delivery based on the IN parameter p_action value UNASSIGN. If the action is UNASSIGN, there is no need to pass a Delivery ID or a Delivery Name as the procedure will just unassign all the Delivery Details from the Deliveries.

For ASSIGN, the procedure checks the Delivery Detail status, Delivery status, group by attributes defined in Shipping Parameters, Ship To Locations and Ship From Locations. If one of the Delivery Details is assigned to a delivery, the procedure continues to the next Delivery Detail, performs ASSIGN action to the Delivery Details which are not already assigned, and returns an error in the return status. The calling procedure decides whether to commit or rollback. ASSIGN is allowed only for the Deliveries in the status of OPEN or PACKED and not planned.

Security rules check if ship-to and ship-from location information and other optional group by attributes are common between delivery details and delivery.

For UNASSIGN, if one of the Delivery Details is unassigned already, return status posts a warning, and the action continues to the next Delivery Detail. To allow UNASSIGN action, the Delivery status must be OPEN and not planned. If there is freight cost for the Delivery Detail, a warning is posted in the return status.

Procedure Parameter Descriptions

WSH_DELIVERY_DETAILS_PUB.DETAIL_TO_DELIVERY

The following chart describes all parameters used by the public procedure WSH_DELIVERY_DETAILS_PUB.DETAIL_TO_DELIVERY. All of the inbound and outbound parameters are listed. Additional information on these parameters follows.

Table 4–20 WSH_DELIVERY_DETAILS_PUB.DETAIL_TO_DELIVERY Parameters

Parameter	Usage	Type	Required
p_api_version_number	IN	Number	x
p_init_msg_list	IN	Varchar2	-
x_return_status	OUT	Varchar2	-
x_msg_count	OUT	Number	-
x_msg_data	OUT	Varchar2	-
p_TabOfDelDets	IN	Table	x
p_action	IN	Varchar2	x
p_delivery_id	OUT	Number	-
p_delivery_name	OUT	Varchar2	-

p_api_version_number

Compares the incoming API call's version number with the current version number. An error is returned if the version numbers are incompatible.

p_init_msg_list

Requests that the API initialize the message list on your behalf. If the x_msg_count is greater than 1, then the list of messages must be retrieved using the call FND_MSG_PUB.GET. The values are:

- p_msg_index => I
- p_encoded => F
- p_data => 1_message
- p_msg_index_out => 1_msg_index_out

where 1_message and 1_msg_index_out are local variables of types Varchar2(2000) and Number respectively.

Default Value: FND_API.G_FALSE

x_return_status

Requests that the API return the status of the data for you after it completes its function. Valid values include:

- Success: FND_API.G_RET_STS_SUCCESS
- Error: FND_API.G_RET_STS_ERROR
- Unexpected Error: FND_API.G_RET_STS_UNEXP_ERROR

x_msg_count

Indicates number of error messages API has encountered.

x_msg_data

Returns error message text. If the x_msg_count is equal to 1, then this contains the actual message.

p_TabOfDelDets

Table of Delivery Detail ID's of type WSH_UTIL_CORE.ID_TAB_TYPE which is a table of type Number indexed by binary integers.

p_action

Valid values for the Action are ASSIGN and UNASSIGN.

If the action is ASSIGN, the user needs to pass either the Delivery ID or the Delivery Name to identify which Delivery the Delivery Details are assigned to. If both the Delivery ID and Delivery Name are passed in, only the Delivery ID will be used to identify the Delivery, the Delivery Name will be ignored.

p_delivery_id

The Delivery ID to identify the Delivery, not needed if the action is UNASSIGN.

p_delivery_name

The Delivery Name to identify the Delivery, not needed if the action is UNASSIGN

Error Handling

Refer to parameters p_init_msg_list, x_msg_count, and x_msg_data on retrieving error messages.

Split_Line API Features

The Split_Line API has the following features:

The Split_Line procedure enables you to split a Delivery Detail into two Delivery Details. A Delivery Detail needs to be split when ship partially, or when a backorder occurs.

Functional Overview

This API splits a Delivery Detail into two Delivery Details. Most of the fields in the new Delivery Detail have the same values except that the requested quantity is the split quantity, the shipped quantity is NULL, the backorder quantity is NULL and that the split_from_detail_id points to the original Delivery Detail. After the split, the calling procedure needs to make another call to update the shipped quantity and backordered quantity to complete the whole transaction.

This procedure checks that the Delivery Detail is valid, the split quantity is not greater than the requested quantity before calling WSH_DELIVERY_DETAILS_ACTIONS.Split_Delivery_Details. Most of the validation is performed in WSH_DELIVERY_DETAILS_ACTIONS.Split_Delivery_Details, which checks that the quantity meets the decimal quantity standard.

Procedure Parameter Descriptions

WSH_DELIVERY_DETAILS_PUB.SPLIT_LINE

The following chart describes all parameters used by the public procedure WSH_DELIVERY_DETAILS_PUB.SPLIT_LINE. All of the inbound and outbound parameters are listed. Additional information on these parameters follows.

Table 4–21 WSH_DELIVERY_DETAILS_PUB.SPLIT_LINE Parameters

Parameter	Usage	Type	Required
p_api_version_number	IN	Number	x
p_init_msg_list	IN	Varchar2	-
p_commit	IN	Varchar2	-
p_validation_level	IN	Number	-
x_return_status	OUT	Varchar2	-
x_msg_count	OUT	Number	-

Table 4–21 WSH_DELIVERY_DETAILS_PUB.SPLIT_LINE Parameters

Parameter	Usage	Type	Required
x_msg_data	OUT	Varchar2	-
p_from_detail_id	IN	Varchar2	x
x_new_detail_id	OUT	Number	-
x_split_quantity	IN OUT	Number	-

p_api_version_number

Compares the incoming API call's version number with the current version number. An error is returned if the version numbers are incompatible.

p_init_msg_list

Requests that the API initialize the message list. If the x_msg_count is greater than 1, then the list of messages must be retrieved using the call FND_MSG_PUB.GET. The values are:

- p_msg_index => I
- p_encoded => F
- p_data => 1_message
- p_msg_index_out => 1_msg_index_out

where 1_message and 1_msg_index_out are local variables of types Varchar2(2000) and Number respectively.

Default Value: FND_API.G_FALSE.

p_commit

Commits the transaction if this parameter is set to TRUE.

Default Value: FND_API.G_FALSE

x_return_status

Requests that the API return the status of the data for you after it completes its function. Valid values include:

- Success: FND_API.G_RET_STS_SUCCESS
- Error: FND_API.G_RET_STS_ERROR

- Unexpected Error: FND_API.G_RET_STS_UNEXP_ERROR

x_msg_count

Indicates number of error messages API has encountered.

x_msg_data

Returns error message text. If the x_msg_count is equal to 1, then this contains the actual message.

p_from_detail_id

The Delivery Detail ID of the line to be split.

x_new_detail_id

Delivery Detail ID of the new line being created after splitting the original line.

x_split_quantity

Quantity out of the original Delivery Detail quantity allocated to the new Delivery Detail created after the split.

Error Handling

Refer to parameters p_init_msg_list, x_msg_count, and x_msg_data on retrieving error messages.

Update_Shipping_Attributes API Features

The Update_Shipping API has the following features:

The Update_Shipping_Attributes procedure enables you to modify data in wsh_delivery_details.

Functional Overview

This procedure is called from the source system to modify data in wsh_delivery_details to reflect any changes made by the source system including pick release, ship confirm, split source line, and other update activities.

The procedure validates the input parameters and updates the delivery details.

Procedure Parameter Descriptions

WSH_DELIVERY_DETAILS_PUB.SPLIT_LINE

The following chart describes all parameters used by the public procedure WSH_DELIVERY_DETAILS_PUB.SPLIT_LINE. All of the inbound and outbound parameters are listed. Additional information on these parameters follows.

Table 4–22 WSH_DELIVERY_DETAILS_PUB.SPLIT_LINE Parameters

Parameter	Usage	Type	Required
p_api_version_number	IN	Number	x
p_init_msg_list	IN	Varchar2	-
x_return_status	OUT	Varchar2	-
x_msg_count	OUT	Number	-
x_msg_data	OUT	Varchar2	-
p_changed_attributes	IN	Record	x
p_source_code	IN	Varchar2	x
p_serial_range_tab	IN	wsh_delivery_details_grp.serialrangetype	-

p_api_version_number

Compares the incoming API call's version number with the current version number. An error is returned if the version numbers are incompatible.

p_init_msg_list

Requests that the API initialize the message list on your behalf. If the x_msg_count is greater than 1, then the list of messages must be retrieved using the call FND_MSG_PUB.GET. The values are:

- p_msg_index => I
- p_encoded => F
- p_data => 1_message
- p_msg_index_out => 1_msg_index_out

where 1_message and 1_msg_index_out are local variables of types Varchar2(2000) and Number respectively.

Default Value: FND_API.G_FALSE.

x_return_status

Requests that the API return the status of the data for you after it completes its function. Valid values include:

- Success: FND_API.G_RET_STS_SUCCESS
- Error: FND_API.G_RET_STS_ERROR
- Unexpected Error: FND_API.G_RET_STS_UNEXP_ERROR

x_msg_count

Indicates number of error messages API has encountered.

x_msg_data

Returns error message text. If the x_msg_count is equal to 1, then this contains the actual message.

p_changed_attributes

Attributes of ChangedAttributesTabType that are updated in wsh_delivery_details. WSH_DELIVERY_DETAILS_PUB.ChangedAttributesTabType is a table of ChangedAttributesRecType indexed by Binary_Integer. Definition of ChangedAttributesRecType follows.

p_source_code

Code for source system which updates wsh_delivery_details table. Source system can only update delivery details that are created by the same source system. For Update_Shipping_Attributes this should be OE.

p_serial_range_tab

This is of type WSH_DELIVERY_DETAILS_GRP.serialRangeTabType.

Where, WSH_DELIVERY_DETAILS_GRP.serialRangeTabType is of type

TYPE serialRangeTabType IS TABLE OF serialRangeRecType

INDEX BY BINARY_INTEGER;

Table 4-23 p_serial_range_tab

Attribute	Type	Default Value
delivery_detail_id	NUMBER	NULL
from_serial_number	VARCHAR2	NULL
to_serial_number	VARCHAR2	NULL
quantity	NUMBER	NULL

delivery_detail_id

The delivery detail id for which the serial numbers are entered.

from_serial_number

The starting serial number.

to_serial_number

The ending serial number.

quantity

The serial quantity, between the from_serial_number and the to_serial_number.

Record Parameter Descriptions**ChangedAttributesRecType RECORD DEFINITION**

To encapsulate WSH_DELIVERY_DETAILS table definition and Value column equivalents for ID columns in a PL/SQL record, define ChangedAttributesRecType.

Table 4–24 *ChangedAttributesRecType RECORD DEFINITION*

Attribute	Type	Default Value
source_header_id	Number	fnd_api.g_miss_num
source_line_id	Number	fnd_api.g_miss_num
sold_to_org_id	Number	fnd_api.g_miss_num
customer_number	Number	fnd_api.g_miss_char
sold_to_contact_id	Number	fnd_api.g_miss_num
ship_from_org_id	Number	fnd_api.g_miss_char
ship_from_org_code	Varchar2(3)	fnd_api.g_miss_char
ship_to_org_id	Number	fnd_api.g_miss_num
ship_to_org_code	Varchar2(3)	fnd_api.g_miss_char
ship_to_contact_id	Number	fnd_api.g_miss_num
deliver_to_org_id	Number	fnd_api.g_miss_num
deliver_to_org_code	Varchar2(3)	fnd_api.g_miss_char
deliver_to_contact_id	Number	fnd_api.g_miss_num
intmed_ship_to_org_id	Number	fnd_api.g_miss_num
intmed_ship_to_org_code	Varchar2(3)	fnd_api.g_miss_char
intmed_ship_to_contact_id	Number	fnd_api.g_miss_num
ship_tolerance_above	Number	fnd_api.g_miss_num
ship_tolerance_below	Number	fnd_api.g_miss_num
ordered_quantity	Number	fnd_api.g_miss_num
ordered_quantity2	Number	fnd_api.g_miss_num
order_quantity_uom	Varchar2(3)	fnd_api.g_miss_char
ordered_quantity_uom2	Varchar2(25)	fnd_api.g_miss_char
ordered_qty_unit_of_measure	Varchar2(25)	fnd_api.g_miss_char
ordered_qty_unit_of_measure2	Varchar2(25)	fnd_api.g_miss_char
subinventory	Varchar2(10)	fnd_api.g_miss_char
prefered_grade	Varchar2(4)	fnd_api.g_miss_char

Table 4–24 ChangedAttributesRecType RECORD DEFINITION

Attribute	Type	Default Value
revision	Varchar2(3)	fnd_api.g_miss_char
lot_number	Varchar2(30)	fnd_api.g_miss_char
sublot_number	Varchar2(30)	fnd_api.g_miss_char
customer_requested_lot_flag	Varchar2(1)	fnd_api.g_miss_char
serial_number	Varchar2(30)	fnd_api.g_miss_char
locator_id	Number	fnd_api.g_miss_num
date_requested	Date	fnd_api.g_miss_date
date_scheduled	Date	fnd_api.g_miss_date
master_container_item_id	Number	fnd_api.g_miss_num
detail_container_item_id	Number	fnd_api.g_miss_num
shipping_method_code	Varchar2(30)	fnd_api.g_miss_char
carrier_id	Number	fnd_api.g_miss_num
freight_terms_code	Varchar2(30)	fnd_api.g_miss_char
freight_terms_name	Varchar2(30)	fnd_api.g_miss_char
freight_carrier_code	Varchar2(30)	fnd_api.g_miss_char
shipment_priority_code	Varchar2(30)	fnd_api.g_miss_char
fob_code	Varchar2(30)	fnd_api.g_miss_char
fob_name	Varchar2(30)	fnd_api.g_miss_char
dep_plan_required_flag	Varchar2(1)	fnd_api.g_miss_char
customer_prod_seq	Varchar2(50)	fnd_api.g_miss_char
customer_dock_code	Varchar2(30)	fnd_api.g_miss_char
gross_weight	Number	fnd_api.g_miss_num
net_weight	Number	fnd_api.g_miss_num
weight_uom_code	Varchar2(3)	fnd_api.g_miss_char
weight_uom_desc	Varchar2(50)	fnd_api.g_miss_char
volume	Number	fnd_api.g_miss_num
volume_uom_code	Varchar2(3)	fnd_api.g_miss_char

Table 4–24 ChangedAttributesRecType RECORD DEFINITION

Attribute	Type	Default Value
volume_uom_desc	Varchar2(50)	fnd_api.g_miss_char
top_model_line_id	Number	fnd_api.g_miss_num
ship_set_id	Number	fnd_api.g_miss_num
ato_line_id	Number	fnd_api.g_miss_num
arrival_set_id	Number	fnd_api.g_miss_num
ship_model_complete_flag	Varchar2(1)	fnd_api.g_miss_char
cust_po_number	Varchar2(50)	fnd_api.g_miss_char
released_status	Varchar2(1)	fnd_api.g_miss_char
packing_instructions	Varchar2(2000)	fnd_api.g_miss_char
shipping_instructions	Varchar2(2000)	fnd_api.g_miss_char
container_name	Varchar2(50)	fnd_api.g_miss_char
container_flag	Varchar2(1)	fnd_api.g_miss_char
delivery_detail_id	Number	fnd_api.g_miss_num
shipped_quantity	Number	fnd_api.g_miss_num
shipped_quantity2	Number	fnd_api.g_miss_num
cycle_count_quantity	Number	fnd_api.g_miss_num
cycle_count_quantity2	Number	fnd_api.g_miss_num
tracking_number	Varchar2(30)	fnd_api.g_miss_char
attribute1	Varchar2(150)	fnd_api.g_miss_char
attribute2	Varchar2(150)	fnd_api.g_miss_char
attribute3	Varchar2(150)	fnd_api.g_miss_char
attribute4	Varchar2(150)	fnd_api.g_miss_char
attribute5	Varchar2(150)	fnd_api.g_miss_char
attribute6	Varchar2(150)	fnd_api.g_miss_char
attribute7	Varchar2(150)	fnd_api.g_miss_char
attribute8	Varchar2(150)	fnd_api.g_miss_char
attribute9	Varchar2(150)	fnd_api.g_miss_char

Table 4–24 ChangedAttributesRecType RECORD DEFINITION

Attribute	Type	Default Value
attribute10	Varchar2(150)	fnd_api.g_miss_char
attribute11	Varchar2(150)	fnd_api.g_miss_char
attribute12	Varchar2(150)	fnd_api.g_miss_char
attribute13	Varchar2(150)	fnd_api.g_miss_char
attribute14	Varchar2(150)	fnd_api.g_miss_char
attribute15	Varchar2(150)	fnd_api.g_miss_char
to_serial_number	Varchar2(30)	fnd_api.g_miss_char

Record Parameter Attribute Validations

source_header_id

Should indicate the source of the header content, for example OM or OKE.

sold_to_org_id

Should be a valid unique element of ra_customers.customer_id.

customer_number

Should be a valid unique element of ra_customers.customer_number.

sold_to_contact_id

Should indicate the person to contact at the sold to location.

ship_from_org_id

Should be a valid element of org_organization_definitions.organization_id.

ship_from_org_code

Should be a valid element of org_organization_definitions.organization_code.

ship_to_org_id

Should be a valid element of org_organization_definitions.organization_id.

ship_to_org_code

Should be a valid element of org_organization_definitions.organization_code.

ship_to_contact_id

Should indicate the person to contact at the ship to location.

deliver_to_org_id

Should be a valid element of org_organization_definitions.organization_id.

deliver_to_org_code

Should be a valid element of org_organization_definitions.organization_code.

deliver_to_contact_id

Should indicate the person to contact at the delivery to location.

intmed_ship_to_org_id

Should be a valid element of org_organization_definitions.organization_id.

intmed_ship_to_org_code

Should be a valid element of org_organization_definitions.organization_code.

intmed_ship_to_contact_id

Should indicate the person to contact at the intermediate ship to location.

ship_tolerance_above

Should be a non-negative number.

ship_tolerance_below

Should be a non-negative number.

ordered_quantity

Should be a non-negative whole number.

ordered_quantity2

Should be a non-negative whole number.

order_quantity_uom

Should be a valid element of mtl_items_uoms_view.uom_code.

ordered_quantity_uom2

Should be a valid element of mtl_items_uoms_view.uom_code.

preferred_grade

Should indicate the preferred grade for the line item.

ordered_qty_unit_of_measure

Should be a valid element of mtl_items_uoms_view.unit_of_measure.

ordered_qty_unit_of_measure2

Should be a valid element of mtl_items_uoms_view.unit_of_measure.

subinventory

Should indicate the subinventory from which the line item is picked.

revision

Should indicate the revision number for the line item.

lot_number

Should indicate the lot number for the line item.

customer_requested_lot_flag

Should indicate that the line item requires a customer requested lot number.

serial_number

Should indicate the serial number for the line item.

locator_id

Should indicate the inventory locator from which the line item is picked.

date_requested

Should indicate the date requested for the line item.

date_scheduled

Should indicate the date scheduled for the line item.

master_container_item_id

Should indicate the master container name into which the line is packed.

detail_container_item_id

Should indicate the parent container name into which the line is packed.

shipping_method_code

Should be a valid element of fnd_lookup_values_vl.lookup_code for lookup_type SHIP_METHOD.

carrier_id

Should indicate the carrier name transporting the line item.

freight_terms_code

Should be a valid element of fnd_lookup_values_vl.lookup_code for lookup_type FREIGHT_TERMS.

freight_terms_name

Should be a valid element of fnd_lookup_values_vl.meaning for lookup_type FREIGHT_TERMS.

freight_carrier_code

Should indicate the freight carrier code for the carrier transporting the line item.

shipment_priority_code

Should indicate the shipment priority assigned to the line item.

fob_code

Should be a valid element of fnd_lookup_values_vl.lookup_code for lookup_type FOB.

fob_name

Should be a valid element of fnd_lookup_values_vl.meaning for lookup_type FOB.

dep_plan_required_flag

Should be Y or N.

customer_prod_seq

Should indicate the customer production sequence number assigned to the line item.

customer_dock_code

Should indicate the customer dock code for the line item.

gross_weight

Should be a non-negative number.

net_weight

Should be a non-negative number.

weight_uom_code

Should be a valid element of mtl_units_of_measure.uom_code for a weight uom class

weight_uom_desc

Should be a valid element of mtl_units_of_measure.unit_of_measure for a weight uom class.

volume

Should be a non-negative number.

volume_uom_code

Should be a valid element of mtl_units_of_measure.uom_code for a volume uom class.

volume_uom_desc

Should be a valid element of mtl_units_of_measure.unit_of_measure for a volume uom class.

top_model_line_id

If the item is included as part of a model, this should indicate the top model line name for the line item.

ship_set_id

If the item is included as part of a ship set, this should indicate the ship set number to which the line item is assigned.

ato_line_id

If the item is included as part of a ATO model, this should indicate the model line name to which the line item is assigned.

arrival_set_id

If the item is included as part of a arrival ship set, this should indicate the arrival set number to which the line item is assigned.

ship_model_complete_flag

Should be Y or N.

cust_po_number

Should indicate the purchase order number for the line item.

released_status

Should be one of R, S, Y, or X.

packing_instructions

Should indicate the packing instructions included for the line item.

shipping_instructions

Should indicate the shipping instructions included for the line item.

container_name

Should indicate the container name of the container.

container_flag

Should indicate that the line item is a container.

delivery_detail_id

Should indicate the delivery detail id for the line.

tracking_number

Should indicate the tracking number for the line item.

shipped_quantity

Should be a non-negative number.

shipped_quantity2

Should be a non-negative number.

cycle_count_quantity

Should be a non-negative number.

cycle_count_quantity2

Should be a non-negative number.

Error Handling

Refer to parameters p_init_msg_list, x_msg_count, and x_msg_data on retrieving error messages.

Autocreate_Deliveries API Features

The Autocreate_Deliveries API has the following features:

The Autocreate_Deliveries procedure is called from the source system to automatically create deliveries for multiple delivery lines.

The procedure checks that the attribute parameters passed in are valid and perform value to ID conversion while validating.

Procedure Parameter Descriptions

WSH_DELIVERY_DETAILS_PUB.AUTOCREATE_DELIVERIES

The following chart describes all parameters used by the public procedure WSH_DELIVERY_DETAILS_PUB.AUTOCREATE_DELIVERIES. All of the inbound and outbound parameters are listed. Additional information on these parameters follows.

Table 4–25 WSH_DELIVERY_DETAILS_PUB.AUTOCREATE_DELIVERIES Parameters

Parameter	Usage	Type	Required
p_api_version_number	IN	Number	x
p_init_msg_list	IN	Varchar2	-
p_commit	IN	Varchar2	-
x_return_status	OUT	Varchar2	-
x_msg_count	OUT	Number	-
x_msg_data	OUT	Varchar2	-
p_line_rows	IN	Table	x
x_del_rows	OUT	Table	-

p_api_version_number

Compares the incoming API call's version number with the current version number. An error is returned if the version numbers are incompatible.

p_init_msg_list

Requests that the API initializes the message list on your behalf. If the x_msg_count is greater than 1, then the list of messages must be retrieved using the call FND_MSG_PUB.GET. The values are:

- `p_msg_index => I`
- `p_encoded => F`
- `p_data => 1_message`
- `p_msg_index_out => 1_msg_index_out`

where `1_message` and `1_msg_index_out` are local variables of types `Varchar2(2000)` and `Number` respectively.

Default Value: `FND_API.G_FALSE`.

p_commit

Commits the transaction if this parameter is set to `TRUE`.

Default Value: `FND_API.G_FALSE`

x_return_status

Requests that the API return the status of the data for you after it completes its function. Valid values include:

- Success: `FND_API.G_RET_STS_SUCCESS`
- Error: `FND_API.G_RET_STS_ERROR`
- Unexpected Error: `FND_API.G_RET_STS_UNEXP_ERROR`

x_msg_count

Indicates number of error messages API has encountered.

x_msg_data

Returns error message text. If the `x_msg_count` is equal to 1, then this contains the actual message.

p_line_rows

Table of Delivery Details of type `WSH_UTIL_CORE.ID_TAB_TYPE` which is a table of type `Number` indexed by binary integers.

x_del_rows

Table of Deliveries of type `WSH_UTIL_CORE.ID_TAB_TYPE` which is a table of type `Number` indexed by binary integers.

Error Handling

Refer to parameters `p_init_msg_list`, `x_msg_count`, and `x_msg_data` on retrieving error messages.

Autocreate_Del_Trip API Features

The Autocreate_Del_Trip API has the following features:

The Autocreate_Del_Trip procedure is called from the source system to automatically create trips and deliveries for multiple delivery lines.

The procedure also checks that the attribute parameters passed in are valid and perform value to ID conversion while validating.

Procedure Parameter Descriptions

WSH_DELIVERY_DETAILS_PUB.AUTOCREATE_DEL_TRIP

The following chart describes all parameters used by the public procedure WSH_DELIVERY_DETAILS_PUB.AUTOCREATE_DEL_TRIP. All of the inbound and outbound parameters are listed. Additional information on these parameters follows.

Table 4–26 WSH_DELIVERY_DETAILS_PUB.AUTOCREATE_DEL_TRIP Parameters

Parameter	Usage	Type	Required
p_api_version_number	IN	Number	x
p_init_msg_list	IN	Varchar2	-
p_commit	IN	Varchar2	-
x_return_status	OUT	Varchar2	-
x_msg_count	OUT	Number	-
x_msg_data	OUT	Varchar2	-
p_line_rows	IN	Table	x
x_del_rows	OUT	Table	x
x_trip_id	OUT	Number	x
x_trip_name	OUT	Varchar2	x

p_api_version_number

Compares the incoming API call's version number with the current version number. An error is returned if the version numbers are incompatible.

p_init_msg_list

Requests that the API initialize the message list on your behalf. If the x_msg_count is greater than 1, then the list of messages must be retrieved using the call FND_MSG_PUB.GET. The values are:

- p_msg_index => I
- p_encoded => F
- p_data => 1_message
- p_msg_index_out => 1_msg_index_out

where 1_message and 1_msg_index_out are local variables of types Varchar2(2000) and Number respectively.

Default Value: FND_API.G_FALSE.

p_commit

Commits the transaction if this parameter is set to TRUE.

Default Value: FND_API.G_FALSE.

x_return_status

Requests that the API return the status of the data for you after it completes its function. Valid values include:

- Success: FND_API.G_RET_STS_SUCCESS
- Error: FND_API.G_RET_STS_ERROR
- Unexpected Error: FND_API.G_RET_STS_UNEXP_ERROR

x_msg_count

Indicates number of error messages API has encountered.

x_msg_data

Returns error message text. If the x_msg_count is equal to 1, then this contains the actual message.

p_line_rows

Table of Delivery Details of type WSH_UTIL_CORE.ID_TAB_TYPE which is a table of type Number indexed by binary integers.

x_del_rows

Table of Deliveries of type WSH_UTIL_CORE.ID_TAB_TYPE which is a table of type Number indexed by binary integers.

x_trip_id

ID of autocreated trip.

x_trip_name

Name of autocreated trip.

Error Handling

Refer to parameters p_init_msg_list, x_msg_count, and x_msg_data on retrieving error messages.

Container Public Application Program Interface

The Container Public Application Program Interface (API) is a public API that consists of the four following procedures in package **WSH_CONTAINER_PUB**:

Create_Containers: Enables you to create a new container record.

Update_Container: Updates an existing container record.

Auto_Pack: Lets you autopack containers.

Container_Actions: Enables you to perform certain actions on a container.

This section describes how to use the Container Public API and how it functions in Oracle Shipping Execution.

Create_Containers API Features

The Create_Containers API has the following features.

The Create_Containers procedure takes in a container item id or container item name and other necessary parameters to create one or more containers and creates the required containers. It returns a table of container instance ids (delivery detail ids) along with the standard out parameters.

Functional Overview

This API creates a new container record in WSH_DELIVERY_DETAILS. It Inserts container information into WSH_DELIVERY_DETAILS and returns a table of the newly created container IDs.

Procedure Parameter Descriptions

WSH_CONTAINER_PUB.CREATE_CONTAINERS

The following chart describes all parameters used by the public procedure WSH_CONTAINER_PUB.CREATE_CONTAINERS. All of the inbound and outbound parameters are listed. Additional information on these parameters follows.

Table 4–27 WSH_CONTAINER_PUB.CREATE_CONTAINERS Parameters

Parameter	Usage	Type	Required
p_api_version_number	IN	Number	x
p_init_msg_list	IN	Varchar2	-

Table 4–27 WSH_CONTAINER_PUB.CREATE_CONTAINERS Parameters

Parameter	Usage	Type	Required
p_commit	IN	Varchar2	-
p_validation_level	IN	Number	-
x_return_status	OUT	Varchar2	-
x_msg_count	OUT	Number	-
x_msg_data	OUT	Varchar2	-
p_container_item_id	IN	Number	-
p_container_item_name	IN	Varchar2	x
p_organization_id	IN	Number	-
p_organization_code	IN	Varchar2	-
p_name_prefix	IN	Varchar2	-
p_name_suffix	IN	Varchar2	-
p_base_number	IN	Number	x
p_num_digits	IN	Number	-
p_quantity	IN	Number	-
p_container_name	IN	Varchar2	-
p_action	IN	Varchar2	-
x_container_ids	OUT	Table	-

p_api_version_number

Compares the incoming API call's version number with the current version number. An error is returned if the version numbers are incompatible.

p_init_msg_list

Requests that the API initialize the message list on your behalf. If the x_msg_count is greater than 1, then the list of messages must be retrieved using the call FND_MSG_PUB.GET. The values are:

- p_msg_index => I
- p_encoded => F
- p_data => 1_message

- `p_msg_index_out => 1_msg_index_out`

where `1_message` and `1_msg_index_out` are local variables of types `Varchar2(2000)` and `Number` respectively.

Default Value: `FND_API.G_FALSE`

x_return_status

Requests that the API return the status of the data for you after it completes its function. Valid values include:

- Success: `FND_API.G_RET_STS_SUCCESS`
- Error: `FND_API.G_RET_STS_ERROR`
- Unexpected Error: `FND_API.G_RET_STS_UNEXP_ERROR`

x_msg_count

Indicates number of error messages API has encountered.

x_msg_data

Returns error message text. If the `x_msg_count` is equal to 1, then this contains the actual message.

p_container_item_id

Key flexfield Id for the container.

p_container_item_name

Flexfield name for the container.

p_organization_id

Organization ID for the container.

p_organization_code

Organization code for the container.

p_name_prefix

Prefix of the container name.

p_name_suffix

Suffix of the container name.

p_name_prefix

Prefix of the container name.

p_base_number

Starting number for the numeric portion of the container name.

p_num_digits

Precision for the number of digits.

p_quantity

Number of containers created.

p_container_name

Container name if creating just one container.

x_container_ids

Table of the newly created container IDs of type WSH_UTIL_CORE.ID_TAB_TYPE which is a table of type Number indexed by binary integers.

Error Handling

Refer to parameters p_init_msg_list, x_msg_count, and x_msg_data on retrieving error messages.

Update Container API Features

The Trip_Action API has the following features:

The Update_Container procedure enables you to update an existing container. It accepts as IN parameters the container information and the name/ID of the container being updated.

Functional Overview

The API updates an existing container record in WSH_DELIVERY_DETAILS with the attributes input in the container rec type. The name or the ID of the container being updated is passed as IN parameters p_container_name and p_cont_instance_id respectively.

The API validates the input record attributes before updating a container record.

Table 4–28 WSH_DELIVERY_DETAILS

Parameter	Usage	Type	Required
p_api_version_number	IN	Number	x
p_init_msg_list	IN	Varchar2	-
p_commit	IN	Varchar2	-
p_validation_level	IN	Varchar2	-
x_return_status	OUT	Varchar2	-
x_msg_count	OUT	Number	-
x_msg_data	OUT	Varchar2	-
p_container_rec	IN	Record	x

p_api_version_number

Compares the incoming API call's version number with the current version number. An error is returned if the version numbers are incompatible.

p_init_msg_list

Requests that the API initialize the message list on your behalf. If the x_msg_count is greater than 1, then the list of messages must be retrieved using the call FND_MSG_PUB.GET. The values are:

- p_msg_index => I

- `p_encoded => F`
- `p_data => 1_message`
- `p_msg_index_out => 1_msg_index_out`

where `1_message` and `1_msg_index_out` are local variables of types `Varchar2(2000)` and `Number` respectively.

Default Value: `FND_API.G_FALSE`

x_return_status

Requests that the API return the status of the data for you after it completes its function. Valid values include:

- Success: `FND_API.G_RET_STS_SUCCESS`
- Error: `FND_API.G_RET_STS_ERROR`
- Unexpected Error: `FND_API.G_RET_STS_UNEXP_ERROR`

x_msg_count

Indicates number of error messages API has encountered.

x_msg_data

Returns error message text. If the `x_msg_count` is equal to 1, then this contains the actual message.

p_container_rec

Attributes of the container entity of type `ChangedAttributeRecType`. These attributes are updated in `WSH_DELIVERY_DETAILS`. Definition of `ChangedAttributeRecType` follows.

Record Parameter Descriptions

TRIP_STOP_PUB_REC_TYPE RECORD DEFINITION

To encapsulate `WSH_DELIVERY_DETAILS` table definition and Value column equivalents for ID columns in a PL/SQL record, define `ChangedAttributeRecType` and use this to pass trip information to the `Update_Container` routine.

Table 4–29 TRIP_STOP_PUB_REC_TYPE RECORD DEFINITION

Attribute	Type	Default Value
source_header_id	Number	fnd_api.g_miss_num
source_line_id	Number	fnd_api.g_miss_num
sold_to_org_id	Number	fnd_api.g_miss_num
customer_number	Number	fnd_api.g_miss_char
sold_to_contact_id	Number	fnd_api.g_miss_num
ship_from_org_id	Number	fnd_api.g_miss_char
ship_from_org_code	Varchar2(3)	fnd_api.g_miss_char
ship_to_org_id	Number	fnd_api.g_miss_num
ship_to_org_code	Varchar2(3)	fnd_api.g_miss_char
ship_to_contact_id	Number	fnd_api.g_miss_num
deliver_to_org_id	Number	fnd_api.g_miss_num
deliver_to_org_code	Varchar2(3)	fnd_api.g_miss_char
deliver_to_contact_id	Number	fnd_api.g_miss_num
intmed_ship_to_org_id	Number	fnd_api.g_miss_num
intmed_ship_to_org_code	Varchar2(3)	fnd_api.g_miss_char
intmed_ship_to_contact_id	Number	fnd_api.g_miss_num
ship_tolerance_above	Number	fnd_api.g_miss_num
ship_tolerance_below	Number	fnd_api.g_miss_num
ordered_quantity	Number	fnd_api.g_miss_num
order_quantity_uom	Varchar2(3)	fnd_api.g_miss_char
ordered_qty_unit_of_measure	Varchar2(25)	fnd_api.g_miss_char
subinventory	Varchar2(10)	fnd_api.g_miss_char
revision	Varchar2(3)	fnd_api.g_miss_char
lot_number	Varchar2(30)	fnd_api.g_miss_char
customer_requested_lot_flag	Varchar2(1)	fnd_api.g_miss_char

Table 4–29 TRIP_STOP_PUB_REC_TYPE RECORD DEFINITION

Attribute	Type	Default Value
serial_number	Varchar2(30)	fnd_api.g_miss_char
locator_id	Number	fnd_api.g_miss_num
date_requested	Date	fnd_api.g_miss_date
date_scheduled	Date	fnd_api.g_miss_date
master_container_item_id	Number	fnd_api.g_miss_num
detail_container_item_id	Number	fnd_api.g_miss_num
shipping_method_code	Varchar2(30)	fnd_api.g_miss_char
carrier_id	Number	fnd_api.g_miss_num
freight_terms_code	Varchar2(30)	fnd_api.g_miss_char
freight_terms_name	Varchar2(30)	fnd_api.g_miss_char
freight_carrier_code	Varchar2(30)	fnd_api.g_miss_char
shipment_priority_code	Varchar2(30)	fnd_api.g_miss_char
fob_code	Varchar2(30)	fnd_api.g_miss_char
fob_name	Varchar2(30)	fnd_api.g_miss_char
dep_plan_required_flag	Varchar2(1)	fnd_api.g_miss_char
customer_prod_seq	Varchar2(50)	fnd_api.g_miss_char
customer_dock_code	Varchar2(30)	fnd_api.g_miss_char
gross_weight	Number	fnd_api.g_miss_num
net_weight	Number	fnd_api.g_miss_num
weight_uom_code	Varchar2(3)	fnd_api.g_miss_char
weight_uom_desc	Varchar2(50)	fnd_api.g_miss_char
volume	Number	fnd_api.g_miss_num
volume_uom_code	Varchar2(3)	fnd_api.g_miss_char
volume_uom_desc	Varchar2(50)	fnd_api.g_miss_char
top_model_line_id	Number	fnd_api.g_miss_num
ship_set_id	Number	fnd_api.g_miss_num
ato_line_id	Number	fnd_api.g_miss_num

Table 4–29 TRIP_STOP_PUB_REC_TYPE RECORD DEFINITION

Attribute	Type	Default Value
arrival_set_id	Number	fnd_api.g_miss_num
ship_model_complete_flag	Varchar2(1)	fnd_api.g_miss_char
cust_po_number	Varchar2(50)	fnd_api.g_miss_char
released_status	Varchar2(1)	fnd_api.g_miss_char
packing_instructions	Varchar2(2000)	fnd_api.g_miss_char
shipping_instructions	Varchar2(2000)	fnd_api.g_miss_char
container_name	Varchar2(50)	fnd_api.g_miss_char
container_flag	Varchar2(1)	fnd_api.g_miss_char
delivery_detail_id	Number	fnd_api.g_miss_num

Record Parameter Attribute Validations

sold_to_org_id

Should be a valid unique element of ra_customers.customer_id.

customer_number

Should be a valid unique element of ra_customers.customer_number

ship_from_org_id

Should be a valid element of org_organization_definitions.organization_id.

ship_from_org_code

Should be a valid element of org_organization_definitions.organization_code.

ship_to_org_id

Should be a valid element of org_organization_definitions.organization_id.

ship_to_org_code

Should be a valid element of org_organization_definitions.organization_code.

delier_to_org_id

Should be a valid element of org_organization_definitions.organization_id.

deliver_to_org_code

Should be a valid element of org_organization_definitions.organization_code.

intmed_ship_to_org_id

Should be a valid element of org_organization_definitions.organization_id.

intmed_ship_to_org_code

Should be a valid element of org_organization_definitions.organization_code.

ship_tolerance_above

Should be a non-negative number.

ship_tolerance_below

Should be a non-negative number.

ordered_quantity

Should be a non-negative whole number.

order_quantity_uom

Should be a valid element of mtl_items_uoms_view.uom_code.

ordered_quantity_unit_of_measure

Should be a valid element of mtl_items_uoms_view.unit_of_measure.

shipping_method_code

Should be a valid element of fnd_lookup_values_vl.lookup_code for lookup_type SHIP_METHOD.

freight_terms_code

Should be a valid element of fnd_lookup_values_vl.lookup_code for lookup_type FREIGHT_TERMS.

freight_terms_name

Should be a valid element of fnd_lookup_values_vl.meaning for lookup_type FREIGHT_TERMS.

fob_code

Should be a valid element of fnd_lookup_values_vl.lookup_code for lookup_type FOB.

fob_name

Should be a valid element of fnd_lookup_values_vl.meaning for lookup_type FOB.

dep_plan_required_flag

Should be Y or N.

gross_weight

Should be a non-negative number.

net_weight

Should be a non-negative number.

weight_uom_code

Should be a valid element of mtl_units_of_measure.uom_code for a weight uom class

weight_uom_desc

Should be a valid element of mtl_units_of_measure.unit_of_measure for a weight uom class.

volume

Should be a non-negative number.

volume_uom_code

Should be a valid element of mtl_units_of_measure.uom_code for a volume uom class.

volume_uom_desc

Should be a valid element of mtl_units_of_measure.unit_of_measure for a volume uom class.

ship_model_complete_flag

Should be Y or N.

released_status

Should be one of R, S, Y or X.

Error Handling

Refer to parameters p_init_msg_list, x_msg_count, and x_msg_data on retrieving error messages.

Auto_Pack API Features

The Auto_Pack API has the following features.

The Auto_Pack procedure takes in a table of ids of either delivery lines or container or deliveries and autopacks the lines/containers/deliveries into detail containers and returns a table of container instance ids created during the autopacking process

Functional Overview

This API takes in a table of ids of either delivery lines or container or deliveries and autopacks the lines/containers/deliveries into detail containers. The grouping id table is used only if the input table of entities are lines or containers only. The packing of lines and containers into parent containers is determined by the grouping id for each line/container.

If the grouping id table is not input, the API determines the grouping ids for the lines/containers based on the grouping attributes of the lines/containers. The lines/containers are then autopacked into detail containers and the detail containers are packed into parent/master containers based on whether the p_pack_cont_flag is set to 'Y' or 'N'. The API returns a table of container instance ids created during the autopacking operation. If the detail containers are packed into parent containers, the output table of ids will contain both the detail and parent containers' delivery detail ids.

Procedure Parameter Descriptions

WSH_CONTAINER_PUB.AUTO_PACK

The following chart describes all parameters used by the public procedure WSH_CONTAINER_PUB.AUTO_PACK. All of the inbound and outbound parameters are listed. Additional information on these parameters follows.

Table 4–30 WSH_CONTAINER_PUB.AUTO_PACK Parameters

Parameter	Usage	Type	Required
p_api_version_number	IN	Number	x
p_init_msg_list	IN	Varchar2	-
p_commit	IN	Varchar2	-
p_validation_level	IN	Number	-
x_return_status	OUT	Varchar2	-

Table 4–30 WSH_CONTAINER_PUB.AUTO_PACK Parameters

Parameter	Usage	Type	Required
x_msg_count	OUT	Number	-
x_msg_data	OUT	Varchar2	-
p_entity_tab	IN	Table	-
p_entity_type	IN	Varchar2	x
p_group_id_tab	IN	Table	-
p_pack_cont_flag	IN	Varchar2	x
x_cont_inst_tab	IN	Table	-

p_api_version_number

Compares the incoming API call's version number with the current version number. An error is returned if the version numbers are incompatible.

p_init_msg_list

Requests that the API initialize the message list on your behalf. If the x_msg_count is greater than 1, then the list of messages must be retrieved using the call FND_MSG_PUB.GET. The values are:

- p_msg_index => I
- p_encoded => F
- p_data => 1_message
- p_msg_index_out => 1_msg_index_out

where 1_message and 1_msg_index_out are local variables of types Varchar2(2000) and Number respectively.

Default Value: FND_API.G_FALSE

x_return_status

Requests that the API return the status of the data for you after it completes its function. Valid values include:

- Success: FND_API.G_RET_STS_SUCCESS
- Error: FND_API.G_RET_STS_ERROR
- Unexpected Error: FND_API.G_RET_STS_UNEXP_ERROR

x_msg_count

Indicates number of error messages API has encountered.

x_msg_data

Returns error message text. If the x_msg_count is equal to 1, then this contains the actual message.

p_entity_tab

Table of ids of either lines or containers or deliveries that need to be autopacked of type WSH_UTIL_CORE.ID_TAB_TYPE which is a table of type Number indexed by binary integers.

p_entity_type

Type of entity id contained in the entity_tab that needs to be autopacked ('L' - lines, 'C' - containers or 'D' - deliveries).

p_group_id_tab

Table of ids (numbers that determine the grouping of lines for packing into containers) of type WSH_UTIL_CORE.ID_TAB_TYPE which is a table of type Number indexed by binary integers.

p_pack_cont_flag

A 'Y' or 'N' value to determine whether to autopack the detail containers that are created into parent containers.

x_cont_inst_tab

Table of container IDs created during the autopacking process of type WSH_UTIL_CORE.ID_TAB_TYPE which is a table of type Number indexed by binary integers.

Error Handling

Refer to parameters p_init_msg_list, x_msg_count, and x_msg_data on retrieving error messages.

Container_Actions API Features

The Container_Actions API has the following features.

The Container_Actions procedure enables you to carry out actions on a container. It accepts as IN parameters a table of delivery detail ids, an action code, and any additional parameters needed for specific actions, and returns a completion status.

Functional Overview

This API takes in a table of delivery detail ids and name and/or delivery detail id of the container to pack. If the action code is assigned then delivery id and delivery name must be specified. The API determines what action to perform based on the action code and then calls appropriate private pack/assign/unpack/unassign API. The input table of ids could be lines or containers. The delivery lines and containers are separated from the input table and validated before the appropriate private APIs are called.

Procedure Parameter Descriptions

WSH_CONTAINER_PUB.CONTAINER_ACTIONS

The following chart describes all parameters used by the public procedure WSH_CONTAINER_PUB.CONTAINER_ACTIONS. All of the inbound and outbound parameters are listed. Additional information on these parameters follows.

Table 4–31 WSH_CONTAINER_PUB.CONTAINER_ACTIONS Parameters

Parameter	Usage	Type	Required
p_api_version_number	IN	Number	x
p_init_msg_list	IN	Varchar2	-
p_commit	IN	Varchar2	-
p_validation_level	IN	Number	-
x_return_status	OUT	Varchar2	-
x_msg_count	OUT	Number	-
x_msg_data	OUT	Varchar2	-
p_detail_tab	IN	Table	x
p_container_instance_id	IN	Number	-

Table 4–31 WSH_CONTAINER_PUB.CONTAINER_ACTIONS Parameters

Parameter	Usage	Type	Required
p_container_flag	IN	Table	-
p_delivery_flag	IN	Varchar2	-
p_delivery_id	IN	Number	-
p_delivery_name	IN	Varchar2	-
p_action_code	IN	Varchar2	-

p_api_version_number

Compares the incoming API call's version number with the current version number. An error is returned if the version numbers are incompatible.

p_init_msg_list

Requests that the API initialize the message list on your behalf. If the x_msg_count is greater than 1, then the list of messages must be retrieved using the call FND_MSG_PUB.GET. The values are:

- p_msg_index => I
- p_encoded => F
- p_data => 1_message
- p_msg_index_out => 1_msg_index_out

where 1_message and 1_msg_index_out are local variables of types Varchar2(2000) and Number respectively.

Default Value: FND_API.G_FALSE

x_return_status

Requests that the API return the status of the data for you after it completes its function. Valid values include:

- Success: FND_API.G_RET_STS_SUCCESS
- Error: FND_API.G_RET_STS_ERROR
- Unexpected Error: FND_API.G_RET_STS_UNEXP_ERROR

x_msg_count

Indicates number of error messages API has encountered.

x_msg_data

Returns error message text. If the x_msg_count is equal to 1, then this contains the actual message.

p_detail_tab

Input table of delivery detail ids of type WSH_UTIL_CORE.ID_TAB_TYPE which is a table of type Number indexed by binary integers.

p_container_instance_id

Delivery detail ID of parent container being packed

p_container_name

Container name if ID is not known

p_container_flag

Y or N depending on whether to unpack or not respectively.

p_delivery_flag

Y or N depending on whether the container needs to be unassigned from a delivery or not respectively.

p_delivery_id

Delivery ID the container is assigned to.

p_delivery_name

Delivery name the container is assigned to.

p_action_code

Action code Pack, Assign, Unpack, or Unassign to specify which action to perform.

Error Handling

Refer to parameters p_init_msg_list, x_msg_count, and x_msg_data on retrieving error messages.

Freight Costs Public Application Program Interface

The Freight Costs Public Application Program Interface (API) is a public API that consists of the following three procedures in package WSH_FREIGHT_COSTS_PUB:

Create_Update_Freight_Costs: Enables you to create a new freight cost record and update an existing freight cost record.

Validate_Freight_Cost_Type: Validates that the freight cost type exists.

Delete_Freight_Costs: Enables you to delete a freight cost record.

This section describes how to use the Freight Costs Public API and how it functions in Oracle Shipping Execution.

Create_Update_Freight_Costs API Features

The Create_Update_Freight_Cost API has the following features.

The Create_Update_Freight_Costs procedure enables you to create a new freight cost record or update an existing freight cost record in WSH_FREIGHT_COSTS table. The FREIGHT_COST_ID, and return status of a new freight cost record are passed as OUT parameters, while a freight cost record of freight cost information is passed as an IN parameter.

Functional Overview

This API creates a new freight cost record in WSH_FREIGHT_COSTS as specified by IN parameter p_action_code value CREATE. It inserts the freight cost information into WSH_FREIGHT_COSTS and returns the FREIGHT_COST_ID of the new freight cost record. It also updates an existing freight cost record in WSH_FREIGHT_COSTS as specified by IN parameter p_action_code value UPDATE.

The API validates freight information such as freight cost type, unit amount, conversion rate, and currency before performing the actions of creating or updating a freight cost record. It also checks that the insert or update statements were successful, and if not returns an error.

Also, ensure that only one shipping entity—a trip, stop, delivery, delivery leg, or delivery detail—is passed in to be associated with a freight cost record. Each freight cost must be associated with only one shipping entity.

Procedure Parameter Descriptions

WSH_FREIGHT_COSTS_PUB.CREATE_UPDATE_FREIGHT_COSTS

The following chart describes all parameters used by the public procedure WSH_FREIGHT_COSTS_PUB.CREATE_UPDATE_FREIGHT_COSTS. All of the inbound and outbound parameters are listed. Additional information on these parameters follows.

Table 4–32 WSH_FREIGHT_COSTS_PUB.CREATE_UPDATE_FREIGHT_COSTS Parameters

Parameter	Usage	Type	Required
p_api_version_number	IN	Number	x
p_init_msg_list	IN	Varchar2	-
x_return_status	OUT	Varchar2	-
x_msg_count	OUT	Number	-
x_msg_data	OUT	Varchar2	-
p_pub_freight_costs	IN OUT	Record	x
p_action_code	IN	Varchar2	x
x_freight_cost_id	OUT	Number	-

p_api_version_number

Compares the incoming API call's version number with the current version number. An error is returned if the version numbers are incompatible.

p_init_msg_list

Requests that the API initialize the message list on your behalf. If the x_msg_count is greater than 1, then the list of messages must be retrieved using the call FND_MSG_PUB.GET. The values are:

- p_msg_index => I
- p_encoded => F
- p_data => 1_message
- p_msg_index_out => 1_msg_index_out

where 1_message and 1_msg_index_out are local variables of types Varchar2(2000) and Number respectively.

Default Value: FND_API.G_FALSE

x_return_status

Requests that the API return the status of the data for you after it completes its function. Valid values include:

- Success: FND_API.G_RET_STS_SUCCESS
- Error: FND_API.G_RET_STS_ERROR
- Unexpected Error: FND_API.G_RET_STS_UNEXP_ERROR

x_msg_count

Indicates number of error messages API has encountered.

x_msg_data

Returns error message text. If the x_msg_count is equal to 1, then this contains the actual message.

p_pub_freight_costs

Attributes of the freight cost entity of type PubFreightCostRecType. These attributes are inserted/updated in WSH_FREIGHT_COSTS. Definition of PubFreightCostRecType follows.

p_action_code

Specifies whether API should create a new freight cost record or update existing freight cost record information based on its values CREATE or UPDATE.

x_freight_cost_id

ID of new freight cost record being created.

Record Parameter Descriptions

PubFreightCostRecType RECORD DEFINITION

To encapsulate WSH_FREIGHT_COSTS table definition and Value column equivalents for ID columns in a PL/SQL record, define PubFreightCostRecType and use this to pass freight cost record information to the Create_Update_Freight_Cost routine.

Table 4–33 PubFreightCostRecType RECORD DEFINITION

Attribute	Type	Default Value
freight_cost_id	Number	fnd_api.g_miss_num
freight_cost_type_id	Number	fnd_api.g_miss_num
unit_amount	Number	fnd_api.g_miss_num
currency_code	Varchar2(15)	fnd_api.g_miss_char
conversion_date	Date	fnd_api.g_miss_date
conversion_rate	Number	fnd_api.g_miss_num
conversion_type_code	Varchar2(30)	fnd_api.g_miss_char
trip_id	Number	fnd_api.g_miss_num
trip_name	Varchar2(30)	fnd_api.g_miss_char
stop_id	Number	fnd_api.g_miss_num
stop_location_id	Number	fnd_api.g_miss_num
planned_dep_date	Date	fnd_api.g_miss_date
delivery_id	Number	fnd_api.g_miss_num
delivery_name	Varchar2(30)	fnd_api.g_miss_char
delivery_leg_id	Number	fnd_api.g_miss_num
delivery_detail_id	Number	fnd_api.g_miss_num
attribute_category	Varchar2(150)	fnd_api.g_miss_char
attribute1	Varchar2(150)	fnd_api.g_miss_char
attribute2	Varchar2(150)	fnd_api.g_miss_char
attribute3	Varchar2(150)	fnd_api.g_miss_char
attribute4	Varchar2(150)	fnd_api.g_miss_char

Table 4–33 *PubFreightCostRecType* RECORD DEFINITION

Attribute	Type	Default Value
attribute5	Varchar2(150)	fnd_api.g_miss_char
attribute6	Varchar2(150)	fnd_api.g_miss_char
attribute7	Varchar2(150)	fnd_api.g_miss_char
attribute8	Varchar2(150)	fnd_api.g_miss_char
attribute9	Varchar2(150)	fnd_api.g_miss_char
attribute10	Varchar2(150)	fnd_api.g_miss_char
attribute11	Varchar2(150)	fnd_api.g_miss_char
attribute12	Varchar2(150)	fnd_api.g_miss_char
attribute13	Varchar2(150)	fnd_api.g_miss_char
attribute14	Varchar2(150)	fnd_api.g_miss_char
attribute15	Varchar2(150)	fnd_api.g_miss_char
creation_date	Date	fnd_api.g_miss_date
created_by	Number	fnd_api.g_miss_num
last_update_date	Date	fnd_api.g_miss_date
last_updated_by	Number	fnd_api.g_miss_num
last_update_login	Date	fnd_api.g_miss_date
program_application_id	Number	fnd_api.g_miss_num
program_id	Number	fnd_api.g_miss_num
program_update_date	Date	fnd_api.g_miss_date
request_id	Number	fnd_api.g_miss_num
freight_cost_type	Varchar2(30)	fnd_api.g_miss_char
action_code	Varchar2(30)	fnd_api.g_miss_char

Record Parameter Attribute Validations

freight_cost_id

Should be a unique valid element of wsh_freight_cost_types.freight_cost_id.

freight_cost_type_id

Should be a valid element of wsh_freight_cost_types.freight_cost_type_id.

unit_amount

Should be a non-negative number.

currency_code

Should be a valid element of fnd_currencies_vl.currency_code.

conversion_rate

Should be a non-negative number.

trip_id

Should be a valid unique element of wsh_trips.trip_id.

trip_name

Should be a valid unique element of wsh_trips.name

stop_id

Should be a valid unique element of wsh_trip_stops.stop_id.

stop_location_id

Should be a valid element of wsh_trip_stops.stop_location_id.

planned_dep_date

Should be a valid element of wsh_trip_stops.planned_departure_date.

delivery_id

Should be a valid unique element of wsh_new_deliveries.delivery_id.

delivery_name

Should be a valid unique element of wsh_new_deliveries.name

delivery_leg_id

Should be a valid element of wsh_delivery_legs.delivery_leg_id.

delivery_detail_id

Should be a valid element of wsh_delivery_details.delivery_detail_id.

freight_cost_type

Should be a valid element of wsh_freight_cost_types.name.

Error Handling

Refer to parameters p_init_msg_list, x_msg_count, and x_msg_data on retrieving error messages.

Validate_Freight_Cost_Type API Features

The Validate_Freight_Cost_Type API has the following features.

The Validate_Freight_Cost_Type procedure enables you to validate an existing freight cost type from the WSH_FREIGHT_COST_TYPES table. The freight cost type is passed as an IN parameter, while the freight cost id and a return status are passed as an OUT parameter.

Procedure Parameter Descriptions

WSH_FREIGHT_COSTS_PUB.DELETE_FREIGHT_COSTS

The following chart describes all parameters used by the public procedure WSH_FREIGHT_COSTS_PUB.DELETE_FREIGHT_COSTS. All of the inbound and outbound parameters are listed. Additional information on these parameters follows.

Table 4–34 *WSH_FREIGHT_COSTS_PUB.DELETE_FREIGHT_COSTS Parameters*

Parameter	Usage	Type	Required
p_freight_cost_type	IN	Varchar2	x
x_freight_cost_type_id	IN OUT	Varchar2	-
x_return_status	OUT	Varchar2	-

p_freight_cost_type

Freight cost type that needs to be validated.

x_freight_cost_type_id

Type Id of the freight cost being validated.

x_return_status

Requests that the API return the status of the data for you after it completes its function. Valid values include:

- Success: FND_API.G_RET_STS_SUCCESS
- Error: FND_API.G_RET_STS_ERROR
- Unexpected Error: FND_API.G_RET_STS_UNEXP_ERROR.

Delete_Freight_Costs API Features

The Delete_Freight_Costs API has the following features.

The Delete_Freight_Costs procedure enables you to delete an existing freight cost record from the WSH_FREIGHT_COSTS table. A freight cost record of freight cost information of the freight cost record being deleted is passed as an IN parameter.

Functional Overview

The Delete_Freight_Costs procedure enables you to delete a new freight cost record from WSH_FREIGHT_COSTS table. It identifies the freight record being deleted by using the freight_cost_id attribute in the freight cost record that is being passed as an IN parameter.

Procedure Parameter Descriptions

WSH_FREIGHT_COSTS_PUB.DELETE_FREIGHT_COSTS

The following chart describes all parameters used by the public procedure WSH_FREIGHT_COSTS_PUB.DELETE_FREIGHT_COSTS. All of the inbound and outbound parameters are listed. Additional information on these parameters follows.

Table 4–35 WSH_FREIGHT_COSTS_PUB.DELETE_FREIGHT_COSTS Parameters

Parameter	Usage	Type	Required
p_api_version_number	IN	Number	x
p_init_msg_list	IN	Varchar2	-
x_return_status	OUT	Varchar2	-
x_msg_count	OUT	Number	-
x_msg_data	OUT	Varchar2	-
p_pub_freight_costs	IN OUT	Record	x

p_api_version_number

Compares the incoming API call's version number with the current version number. An error is returned if the version numbers are incompatible.

p_init_msg_list

Requests that the API initialize the message list on your behalf. If the x_msg_count is greater than 1, then the list of messages must be retrieved using the call FND_MSG_PUB.GET. The values are:

- p_msg_index => I
- p_encoded => F
- p_data => 1_message
- p_msg_index_out => 1_msg_index_out

where 1_message and 1_msg_index_out are local variables of types Varchar2(2000) and Number respectively.

Default Value: FND_API.G_FALSE

x_return_status

Requests that the API return the status of the data after it completes its function. Valid values include:

- Success: FND_API.G_RET_STS_SUCCESS
- Error: FND_API.G_RET_STS_ERROR
- Unexpected Error: FND_API.G_RET_STS_UNEXP_ERROR

x_msg_count

Indicates number of error messages API has encountered.

x_msg_data

Returns error message text. If the x_msg_count is equal to 1, then this contains the actual message.

p_pub_freight_costs

Attributes of the freight cost entity of type PubFreightCostRecType. These attributes are inserted/updated in WSH_FREIGHT_COSTS. Definition of PubFreightCostRecType follows.

Record Parameter Descriptions

PubFreightCostRecType RECORD DEFINITION

To encapsulate WSH_FREIGHT_COSTS table definition and Value column equivalents for ID columns in a PL/SQL record, we define PubFreightCostRecType and use this to pass Freight Cost information to the Delete_Freight_Cost routine.

Table 4–36 PubFreightCostRecType RECORD DEFINITION

Attribute	Type	Default Value
freight_cost_id	Number	fnd_api.g_miss_num
freight_cost_type_id	Number	fnd_api.g_miss_num
unit_amount	Number	fnd_api.g_miss_num
currency_code	Varchar2(15)	fnd_api.g_miss_char
conversion_date	Date	fnd_api.g_miss_date
conversion_rate	Number	fnd_api.g_miss_num
conversion_type_code	Varchar2(30)	fnd_api.g_miss_char
trip_id	Number	fnd_api.g_miss_num
trip_name	Varchar2(30)	fnd_api.g_miss_char
stop_id	Number	fnd_api.g_miss_num
stop_location_id	Number	fnd_api.g_miss_num
planned_dep_date	Date	fnd_api.g_miss_date
delivery_id	Number	fnd_api.g_miss_num
delivery_name	Varchar2(30)	fnd_api.g_miss_char
delivery_leg_id	Number	fnd_api.g_miss_num
delivery_detail_id	Number	fnd_api.g_miss_num
attribute_category	Varchar2(150)	fnd_api.g_miss_char
attribute1	Varchar2(150)	fnd_api.g_miss_char
attribute2	Varchar2(150)	fnd_api.g_miss_char
attribute3	Varchar2(150)	fnd_api.g_miss_char
attribute4	Varchar2(150)	fnd_api.g_miss_char

Table 4–36 PubFreightCostRecType RECORD DEFINITION

Attribute	Type	Default Value
attribute5	Varchar2(150)	fnd_api.g_miss_char
attribute6	Varchar2(150)	fnd_api.g_miss_char
attribute7	Varchar2(150)	fnd_api.g_miss_char
attribute8	Varchar2(150)	fnd_api.g_miss_char
attribute9	Varchar2(150)	fnd_api.g_miss_char
attribute10	Varchar2(150)	fnd_api.g_miss_char
attribute11	Varchar2(150)	fnd_api.g_miss_char
attribute12	Varchar2(150)	fnd_api.g_miss_char
attribute13	Varchar2(150)	fnd_api.g_miss_char
attribute14	Varchar2(150)	fnd_api.g_miss_char
attribute15	Varchar2(150)	fnd_api.g_miss_char
creation_date	Date	fnd_api.g_miss_date
created_by	Number	fnd_api.g_miss_num
last_update_date	Date	fnd_api.g_miss_date
last_updated_by	Number	fnd_api.g_miss_num
last_update_login	Date	fnd_api.g_miss_date
program_application_id	Number	fnd_api.g_miss_num
program_id	Number	fnd_api.g_miss_num
program_update_date	Date	fnd_api.g_miss_date
request_id	Number	fnd_api.g_miss_num
freight_cost_type	Varchar2(30)	fnd_api.g_miss_char
action_code	Varchar2(30)	fnd_api.g_miss_char

Record Parameter Attribute Validations

freight_cost_id

Should be a valid element of wsh_freight_cost_types.freight_cost_id.

freight_cost_type_id

Should be a valid element of wsh_freight_cost_types.freight_cost_type_id.

unit_amount

Should be a non-negative number.

currency_code

Should be a valid element of fnd_currencies_vl.currency_code.

conversion_rate

Should be a non-negative number.

trip_id

Should be a valid unique element of wsh_trips.trip_id.

trip_name

Should be a valid unique element of wsh_trips.name

stop_id

Should be a valid unique element of wsh_trip_stops.stop_id.

stop_location_id

Should be a valid element of wsh_trip_stops.stop_location_id.

planned_dep_date

Should be a valid element of wsh_trip_stops.planned_departure_date.

delivery_id

Should be a valid unique element of wsh_new_deliveries.delivery_id.

delivery_name

Should be a valid unique element of wsh_new_deliveries.name

delivery_leg_id

Should be a valid element of wsh_delivery_legs.delivery_leg_id.

delivery_detail_id

Should be a valid element of wsh_delivery_details.delivery_detail_id.

freight_cost_type

Should be a valid element of wsh_freight_cost_types.name.

Error Handling

Refer to parameters p_init_msg_list, x_msg_count, and x_msg_data on retrieving error messages.

Pick Release Application Program Interface

The Pick Release Public Application Program Interface (API) is a public API that consists of the following procedures in package WSH_PICKING_BATCHES_PUB:

Create_Batch: Enables you to create a new pick release batch.

Release_Batch: Enables you to release a batch.

This section describes how to use the Pick Release Public API and how it functions in Oracle Shipping Execution.

Note: When pick releasing using the WSH_PICKING_BATCHES_PUB.CREATE_BATCH API, the default value for the attribute append_flag is N. To enable Append Deliveries during pick release from the API, the value of append_flag must be set to Y or NULL.

If append_flag is NULL, and the batch_id is specified, then WSH_PICKING_BATCHES_PUB.CREATE_BATCH will get the value of append_flag from the picking batch.

If append_flag is NULL or Y, then all of the following conditions must be met:

1. The organization_id is specified and append_limit of the organization in the Shipping Parameters window is not Do Not Append.
2. The autocreate_delivery_flag is Y and ac_delivery_criteria is N.
3. The auto_pick_confirm_flag is N.
4. The ship_confirm_rule_id is NULL.
5. The autopack_flag is N.

Also, if you run this API with p_release_mode ONLINE, when calling WSH_PICKING_BATCHES_PUB.Release_Batch, it is not recommended that you run pick release on the same set of delivery lines with append_flag set to Y because may result in unpredictable results. The existing deliveries selected for appending may differ.

Create_Batch API Features

The Create_Batch API enables you to create a new pick release batch.

Functional Overview

This API creates a new pick release batch with a unique batch ID that is then used by the Release_Batch API to release the batch.

Procedure Parameter Descriptions

WSH_PICKING_BATCHES_PUB.CREATE_BATCH

The following chart describes all parameters used by the public procedure WSH_PICKING_BATCHES_PUB.CREATE_BATCH. All of the inbound and outbound parameters are listed. Additional information on these parameters follows.

Table 4–37 WSH_PICKING_BATCHES_PUB.CREATE_BATCH Parameters

Parameter	Usage	Type	Required
p_api_version_number	IN	Number	x
p_init_msg_list	IN	Varchar2	-
p_commit	IN	Varchar2	-
x_return_status	OUT	Varchar2	-
x_msg_count	OUT	Number	-
x_msg_data	Out	Varchar2	-
p_rule_id	IN	Number	-
p_rule_name	IN	Varchar2	-
p_batch_rec	IN	Record	x
p_batch_prefix	IN	Varchar2	-
x_batch_id	OUT	Number	-

p_api_version_number

Compares the incoming API call's version number with the current version number. An error is returned if the version numbers are incompatible.

p_init_msg_list

Requests that the API initialize the message list on your behalf. If the x_msg_count is greater than 1, then the list of messages must be retrieved using the call FND_MSG_PUB.GET. The values are:

- `p_msg_index => I`
- `p_encoded => F`
- `p_data => 1_message`
- `p_msg_index_out => 1_msg_index_out`

where `1_message` and `1_msg_index_out` are local variables of types `Varchar2(2000)` and `Number` respectively.

Default Value: `FND_API.G_FALSE`

p_commit

Requests that the API update information for you after it completes its function.

Default Value: `FND_API.G_FALSE`

x_return_status

Requests that the API return the status of the data after it completes its function.

Valid values include:

- Success: `FND_API.G_RET_STS_SUCCESS`
- Error: `FND_API.G_RET_STS_ERROR`
- Unexpected Error: `FND_API.G_RET_STS_UNEXP_ERROR`

x_msg_count

Indicates number of error messages API has encountered.

x_msg_data

Returns error message text. If the `x_msg_count` is equal to 1, then this contains the actual message.

p_rule_id

Indicates the picking rule identification.

p_rule_name

Indicates the picking rule name.

p_batch_rec

Contains all picking batch parameters.

p_batch_prefix

Used to add a prefix to the batch id.

x_batch_id

Returns the batch id that is created.

Record Parameter Descriptions

PubBatchInfoRecType RECORD DEFINITION

To encapsulate WSH_PICKING_BATCHES table definition and Value column equivalents for ID columns in a PL/SQL record, we define PubBatchInfoRecType and use this to pass Pick Release information to the Create_Batch routine.

Table 4–38 PubBatchInfoRecType RECORD DEFINITION

Attribute	Type
Backorders_Only_Flag	Varchar2(1)
Document_Set_Id	Number
Document_Set_Name	Varchar2(30)
Existing_Rsvs_Only_Flag	Varchar2(1)
Shipment_Priority_Code	Varchar2(30)
Ship_Method_Code	Varchar2(30)
Ship_Method_Name	Varchar2(80)
Customer_Id	Number
Customer_Number	Varchar2(30)
Order_Header_Id	Number
Order_Number	Number
Ship_Set_Id	Number
Ship_Set_Number	Varchar2(30)
Inventory_Item_Id	Number
Order_Type_Id	Number
From_Requested_Date	Date
To_Requested_Date	Date
From_Scheduled_Ship_Date	Date
To_Scheduled_Ship_Date	Date

Table 4–38 PubBatchInfoRecType RECORD DEFINITION

Attribute	Type
Ship_To_Location_Id	Number
Ship_To_Location_code	Varchar2(30)
Ship_From_Location_Id	Number
Ship_From_Location_code	Varchar2(30)
Trip_Name	Varchar2(30)
Delivery_Id	Number
Delivery_Name	Varchar2(30)
Include_Planned_Lines	Varchar2(1)
Pick_Grouping_Rule_Id	Number
Pick_Grouping_Rule_Name	Varchar2(30)
Pick_Sequence_Rule_Id	Number
Pick_Sequence_Rule_Name	Varchar2(30)
Autocreate_Delivery_Flag	Varchar2(1)
Attribute_Category	Varchar2(30)
attribute1	Varchar2(150)
attribute2	Varchar2(150)
attribute3	Varchar2(150)
attribute4	Varchar2(150)
attribute5	Varchar2(150)
attribute6	Varchar2(150)
attribute7	Varchar2(150)
attribute8	Varchar2(150)
attribute9	Varchar2(150)
attribute10	Varchar2(150)
attribute11	Varchar2(150)

Table 4–38 PubBatchInfoRecType RECORD DEFINITION

Attribute	Type
attribute12	Varchar2(150)
attribute13	Varchar2(150)
attribute14	Varchar2(150)
attribute15	Varchar2(150)
Autodetail_Pr_Flag	Varchar2(1)
Trip_Stop_Id	Number
Trip_Stop_location_id	Number
Default_Stage_ Subinventory	Varchar2(10)
Default_Stage_Locator_ Id	Number
Pick_From_ Subinventory	Varchar2(10)
Pick_From_locator_Id	Number
Auto_Pick_Confirm_ Flag	Varchar2(1)
Delivery_Detail_Id	Number
Project_Id	Number
Task_Id	Number
Organization_Id	Number
Organization_Code	Varchar2(3)
Ship_Confirm_Rule_Id	Number
Ship_Confirm_Rule_ Name	Varchar2(30)
Autopack_Flag	Varchar2(1)
Autopack_Level	Number
Task_Planning_Flag	Varchar2(1)
Category_Set_ID	Number
Category_ID	Number

Table 4–38 PubBatchInfoRecType RECORD DEFINITION

Attribute	Type
Ship_Set_Smc_Flag	Varchar2(1)
region_ID	Number
zone_ID	Number
ac_Delivery_Criteria	Varchar2(2)
rel_subinventory	Varchar2(10)
append_flag	Varchar2(1)
task_priority	Number

Record Parameter Attribute Validations

backorders_only_flag

Should be a valid element of wsh_picking_batches.Backorders_Only_Flag. Valid entries are I, E, and O.

document_set_id

Should be a valid element of wsh_picking_batches.Document_Set_Id.

document_set_name

Should be a valid element of wsh_report_sets.name.

existing_rsvs_only_flag

Should be a valid element of wsh_picking_batches.Existing_Rsvs_Only_Flag.

shipment_priority_code

Should be a valid element of oe_lookups.lookup_code.

ship_method_code

Should be a valid unique element of wsh_picking_batches.Ship_Method_Code.

ship_method_name

Should be a valid unique element of fnd_lookup_values_vl.meaning.

customer_id

Should be a valid unique element of wsh_picking_batches.Customer_Id.

customer_number

Should be a valid element of ra_customers.customer_number.

order_header_id

Should be a valid element of wsh_picking_batches.Order_Header_Id.

order_number

Should be a valid unique element of oe_order_headers_all.Order_Number.

ship_set_id

Should be a valid unique element of wsh_picking_batches.Ship_Set_Number

ship_set_number

Should be a valid element of oe_sets.set_name.

inventory_item_id

Should be a valid element of wsh_picking_batches.Inventory_Item_Id.

order_type_id

Should be a valid element of wsh_picking_batches.Order_Type_Id.

order_type_name

Should be a valid element of oe_transaction_types_tl.name.

from_requested_date

Starting request date to release.

to_requested_date

Ending request date to release.

from_scheduled_ship_date

Starting schedule date to release.

to_scheduled_ship_date

Ending schedule date to release.

ship_to_location_id

Should be a valid element of wsh_picking_batches.Ship_To_Location_Id.

ship_to_location_code

Should be a valid element of hr_locations_all_tl.location_code.

ship_from_location_id

Should be a valid unique element of wsh_picking_batches.Ship_From_Location_Id.

ship_from_location_code

Should be a valid unique element of hr_locations_all_tl.location_code.

trip_id

Should be a valid element of wsh_picking_batches.Trip_Id.

trip_name

Should be a valid element of wsh_trips.name.

delivery_id

Should be a valid element of wsh_picking_batches.Delivery_Id.

delivery_name

Should be a valid element of wsh_new_deliveries.name.

include_planned_lines

Should be a valid unique element of wsh_picking_batches.Include_Planned_Lines.

pick_grouping_rule_id

Should be a valid unique element of wsh_picking_batches.Pick_Grouping_Rule_Id.

pick_grouping_rule_name

Should be a valid element of wsh_pick_grouping_rules.name.

pick_sequence_rule_id

Should be a valid element of wsh_picking_batches.Pick_Sequence_Rule_Id.

pick_sequence_rule_name

Should be a valid element of wsh_pick_sequence_rules.name.

auto_create_delivery_flag

Should be a valid element of wsh_picking_batches.Autocreate_Delivery_Flag.

attribute_category

Should be a valid unique element of wsh_picking_batches.Attribute_Category.

autodetail_pr_flag

Should be a valid unique element of wsh_picking_batches.Autodetail_Pr_Flag.

trip_stop_id

Should be a valid element of wsh_picking_batches.Trip_Stop_Id.

trip_stop_location_id

Should be a valid element of wsh_trip_stops.Stop_Id.

default_stage_subinventory

Should be a valid element of wsh_picking_batches.Default_Stage_Subinventory.

default_stage_locator_id

Should be a valid element of wsh_picking_batches.Default_Stage_Locator_Id.

pick_from_subinventory

Should be a valid unique element of wsh_picking_batches.Pick_From_Subinventory.

pick_from_locator_id

Should be a valid unique element of wsh_picking_batches.Pick_From_locator_Id.

auto_pick_confirm_flag

Should be a valid element of wsh_picking_batches.Auto_Pick_Confirm_Flag. Valid entries are Y or N.

delivery_detail_id

Should be a valid element of wsh_picking_batches.Delivery_Detail_Id.

project_id

Should be a valid element of wsh_picking_batches.Project_Id.

task_id

Should be a valid element of wsh_picking_batches.Task_Id.

organization_id

Should be a valid unique element of wsh_picking_batches.Organization_Id.

organization_code

Should be a valid unique element of org_organization_definitions.organization_code.

ship_confirm_rule_id

Should be a valid element of wsh_picking_batches.Ship_Confirm_Rule_Id.

ship_confirm_rule_name

Should be a valid element of wsh_Ship_Confirm_rules.name.

autopack_flag

Should be a valid element of wsh_picking_batches.Autopack_Flag.

autopack_level

Should be a valid element of wsh_picking_batches.Autopack_Level. Valid entries are 0, 1, or 2.

task_planning_flag

Should be a valid unique element of wsh_picking_batches.Task_Planning_Flag.

category_set_id

Should be a valid unique element of wsh_picking_batches.Category_Set_ID.

category_id

Should be a valid element of wsh_picking_batches.Category_ID.

ship_set_smc_flag

Should be a valid element of wsh_picking_batches.Ship_Set_Smc_Flag.

region_id

Should be a valid element of wsh_picking_batches.region_id.

zone_id

Should be a valid element of wsh_picking_batches.zone_id.

ac_delivery_criteria

Should be a valid unique element of wsh_picking_batches.ac_delivery_criteria.

rel_subinventory

Should be a valid unique element of wsh_picking_batches.rel_subinventory.

append_flag

Should be a valid element of wsh_picking_batches.append_flag.

task_priority

Should be a valid element of wsh_picking_batches.task_priority.

Error Handling

Refer to parameters p_init_msg_list, x_msg_count, and x_msg_data on retrieving error messages.

Release_Batch API Features

The Release_Batch API enables you to release a newly created pick release batch.

Functional Overview

This API enables you to release a pick release batch that was generated by the Create_Batch API.

Procedure Parameter Descriptions

WSH_PICKING_BATCHES_PUB.RELEASE_BATCH

The following chart describes all parameters used by the public procedure WSH_PICKING_BATCHES_PUB.RELEASE_BATCH. All of the inbound and outbound parameters are listed. Additional information on these parameters follows.

Table 4–39 WSH_PICKING_BATCHES_PUB.RELEASE_BATCH Parameters

Parameter	Usage	Type	Required
p_api_version_number	IN	Number	x
p_init_msg_list	IN	Varchar2	-
p_commit	IN	Varchar2	-
x_return_status	OUT	Varchar2	-
x_msg_count	OUT	Number	-
x_msg_data	OUT	Varchar2	-
p_batch_id	IN	Number	-
p_batch_name	IN	Varchar2	-
p_log_level	IN	Number	-
p_release_mode	IN	Varchar2	-
x_request_id	OUT	Number	-

p_api_version_number

Compares the incoming API call's version number with the current version number. An error is returned if the version numbers are incompatible.

p_init_msg_list

Requests that the API initialize the message list on your behalf. If the x_msg_count is greater than 1, then the list of messages must be retrieved using the call FND_MSG_PUB.GET. The values are:

- p_msg_index => I
- p_encoded => F
- p_data => 1_message
- p_msg_index_out => 1_msg_index_out

where 1_message and 1_msg_index_out are local variables of types Varchar2(2000) and Number respectively.

Default Value: FND_API.G_FALSE

p_commit

Requests that the API update information for you after it completes its function.

Default Value: FND_API.G_FALSE

x_return_status

Requests that the API return the status of the data after it completes its function.

Valid values include:

- Success: FND_API.G_RET_STS_SUCCESS
- Error: FND_API.G_RET_STS_ERROR
- Unexpected Error: FND_API.G_RET_STS_UNEXP_ERROR

x_msg_count

Indicates number of error messages API has encountered.

x_msg_data

Returns error message text. If the x_msg_count is equal to 1, then this contains the actual message.

p_batch_id

Indicates the picking batch identification which is used to get the batch information from wsh_picking_batches.

p_batch_name

Indicates the picking batch name which is used to get the batch information from wsh_picking_batches.

p_log_level

Controls the log message generated by the concurrent pick release process.

p_release_mode

Controls the release mode of either Concurrent or Online. Default = Concurrent.

x_request_id

Returns the request id for the concurrent pick release request.

Migration from Open Interfaces to Public APIs

Overview

This section provides information about migrating your shipment process from the Releases 10.7 and 11 Delivery-based Ship Confirm Open Interface to the Release 11*i* public APIs. It contains the following:

- A description of the Release 11 Ship Confirm open interface process.
- A description of the Release 11*i* ship confirm process using APIs.
- Mapping details from Release 11 open interface to Release 11*i* APIs.
- An illustration of the Release 11*i* APIs.

Releases 10.7 and 11 Ship Confirm Open Interface

The Delivery-based Ship Confirm open interface loads external shipping data into Oracle Shipping Execution tables and closes the delivery without using the Ship Confirm-Delivery or Ship Confirm-Departure user interfaces.

The Delivery-based Ship Confirm open interface process is:

1. Enter data in the delivery, packed containers, picking line details, and freight charges interface tables via an external device, for example, a bar code reader.
2. Run the Delivery-based Ship Confirm Open Interface concurrent process from the Standard Request Submission user interface.
3. Check the interface tables for errors during processing then modify and resubmit them or remove them. The open interface removes successful transactions from the interface tables and leaves transactions with errors.

Release 11*i* Ship Confirm Using Public APIs

Shipping public APIs load external shipping data into Oracle Shipping Execution tables and close the delivery without using the Shipping Transaction Form.

The Shipping Execution public API process is :

1. Call WSH_DELIVERY_DETAILS_PUB.UPDATE_SHIPPING_ATTRIBUTES to modify data in WSH DELIVERY DETAILS. Setting the parameters of this API is similar to populating the WSH_PICKING_DETAILS_INTERFACE table in Release 11.

- 2. Call WSH_FREIGHT_COSTS_PUB.CREATE_UPDATE_FREIGHT_COSTS to create or update freight cost records in WSH_FREIGHT_COSTS. Setting the parameters of this API is similar to populating the WSH_FREIGHT_CHARGES_INTERFACE table in Release 11.
- 3. Call WSH_CONTAINER_PUB.CREATE_CONTAINERS to update the container information. Setting the parameters of this API is similar to populating the WSH_PACKED_CONTAINER_INTERFACE table.
- 4. Call WSH_DELIVERIES_PUB.CONFIRM_DELIVERY to carry out different actions on the delivery and WSH_DELIVERIES_PUB.CREATE_UPDATE_DELIVERY to update the delivery information. Setting the parameters of these APIs is similar to populating the WSH_DELIVERIES_INTERFACE table.

This table relates Release 11 open interface tables to Release 11*i* public APIs.

Table 4–40 Open Interface and Public API Comparison

Release 11 Interface Tables	Release 11/ Public API
WSH_PICKING_DETAILS_INTERFACE	WSH_DELIVERY_DETAILS_PUB.UPDATE_SHIPPING_ATTRIBUTES
WSH_FREIGHT_CHARGES_INTERFACE	WSH_FREIGHT_COSTS_PUB.CREATE_UPDATE_FREIGHT_COSTS
WSH_PACKED_CONTAINER_INTERFACE	WSH_CONTAINER_PUB.CREATE_CONTAINERS
WSH_DELIVERIES_INTERFACE	WSH_DELIVERIES_PUB.DELIVERY_ACTION and WSH_DELIVERIES_PUB.CREATE_UPDATE_DELIVERY

In Release 11*i*, you do not need to populate any interface tables or to launch any concurrent program to load data into interface tables.

WSH_CONTAINER_PUB.CREATE_CONTAINERS creates a container (LPN). To pack a delivery detail with a container call WSH_CONTAINER_PUB.container_actions.

Ship Confirm Mapping

This section details mapping of Release 11 DBSCOI interface tables and columns to the Release 11*i* Shipping public API parameters.

WSH_DELIVERY_DETAILS_PUB.UPDATE_SHIPPING_ATTRIBUTES

WSH_DELIVERY_DETAILS_PUB.UPDATE_SHIPPING_ATTRIBUTES is called from the source system to modify data in WSH_DELIVERY_DETAILS and is detailed in another section of this chapter.

A key parameter is p_changed_attributes which:

- It is of type WSH_DELIVERY_DETAILS_PUB.ChangedAttributesTabType
- Is a table of ChangedAttributesRecType indexed by Binary_Integer.
- Specifies the modifications for the WSH_DELIVERY_DETAILS table.

This chart shows the mapping between the columns in WSH_PICKING_DETAILS_INTERFACE table (from Release 11) and the parameters of ChangedAttributesRecType.

Table 4–41 UPDATE_SHIPPING_ATTRIBUTES Mapping

WSH_PICKING_DETAILS_INTERFACE Column	ChangedAttributesRecType Parameters
TRANSACTION_ID	-
PROCESS_FLAG	-
PICKING_LINE_DETAIL_ID	DELIVERY_DETAIL_ID
SOURCE_CODE	-
SOURCE_HEADER_ID	SOURCE_HEADER_ID
TRANSACTION_MODE	-
TRANSACTION_TYPE_ID	-
INVENTORY_ITEM_ID	-
WAREHOUSE_ID	-
SUBINVENTORY	SUBINVENTORY
LOT_NUMBER	LOT_NUMBER
REVISION	REVISION
LOCATOR_ID	LOCATOR_ID
SERIAL_NUMBER	SERIAL_NUMBER
SHIPPED_QUANTITY	SHIPPED_QUANTITY

Table 4–41 UPDATE_SHIPPING_ATTRIBUTES Mapping

WSH_PICKING_DETAILS_INTERFACE Column	ChangedAttributesRecType Parameters
-	CYCLE_COUNT_QUANTITY (This specifies the backorder quantity at ship confirm; the quantity not shipped backordered becomes staged. This must be added when you enter Shipped Quantity).
TRANSACTION_UOM	-
TRANSACTION_DATE	-
ATTRIBUTE_CATEGORY	ATTRIBUTE_CATEGORY
ATTRIBUTE1-15	ATTRIBUTE1-15
CREATED_BY	-
CREATION_DATE	-
LAST_UPDATED_BY	-
LAST_UPDATE_DATE	-
LAST_UPDATE_LOGIN	-
ERROR_EXPLANATION	-
ERROR_CODE	-
CONTAINER_SEQUENCE	-
CONTAINER_ID	-

WSH_DELIVERIES_PUB.DELIVERY_ACTION

WSH_DELIVERIES_PUB.DELIVERY_ACTION is called to carry out different actions on a delivery and is described in another section of this chapter. The important parameters specify the action to be performed on the delivery; they are p_action_code and p_sc_action_flag.

WSH_DELIVERIES_INTERFACE

WSH_DELIVERIES_INTERFACE has some columns which indicate the action to be performed on the delivery and some columns which update the delivery information, such as Waybill Number and attributes.

WSH_DELIVERIES_PUB.DELIVERY_ACTION and CREATE_UPDATE_DELIVERY

WSH_DELIVERIES_PUB.DELIVERY_ACTION is to be called to perform an action on the delivery and WSH_DELIVERIES_PUB.CREATE_UPDATE_DELIVERY is called to update the delivery information. Parameter p_delivery_info of CREATE_UPDATE_DELIVERY specifies the updates for WSH_NEW_DELIVERIES.

This chart shows the mapping between the columns in WSH_DELIVERIES_INTERFACE table (from Release 11) and the parameters to the procedure WSH_DELIVERIES_PUB.DELIVERY_ACTION.

Table 4–42 DELIVERY_ACTION Mapping

WSH_DELIVERIES_INTERFACE	WSH_DELIVERIES_PUB.DELIVERY_ACTION Parameters
TRANSACTION_ID	-
PROCESS_FLAG	-
REQUEST_ID	-
DELIVERY_ID	P_delivery_id
DELIVERY_NAME	P_delivery_name
ORGANIZATION_ID	-
ORGANIZATION_CODE	-
ACTION_CODE	P_action_code, p_sc_action_flag
REPORT_SET_ID	P_sc_report_set_id
REPORT_SET	P_sc_report_set_name
DATE_CLOSED	P_sc_actual_dep_date
SEQUENCE_NUMBER	-
CUSTOMER_ID	-
CUSTOMER_NUMBER	-

This chart shows the mapping between the columns in WSH_DELIVERIES_INTERFACE table (from Release 11) and the parameters to the procedure WSH_DELIVERIES_PUB.CREATE_UPDATE_DELIVERY.

Table 4–43 CREATE_UPDATE_DELIVERY Mapping

WSH_DELIVERIES_INTERFACE	p_delivery_info (DELIVERY_PUB_REC_TYPE) Attribute
LOADING_ORDER_FLAG	Loading_order_flag
LOADING_ORDER_DESC	Loading_order_desc
FREIGHT_CARRIER_CODE	Carrier_code
ULTIMATE_SHIP_TO_ID	Ultimate_dropoff_location_id
FREIGHT_TERMS_CODE	Freight_terms_code
CURRENCY_CODE	Currency_code
CURRENCY_NAME	Currency_name
FOB_CODE	Fob_code
ERROR_EXPLANATION	-
ERROR_CODE	-
ACTION_CODE	-
INTERMEDIATE_SHIP_TO_ID	Intmed_ship_to_location_id
POOLED_SHIP_TO_ID	Pooled_ship_to_location_id
WAYBILL_NUM	Waybill
GROSS_WEIGHT	Gross_weight
WEIGHT_UOM_CODE	Weight_uom_code
WEIGHT_UNIT_OF_MEASURE	-
VOLUME	Volume
VOLUME_UOM_CODE	Volume_uom_code
VOLUME_UNIT_OF_MEASURE	-
PICKED_BY_ID	-
ATTRIBUTE_CATEGORY	Attribute_category
ATTRIBUTE1-15	Attribute1-15
GLOBAL_ATTRIBUTE_CATEGORY	global_attribute_category
GLOBAL_ATTRIBUTE1-15	Global_attribute1-15

Action Codes

This chart shows the mapping between the Release 11 action codes and the Release 11i public API parameters of WSH_DELIVERIES_PUB.DELIVERY_ACTION.

Table 4–44 WSH_DELIVERIES_PUB.DELIVERY_ACTION Parameters

R11 Action Code	R11 Description	R11i P_action_code	R11i P_sc_action_flag	R11i Other Parameters
NULL	Do not perform ship confirm	-	-	-
1	PACK ENTERED , UNASSIGN REMAINDER	PACK	-	-
2	PACK ALL, UNASSIGN REMAINDER	PACK	-	-
3	SHIP ENTERED, UNASSIGN REMAINDER	CONFIRM	B	cycle_count_quantity (of ChangedAttributesRecType) = 0 (Refer to the UPDATE_SHIPPING_ATTRIBUTES Mapping table for details on cycle_count_quantity), p_sc_stage_delivery = N
4	SHIP ALL, UNASSIGN REMAINDER	CONFIRM	A	-
5	BACKORDER ALL	CONFIRM	C	-
6	SHIP ENTERED, CLOSE DEPARTURE	CONFIRM	B	cycle_count_quantity (of ChangedAttributesRecType) = 0, p_sc_stage_delivery = N, p_sc_close_trip_flag = Y

Table 4–44 WSH_DELIVERIES_PUB.DELIVERY_ACTION Parameters

R11 Action Code	R11 Description	R11/ P_action_ code	R11/ P_sc_action_ flag	R11/ Other Parameters
7	SHIP ALL, CLOSE DEPARTURE	CONFIRM	A	p_sc_close_trip_flag = Y
8	BACKORDER ALL, CLOSE DEPARTURE	CONFIRM	C	p_sc_close_trip_flag = Y
9	PACK ENTERED	-	-	-
10	PACK ALL	PACK	-	-
11	SHIP ENTERED , BACKORDER REMAINDER	CONFIRM	B	-
12	SHIP ALL, BACKORDER REMAINDER	CONFIRM	A	-
13	SHIP ENTERED, CLOSE DEPARTURE, BACKORDER REMAINDER	CONFIRM	B	p_sc_close_trip_flag= Y

In Release 11, if you populate data only in WSH_DELIVERIES_INTERFACE with action code 3, the entire quantity in all the WSH_PICKING_DETAILS is backordered. If the shipped quantity is not specified in WSH_PICKING_DETAILS_INTERFACE, the entire quantity is backordered.

In Release 11*i*, you can control the action to be taken on unspecified quantities. If there is no call to UPDATE_SHIPPING_ATTRIBUTES before confirming the delivery, the shipped quantity is not specified. However, you can control the actions taken on unspecified quantities through the parameter p_sc_action_flag of WSH_DELIVERIES_PUB.DELIVERY_ACTION.

The data model for containers in Release 11 and Release 11*i* is different. In Release 11, the packed containers have an entry in WSH_PACKED_CONTAINERS with the quantity. In Release 11*i*, the container information is stored in WSH_DELIVERY_DETAILS and this table does not have a quantity field. If a delivery details requires two containers then the process creates two container instances and splits the delivery details.

In Release 11*i*, the auto-packing feature does packing based on the defaults. For more information, see *Oracle Shipping Execution User's Guide*.

In Release 11, the data elements `ultimate_ship_to_id`, `intermediate_ship_to_id`, and `pooled_ship_to_id_locations` are the site use IDs. In Release 11*i*, these data elements correspond to the location IDs and you need to pass the location IDs. Generally, ship from (internal) locations have `location_code` and ship to (external) locations need `location_id`.

Example of Shipping Execution Public APIs

This example shows:

- A delivery detail for which we ship one and backorder the remainder.
- A delivery detail for which we backorder the entire quantity.backordered.
- A delivery detail for which we ship partially and unassign the remaining quantity without backordering (action code 3).

Package Specification

```
Create or replace package WSH_SHIP_API as
procedure SHIP_CONFIRM_EXAMPLE1(
  x_return_status OUT VARCHAR2,
  x_msg_count OUT NUMBER,
  x_msg_data OUT VARCHAR2);
END WSH_SHIP_API;
/
```

Package Body

```
Create or replace package body WSH_SHIP_API as
/*
```

Example 1:

This procedure can be used as an example on how to ship confirm a delivery consisting of delivery details that have already been pick released and assigned to a delivery.

Call `WSH_DELIVERY_DETAILS_PUB.Update_Shipping_Attributes` API to update the corresponding delivery details to ship partial quantity and backorder the remaining qty.

Call `WSH_FREIGHT_COSTS_PUB.Create_Update_Freight_Costs` API to create freight costs for the delivery that the details have been assigned to.

Call `WSH_CONTAINER_PUB.Auto_Pack` to pack the delivery lines into a container.

Call `WSH_DELIVERIES_PUB.Delivery_Action` API to ship confirm the delivery.

```
*/
```

```

procedure SHIP_CONFIRM_EXAMPLE(
    x_return_status OUT VARCHAR2,
    x_msg_count OUT NUMBER,
    x_msg_data OUT VARCHAR2) IS
/*
--Standard Parameters.
p_api_version_number NUMBER;
init_msg_list VARCHAR2(30);
x_msg_details VARCHAR2(3000);
x_msg_summary VARCHAR2(3000);
p_validation_level NUMBER;
p_commit VARCHAR2(30);

--Parameters for WSH_DELIVERY_DETAILS_PUB.Update_Shipping_Attributes.
source_code VARCHAR2(15);
changed_attributes WSH_DELIVERY_DETAILS_PUB.ChangedAttributeTabType;

--Parameters for WSH_CONTAINER_PUB.Auto_Pack.
p_entity_tab WSH_UTIL_CORE.id_tab_type;
p_entity_type VARCHAR2(30);
p_group_id_tab WSH_UTIL_CORE.id_tab_type;
p_pack_cont_flag varchar2(30);
x_cont_inst_tab WSH_UTIL_CORE.id_tab_type;

--Parameters for WSH_FREIGHT_COSTS_PUB.Create_Update_Freight_Costs.
action_code VARCHAR2(15);
pub_freight_costs WSH_FREIGHT_COSTS_PUB.PubFreightCostRecType;
freight_cost_id NUMBER;

--Parameters for WSH_DELIVERIES_PUB.Delivery_Action.
p_action_code VARCHAR2(15);
p_delivery_id NUMBER;
p_delivery_name VARCHAR2(30);
p_asg_trip_id NUMBER;
p_asg_trip_name VARCHAR2(30);
p_asg_pickup_stop_id NUMBER;
p_asg_pickup_loc_id NUMBER;
p_asg_pickup_loc_code VARCHAR2(30);
p_asg_pickup_arr_date DATE;
p_asg_pickup_dep_date DATE;
p_asg_dropoff_stop_id NUMBER;
p_asg_dropoff_loc_id NUMBER;
p_asg_dropoff_loc_code VARCHAR2(30);
p_asg_dropoff_arr_date DATE;

```

```
p_asg_dropoff_dep_date DATE;
p_sc_action_flag VARCHAR2(10);
p_sc_close_trip_flag VARCHAR2(10);
p_sc_create_bol_flag VARCHAR2(10);
p_sc_stage_del_flag VARCHAR2(10);
p_sc_trip_ship_method VARCHAR2(30);
p_sc_actual_dep_date VARCHAR2(30);
p_sc_report_set_id NUMBER;
p_sc_report_set_name VARCHAR2(60);
p_wv_override_flag VARCHAR2(10);
x_trip_id VARCHAR2(30);
x_trip_name VARCHAR2(30);

/*Handle exceptions*/
fail_api EXCEPTION;

BEGIN

/* Initialize return status*/
x_return_status := WSH_UTIL_CORE.G_RET_STS_SUCCESS;

/* Call this procedure to initialize applications parameters. To determine
parameter values, refer to the Application Paramater Initialization section of
this chapter. */
FND_GLOBAL.APPS_INITIALIZE(user_id => 1001594
                           ,resp_id =>52892
                           ,resp_appl_id =>660);

/* Values for updating delivery details to ship the partial qty.
*/

source_code := 'OE'; -- The only source code that should be used by the API

/* The values to be set for action_code 11*/
changed_attributes(1).delivery_detail_id := 13431;
changed_attributes(1).shipped_quantity := 1;

/* The second Delivery Detail that needs to be backordered */
changed_attributes(2).delivery_detail_id := 13432;
changed_attributes(2).shipped_quantity := 0;

/* The corresponding values to be set for action_code 3 are
changed_attributes(1).delivery_detail_id := <del_detail_id>;
```

```
changed_attributes(1).shipped_quantity := 1;
changed_attributes(1).cycle_count_quantity := 0;
*/

--Call to WSH_DELIVERY_DETAILS_PUB.Update_Shipping_Attributes.

WSH_DELIVERY_DETAILS_PUB.Update_Shipping_Attributes(
    p_api_version_number => 1.0,
    p_init_msg_list => init_msg_list,
    p_commit => p_commit,
    x_return_status => x_return_status,
    x_msg_count => x_msg_count,
    x_msg_data => x_msg_data,
    p_changed_attributes => changed_attributes,
    p_source_code => source_code);

if (x_return_status <> WSH_UTIL_CORE.G_RET_STS_SUCCESS) then
    raise fail_api;
end if;

/*Assign Values to the Parameters for creating the freight costs.*/

pub_freight_costs.freight_cost_type_id := 1;
pub_freight_costs.unit_amount := 20;
pub_freight_costs.currency_code := 'USD';
pub_freight_costs.delivery_id := 5341;

--Call to WSH_FREIGHT_COSTS_PUB.Create_Update_Freight_Costs.

WSH_FREIGHT_COSTS_PUB.Create_Update_Freight_Costs(
    p_api_version_number => 1.0,
    p_init_msg_list => init_msg_list,
    p_commit => p_commit,
    x_return_status => x_return_status,
    x_msg_count => x_msg_count,
    x_msg_data => x_msg_data,
    p_pub_freight_costs => pub_freight_costs,
    p_action_code => 'CREATE',
    x_freight_cost_id => freight_cost_id);

if (x_return_status <> WSH_UTIL_CORE.G_RET_STS_SUCCESS) then
    raise fail_api;
end if;
```

```

/* Values for autopacking the delivery details to a container.
*/
p_entity_tab(1) := 13431;
p_entity_tab(2) := 13432;
p_entity_tab(3) := 13434;
--Call to WSH_CONTAINER_PUB.Auto_Pack
WSH_CONTAINER_PUB.Auto_Pack(
p_api_version => 1.0,
p_init_msg_list => init_msg_list,
p_commit => p_commit,
p_validation_level => p_validation_level,
x_return_status => x_return_status,
x_msg_count => x_msg_count,
x_msg_data => x_msg_data,
p_entity_tab => p_entity_tab,
p_entity_type => 'L',
p_group_id_tab => p_group_id_tab,
p_pack_cont_flag => p_pack_cont_flag,
x_cont_inst_tab => x_cont_inst_tab);
if (x_return_status <> WSH_UTIL_CORE.G_RET_STS_SUCCESS) then
raise fail_api;
end if;

/* Values for Ship Confirming the delivery.
*/
p_action_code := 'CONFIRM'; -- The action code for ship confirm
p_delivery_id := 5341; -- The delivery that needs to be confirmed
p_delivery_name := '5341'; -- The delivery name,
p_sc_action_flag := 'S'; -- Ship entered quantity.

/* Values to be set for action code 3 are
*/
p_action_code := 'CONFIRM'; -- The action code for ship confirm
p_delivery_id := <delivery_id>; -- The delivery that needs to be confirmed
p_delivery_name := <delivery_name>; -- The delivery name,
p_sc_action_flag := 'S'; -- Ship entered quantity.
p_sc_stage_del_flag := 'N'; --
p_sc_trip_ship_method := 'GROUND'; -- The ship method code

/*Call to WSH_DELIVERIES_PUB.Delivery_Action. */

WSH_DELIVERIES_PUB.Delivery_Action(

```

```

p_api_version_number => 1.0,
p_init_msg_list => init_msg_list,
x_return_status => x_return_status,
x_msg_count => x_msg_count,
x_msg_data => x_msg_data,
p_action_code => p_action_code,
p_delivery_id => p_delivery_id,
p_delivery_name => p_delivery_name,
p_asg_trip_id => p_asg_trip_id,
p_asg_trip_name => p_asg_trip_name,
p_asg_pickup_stop_id => p_asg_pickup_stop_id,
p_asg_pickup_loc_id => p_asg_pickup_loc_id,
p_asg_pickup_loc_code => p_asg_pickup_loc_code,
p_asg_pickup_arr_date => p_asg_pickup_arr_date,
p_asg_pickup_dep_date => p_asg_pickup_dep_date,
p_asg_dropoff_stop_id => p_asg_dropoff_stop_id,
p_asg_dropoff_loc_id => p_asg_dropoff_loc_id,
p_asg_dropoff_loc_code => p_asg_dropoff_loc_code,
p_asg_dropoff_arr_date => p_asg_dropoff_arr_date,
p_asg_dropoff_dep_date => p_asg_dropoff_dep_date,
p_sc_action_flag => p_sc_action_flag,
p_sc_close_trip_flag => p_sc_close_trip_flag,
p_sc_create_bol_flag => p_sc_create_bol_flag,
p_sc_stage_del_flag => p_sc_stage_del_flag,
p_sc_trip_ship_method => p_sc_trip_ship_method,
p_sc_actual_dep_date => p_sc_actual_dep_date,
p_sc_report_set_id => p_sc_report_set_id,
p_sc_report_set_name => p_sc_report_set_name,
p_wv_override_flag => p_wv_override_flag,
x_trip_id => x_trip_id,
x_trip_name => x_trip_name);
if (x_return_status <> WSH_UTIL_CORE.G_RET_STS_SUCCESS) then
raise fail_api;
end if;

exception
when fail_api then
WSH_UTIL_CORE.get_messages('Y', x_msg_summary, x_msg_details, x_msg_count);
if x_msg_count > 1 then
    x_msg_data := x_msg_summary || x_msg_details;
else
    x_msg_data := x_msg_summary;
end if;
END SHIP_CONFIRM_EXAMPLE;
```

Oracle Advanced Pricing Open Interfaces

This chapter contains information about the following Oracle Advanced Pricing open interfaces and application program interfaces (APIs):

- [Agreements Public Application Program Interface](#) on page 5-3
- [Attribute Mapping Application Program Interface](#) on page 5-49
- [Business Object for Modifier Setup Application Program Interface](#) on page 5-60
- [Business Object for Pricing Formulas Application Program Interface](#) on page 5-133
- [Business Object for Pricing Limits Application Program Interface](#) on page 5-157
- [Get Currency Application Program Interface](#) on page 5-177
- [Get Custom Price Application Program Interface](#) on page 5-179
- [Get Price List Application Program Interface](#) on page 5-185
- [Multi-Currency Conversion Setup Application Program Interface](#) on page 5-187
- [Price List Setup Application Program Interface](#) on page 5-199
- [Price Request Application Program Interface](#) on page 5-255
- [QP_ATTRIBUTES_PUB Application Program Interface](#) on page 5-293
- [QP_ATTR_MAPPING_PUB Application Program Interface](#) on page 5-303
- [Qualifiers Application Program Interface](#) on page 5-308
- [Reverse Limits Application Program Interface](#) on page 5-328
- [Round Price Application Program Interface](#) on page 5-333
- [Validate_Price_list_Curr_code Application Program Interface](#) on page 5-336

Key of Short Names

A key of the short names and definitions used in the API tables are provided in the following table:

Table 5–1 Key of Short Names

Short name	Definition
Drv	Derived
Req	Required Yes this is a required parameter No this is an optional parameter
N/A (no entry)	No value/not applicable

Agreements Public Application Program Interface

This section explains how to use the Agreements Public API and how it functions in Oracle Advanced Pricing. The Agreements Public package consists of entities to support creating and maintaining agreements.

Functional Overview

Process_Agreement processes inserts, updates, and deletes records related to agreements.

Setting Up and Parameter Descriptions

The following tables describe all parameters used by the public Agreements Public API. All of the inbound and outbound parameters are listed. Additional information on these parameters follows.

PROCESS_AGREEMENT

The following table shows the parameters for this structure.

Table 5–2 PROCESS_AGREEMENT Parameters

Parameter	Usage	Type	Req	Drv
p_api_version_number	IN	Number	No	No
p_init_msg_list	IN	Varchar2	No	No
p_return_values	IN	Varchar2	No	No
p_commit	IN	Varchar2	No	No
x_return_status	OUT	Varchar2	No	No
x_msg_count	OUT	Number	No	No
x_msg_data	OUT	Varchar2	No	No
p_Agreement_rec	IN	Agreement_Rec_Type	No	No
p_Agreement_value_rec	IN	Agreement_Val_Rec_Type	No	No
p_Price_LHeader_rec	IN	QP_Price_List_PUB.Price_List_Rec_Type	No	No
p_Price_LHeader_val_rec	IN	QP_Price_List_PUB.Price_List_Val_Rec_Type	No	No
p_Price_LLine_tbl	IN	QP_Price_List_PUB.Price_List_Line_Tbl_Type	No	No
p_Price_LLine_val_tbl	IN	QP_Price_List_PUB.Price_List_Val_Tbl_Type	No	No

Table 5–2 PROCESS_AGREEMENT Parameters

Parameter	Usage	Type	Req	Drv
p_Pricing_Attr_tbl	IN	QP_Price_List_PUB.Pricing_Attr_Tbl_Type	No	No
p_Pricing_Attr_val_tbl	IN	QP_Price_List_PUB.Pricing_Attr_Val_Tbl_Type	No	No
x_Agreement_rec	OUT	Agreement_Rec_Type	No	No
x_Agreement_val_rec	OUT	Agreement_Val_Rec_Type	No	No
x_Price_LHeader_rec	OUT	QP_Price_List_PUB.Price_List_Rec_Type	No	No
x_Price_LHeader_val_rec	OUT	QP_Price_List_PUB.Price_List_Val_Rec_Type	No	No
x_Price_LLine_tbl	OUT	QP_Price_List_PUB.Price_List_Line_Tbl_Type	No	No
x_Price_LLine_val_tbl	OUT	QP_Price_List_PUB.Price_List_Line_Val_Tbl_Type	No	No
x_Pricing_Attr_tbl	OUT	QP_Price_List_PUB.Pricing_Attr_Tbl_Type	No	No
x_Pricing_Attr_val_tbl	OUT	QP_Price_List_PUB.Pricing_Attr_Val_Tbl_Type	No	No

AGREEMENT_REC_TYPE

The following table shows the parameters for this structure.

api_center_ag_01
Table 5–3 AGREEMENT_REC_TYPE

Parameter	Usage	Type	Req	Drv
accounting_rule_id	Null	Number	No	No
agreement_contact_id	Null	Number	No	No
agreement_id	Null	Number	Yes ¹	No
agreement_num	Null	Varchar2(50)	No	No
agreement_source_code	Null	Varchar2(30)	No	No
agreement_type_code	Null	Varchar2(30)	No	No
attribute1	Null	Varchar2(150)	No	No
attribute2	Null	Varchar2(150)	No	No
attribute3	Null	Varchar2(150)	No	No
attribute4	Null	Varchar2(150)	No	No
attribute5	Null	Varchar2(150)	No	No

Table 5–3 AGREEMENT_REC_TYPE

Parameter	Usage	Type	Req	Drv
attribute6	Null	Varchar2(150)	No	No
attribute7	Null	Varchar2(150)	No	No
attribute8	Null	Varchar2(150)	No	No
attribute9	Null	Varchar2(150)	No	No
attribute10	Null	Varchar2(150)	No	No
attribute11	Null	Varchar2(150)	No	No
attribute12	Null	Varchar2(150)	No	No
attribute13	Null	Varchar2(150)	No	No
attribute14	Null	Varchar2(150)	No	No
attribute15	Null	Varchar2(150)	No	No
comments	Null	Varchar2(30)	No	No
context	Null	Varchar2(30)	No	No
created_by	Null	Number	Yes	No
creation_date	Null	Date	Yes	No
sold_to_org_id	Null	Number	No	No
end_date_active	Null	Date	No	No
freight_terms_code	Null	Varchar2(30)	No	No
invoice_contact_id	Null	Number	No	No
invoice_to_org_id	Null	Number	No	No
invoicing_rule_id	Null	Number	No	No
last_updated_by	Null	Number	Yes	No
last_update_date	Null	Date	Yes	No
last_update_login	Null	Number	No	No
name	Null	Varchar2(30)	Yes	No
orig_system_agr_id	Null	Number	No	No
override_arule_flag	Null	Varchar2(1)	Yes	No
override_irule_flag	Null	Varchar2(1)	Yes	No

Table 5–3 AGREEMENT_REC_TYPE

Parameter	Usage	Type	Req	Drv
price_list_id	Null	Number	Yes ²	No
pricing_contract_id	Null	Number	No	No
purchase_order_num	Null	Varchar2(50)	No	No
revision	Null	Varchar2(50)	Yes	No
revision_date	Null	Date	Yes	No
revision_reason_code	Null	Varchar2(30)	No	No
salesrep_id	Null	Number	No	No
ship_method_code	Null	Varchar2(30)	No	No
signature_date	Null	Date	No	No
start_date_active	Null	Date	No	No
term_id	Null	Number	Yes	No
return_status	Null	Varchar2(1)	No	No
db_flag	Null	Varchar2(1)	No	No
operation	Null	Varchar2(30)	Yes	No
tp_attribute1	Null	Varchar2(240)	No	No
tp_attribute2	Null	Varchar2(240)	No	No
tp_attribute3	Null	Varchar2(240)	No	No
tp_attribute4	Null	Varchar2(240)	No	No
tp_attribute5	Null	Varchar2(240)	No	No
tp_attribute6	Null	Varchar2(240)	No	No
tp_attribute7	Null	Varchar2(240)	No	No
tp_attribute8	Null	Varchar2(240)	No	No
tp_attribute9	Null	Varchar2(240)	No	No
tp_attribute10	Null	Varchar2(240)	No	No
tp_attribute11	Null	Varchar2(240)	No	No
tp_attribute12	Null	Varchar2(240)	No	No
tp_attribute13	Null	Varchar2(240)	No	No

Table 5–3 AGREEMENT_REC_TYPE

Parameter	Usage	Type	Req	Drv
tp_attribute14	Null	Varchar2(240)	No	No
tp_attribute15	Null	Varchar2(240)	No	No
tp_attribute_category	Null	Varchar2(30)	No	No

The following table describes notations listed in the preceding table:

Table 5–4 Notations

Note	Description
1	For update and delete
2	If you are not passing a price list record

AGREEMENT_TBL_TYPE

The following table shows the parameters for this structure.

Table 5–5 AGREEMENT_TBL_TYPE Parameters

Parameter	Usage	Type	Req	Drv
Agreement_Rec_Type	Null	Record	No	No

AGREEMENT_VAL_REC_TYPE

The following table shows the parameters for this structure.

Table 5–6 AGREEMENT_VAL_REC_TYPE Parameters

Parameter	Usage	Type	Req	Drv
accounting_rule	Null	Varchar2(240)	Null	Null
agreement_contact	Null	Varchar2(240)	Null	Null
agreement	Null	Varchar2(240)	Null	Null
agreement_type	Null	Varchar2(240)	Null	Null
customer	Null	Varchar2(240)	Null	Null
freight_terms	Null	Varchar2(240)	Null	Null
invoice_contact	Null	Varchar2(240)	Null	Null
invoice_to_site_use	Null	Varchar2(240)	Null	Null
invoicing_rule	Null	Varchar2(240)	Null	Null
override_arule	Null	Varchar2(240)	Null	Null
override_irule	Null	Varchar2(240)	Null	Null
price_list	Null	Varchar2(240)	Null	Null

Table 5–6 AGREEMENT_VAL_REC_TYPE Parameters

Parameter	Usage	Type	Req	Drv
revision_reason	Null	Varchar2(240)	Null	Null
salesrep	Null	Varchar2(240)	Null	Null
ship_method	Null	Varchar2(240)	Null	Null
term	Null	Varchar2(240)	Null	Null

AGREEMENT_VAL_TBL_TYPE

The following table shows the parameters for this structure.

Table 5–7 AGREEMENT_VAL_TBL_TYPE Parameters

Parameter	Usage	Type	Req	Drv
Agreement_Val_Rec_Type	Null	Record	No	No

PRICE_LIST_REC_TYPE

The following table shows the parameters for this structure.

Table 5–8 PRICE_LIST_REC_TYPE Parameters

Parameter	Usage	Type	Req	Drv
attribute1	Null	Varchar2	No	No
attribute2	Null	Varchar2	No	No
attribute3	Null	Varchar2	No	No
attribute4	Null	Varchar2	No	No
attribute5	Null	Varchar2	No	No
attribute6	Null	Varchar2	No	No
attribute7	Null	Varchar2	No	No
attribute8	Null	Varchar2	No	No
attribute9	Null	Varchar2	No	No
attribute10	Null	Varchar2	No	No
attribute11	Null	Varchar2	No	No
attribute12	Null	Varchar2	No	No

Table 5–8 PRICE_LIST_REC_TYPE Parameters

Parameter	Usage	Type	Req	Drv
attribute13	Null	Varchar2	No	No
attribute14	Null	Varchar2	No	No
attribute15	Null	Varchar2	No	No
automatic_flag	Null	Varchar2	No	No
comments	Null	Varchar2	No	No
context	Null	Varchar2	No	No
created_by	Null	Number	No	No
creation_date	Null	Date	No	No
currency_code	Null	Varchar2	Yes	No
discount_lines_flag	Null	Varchar2	No	No
end_active_date	Null	Date	No	No
freight_terms_code	Null	Varchar2	No	No
gsa_indicator	Null	Varchar2	No	No
last_updated_by	Null	Number	No	No
last_update_date	Null	Date	No	No
last_update_login	Null	Number	No	No
list_header_id	Null	Number	No	No
list_type_code	Null	Varchar2	No	No
program_application_id	Null	Number	No	No
program_id	Null	Number	No	No
program_update_date	Null	Date	No	No
prorate_flag	Null	Varchar2	No	No
request_id	Null	Number	No	No
rounding_factor	Null	Number	No	No
ship_method_code	Null	Varchar2	No	No
start_date_active	Null	Date	No	No
terms_id	Null	Number	No	No

Table 5–8 PRICE_LIST_REC_TYPE Parameters

Parameter	Usage	Type	Req	Drv
return_status	Null	Varchar2	No	No
db_flag	Null	Varchar2	No	No
operation	Null	Varchar2	Yes	No
name	Null	Varchar2	Yes	No
description	Null	Varchar2	No	No
version_no	Null	Varchar2	No	No
active_flag	Null	Varchar2	No	No
mobile_download	Null	Varchar2	No	No
currency_header_id	Null	Number	No	No
pte_code	Null	Varchar2	No	Yes
list_source_code	Null	Varchar2	No	No
orig_system_header_ref	Null	Varchar2	No	No
global_flag	Null	Varchar2	No	No
orig_org_id	Null	Number	No	No

PRICE_LIST_TBL_TYPE

The following table shows the parameters for this structure.

Table 5–9 PRICE_LIST_TBL_TYPE Parameters

Parameter	Usage	Type	Req	Drv
Price_List_Rec_Type	Null	Record	No	No

PRICE_LIST_VAL_REC_TYPE

The following table shows the parameters for this structure.

Table 5–10 PRICE_LIST_VAL_REC_TYPE Parameters

Parameter	Usage	Type	Req	Drv
automatic	Null	Varchar2	No	No
currency	Null	Varchar2	No	No

Table 5–10 PRICE_LIST_VAL_REC_TYPE Parameters

Parameter	Usage	Type	Req	Drv
discount_lines	Null	Varchar2	No	No
freight_terms	Null	Varchar2	No	No
list_header	Null	Varchar2	No	No
list_type	Null	Varchar2	No	No
prorate	Null	Varchar2	No	No
ship_method	Null	Varchar2	No	No
terms	Null	Varchar2	No	No
currency_header	Null	Varchar2	No	No
pte	Null	Varchar2	No	No

PRICE_LIST_VAL_TBL_TYPE

The following table shows the parameters for this structure.

Table 5–11 PRICE_LIST_VAL_TBL_TYPE Parameters

Parameter	Usage	Type	Req	Drv
Price_List_Val_Rec_Type	Null	Record	No	No

PRICE_LIST_LINE_REC_TYPE

The following table shows the parameters for this structure:

Table 5–12 PRICE_LIST_LINE_REC_TYPE Parameters

Parameter	Usage	Type	Req	Drv
accrual_qty	Null	Number	No	No
accrual_uom_code	Null	Varchar2	No	No
arithmetic_operator	Null	Varchar2	No	No
attribute1	Null	Varchar2	No	No
attribute2	Null	Varchar2	No	No
attribute3	Null	Varchar2	No	No
attribute4	Null	Varchar2	No	No
attribute5	Null	Varchar2	No	No
attribute6	Null	Varchar2	No	No
attribute7	Null	Varchar2	No	No
attribute8	Null	Varchar2	No	No
attribute9	Null	Varchar2	No	No
attribute10	Null	Varchar2	No	No
attribute11	Null	Varchar2	No	No
attribute12	Null	Varchar2	No	No
attribute13	Null	Varchar2	No	No
attribute14	Null	Varchar2	No	No
attribute15	Null	Varchar2	No	No

Table 5–12 PRICE_LIST_LINE_REC_TYPE Parameters

Parameter	Usage	Type	Req	Drv
automatic_flag	Null	Varchar2	No	No
base_qty	Null	Number	No	No
base_uom_code	Null	Varchar2	No	No
comments	Null	Varchar2	No	No
context	Null	Varchar2	No	No
created_by	Null	Number	No	No
creation_date	Null	Date	No	No
effective_period_uom	Null	Varchar2	No	No
end_date_active	Null	Date	No	No
estim_accrual_rate	Null	Number	No	No
generate_using_formula_id	Null	Number	No	No
inventory_item_id	Null	Number	No	No
last_updated_by	Null	Number	No	No
last_update_date	Null	Date	No	No
last_update_login	Null	Number	No	No
list_header_id	Null	Number	No	No
list_line_id	Null	Number	No	No
list_line_type_code	Null	Varchar2	No	No
list_price	Null	Number	No	No
modifier_level_code	Null	Varchar2	No	No
number_effective_periods	Null	Number	No	No
operand	Null	Number	No	No
organization_id	Null	Number	No	No
override_flag	Null	Varchar2	No	No
percent_price	Null	Number	No	No
price_break_type_code	Null	Varchar2	No	No
price_by_formula_id	Null	Number	No	No

Table 5–12 PRICE_LIST_LINE_REC_TYPE Parameters

Parameter	Usage	Type	Req	Drv
primary_uom_flag	Null	Varchar2	No	No
print_on_invoice_flag	Null	Varchar2	No	No
program_application_id	Null	Number	No	No
program_id	Null	Number	No	No
program_update_date	Null	Date	No	No
rebate_trxn_type_code	Null	Varchar2	No	No
related_item_id	Null	Number	No	No
relationship_type_id	Null	Number	No	No
reprice_flag	Null	Varchar2	No	No
request_id	Null	Number	No	No
revision	Null	Varchar2	No	No
revision_date	Null	Date	No	No
revision_reason_code	Null	Varchar2	No	No
start_date_active	Null	Date	No	No
substitution_attribute	Null	Varchar2	No	No
substitution_context	Null	Varchar2	No	No
substitution_value	Null	Varchar2	No	No
return_status	Null	Varchar2	No	No
db_flag	Null	Varchar2	No	No
operation	Null	Varchar2	No	No
from_rltd_modifier_id	Null	Number	No	No
rltd_modifier_group_no	Null	Number	No	No
rltd_modifier_grp_type	Null	Varchar2	No	No
product_precedence	Null	Number	No	No
price_break_header_index	Null	Number	No	No
list_line_no	Null	Number	No	Yes
qualification_ind	Null	Number	No	Yes

PRICE_LIST_LINE_TBL_TYPE

The following table shows the parameters for this structure.

Table 5–13 PRICE_LIST_LINE_TBL_TYPE Parameters

Parameter	Usage	Type	Req	Drv
Price_List_Line_Rec_Type	Null	Record	No	No

PRICE_LIST_LINE_VAL_REC_TYPE

The following table shows the parameters for this structure.

Table 5–14 PRICE_LIST_LINE_VAL_REC_TYPE Parameters

Parameter	Usage	Type	Req	Drv
accrual_uom	Null	Varchar2	No	No
automatic	Null	Varchar2	No	No
base_uom	Null	Varchar2	No	No
generate_using_formula	Null	Varchar2	No	No
inventory_item	Null	Varchar2	No	No
list_header	Null	Varchar2	No	No
list_line	Null	Varchar2	No	No
list_line_type	Null	Varchar2	No	No
modifier_level	Null	Varchar2	No	No
organization	Null	Varchar2	No	No
override	Null	Varchar2	No	No
price_break_type	Null	Varchar2	No	No
price_by_formula	Null	Varchar2	No	No
primary_uom	Null	Varchar2	No	No
print_on_invoice	Null	Varchar2	No	No
rebate_transaction_type	Null	Varchar2	No	No
related_item	Null	Varchar2	No	Null
relationship_type	Null	Varchar2	No	Null

Table 5–14 PRICE_LIST_LINE_VAL_REC_TYPE Parameters

Parameter	Usage	Type	Req	Drv
reprice	Null	Varchar2	No	Null
revision_reason	Null	Varchar2	No	Null

PRICE_LIST_VAL_TBL_TYPE

The following table shows the parameters for this structure.

Table 5–15 PRICE_LIST_VAL_TBL_TYPE Parameters

Parameter	Usage	Type	Req	Drv
Price_List_Line_Val_Rec_Type	Null	Record	No	No

PRICING_ATTR_REC_TYPE

The following table shows the parameters for this structure.

Table 5–16 PRICING_ATTR_REC_TYPE Parameters

Parameter	Usage	Type	Req	Drv
accumulate_flag	Null	Varchar2	No	No
attribute1	Null	Varchar2	No	No
attribute2	Null	Varchar2	No	No
attribute3	Null	Varchar2	No	No
attribute4	Null	Varchar2	No	No
attribute5	Null	Varchar2	No	No
attribute6	Null	Varchar2	No	No
attribute7	Null	Varchar2	No	No
attribute8	Null	Varchar2	No	No
attribute9	Null	Varchar2	No	No
attribute10	Null	Varchar2	No	No
attribute11	Null	Varchar2	No	No
attribute12	Null	Varchar2	No	No
attribute13	Null	Varchar2	No	No
attribute14	Null	Varchar2	No	No
attribute15	Null	Varchar2	No	No
attribute_grouping_number	Null	Number	No	No
context	Null	Varchar2	No	No

Table 5–16 PRICING_ATTR_REC_TYPE Parameters

Parameter	Usage	Type	Req	Drv
created_by	Null	Number	No	No
creation_date	Null	Date	No	No
excluder_flag	Null	Varchar2	No	No
last_updated_by	Null	Number	No	No
last_update_date	Null	Date	No	No
last_update_login	Null	Number	No	No
list_line_id	Null	Number	No	No
pricing_attribute	Null	Varchar2	No	No
pricing_attribute_context	Null	Varchar2	No	No
pricig_attribute_id	Null	Number	No	No
pricing_attr_value_from	Null	Varchar2	No	No
pricing_attr_value_to	Null	Varchar2	No	No
product_attribute	Null	Varchar2	Yes	No
product_attribute_context	Null	Varchar2	Yes	No
product_attr_value	Null	Varchar2	Yes	No
product_uom_code	Null	Varchar2	Yes	No
program_application_id	Null	Number	No	No
program_id	Null	Number	No	No
program_update_date	Null	Date	No	No
request_id	Null	Number	No	No
pricing_attr_value_from_number	Null	Number	No	Yes
pricing_attr_value_to_number	Null	Number	No	Yes
qualification_ind	Null	Number	No	Yes
return_status	Null	Varchar2	No	No
db_flag	Null	Varchar2	No	No
operation	Null	Varchar2	Yes	No

Table 5–16 PRICING_ATTR_REC_TYPE Parameters

Parameter	Usage	Type	Req	Drv
PRICE_LIST_LINE_index	Null	Number	No	No
from_rltd_modifier_id	Null	Number	No	No
comparison_operator_code	Null	Varchar2	Yes	No
product_attribute_datatype	Null	Varchar2	No	Yes
pricing_attribute_datatype	Null	Varchar2	No	Yes
list_header_id	Null	Number	No	Yes
pricing_phase_id	Null	Number	No	Yes

PRICING_ATTR_TBL_TYPE

The following table shows the parameters for this structure.

Table 5–17 PRICING_ATTR_TBL_TYPE Parameters

Parameter	Usage	Type	Req	Drv
Pricing_Attr_Rec_Type	Null	Record	No	No

PRICING_ATTR_VAL_REC_TYPE

The following table shows the parameters for this structure.

Table 5–18 PRICING_ATTR_VAL_REC_TYPE Parameters

Parameter	Usage	Type	Req	Drv
accumulate	Null	Varchar2	No	No
excluder	Null	Varchar2	No	No
list_line	Null	Varchar2	No	No
pricing_attribute	Null	Varchar2	No	No
product_uom	Null	Varchar2	No	No
pricing_attribute_desc	Null	Varchar2	No	No
pricing_attr_value_from_desc	Null	Varchar2	No	No
pricing_attr_value_to_desc	Null	Varchar2	No	No

PRICE_LIST_VAL_TBL_TYPE

The following table shows the parameters for this structure.

Table 5–19 PRICE_LIST_VAL_TBL_TYPE Parameters

Parameter	Usage	Type	Req	Drv
Pricing_Attr_Val_Rec_ Type	Null	Record	No	No

Validation of Agreements Public API

Standard Validation

Oracle Advanced Pricing validates all required columns in the Agreements Public API. For specific information on the data implied by these columns, see *Oracle Pricing Technical Reference Manual*.

Other Validation

None

Error Handling

If any validation fails, the API will return error status to the calling module. The Agreements Public API processes the rows and reports the values in the following table for every record.

Table 5–20 Error Handling

Condition	PROCESS_STATUS	ERROR_MESSAGE
Success	5	null
Failure	4	actual error message

Example of Agreements Public API

The following sample code demonstrates Agreements Public API.

Insert an agreement and a price list header

```
SET SERVEROUTPUT ON SIZE 200000

declare

    out_return_status varchar2(1) := NULL;
    out_msg_count number := 0;
    out_msg_data varchar2(2000);

    in_Agreement_recOE_Pricing_Cont_PUB.Agreement_Rec_Type;
    in_Agreement_val_recOE_Pricing_Cont_PUB.Agreement_Val_Rec_Type;
    in_price_list_rec QP_PRICE_LIST_PUB.Price_List_Rec_Type;

    in_price_list_val_rec QP_PRICE_LIST_PUB.Price_List_Val_Rec_Type;
```

```

in_price_list_line_tbl QP_PRICE_LIST_PUB.Price_List_Line_Tbl_Type;
in_price_list_line_val_tbl QP_PRICE_LIST_PUB.Price_List_Line_Val_Tbl_Type;
in_pricing_attr_tbl QP_PRICE_LIST_PUB.Pricing_Attr_Tbl_Type;
in_pricing_attr_val_tbl QP_PRICE_LIST_PUB.Pricing_Attr_Val_Tbl_Type;
out_Agreement_rec      OE_Pricing_Cont_PUB.Agreement_Rec_Type;
out_Agreement_val_rec   OE_Pricing_Cont_PUB.Agreement_Val_Rec_Type;
out_price_list_rec      QP_PRICE_LIST_PUB.Price_List_Rec_Type;
out_price_list_val_rec   QP_PRICE_LIST_PUB.Price_List_Val_Rec_Type;
out_price_list_line_tbl QP_PRICE_LIST_PUB.Price_List_Line_Tbl_Type;
out_price_list_line_val_tbl QP_PRICE_LIST_PUB.Price_List_Line_Val_Tbl_Type;
out_pricing_attr_tbl    QP_PRICE_LIST_PUB.Pricing_Attr_Tbl_Type;
out_pricing_attr_val_tbl QP_PRICE_LIST_PUB.Pricing_Attr_Val_Tbl_Type;

K number := 1;

j number := 1;

begin

dbms_output.put_line('Creating an Agreement record');

--Creating an Agreement record

in_Agreement_rec.name := 'TESTING 2001';
in_Agreement_rec.creation_date := sysdate;
in_Agreement_rec.created_by := 1001;
in_Agreement_rec.last_update_date := sysdate;
in_Agreement_rec.last_updated_by := 9001;
in_Agreement_rec.agreement_type_code := 'STANDARD';
in_Agreement_rec.agreement_num := '88';
in_Agreement_rec.revision := '1';
in_Agreement_rec.revision_date := sysdate;
in_Agreement_rec.term_id := 1000;
in_Agreement_rec.OVERRIDE_IRULE_FLAG := 'Y';
in_Agreement_rec.OVERRIDE_ARULE_FLAG := 'Y';
in_Agreement_rec.agreement_id := FND_API.G_MISS_NUM;
in_Agreement_rec.operation      := QP_GLOBALS.G_OPR_CREATE;

/* set the list_header_id to g_miss_num */

in_price_list_rec.list_header_id := FND_API.G_MISS_NUM;
in_price_list_rec.name := 'Sample1-1028 -8987';
in_price_list_rec.list_type_code := 'AGR';
in_price_list_rec.description := 'Sample price list 8987';
in_price_list_rec.currency_code := 'USD';

```

```

        in_price_list_rec.operation := QP_GLOBALS.G_OPR_CREATE;

oe_debug_pub.add('Calling the process Agreements API');

    OE_Pricing_Cont_PUB.Process_Agreement

(
    p_api_version_number=> 1.0
  , p_init_msg_list=> FND_API.G_FALSE
  , p_return_values=> FND_API.G_FALSE
  , p_commit=> FND_API.G_FALSE
  , x_return_status=> out_return_status
  , x_msg_count=> out_msg_count
  , x_msg_data=> out_msg_data
  , p_Agreement_rec=> in_Agreement_rec
  , p_Price_LHeader_rec=> in_price_list_rec
  , p_Price_LLine_tbl=> in_price_list_line_tbl
  , p_Pricing_Attr_tbl=> in_pricing_attr_tbl
  , x_Agreement_rec=> out_Agreement_rec
  , x_Agreement_val_rec=> out_Agreement_val_rec
  , x_Price_LHeader_rec=> out_price_list_rec
  , x_Price_LHeader_val_rec=> out_price_list_val_rec
  , x_Price_LLine_tbl=> out_price_list_line_tbl
  , x_Price_LLine_val_tbl=> out_price_list_line_val_tbl
  , x_Pricing_Attr_tbl=> out_pricing_attr_tbl
  , x_Pricing_Attr_val_tbl=> out_pricing_attr_val_tbl
);

dbms_output.put_line('Agreement ID = '||to_char(out_Agreement_rec.agreement_id));

    IF out_return_status <> FND_API.G_RET_STS_SUCCESS THEN

        RAISE FND_API.G_EXC_UNEXPECTED_ERROR;

    END IF;

oe_debug_pub.add('after process agreement ');

EXCEPTION

    WHEN FND_API.G_EXC_ERROR THEN

        out_return_status := FND_API.G_RET_STS_ERROR;

        --Get message count and data

```



```

        dbms_output.put_line('err msg 1 is : ' || out_msg_data);

    WHEN FND_API.G_EXC_UNEXPECTED_ERROR THEN

        out_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;
        dbms_output.put_line(' msg count 2 is : ' || out_msg_count);
        for k in 1 .. out_msg_count loop
            out_msg_data := oe_msg_pub.get( p_msg_index => k,

p_encoded => 'F'

        );
        -- Get message count and data
        dbms_output.put_line('err msg ' || k || 'is: ' || out_msg_data);

        null;

    end loop;

    WHEN OTHERS THEN

        out_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;

        --Get message count and data
        dbms_output.put_line('err msg 3 is : ' || out_msg_data);

    for k in 1 .. out_msg_count loop
        out_msg_data := oe_msg_pub.get( p_msg_index => k,
        p_encoded => 'F'

    );

    --Get message count and data
    dbms_output.put_line('err msg ' || k || 'is: ' || out_msg_data);
    null;
    end loop;
end;

/
--commit;
--exit;

```

Insert an agreement, a price list header, and lines

```
SET SERVEROUTPUT ON SIZE 200000
```

```

declare

    out_return_status varchar2(1) := NULL;
    out_msg_count number := 0;
    out_msg_data varchar2(2000);
    in_Agreement_rec OE_Pricing_Cont_PUB.Agreement_Rec_Type;
    in_Agreement_val_rec OE_Pricing_Cont_PUB.Agreement_Val_Rec_Type;
    in_price_list_rec QP_PRICE_LIST_PUB.Price_List_Rec_Type;
    in_price_list_val_rec QP_PRICE_LIST_PUB.Price_List_Val_Rec_Type;
    in_price_list_line_tbl QP_PRICE_LIST_PUB.Price_List_Line_Tbl_Type;
    in_price_list_line_val_tbl QP_PRICE_LIST_PUB.Price_List_Line_Val_Tbl_Type;
    in_pricing_attr_tbl QP_PRICE_LIST_PUB.Pricing_Attr_Tbl_Type;
    in_pricing_attr_val_tbl QP_PRICE_LIST_PUB.Pricing_Attr_Val_Tbl_Type;
    out_Agreement_rec OE_Pricing_Cont_PUB.Agreement_Rec_Type;
    out_Agreement_val_rec OE_Pricing_Cont_PUB.Agreement_Val_Rec_Type;
    out_price_list_rec QP_PRICE_LIST_PUB.Price_List_Rec_Type;
    out_price_list_val_rec QP_PRICE_LIST_PUB.Price_List_Val_Rec_Type;
    out_price_list_line_tbl QP_PRICE_LIST_PUB.Price_List_Line_Tbl_Type;
    out_price_list_line_val_tbl QP_PRICE_LIST_PUB.Price_List_Line_Val_Tbl_Type;
    out_pricing_attr_tbl QP_PRICE_LIST_PUB.Pricing_Attr_Tbl_Type;
    out_pricing_attr_val_tbl QP_PRICE_LIST_PUB.Pricing_Attr_Val_Tbl_Type;

    K number := 1;
    j number := 1;

begin

    dbms_output.put_line('Creating an Agreement record');

    -- Creating an Agreement record

    in_Agreement_rec.name := 'TESTING SEAGATE API';
    in_Agreement_rec.creation_date := sysdate;
    in_Agreement_rec.created_by := 1001;
    in_Agreement_rec.last_update_date := sysdate;
    in_Agreement_rec.last_updated_by := 9001;
    in_Agreement_rec.agreement_type_code := 'STANDARD';
    in_Agreement_rec.agreement_num := '100888';
    in_Agreement_rec.revision := '1';

    /* Agreement Number and revision must be unique */
    in_Agreement_rec.revision_date := sysdate;
    in_Agreement_rec.term_id := 1000;
    in_Agreement_rec.OVERRIDE_IRULE_FLAG := 'Y';

```

```

in_Agreement_rec.OVERRIDE_ARULE_FLAG := 'Y';
in_Agreement_rec.agreement_id := FND_API.G_MISS_NUM;
in_Agreement_rec.operation      := QP_GLOBALS.G_OPR_CREATE;

    /* set the list_header_id to g_miss_num */
    in_price_list_rec.list_header_id := FND_API.G_MISS_NUM;
    in_price_list_rec.name := 'seagate pricelist1';
    in_price_list_rec.list_type_code := 'AGR';
    in_price_list_rec.description := 'Sample pr 3';
    in_price_list_rec.currency_code := 'USD';
    in_price_list_rec.operation := QP_GLOBALS.G_OPR_CREATE;

FOR K IN 1..3 LOOP
in_price_list_line_tbl(K).list_line_id := FND_API.G_MISS_NUM;
in_price_list_line_tbl(K).list_line_type_code := 'PLL';
in_price_list_line_tbl(K).operation := QP_GLOBALS.G_OPR_CREATE;
in_price_list_line_tbl(K).operand := 101;
in_price_list_line_tbl(K).arithmetic_operator := 'UNIT_PRICE';
END LOOP;

/*
product_attr_value stores inventory item id -
product_attribute for Item Number is Pricing_Attribute1
product_attribute_context is ITEM. Each line can have one or more pricing
attributes. We use PRICE_LIST_LINE_INDEX to link the child(pricing attributes )
to the parent(line). When you have pricing attributes like color, length, width
etc, populate the fields pricing_attribute_context, pricing_attribute, pricing_
attr_value_from, pricing_attr_value_to and comparison_operator_code ( '=' or
'between') and repeat the product_attr_value and its attribute and context for
each record.

*/

J := 1;

in_pricing_attr_tbl(J).pricing_attribute_id := FND_API.G_MISS_NUM;
in_pricing_attr_tbl(J).list_line_id := FND_API.G_MISS_NUM;
in_pricing_attr_tbl(J).PRODUCT_ATTRIBUTE_CONTEXT := 'ITEM';
in_pricing_attr_tbl(J).PRODUCT_ATTRIBUTE := 'PRICING_ATTRIBUTE1';
in_pricing_attr_tbl(J).PRODUCT_ATTR_VALUE := '62061';
in_pricing_attr_tbl(J).PRODUCT_UOM_CODE := 'Ea';
in_pricing_attr_tbl(J).EXCLUDER_FLAG := 'N';
in_pricing_attr_tbl(J).ATTRIBUTE_GROUPING_NO := 1;
in_pricing_attr_tbl(J).PRICE_LIST_LINE_INDEX := 1;
in_pricing_attr_tbl(J).operation := QP_GLOBALS.G_OPR_CREATE;

```

```

J := J + 1;

in_pricing_attr_tbl(J).pricing_attribute_id := FND_API.G_MISS_NUM;
in_pricing_attr_tbl(J).list_line_id := FND_API.G_MISS_NUM;
in_pricing_attr_tbl(J).PRODUCT_ATTRIBUTE_CONTEXT := 'ITEM';
in_pricing_attr_tbl(J).PRODUCT_ATTRIBUTE := 'PRICING_ATTRIBUTE1';
in_pricing_attr_tbl(J).PRODUCT_ATTR_VALUE := '62081';
in_pricing_attr_tbl(J).PRODUCT_UOM_CODE := 'Ea';
in_pricing_attr_tbl(J).EXCLUDER_FLAG := 'N';
in_pricing_attr_tbl(J).ATTRIBUTE_GROUPING_NO := 1;
in_pricing_attr_tbl(J).PRICE_LIST_LINE_INDEX := 2;
in_pricing_attr_tbl(J).operation := QP_GLOBALS.G_OPR_CREATE;

J := J + 1;
in_pricing_attr_tbl(J).pricing_attribute_id := FND_API.G_MISS_NUM;
in_pricing_attr_tbl(J).list_line_id := FND_API.G_MISS_NUM;
in_pricing_attr_tbl(J).PRODUCT_ATTRIBUTE_CONTEXT := 'ITEM';
in_pricing_attr_tbl(J).PRODUCT_ATTRIBUTE := 'PRICING_ATTRIBUTE1';
in_pricing_attr_tbl(J).PRODUCT_ATTR_VALUE := '62083';
in_pricing_attr_tbl(J).PRODUCT_UOM_CODE := 'Ea';
in_pricing_attr_tbl(J).EXCLUDER_FLAG := 'N';
in_pricing_attr_tbl(J).ATTRIBUTE_GROUPING_NO := 1;
in_pricing_attr_tbl(J).PRICE_LIST_LINE_INDEX := 3;
in_pricing_attr_tbl(J).operation := QP_GLOBALS.G_OPR_CREATE;

oe_debug_pub.add('Calling the process Agreements API');

```

OE_Pricing_Cont_PUB.Process_Agreement

(p_api_version_number	=> 1.0
,	p_init_msg_list	=> FND_API.G_FALSE
,	p_return_values	=> FND_API.G_FALSE
,	p_commit	=> FND_API.G_FALSE
,	x_return_status	=> out_return_status
,	x_msg_count	=> out_msg_count
,	x_msg_data	=> out_msg_data
,	p_Agreement_rec	=> in_Agreement_rec
,	p_Price_LHeader_rec	=> in_price_list_rec
,	p_Price_LLine_tbl	=> in_price_list_line_tbl
,	p_Pricing_Attr_tbl	=> in_pricing_attr_tbl
,	x_Agreement_rec	=> out_Agreement_rec
,	x_Agreement_val_rec	=> out_Agreement_val_rec
,	x_Price_LHeader_rec	=> out_price_list_rec
,	x_Price_LHeader_val_rec	=> out_price_list_val_rec

```

,   x_Price_LLine_tbl           => out_price_list_line_tbl
,   x_Price_LLine_val_tbl       => out_price_list_line_val_tbl
,   x_Pricing_Attr_tbl          => out_pricing_attr_tbl
,   x_Pricing_Attr_val_tbl      => out_pricing_attr_val_tbl
);

IF out_return_status <> FND_API.G_RET_STS_SUCCESS THEN

    RAISE FND_API.G_EXC_UNEXPECTED_ERROR;

END IF;

oe_debug_pub.add('after process agreement ');

EXCEPTION

    WHEN FND_API.G_EXC_ERROR THEN

out_return_status := FND_API.G_RET_STS_ERROR;

--Get message count and data

    dbms_output.put_line('err msg 1 is : ' || out_msg_data);

    WHEN FND_API.G_EXC_UNEXPECTED_ERROR THEN

        out_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;

        dbms_output.put_line(' msg count 2 is : ' || out_msg_count);

        for k in 1 .. out_msg_count loop

out_msg_data := oe_msg_pub.get( p_msg_index => k,

p_encoded => 'F'

        );

        --Get message count and data
        dbms_output.put_line('err msg ' || k || 'is: ' || out_msg_data);

        null;

    end loop;

    WHEN OTHERS THEN

```

```

        out_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;

        --Get message count and data

        dbms_output.put_line('err msg 3 is : ' || out_msg_data);

        for k in 1 .. out_msg_count loop

out_msg_data := oe_msg_pub.get( p_msg_index => k,

p_encoded => 'F'

        );

        --Get message count and data

        dbms_output.put_line('err msg ' || k || 'is: ' || out_msg_data);

        null;
end loop;

end;
/
--commit;
--exit;

```

Create an agreement with a standard price list

This script inserts an agreement which uses any existing standard price list. Therefore, you do not need to pass a price list record.

declare

```

out_return_status varchar2(1) := NULL;
out_msg_count number := 0;
out_msg_data varchar2(2000);
in_Agreement_rec      OE_Pricing_Cont_PUB.Agreement_Rec_Type;
in_Agreement_val_rec  OE_Pricing_Cont_PUB.Agreement_Val_Rec_Type;
in_price_list_rec     QP_PRICE_LIST_PUB.Price_List_Rec_Type;
in_price_list_val_rec QP_PRICE_LIST_PUB.Price_List_Val_Rec_Type;
in_price_list_line_tbl QP_PRICE_LIST_PUB.Price_List_Line_Tbl_Type;
in_price_list_line_val_tbl QP_PRICE_LIST_PUB.Price_List_Line_Val_Tbl_Type;
in_pricing_attr_tbl   QP_PRICE_LIST_PUB.Pricing_Attr_Tbl_Type;
in_pricing_attr_val_tbl QP_PRICE_LIST_PUB.Pricing_Attr_Val_Tbl_Type;

```

```

out_Agreement_rec          OE_Pricing_Cont_PUB.Agreement_Rec_Type;
out_Agreement_val_rec      OE_Pricing_Cont_PUB.Agreement_Val_Rec_Type;
out_price_list_rec         QP_PRICE_LIST_PUB.Price_List_Rec_Type;
out_price_list_val_rec     QP_PRICE_LIST_PUB.Price_List_Val_Rec_Type;
out_price_list_line_tbl    QP_PRICE_LIST_PUB.Price_List_Line_Tbl_Type;
out_price_list_line_val_tbl QP_PRICE_LIST_PUB.Price_List_Line_Val_Tbl_Type;
out_pricing_attr_tbl       QP_PRICE_LIST_PUB.Pricing_Attr_Tbl_Type;
out_pricing_attr_val_tbl   QP_PRICE_LIST_PUB.Pricing_Attr_Val_Tbl_Type;

K number := 1;
j number := 1;

begin

dbms_output.put_line('Creating an Agreement record');

-- Creating an Agreement record

in_Agreement_rec.name := 'Test SEAGATE API 55';
in_Agreement_rec.creation_date := sysdate;
in_Agreement_rec.created_by := 1001;
in_Agreement_rec.last_update_date := sysdate;
in_Agreement_rec.last_updated_by := 9001;
in_Agreement_rec.agreement_type_code := 'STANDARD';
in_Agreement_rec.agreement_num := '55';
in_Agreement_rec.revision := '1';
in_Agreement_rec.revision_date := sysdate;
in_Agreement_rec.term_id := 1000;
in_Agreement_rec.OVERRIDE_IRULE_FLAG := 'Y';
in_Agreement_rec.OVERRIDE_ARULE_FLAG := 'Y';
in_Agreement_rec.agreement_id := FND_API.G_MISS_NUM;
in_Agreement_rec.operation := QP_GLOBALS.G_OPR_CREATE;
in_Agreement_rec.price_list_id := 107811;
/*make sure that you are using a price list for which list_type_code is 'PRL'
which means that it is a standard price list */

oe_debug_pub.add('Calling the process Agreements API');
```

OE_Pricing_Cont_PUB.Process_Agreement

```

( p_api_version_number=> 1.0
, p_init_msg_list=> FND_API.G_FALSE
, p_return_values=> FND_API.G_FALSE
, p_commit=> FND_API.G_FALSE
, x_return_status=> out_return_status
```

```

,   x_msg_count=> out_msg_count
,   x_msg_data=> out_msg_data
,   p_Agreement_rec=> in_Agreement_rec
,   x_Agreement_rec=> out_Agreement_rec
,   x_Agreement_val_rec=> out_Agreement_val_rec
,   x_Price_LHeader_rec=> out_price_list_rec
,   x_Price_LHeader_val_rec=> out_price_list_val_rec
,   x_Price_LLine_tbl=> out_price_list_line_tbl
,   x_Price_LLine_val_tbl=> out_price_list_line_val_tbl
,   x_Pricing_Attr_tbl=> out_pricing_attr_tbl
,   x_Pricing_Attr_val_tbl=> out_pricing_attr_val_tbl
);

IF out_return_status <> FND_API.G_RET_STS_SUCCESS THEN

    RAISE FND_API.G_EXC_UNEXPECTED_ERROR;

END IF;

oe_debug_pub.add('after process agreement ');

EXCEPTION

    WHEN FND_API.G_EXC_ERROR THEN

        out_return_status := FND_API.G_RET_STS_ERROR;

        --Get message count and data

        dbms_output.put_line('err msg 1 is : ' || out_msg_data);

    WHEN FND_API.G_EXC_UNEXPECTED_ERROR THEN

        out_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;
        dbms_output.put_line(' msg count 2 is : ' || out_msg_count);

        for k in 1 .. out_msg_count loop

            out_msg_data := oe_msg_pub.get( p_msg_index => k,

            p_encoded => 'F'

            );

            -- Get message count and data

```



```

        dbms_output.put_line('err msg ' || k || 'is: ' || out_msg_data);

        null;

    end loop;

    WHEN OTHERS THEN

        out_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;

        --Get message count and data

        dbms_output.put_line('err msg 3 is : ' || out_msg_data);

    for k in 1 .. out_msg_count loop
        out_msg_data := oe_msg_pub.get( p_msg_index => k,

            p_encoded => 'F'

        );

        --Get message count and data
        dbms_output.put_line('err msg ' || k || 'is: ' || out_msg_data);
        null;
    end loop;
end;
/
--commit;
--exit;

```

Update an agreement with a standard price list

This script inserts an agreement which uses any existing standard price list. Therefore, you do not need to pass a price list record.

declare

```

    out_return_status varchar2(1) := NULL;
    out_msg_count number := 0;
    out_msg_data varchar2(2000);
    in_Agreement_rec      OE_Pricing_Cont_PUB.Agreement_Rec_Type;
    in_Agreement_val_rec  OE_Pricing_Cont_PUB.Agreement_Val_Rec_Type;
    in_price_list_rec     QP_PRICE_LIST_PUB.Price_List_Rec_Type;
    in_price_list_val_rec QP_PRICE_LIST_PUB.Price_List_Val_Rec_Type;
    in_price_list_line_tbl QP_PRICE_LIST_PUB.Price_List_Line_Tbl_Type;

```

```

in_price_list_line_val_tbl QP_PRICE_LIST_PUB.Price_List_Line_Val_Tbl_Type;
in_pricing_attr_tbl       QP_PRICE_LIST_PUB.Pricing_Attr_Tbl_Type;
in_pricing_attr_val_tbl   QP_PRICE_LIST_PUB.Pricing_Attr_Val_Tbl_Type;
out_Agreement_rec         OE_Pricing_Cont_PUB.Agreement_Rec_Type;
out_Agreement_val_rec     OE_Pricing_Cont_PUB.Agreement_Val_Rec_Type;
out_price_list_rec        QP_PRICE_LIST_PUB.Price_List_Rec_Type;
out_price_list_val_rec    QP_PRICE_LIST_PUB.Price_List_Val_Rec_Type;
out_price_list_line_tbl   QP_PRICE_LIST_PUB.Price_List_Line_Tbl_Type;
out_price_list_line_val_tbl QP_PRICE_LIST_PUB.Price_List_Line_Val_Tbl_Type;
out_pricing_attr_tbl      QP_PRICE_LIST_PUB.Pricing_Attr_Tbl_Type;
out_pricing_attr_val_tbl  QP_PRICE_LIST_PUB.Pricing_Attr_Val_Tbl_Type;
K number := 1;
j number := 1;

begin

dbms_output.put_line('Updating an Agreement record');

-- Updating an Agreement record

in_Agreement_rec.agreement_id := 19034;
in_Agreement_rec.name := 'Test NGUHA 22';
in_Agreement_rec.creation_date := sysdate;
in_Agreement_rec.created_by := 1001;
in_Agreement_rec.last_update_date := sysdate;
in_Agreement_rec.last_updated_by := 9001;
in_Agreement_rec.agreement_type_code := 'STANDARD';
in_Agreement_rec.agreement_num := '22';
in_Agreement_rec.revision := '1';
in_Agreement_rec.revision_date := sysdate;
in_Agreement_rec.term_id := 1000;
in_Agreement_rec.operation := QP_GLOBALS.G_OPR_UPDATE;
in_Agreement_rec.price_list_id := 107811;

oe_debug_pub.add('Calling the process Agreements API');
```

OE_Pricing_Cont_PUB.Process_Agreement

```

( p_api_version_number=> 1.0
, p_init_msg_list=> FND_API.G_FALSE
, p_return_values=> FND_API.G_FALSE
, p_commit=> FND_API.G_FALSE
, x_return_status=> out_return_status
, x_msg_count=> out_msg_count
, x_msg_data=> out_msg_data
```

```

,   p_Agreement_rec=> in_Agreement_rec
,   x_Agreement_rec=> out_Agreement_rec
,   x_Agreement_val_rec=> out_Agreement_val_rec
,   x_Price_LHeader_rec=> out_price_list_rec
,   x_Price_LHeader_val_rec=> out_price_list_val_rec
,   x_Price_LLine_tbl=> out_price_list_line_tbl
,   x_Price_LLine_val_tbl=> out_price_list_line_val_tbl
,   x_Pricing_Attr_tbl=> out_pricing_attr_tbl
,   x_Pricing_Attr_val_tbl=> out_pricing_attr_val_tbl
);

IF out_return_status <> FND_API.G_RET_STS_SUCCESS THEN

    RAISE FND_API.G_EXC_UNEXPECTED_ERROR;

END IF;

oe_debug_pub.add('after process agreement ');

EXCEPTION

    WHEN FND_API.G_EXC_ERROR THEN

        out_return_status := FND_API.G_RET_STS_ERROR;

        --Get message count and data

        dbms_output.put_line('err msg 1 is : ' || out_msg_data);

    WHEN FND_API.G_EXC_UNEXPECTED_ERROR THEN

        out_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;

        dbms_output.put_line(' msg count 2 is : ' || out_msg_count);

        for k in 1 .. out_msg_count loop

            out_msg_data := oe_msg_pub.get( p_msg_index => k,

                p_encoded => 'F'

            );

            --Get message count and data

            dbms_output.put_line('err msg ' || k || 'is: ' || out_msg_data);

```

```

        null;

    end loop;

    WHEN OTHERS THEN

        out_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;

        --Get message count and data
        dbms_output.put_line('err msg 3 is : ' || out_msg_data);

    for k in 1 .. out_msg_count loop
    out_msg_data := oe_msg_pub.get( p_msg_index => k,

        p_encoded => 'F'

    );

    --Get message count and data
    dbms_output.put_line('err msg ' || k || 'is: ' || out_msg_data);

    null;

    end loop;

end;
/
--commit;
--exit;

```

Update agreement record and create agreement price list

declare

```

    out_return_status varchar2(1) := NULL;
    out_msg_count number := 0;
    out_msg_data varchar2(2000);

    in_Agreement_rec          OE_Pricing_Cont_PUB.Agreement_Rec_Type;
    in_Agreement_val_rec OE_Pricing_Cont_PUB.Agreement_Val_Rec_Type;
    in_price_list_rec QP_PRICE_LIST_PUB.Price_List_Rec_Type;
    in_price_list_val_rec QP_PRICE_LIST_PUB.Price_List_Val_Rec_Type;
    in_price_list_line_tbl QP_PRICE_LIST_PUB.Price_List_Line_Tbl_Type;
    in_price_list_line_val_tbl QP_PRICE_LIST_PUB.Price_List_Line_Val_Tbl_Type;
    in_pricing_attr_tbl QP_PRICE_LIST_PUB.Pricing_Attr_Tbl_Type;

```

```

in_pricing_attr_val_tbl QP_PRICE_LIST_PUB.Pricing_Attr_Val_Tbl_Type;
out_Agreement_rec      OE_Pricing_Cont_PUB.Agreement_Rec_Type;
out_Agreement_val_rec   OE_Pricing_Cont_PUB.Agreement_Val_Rec_Type;
out_price_list_rec      QP_PRICE_LIST_PUB.Price_List_Rec_Type;
out_price_list_val_rec  QP_PRICE_LIST_PUB.Price_List_Val_Rec_Type;
out_price_list_line_tbl QP_PRICE_LIST_PUB.Price_List_Line_Tbl_Type;
out_price_list_line_val_tbl QP_PRICE_LIST_PUB.Price_List_Line_Val_Tbl_Type;
out_pricing_attr_tbl    QP_PRICE_LIST_PUB.Pricing_Attr_Tbl_Type;
out_pricing_attr_val_tbl QP_PRICE_LIST_PUB.Pricing_Attr_Val_Tbl_Type;

K number := 1;
j number := 1;

begin

dbms_output.put_line('Updating an Agreement record');

-- Updating an Agreement record

in_Agreement_rec.agreement_id :=19034;
in_Agreement_rec.name := 'Test 2002';
in_Agreement_rec.last_update_date := sysdate;
in_Agreement_rec.last_updated_by := 9001;
in_Agreement_rec.agreement_type_code := 'STANDARD';
in_Agreement_rec.revision_date := sysdate;
in_Agreement_rec.term_id := 1897;
in_Agreement_rec.operation      := QP_GLOBALS.G_OPR_UPDATE;

-- We are updating the agreement and creating a new AGR price list

/* set the list_header_id to g_miss_num */

in_price_list_rec.list_header_id := FND_API.G_MISS_NUM;
in_price_list_rec.name := 'SEAGATE NEW AGR 1';
in_price_list_rec.list_type_code := 'AGR';
in_price_list_rec.description := 'Sample price list 3';
in_price_list_rec.currency_code := 'USD';
in_price_list_rec.operation := QP_GLOBALS.G_OPR_CREATE;

FOR K IN 1..3 LOOP
in_price_list_line_tbl(K).list_line_id := FND_API.G_MISS_NUM;
in_price_list_line_tbl(K).list_line_type_code := 'PLL';
in_price_list_line_tbl(K).operation := QP_GLOBALS.G_OPR_CREATE;
in_price_list_line_tbl(K).operand := 101;
in_price_list_line_tbl(K).arithmetic_operator := 'UNIT_PRICE';

```

```

END LOOP;

/*
product_attr_value stores inventory item id -
product_attribute for Item Number is Pricing_Attribute1
product_attribute_context is ITEM. Each line can have one or more pricing
attributes. We use PRICE_LIST_LINE_INDEX to link the child(pricing attributes )
to the parent(line). When you have pricing attributes like color, length, width
etc,populate the fields pricing_attribute_context, pricing_attribute, pricing_
attr_value_from, pricing_attr_value_to and comparison_operator_code ( '=' or
'between') and repeat the product_attr_value and its attribute and context for
each record.

*/

J := 1;

in_pricing_attr_tbl(J).pricing_attribute_id := FND_API.G_MISS_NUM;
in_pricing_attr_tbl(J).list_line_id := FND_API.G_MISS_NUM;
in_pricing_attr_tbl(J).PRODUCT_ATTRIBUTE_CONTEXT := 'ITEM';
in_pricing_attr_tbl(J).PRODUCT_ATTRIBUTE := 'PRICING_ATTRIBUTE1';
in_pricing_attr_tbl(J).PRODUCT_ATTR_VALUE := '62061';
--(This is the inventory_item_id)
in_pricing_attr_tbl(J).PRODUCT_UOM_CODE := 'Ea';
in_pricing_attr_tbl(J).EXCLUDER_FLAG := 'N';
in_pricing_attr_tbl(J).ATTRIBUTE_GROUPING_NO := 1;
in_pricing_attr_tbl(J).PRICE_LIST_LINE_INDEX := 1;
in_pricing_attr_tbl(J).operation := QP_GLOBALS.G_OPR_CREATE;

J := J + 1;

in_pricing_attr_tbl(J).pricing_attribute_id := FND_API.G_MISS_NUM;
in_pricing_attr_tbl(J).list_line_id := FND_API.G_MISS_NUM;
in_pricing_attr_tbl(J).PRODUCT_ATTRIBUTE_CONTEXT := 'ITEM';
in_pricing_attr_tbl(J).PRODUCT_ATTRIBUTE := 'PRICING_ATTRIBUTE1';
in_pricing_attr_tbl(J).PRODUCT_ATTR_VALUE := '62081';
--(This is the inventory_item_id)

in_pricing_attr_tbl(J).PRODUCT_UOM_CODE := 'Ea';
in_pricing_attr_tbl(J).EXCLUDER_FLAG := 'N';
in_pricing_attr_tbl(J).ATTRIBUTE_GROUPING_NO := 1;
in_pricing_attr_tbl(J).PRICE_LIST_LINE_INDEX := 2;
in_pricing_attr_tbl(J).operation := QP_GLOBALS.G_OPR_CREATE;

J := J + 1;

```

```

in_pricing_attr_tbl(J).pricing_attribute_id := FND_API.G_MISS_NUM;
in_pricing_attr_tbl(J).list_line_id := FND_API.G_MISS_NUM;
in_pricing_attr_tbl(J).PRODUCT_ATTRIBUTE_CONTEXT := 'ITEM';
in_pricing_attr_tbl(J).PRODUCT_ATTRIBUTE := 'PRICING_ATTRIBUTE1';
in_pricing_attr_tbl(J).PRODUCT_ATTR_VALUE := '62083';
--(This is the inventory_item_id)
in_pricing_attr_tbl(J).PRODUCT_UOM_CODE := 'Ea';
in_pricing_attr_tbl(J).EXCLUDER_FLAG := 'N';
in_pricing_attr_tbl(J).ATTRIBUTE_GROUPING_NO := 1;
in_pricing_attr_tbl(J).PRICE_LIST_LINE_INDEX := 3;
in_pricing_attr_tbl(J).operation := QP_GLOBALS.G_OPR_CREATE;

oe_debug_pub.add('Calling the process Agreements API');

```

OE_Pricing_Cont_PUB.Process_Agreement

```

( p_api_version_number=> 1.0
, p_init_msg_list=> FND_API.G_FALSE
, p_return_values=> FND_API.G_FALSE
, p_commit=> FND_API.G_FALSE
, x_return_status=> out_return_status
, x_msg_count=> out_msg_count
, x_msg_data=> out_msg_data
, p_Agreement_rec=> in_Agreement_rec
, p_Price_LHeader_rec=> in_price_list_rec
, p_Price_LLine_tbl=> in_price_list_line_tbl
, p_Pricing_Attr_tbl=> in_pricing_attr_tbl
, x_Agreement_rec=> out_Agreement_rec
, x_Agreement_val_rec=> out_Agreement_val_rec
, x_Price_LHeader_rec=> out_price_list_rec
, x_Price_LHeader_val_rec=> out_price_list_val_rec
, x_Price_LLine_tbl=> out_price_list_line_tbl
, x_Price_LLine_val_tbl=> out_price_list_line_val_tbl
, x_Pricing_Attr_tbl=> out_pricing_attr_tbl
, x_Pricing_Attr_val_tbl=> out_pricing_attr_val_tbl
);

IF out_return_status <> FND_API.G_RET_STS_SUCCESS THEN

    RAISE FND_API.G_EXC_UNEXPECTED_ERROR;

END IF;

oe_debug_pub.add('after process agreement ');

```

```

EXCEPTION

    WHEN FND_API.G_EXC_ERROR THEN

        out_return_status := FND_API.G_RET_STS_ERROR;

        --Get message count and data

        dbms_output.put_line('err msg 1 is : ' || out_msg_data);

    WHEN FND_API.G_EXC_UNEXPECTED_ERROR THEN

        out_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;
        dbms_output.put_line(' msg count 2 is : ' || out_msg_count);

        for k in 1 .. out_msg_count loop

            out_msg_data := oe_msg_pub.get( p_msg_index => k,
p_encoded => 'F'
            );
            -- Get message count and data
            dbms_output.put_line('err msg ' || k || 'is: ' || out_msg_data);
            null;

        end loop;

    WHEN OTHERS THEN

        out_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;
        --Get message count and data
        dbms_output.put_line('err msg 3 is : ' || out_msg_data);

        for k in 1 .. out_msg_count loop
            out_msg_data := oe_msg_pub.get( p_msg_index => k,

p_encoded => 'F'

            );

            --Get message count and data

            dbms_output.put_line('err msg ' || k || 'is: ' || out_msg_data);
            null;
        end loop;

```



```
end;
/
--commit;
--exit;
```

Update an agreement record and update an agreement price list header

This example also creates three price list lines on the agreement price list.

```
SET SERVEROUTPUT ON SIZE 200000
```

```
declare
```

```
out_return_status varchar2(1) := NULL;
out_msg_count number := 0;
out_msg_data varchar2(2000);
in_Agreement_rec      OE_Pricing_Cont_PUB.Agreement_Rec_Type;
in_Agreement_val_rec  OE_Pricing_Cont_PUB.Agreement_Val_Rec_Type;
in_price_list_rec     QP_PRICE_LIST_PUB.Price_List_Rec_Type;
in_price_list_val_rec QP_PRICE_LIST_PUB.Price_List_Val_Rec_Type;
in_price_list_line_tbl QP_PRICE_LIST_PUB.Price_List_Line_Tbl_Type;
in_price_list_line_val_tbl QP_PRICE_LIST_PUB.Price_List_Line_Val_Tbl_Type;
in_pricing_attr_tbl   QP_PRICE_LIST_PUB.Pricing_Attr_Tbl_Type;
in_pricing_attr_val_tbl QP_PRICE_LIST_PUB.Pricing_Attr_Val_Tbl_Type;
out_Agreement_rec      OE_Pricing_Cont_PUB.Agreement_Rec_Type;
out_Agreement_val_rec  OE_Pricing_Cont_PUB.Agreement_Val_Rec_Type;
out_price_list_rec     QP_PRICE_LIST_PUB.Price_List_Rec_Type;
out_price_list_val_rec QP_PRICE_LIST_PUB.Price_List_Val_Rec_Type;
out_price_list_line_tbl QP_PRICE_LIST_PUB.Price_List_Line_Tbl_Type;
out_price_list_line_val_tbl QP_PRICE_LIST_PUB.Price_List_Line_Val_Tbl_Type;
out_pricing_attr_tbl   QP_PRICE_LIST_PUB.Pricing_Attr_Tbl_Type;
out_pricing_attr_val_tbl QP_PRICE_LIST_PUB.Pricing_Attr_Val_Tbl_Type;
```

```
    K number := 1;
    j number := 1;
```

```
begin
```

```
  dbms_output.put_line('Creating an Agreement record');
```

```
  --Updating an Agreement record
```

```
  in_Agreement_rec.agreement_id :=19034;
  in_Agreement_rec.name :='Test SEAGATE 2001';
  in_Agreement_rec.creation_date :=sysdate;
```

```

in_Agreement_rec.created_by := 1001;
in_Agreement_rec.last_update_date := sysdate;
in_Agreement_rec.last_updated_by := 9001;
in_Agreement_rec.agreement_type_code := 'STANDARD';
in_Agreement_rec.agreement_num := '22';
in_Agreement_rec.revision := '1';
in_Agreement_rec.revision_date := sysdate;
in_Agreement_rec.term_id := 1000;
in_Agreement_rec.OVERRIDE_IRULE_FLAG := 'N';
in_Agreement_rec.OVERRIDE_ARULE_FLAG := 'Y';
in_Agreement_rec.operation      := QP_GLOBALS.G_OPR_UPDATE;
in_Agreement_rec.price_list_id := 107811;

    /* set the list_header_id to g_miss_num */
    in_price_list_rec.list_header_id := 107811;
    in_price_list_rec.name := 'SEAGATE P LIST 2002';
    in_price_list_rec.list_type_code := 'AGR';
    in_price_list_rec.description := 'Sample TESTING AGREEMENT';
    in_price_list_rec.currency_code := 'USD';
    in_price_list_rec.operation      := QP_GLOBALS.G_OPR_UPDATE;

FOR K IN 1..3 LOOP
in_price_list_line_tbl(K).list_line_id := FND_API.G_MISS_NUM;
in_price_list_line_tbl(K).list_line_type_code := 'PLL';
in_price_list_line_tbl(K).operation := QP_GLOBALS.G_OPR_CREATE;
in_price_list_line_tbl(K).operand := 101;
in_price_list_line_tbl(K).arithmetic_operator := 'UNIT_PRICE';
END LOOP;

/*
product_attr_value stores inventory item id -
product_attribute for Item Number is Pricing_Attribute1
product_attribute_context is ITEM .
Each line can have one or more pricing attributes. we use
PRICE_LIST_LINE_INDEX to link the child(pricing attributes )
to the parent(line).
When you have pricing attributes like color, length, width etc,
populate the fields pricing_attribute_context, pricing_attribute
, pricing_attr_value_from, pricing_attr_value_to and
comparison_operator_code ( '=' or 'between') and repeat the
product_attr_value and its attribute and context for each record.
*/

J := 1;

```

```

in_pricing_attr_tbl(J).pricing_attribute_id := FND_API.G_MISS_NUM;
in_pricing_attr_tbl(J).list_line_id := FND_API.G_MISS_NUM;
in_pricing_attr_tbl(J).PRODUCT_ATTRIBUTE_CONTEXT := 'ITEM';
in_pricing_attr_tbl(J).PRODUCT_ATTRIBUTE := 'PRICING_ATTRIBUTE1';
in_pricing_attr_tbl(J).PRODUCT_ATTR_VALUE := '62061';
in_pricing_attr_tbl(J).PRODUCT_UOM_CODE := 'Ea';
in_pricing_attr_tbl(J).EXCLUDER_FLAG := 'N';
in_pricing_attr_tbl(J).ATTRIBUTE_GROUPING_NO := 1;
in_pricing_attr_tbl(J).PRICE_LIST_LINE_INDEX := 1;
in_pricing_attr_tbl(J).operation := QP_GLOBALS.G_OPR_CREATE;

J := J + 1;

in_pricing_attr_tbl(J).pricing_attribute_id := FND_API.G_MISS_NUM;
in_pricing_attr_tbl(J).list_line_id := FND_API.G_MISS_NUM;
in_pricing_attr_tbl(J).PRODUCT_ATTRIBUTE_CONTEXT := 'ITEM';
in_pricing_attr_tbl(J).PRODUCT_ATTRIBUTE := 'PRICING_ATTRIBUTE1';
in_pricing_attr_tbl(J).PRODUCT_ATTR_VALUE := '62081';
in_pricing_attr_tbl(J).PRODUCT_UOM_CODE := 'Ea';
in_pricing_attr_tbl(J).EXCLUDER_FLAG := 'N';
in_pricing_attr_tbl(J).ATTRIBUTE_GROUPING_NO := 1;
in_pricing_attr_tbl(J).PRICE_LIST_LINE_INDEX := 2;
in_pricing_attr_tbl(J).operation := QP_GLOBALS.G_OPR_CREATE;

J := J + 1;

in_pricing_attr_tbl(J).pricing_attribute_id := FND_API.G_MISS_NUM;
in_pricing_attr_tbl(J).list_line_id := FND_API.G_MISS_NUM;
in_pricing_attr_tbl(J).PRODUCT_ATTRIBUTE_CONTEXT := 'ITEM';
in_pricing_attr_tbl(J).PRODUCT_ATTRIBUTE := 'PRICING_ATTRIBUTE1';
in_pricing_attr_tbl(J).PRODUCT_ATTR_VALUE := '62083';
in_pricing_attr_tbl(J).PRODUCT_UOM_CODE := 'Ea';
in_pricing_attr_tbl(J).EXCLUDER_FLAG := 'N';
in_pricing_attr_tbl(J).ATTRIBUTE_GROUPING_NO := 1;
in_pricing_attr_tbl(J).PRICE_LIST_LINE_INDEX := 3;
in_pricing_attr_tbl(J).operation := QP_GLOBALS.G_OPR_CREATE;

oe_debug_pub.add('Calling the process Agreements API');

```

OE_Pricing_Cont_PUB.Process_Agreement

```

(   p_api_version_number=> 1.0
,   p_init_msg_list=> FND_API.G_FALSE
,   p_return_values=> FND_API.G_FALSE
,   p_commit=> FND_API.G_FALSE
,   x_return_status=> out_return_status

```

```

,   x_msg_count=> out_msg_count
,   x_msg_data=> out_msg_data
,   p_Agreement_rec=> in_Agreement_rec
,   p_Price_LHeader_rec=> in_price_list_rec
,   p_Price_LLine_tbl=> in_price_list_line_tbl
,   p_Pricing_Attr_tbl=> in_pricing_attr_tbl
,   x_Agreement_rec=> out_Agreement_rec
,   x_Agreement_val_rec=> out_Agreement_val_rec
,   x_Price_LHeader_rec=> out_price_list_rec
,   x_Price_LHeader_val_rec=> out_price_list_val_rec
,   x_Price_LLine_tbl=> out_price_list_line_tbl
,   x_Price_LLine_val_tbl=> out_price_list_line_val_tbl
,   x_Pricing_Attr_tbl=> out_pricing_attr_tbl
,   x_Pricing_Attr_val_tbl=> out_pricing_attr_val_tbl
);

IF out_return_status <> FND_API.G_RET_STS_SUCCESS THEN

    RAISE FND_API.G_EXC_UNEXPECTED_ERROR;

END IF;

oe_debug_pub.add('after process agreement ');

EXCEPTION

    WHEN FND_API.G_EXC_ERROR THEN

        out_return_status := FND_API.G_RET_STS_ERROR;

        -- Get message count and data

        dbms_output.put_line('err msg 1 is : ' || out_msg_data);

    WHEN FND_API.G_EXC_UNEXPECTED_ERROR THEN

        out_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;
        dbms_output.put_line(' msg count 2 is : ' || out_msg_count);

        for k in 1 .. out_msg_count loop

            out_msg_data := oe_msg_pub.get( p_msg_index => k,

            p_encoded => 'F'

```

```

);

-- Get message count and data
dbms_output.put_line('err msg ' || k || 'is: ' || out_msg_data);

null;

end loop;

WHEN OTHERS THEN

    out_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;
--Get message count and data
    dbms_output.put_line('err msg 3 is : ' || out_msg_data);

for k in 1 .. out_msg_count loop
out_msg_data := oe_msg_pub.get( p_msg_index => k,

    p_encoded => 'F'

);

--Get message count and data

dbms_output.put_line('err msg ' || k || 'is: ' || out_msg_data);
null;
end loop;
end;

/

--commit;

--exit;

```

Delete an agreement

You only need to specify the Agreement_id. If the agreement has a price list that is not used by any order or any other agreement, then the price list will also be deleted.

```

SET SERVEROUTPUT ON SIZE 200000

/* This scripts DELETES an agreement */

```

```

declare
    out_return_status varchar2(1) := NULL;
    out_msg_count number := 0;
    out_msg_data varchar2(2000);
    in_Agreement_rec      OE_Pricing_Cont_PUB.Agreement_Rec_Type;
    in_Agreement_val_rec OE_Pricing_Cont_PUB.Agreement_Val_Rec_Type;
    in_price_list_rec QP_PRICE_LIST_PUB.Price_List_Rec_Type;
    in_price_list_val_rec QP_PRICE_LIST_PUB.Price_List_Val_Rec_Type;
    in_price_list_line_tbl QP_PRICE_LIST_PUB.Price_List_Line_Tbl_Type;
    in_price_list_line_val_tbl QP_PRICE_LIST_PUB.Price_List_Line_Val_Tbl_Type;
    in_pricing_attr_tbl QP_PRICE_LIST_PUB.Pricing_Attr_Tbl_Type;
    in_pricing_attr_val_tbl QP_PRICE_LIST_PUB.Pricing_Attr_Val_Tbl_Type;
    out_Agreement_rec      OE_Pricing_Cont_PUB.Agreement_Rec_Type;
    out_Agreement_val_rec OE_Pricing_Cont_PUB.Agreement_Val_Rec_Type;
    out_price_list_rec QP_PRICE_LIST_PUB.Price_List_Rec_Type;
    out_price_list_val_rec QP_PRICE_LIST_PUB.Price_List_Val_Rec_Type;
    out_price_list_line_tbl QP_PRICE_LIST_PUB.Price_List_Line_Tbl_Type;
    out_price_list_line_val_tbl QP_PRICE_LIST_PUB.Price_List_Line_Val_Tbl_Type;
    out_pricing_attr_tbl QP_PRICE_LIST_PUB.Pricing_Attr_Tbl_Type;
    out_pricing_attr_val_tbl QP_PRICE_LIST_PUB.Pricing_Attr_Val_Tbl_Type;

    K number := 1;

    j number := 1;

begin

    dbms_output.put_line('DELETING an Agreement record');

    -- DELETING an Agreement record

    in_Agreement_rec.agreement_id :=19121;
    in_Agreement_rec.operation      := QP_GLOBALS.G_OPR_DELETE;

    oe_debug_pub.add('Calling the process Agreements API');

```

OE_Pricing_Cont_PUB.Process_Agreement

```

(   p_api_version_number=> 1.0
,   p_init_msg_list=> FND_API.G_FALSE
,   p_return_values=> FND_API.G_FALSE
,   p_commit=> FND_API.G_FALSE
,   x_return_status=> out_return_status
,   x_msg_count=> out_msg_count

```

```

,   x_msg_data=> out_msg_data
,   p_Agreement_rec=> in_Agreement_rec
,   p_Price_LHeader_rec=> in_price_list_rec
,   p_Price_LLine_tbl=> in_price_list_line_tbl
,   p_Pricing_Attr_tbl=> in_pricing_attr_tbl
,   x_Agreement_rec=> out_Agreement_rec
,   x_Agreement_val_rec=> out_Agreement_val_rec
,   x_Price_LHeader_rec=> out_price_list_rec
,   x_Price_LHeader_val_rec=> out_price_list_val_rec
,   x_Price_LLine_tbl=> out_price_list_line_tbl
,   x_Price_LLine_val_tbl=> out_price_list_line_val_tbl
,   x_Pricing_Attr_tbl=> out_pricing_attr_tbl
,   x_Pricing_Attr_val_tblout_pricing_attr_val_tbl

);

IF out_return_status <> FND_API.G_RET_STS_SUCCESS THEN

    RAISE FND_API.G_EXC_UNEXPECTED_ERROR;

END IF;

oe_debug_pub.add('after process agreement ');

EXCEPTION

    WHEN FND_API.G_EXC_ERROR THEN

        out_return_status := FND_API.G_RET_STS_ERROR;

        Get message count and data

        dbms_output.put_line('err msg 1 is : ' || out_msg_data);

    WHEN FND_API.G_EXC_UNEXPECTED_ERROR THEN

        out_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;
        dbms_output.put_line(' msg count 2 is : ' || out_msg_count);

        for k in 1 .. out_msg_count loop

out_msg_data := oe_msg_pub.get( p_msg_index => k,

p_encoded => 'F'

);

```

```

        --Get message count and data

        dbms_output.put_line('err msg ' || k || 'is: ' || out_msg_data);

null;

end loop;

    WHEN OTHERS THEN

        out_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;

        --Get message count and data
        dbms_output.put_line('err msg 3 is : ' || out_msg_data);

    for k in 1 .. out_msg_count loop
        out_msg_data := oe_msg_pub.get( p_msg_index => k,
            p_encoded => 'F'

        );

        -- Get message count and data
        dbms_output.put_line('err msg ' || k || 'is: ' || out_msg_data);
    null;
    end loop;

end;
/
--commit;
--exit;

```


Attribute Mapping Application Program Interface

This section explains how to use the Attribute Mapping APIs and how it functions in Oracle Advanced Pricing. Currently, in the new model, there are three Attribute Mapping packages. They are :

- QP_ATTR_MAP_PUB
- QP_ATTRIBUTES_PUB
- QP_ATTR_MAPPING_PUB

Functional Overview

The Public package QP_ATTR_MAP_PUB is a Business Object API, based on the following tables. QP_LOOKUPS(Type : QP_PTE_TYPE), QP_PTE_SOURCE_SYSTEMS, QP_PTE_REQUEST_TYPES_B/TL, QP_PTE_SEGMENTS and QP_ATTRIBUTE_SOURCING. The QP_ATTR_MAP_PUB model is as shown below :

The Object Name is Attr_Map and the relationship of the tables is shown below.

The package QP_ATTR_MAP_PUB contains the following APIs and record type definitions:

- The API processes one record with each call. The calling application must populate the global records that referenced in the attribute mapping rules. The output of the API is a PL/SQL table with each record having the context name, attribute name, and attribute value.
- Pte_Val_Rec_Type
- Pte_Val_Rec_Type
- Rqt_Rec_Type
- Rqt_Tbl_Type
- Rqt_Val_Rec_Type
- Ssc_Rec_Type
- Ssc_Tbl_Type
- Ssc_Val_Rec_Type
- Psg_Rec_Type
- Psg_Val_Rec_Type
- Sou_Rec_Type

- Sou_Val_Rec_Type
- PROCEDURE Process_Attr_Mapping
- PROCEDURE Lock_Attr_Mapping
- PROCEDURE Get_Attr_Mapping

Setting Up and Parameter Descriptions

The following chart describes all parameters used by the public Attribute Mapping API QP_ATTR_MAP_PUB. All of the inbound and outbound parameters are listed. Additional information on these parameters may follow.

Procedure **PROCESS_ATTR_MAPPING**

The following table shows the parameters for this structure. This API will add, update or delete a Source System, Request type, PTE-Attribute link, Attribute Mapping rules depending on the input parameter.

Table 5–21 PROCESS_ATTR_MAPPING Parameters

Parameter	Usage	Type	Req	Drv
p_api_version_number	In	Number	Yes	No
p_init_msg_list	In	Varchar2	No	No
p_return_values	In	Varchar2	No	No
p_commit	In	Varchar2	No	No
x_return_status	Out	Number	No	No
x_msg_count	Out	Varchar2	No	No
x_msg_data	Out	Pte_Rec_Type	No	No
p_PTE_rec	In	Pte_Val_Rec_Type	No	No
p_PTE_val_rec	In	Pte_Val_Rec_Type	No	No
p_RQT_tbl	In	Rqt_Tbl_Type	No	No
p_RQT_val_tbl	In	Rqt_Val_Tbl_Type	No	No
p_SSC_tbl	In	Ssc_Tbl_Type	No	No
p_SSC_val_tbl	In	Ssc_Val_Tbl_Type	No	No
p_PSG_tbl	In	Psg_Tbl_Type	No	No

Table 5–21 PROCESS_ATTR_MAPPING Parameters

Parameter	Usage	Type	Req	Drv
p_PSG_val_tbl	In	Psg_Val_Tbl_Type	No	No
p_SOU_tbl	In	Sou_Tbl_Type	No	No
p_SOU_val_tbl	In	Sou_Val_Tbl_Type	No	No
x_PTE_rec	Out	Pte_Rec_Type	No	No
x_PTE_val_rec	Out	Pte_Val_Rec_Type	No	No
x_RQT_tbl	Out	Rqt_Tbl_Type	No	No
x_RQT_val_tbl	Out	Rqt_Val_Tbl_Type	No	No
x_SSC_tbl	Out	Ssc_Tbl_Type	No	No
x_SSC_val_tbl	Out	Ssc_Val_Tbl_Type	No	No
x_PSG_tbl	Out	Psg_Tbl_Type	No	No
x_PSG_val_tbl	Out	Psg_Val_Tbl_Type	No	No
x_SOU_tbl	Out	Sou_Tbl_Type	No	No
x_SOU_val_tbl	Out	Sou_Val_Tbl_Type	No	No

p_api_version_number

This the version number of the API.

P_PTE_REC

The following table shows the parameters for this structure.

Table 5–22 P_PTE_REC Parameters

Parameter	Usage	Type	Req	Drv
Description	Null	Varchar2	No	No
Enabled_flag	Null	Varchar2	No	No
End_date_active	Null	Date	No	No
Lookuop_code	Null	Varchar2	No	No
Lookup_type	Null	Varchar2	No	No
meaning	Null	Varchar2	No	No
Start_date_active	Null	Date	No	No

P_PTE_VAL_REC

The following table shows the parameters for this structure.

Table 5–23 P_PTE_VAL_REC Parameters

Parameter	Usage	Type	Req	Drv
enabled	Null	Varchar2	No	No
lookup	Null	Varchar2	No	No

The following table shows the parameters for the tables and their structures.

Table 5–24 Tables and Structures

Table Name	Usage	Type
P_RQT_TBL	Null	TABLE OF Rqt_Rec_Type
P_RQT_VAL_TBL	Null	TABLE OF Pte_Val_Rec_Type
P_SSC_TBL	Null	TABLE OF Ssc_Rec_Type
P_SSC_VAL_TBL	Null	TABLE OF Ssc_Rec_Type
P_PSG_TBL	Null	TABLE OF Psg_Rec_Type
P_PSG_VAL_TBL	Null	TABLE OF Psg_Val_Rec_Type
P_SOU_TBL	Null	TABLE OF Psg_Sou_Type
P_SOU_VAL_TBL	Null	TABLE OF Psg_Sou_Val_Type

X_PTE_REC_TYPE

The following table shows the parameters for this structure.

Table 5–25 X_PTE_REC_TYPE Parameters

Parameter	Usage	Type	Req	Drv
Description	Null	Varchar2	No	No
Enabled_flag	Null	Varchar2	No	No
End_date_active	Null	Date	No	No
Lookuop_code	Null	Varchar2	No	No
Lookup_type	Null	Varchar2	No	No

Table 5–25 X_PTE_REC_TYPE Parameters

Parameter	Usage	Type	Req	Drv
meaning	Null	Varchar2	No	No
Start_date_active	Null	Date	No	No

X_PTE_VAL_REC

The following table shows the parameters for this structure.

Table 5–26 X_PTE_VAL_REC Parameters

Parameter	Usage	Type	Req	Drv
enabled	Null	Varchar2	No	No
lookup	Null	Varchar2	No	No

The following table shows the parameters for the following table structures (no parameters are used for the following):

Table 5–27 Table Parameters

Table Name	Usage	Type	Req	Drv
X_RQT_TBL	Null	TABLE OF Rqt_Rec_Type	No	No
X_RQT_VAL_TBL	Null	TABLE OF Pte_Val_Rec_Type	No	No
P_SSC_TBL	Null	TABLE OF Ssc_Rec_Type	No	No
P_SSC_VAL_TBL	Null	TABLE OF Ssc_Rec_Type	No	No
X_PSG_TBL	Null	TABLE OF Psg_Rec_Type	No	No
X_PSG_VAL_TBL	Null	TABLE OF Psg_Val_Rec_Type	No	No
X_SOU_TBL	Null	TABLE OF Psg_Sou_Type	No	No
X_SOU_VAL_TBL	Null	TABLE OF Psg_Sou_Val_Type	No	No

Procedure LOCK_ATTR_MAPPING

The following table shows the parameters for this structure. User can use this API to lock PTE-Attribute link and all its Attribute Mapping rules from getting updated by a different user concurrently.

Table 5–28 LOCK_ATTR_MAPPING Parameters

Parameter	Usage	Type	Req	Drv
p_api_version_number	In	Number	Yes	No
p_init_msg_list	In	Varchar2	No	No
p_return_values	In	Varchar2	No	No
x_return_status	Out	Number	No	No
x_msg_count	Out	Varchar2	No	No
x_msg_data	Out	Pte_Rec_Type	No	No
p_PTE_rec	In	Pte_Val_Rec_Type	No	No
p_PTE_val_rec	In	Pte_Val_Rec_Type	No	No
p_RQT_tbl	In	Rqt_Tbl_Type	No	No
p_RQT_val_tbl	In	Rqt_Val_Tbl_Type	No	No
p_SSC_tbl	In	Ssc_Tbl_Type	No	No
p_SSC_val_tbl	In	Ssc_Val_Tbl_Type	No	No
p_PSG_tbl	In	Psg_Tbl_Type	No	No
p_PSG_val_tbl	In	Psg_Val_Tbl_Type	No	No
p_SOU_tbl	In	Sou_Tbl_Type	No	No
p_SOU_val_tbl	In	Sou_Val_Tbl_Type	No	No
x_PTE_rec	Out	Pte_Rec_Type	No	No
x_PTE_val_rec	Out	Pte_Val_Rec_Type	No	No
x_RQT_tbl	Out	Rqt_Tbl_Type	No	No
x_RQT_val_tbl	Out	Rqt_Val_Tbl_Type	No	No
x_SSC_tbl	Out	Ssc_Tbl_Type	No	No
x_SSC_val_tbl	Out	Ssc_Val_Tbl_Type	No	No
x_PSG_tbl	Out	Psg_Tbl_Type	No	No
x_PSG_val_tbl	Out	Psg_Val_Tbl_Type	No	No
x_SOU_tbl	Out	Sou_Tbl_Type	No	No
x_SOU_val_tbl	Out	Sou_Val_Tbl_Type	No	No

p_api_version_number

This the version number of the API.

P_PTE_REC_TYPE

The following table shows the parameters for this structure.

Table 5–29 P_PTE_REC_TYPE Parameters

Parameter	Usage	Type	Req	Drv
Description	Null	Varchar2	No	No
Enabled_flag	Null	Varchar2	No	No
End_date_active	Null	Date	No	No
Lookuop_code	Null	Varchar2	No	No
Lookup_type	Null	Varchar2	No	No
meaning	Null	Varchar2	No	No
Start_date_active	Null	Date	No	No

P_PTE_VAL_REC

The following table shows the parameters for this structure.

Table 5–30 P_PTE_VAL_REC Parameters

Parameter	Usage	Type	Req	Drv
enabled	Null	Varchar2	No	No
lookup	Null	Varchar2	No	No

The following table shows the parameters for the following table structures:

Table 5–31 Table Structure Parameters

Table Name	Usage	Type
P_RQT_TBL	Null	TABLE OF Rqt_Rec_Type
P_RQT_VAL_TBL	Null	TABLE OF Pte_Val_Rec_Type
P_SSC_TBL	Null	TABLE OF Ssc_Rec_Type
P_SSC_VAL_TBL	Null	TABLE OF Ssc_Rec_Type

Table 5–31 Table Structure Parameters

Table Name	Usage	Type
P_PSG_TBL	Null	TABLE OF Psg_Rec_Type
P_PSG_VAL_TBL	Null	TABLE OF Psg_Val_Rec_Type
P_SOU_TBL	Null	TABLE OF Psg_Sou_Type
P_SOU_VAL_TBL	Null	TABLE OF Psg_Sou_Val_Type

X_PTE_REC_TYPE

The following table shows the parameters for this structure.

Table 5–32 X_PTE_REC_TYPE Parameters

Parameter	Usage	Type	Req	Drv
Description	Null	Varchar2	No	No
Enabled_flag	Null	Varchar2	No	No
End_date_active	Null	Date	No	No
Lookuop_code	Null	Varchar2	No	No
Lookup_type	Null	Varchar2	No	No
meaning	Null	Varchar2	No	No
Start_date_active	Null	Date	No	No

X_PTE_VAL_REC

The following table shows the parameters for this structure.

Table 5–33 X_PTE_VAL_REC Parameters

Parameter	Usage	Type	Req	Drv
enabled	Null	Varchar2	No	No
lookup	Null	Varchar2	No	No

The following table shows the parameters for the following table structures:

Table 5–34 Table Structure Parameters

Table Name	Usage	Type
X_RQT_TBL	Null	TABLE OF Rqt_Rec_Type
X_RQT_VAL_TBL	Null	TABLE OF Pte_Val_Rec_Type
P_SSC_TBL	Null	TABLE OF Ssc_Rec_Type
P_SSC_VAL_TBL	Null	TABLE OF Psg_Rec_Type
X_PSG_TBL	Null	TABLE OF Psg_Rec_Type
X_PSG_VAL_TBL	Null	TABLE OF Psg_Val_Rec_Type
X_SOU_TBL	Null	TABLE OF Psg_Sou_Type
X_SOU_VAL_TBL	Null	TABLE OF Psg_Sou_Val_Type

Procedure GET_ATTR_MAPPING

The following table shows the parameters for this structure. This API will fetch the records for an Source System, Request Type, PTE_Attribute link, Attribute Mapping rules given the input parameter.

Table 5–35 GET_ATTR_MAPPING Parameters

Parameter	Usage	Type	Req	Drv
p_api_version_number	In	Number	Yes	No
p_init_msg_list	In	Varchar2	No	No
p_return_values	In	Varchar2	No	No
x_return_status	Out	Number	No	No
x_msg_count	Out	Varchar2	No	No
x_msg_data	Out	Pte_Rec_Type	No	No
p_lookup_code	In	Varchar2	No	No
p_lookup	In	Varchar2	No	No
x_PTE_rec	Out	Pte_Rec_Type	No	No
x_PTE_val_rec	Out	Pte_Val_Rec_Type	No	No
x_RQT_tbl	Out	Rqt_Tbl_Type	No	No
x_RQT_val_tbl	Out	Rqt_Val_Tbl_Type	No	No

Table 5–35 GET_ATTR_MAPPING Parameters

Parameter	Usage	Type	Req	Drv
x_SSC_tbl	Out	Ssc_Tbl_Type	No	No
x_SSC_val_tbl	Out	Ssc_Val_Tbl_Type	No	No
x_PSG_tbl	Out	Psg_Tbl_Type	No	No
x_PSG_val_tbl	Out	Psg_Val_Tbl_Type	No	No
x_SOU_tbl	Out	Sou_Tbl_Type	No	No
x_SOU_val_tbl	Out	Sou_Val_Tbl_Type	No	No

p_api_version_number

This the version number of the API.

p_lookup_code

This is the code for the Pricing Transaction Entity; for example, ORDFUL for Order Fulfillment, DEMAND for Demand Planning etc.

p_lookup

This is the type of the Pricing Lookups. For Pricing Transaction Entities, it is QP_PTE_TYPE.

X_PTE_REC_TYPE

The following table shows the parameters for this structure.

Table 5–36 X_PTE_REC_TYPE Parameters

Parameter	Usage	Type	Req	Drv
Description	Null	Varchar2	No	No
Enabled_flag	Null	Varchar2	No	No
End_date_active	Null	Date	No	No
Lookuop_code	Null	Varchar2	No	No
Lookup_type	Null	Varchar2	No	No
meaning	Null	Varchar2	No	No
Start_date_active	Null	Date	No	No

X_PTE_VAL_REC

The following table shows the parameters for this structure.

Table 5–37 X_PTE_VAL_REC Parameters

Parameter	Usage	Type	Req	Drv
enabled	Null	Varchar2	No	No
lookup	Null	Varchar2	No	No

The following table shows the parameters for the following table structures:

Table 5–38 Table Structure Parameters

Table Name	Usage	Type
X_RQT_TBL	Null	TABLE OF Rqt_Rec_Type
X_RQT_VAL_TBL	Null	TABLE OF Pte_Val_Rec_Type
P_SSC_TBL	Null	TABLE OF Ssc_Rec_Type
P_SSC_VAL_TBL	Null	TABLE OF Ssc_Rec_Type
X_PSG_TBL	Null	TABLE OF Psg_Rec_Type
X_PSG_VAL_TBL	Null	TABLE OF Psg_Val_Rec_Type
X_SOU_TBL	Null	TABLE OF Psg_Sou_Type
X_SOU_VAL_TBL	Null	TABLE OF Psg_Sou_Val_Type

Business Object for Modifier Setup Application Program Interface

This section explains how to use the Business Object for Modifier Setup API and how it functions in Oracle Advanced Pricing. The Business Object for Modifier Setup package consists of entities to set up modifiers.

Functional Overview

The package QP_Modifiers_PUB.Process Modifiers contains the following public record type and table of records entities:

- **Process_Modifiers:** QP_Modifiers_PUB.Process_Modifiers is a Public API. It takes two record types and six table types as input parameters. Use this API to insert, update and delete modifiers. Use it to set up a modifier list header for a given p_MODIFIER_LIST_rec record structure.

You can:

- Set up multiple modifier lines by giving multiple modifier definitions in the p_MODIFIERS_tbl table structure.
- Attach multiple qualifiers either at the header level (modifier list) or at the line level (modifier) by giving multiple qualifiers in the p_QUALIFIERS_tbl table structure.
- Attach multiple pricing attributes to modifier lines by giving the pricing attributes in the p_PRICING_ATTR_tbl table structure.
- **Modifier_List_Rec_Type:** Corresponds to the columns in the modifier header tables QP_LIST_HEADERS_B and QP_LIST_HEADERS_TL.
- **Modifier_List_Tbl_Type**
- **Modifier_List_Val_Rec_Type:** Corresponds to the columns in the modifier header table QP_LIST_HEADERS_B.
- **Modifier_List_Val_Tbl_Type**
- **Modifiers_Rec_Type:** Corresponds to the columns in the modifier and related modifiers tables QP_LIST_LINES and QP_RLTD_MODIFIERS.
- **Modifiers_Tbl_Type**
- **Modifiers_Val_Rec_Type:** Corresponds to the columns in the modifier table QP_LIST_LINES.
- **Modifiers_Val_Tbl_Type**

- Qualifiers_Rec_Type: Corresponds to the columns in the qualifier table QP_QUALIFIERS.
- Qualifiers_Tbl_Type
- Qualifiers_Val_Rec_Type: Corresponds to the columns in the qualifier table QP_QUALIFIERS.
- Qualifiers_Val_Tbl_Type
- Pricing_Attr_Rec_Type: Corresponds to the columns in the pricing attributes table QP_PRICING_ATTRIBUTES.
- Pricing_Attr_Tbl_Type
- Pricing_Attr_Val_Rec_Type: Corresponds to the columns in the pricing attributes table QP_PRICING_ATTRIBUTES.
- Pricing_Attr_Val_Tbl_Type

Setting Up and Parameter Descriptions

The following chart describes all parameters used by the public Business Object for Modifier Setup. All of the inbound and outbound parameters are listed. Additional information on these parameters may follow.

PROCESS_MODIFIERS

The following table shows the parameters for this structure.

Table 5–39 PROCESS_MODIFIERS Parameters

Parameter	Usage	Type	Req	Drv
p_api_version_number	In	Number	No	No
p_init_msg_list	In	Varchar2	No	No
p_return_values	In	Varchar2	No	No
p_commit	In	Varchar2	No	No
x_return_status	Out	Varchar2	No	No
x_msg_count	Out	Varchar2	No	No
x_msg_data	Out	Varchar2	No	No
p_MODIFIER_LIST_rec	In	Modifier_List_Rec_Type	No	No
p_MODIFIER_LIST_val_rec	In	Modifier_List_Val_Rec_Type	No	No

Table 5–39 PROCESS_MODIFIERS Parameters

Parameter	Usage	Type	Req	Drv
p_MODIFIERS_tbl	In	Modifiers_Tbl_Type	No	No
p_MODIFIERS_val_tbl	In	Modifiers_Val_Tbl_Type	No	No
p_QUALIFIERS_tbl	In	Qualifiers_Tbl_Type	No	No
p_QUALIFIERS_val_tbl	In	Qualifiers_Val_Tbl_Type	No	No
p_PRICING_ATTR_tbl	In	Pricing_Attr_Tbl_Type	No	No
p_PRICING_ATTR_val_tbl	In	Pricing_Attr_Val_Tbl_Type	No	No
x_MODIFIER_LIST_rec	Out	Modifier_List_Rec_Type	No	No
x_MODIFIER_LIST_val_rec	Out	Modifier_List_Val_Rec_Type	No	No
x_MODIFIERS_tbl	Out	Modifiers_Tbl_Type	No	No
x_MODIFIERS_val_tbl	Out	Modifiers_Val_Tbl_Type	No	No
x_QUALIFIERS_tbl	Out	Qualifiers_Tbl_Type	No	No
x_QUALIFIERS_val_tbl	Out	Qualifiers_Val_Tbl_Type	No	No
x_PRICING_ATTR_tbl	Out	Pricing_Attr_Tbl_Type	No	No
x_PRICING_ATTR_val_tbl	Out	Pricing_Attr_Val_Tbl_Type	No	No

p_init_sg_list

Default Value: FND_API.G_FALSE

p_return_values

Default Value: FND_API.G_FALSE

p_commit

Default Value: FND_API.G_FALSE

p_MODIFIER_LIST_rec

Default Value: G_MISS_MODIFIER_LIST_REC

p_MODIFIER_LIST_val_rec

Default Value: G_MISS_MODIFIER_LIST_VAL_REC

p_MODIFIERS_tbl

Default Value: G_MISS_MODIFIERS_TBL

p_MODIFIERS_val_tbl

Default Value: G_MISS_MODIFIERS_VAL_TBL

p_QUALIFIERS_tbl

Default Value: G_MISS_QUALIFIERS_TBL

p_QUALIFIERS_val_tbl

Default Value: G_MISS_QUALIFIERS_VAL_TBL

p_PRICING_ATTR_tbl

Default Value: G_MISS_PRICING_ATTR_TBL

p_PRICING_ATTR_val_tbl

Default Value: G_MISS_PRICING_ATTR_VAL_TBL

MODIFIER_LIST_REC_TYPE

The following table shows the parameters for this structure.

Table 5–40 MODIFIER_LIST_REC_TYPE Parameters

Parameter	Usage	Type	Req	Drv
attribute1	Null	Varchar2	No	No
attribute2	Null	Varchar2	No	No
attribute3	Null	Varchar2	No	No
attribute4	Null	Varchar2	No	No
attribute5	Null	Varchar2	No	No
attribute6	Null	Varchar2	No	No
attribute7	Null	Varchar2	No	No
attribute8	Null	Varchar2	No	No
attribute9	Null	Varchar2	No	No
attribute10	Null	Varchar2	No	No
attribute11	Null	Varchar2	No	No

Table 5–40 MODIFIER_LIST_REC_TYPE Parameters

Parameter	Usage	Type	Req	Drv
attribute12	Null	Varchar2	No	No
attribute13	Null	Varchar2	No	No
attribute14	Null	Varchar2	No	No
attribute15	Null	Varchar2	No	No
automatic_flag	Null	Varchar2	Yes	No
comments	Null	Varchar2	No	No
context	Null	Varchar2	No	No
created_by	Null	Number	No	No
creation_date	Null	Date	No	No
currency_code	Null	Varchar2	Yes	No
discount_lines_flag	Null	Varchar2	No	No
end_date_active	Null	Date	Yes	No
freight_terms_code	Null	Varchar2	No	No
gsa_indicator	Null	Varchar2	No	No
last_updated_by	Null	Number	No	No
last_update_date	Null	Date	No	No
last_update_login	Null	Number	No	No
list_header_id	Null	Number	No	No
list_type_code	Null	Varchar2	Yes	No
program_application_id	Null	Number	No	No
program_id	Null	Number	No	No
program_update_date	Null	Date	No	No
prorate_flag	Null	Varchar2	No	No
request_id	Null	Number	No	No
rounding_factor	Null	Number	No	No
ship_method_code	Null	Varchar2	No	No
start_date_active	Null	Date	No	No

Table 5–40 MODIFIER_LIST_REC_TYPE Parameters

Parameter	Usage	Type	Req	Drv
terms_id	Null	Number	No	No
source_system_code	Null	Varchar2	Yes	No
active_flag	Null	Varchar2	Yes	No
parent_list_header_id	Null	Number	No	No
start_date_active_first	Null	Date	No	No
end_date_active_first	Null	Date	No	No
active_date_first_type	Null	Varchar2	No	No
start_date_active_second	Null	Date	No	No
end_date_active_second	Null	Date	No	No
active_date_second_type	Null	Varchar2	No	No
ask_for_flag	Null	Varchar2	No	No
return_status	Null	Varchar2	No	No
db_flag	Null	Varchar2	No	No
version_no	Null	Varchar2	Yes	No
operation	Null	Varchar2	Yes	No
name	Null	Varchar2	Yes	No
pte_code	Null	Varchar2	Yes	No
description	Null	Varchar2	Yes	No

attribute1-15

Default Value: FND_API.G_MISS_CHAR

automatic_flag

Default Value: FND_API.G_MISS_CHAR

comments

Default Value: FND_API.G_MISS_CHAR

context

Default Value: FND_API.G_MISS_CHAR

created_by

Default Value: FND_API.G_MISS_NUM

creation_date

Default Value: FND_API.G_MISS_DATE

currency_code

Default Value: FND_API.G_MISS_CHAR

discount_lines_flag

Default Value: FND_API.G_MISS_CHAR

end_date_active

Default Value: FND_API.G_MISS_DATE

freight_terms_code

Default Value: FND_API.G_MISS_CHAR

gsa_indicator

Default Value: FND_API.G_MISS_CHAR

last_updated_by

Default Value: FND_API.G_MISS_NUM

last_update_date

Default Value: FND_API.G_MISS_DATE

last_update_login

Default Value: FND_API.G_MISS_NUM

list_header_id

Default Value: FND_API.G_MISS_NUM

list_type_code

Default Value: FND_API.G_MISS_CHAR

program_application_id

Default Value: FND_API.G_MISS_NUM

program_id

Default Value: FND_API.G_MISS_NUM

program_update_date

Default Value: FND_API.G_MISS_DATE

prorate_flag

Default Value: FND_API.G_MISS_CHAR

request_id

Default Value: FND_API.G_MISS_NUM

rounding_factor

Default Value: FND_API.G_MISS_NUM

ship_method_code

Default Value: FND_API.G_MISS_CHAR

start_date_active

Default Value: FND_API.G_MISS_DATE

terms_id

Default Value: FND_API.G_MISS_NUM

source_system_code

Default Value: FND_API.G_MISS_CHAR

active_flag

Default Value: FND_API.G_MISS_CHAR

parent_list_header_id

Default Value: FND_API.G_MISS_NUM

start_date_active_first

Default Value: FND_API.G_MISS_DATE

end_date_active_first

Default Value: FND_API.G_MISS_DATE

active_date_first_type

Default Value: FND_API.G_MISS_CHAR

start_date_active_second

Default Value: FND_API.G_MISS_DATE

end_date_active_second

Default Value: FND_API.G_MISS_DATE

active_date_second_type

Default Value: FND_API.G_MISS_CHAR

ask_for_flag

Default Value: FND_API.G_MISS_CHAR

return_status

Default Value: FND_API.G_MISS_CHAR

db_flag

Default Value: FND_API.G_MISS_CHAR

operation

Default Value: FND_API.G_MISS_CHAR

name

Default Value: FND_API.G_MISS_CHAR

description

Default Value: FND_API.G_MISS_CHAR

MODIFIER_LIST_TBL_TYPE The following table shows the parameters for this structure.

Table 5–41 MODIFIER_LIST_TBL_TYPE Parameters

Parameter	Usage	Type	Req	Drv
Modifier_List_Rec_Type	Null	Record	No	No

MODIFIER_LIST_VAL_REC_TYPE

The following table shows the parameters for this structure.

Table 5–42 MODIFIER_LIST_VAL_REC_TYPE Parameters

Parameter	Usage	Type	Req	Drv
automatic	Null	Varchar2	No	No
currency	Null	Varchar2	No	No
discount_lines	Null	Varchar2	No	No
freight_terms	Null	Varchar2	No	No
list_header	Null	Varchar2	No	No
list_type	Null	Varchar2	No	No
prorate	Null	Varchar2	No	No
ship_method	Null	Varchar2	No	No
terms	Null	Varchar2	No	No

automatic

Default Value: FND_API.G_MISS_CHAR

currency

Default Value: FND_API.G_MISS_CHAR

discount_lines

Default Value: FND_API.G_MISS_CHAR

freight_terms

Default Value: FND_API.G_MISS_CHAR

list_header

Default Value: FND_API.G_MISS_CHAR

list_type

Default Value: FND_API.G_MISS_CHAR

prorate

Default Value: FND_API.G_MISS_CHAR

ship_method

Default Value: FND_API.G_MISS_CHAR

terms

Default Value: FND_API.G_MISS_CHAR

MODIFIER_LIST_VAL_TBL_TYPE

The following table shows the parameters for this structure.

Table 5–43 MODIFIER_LIST_VAL_TBL_TYPE Parameters

Parameter	Usage	Type	Req	Drv
Modifier_List_Val_Rec_Type	Null	Record	No	No

MODIFIERS_REC_TYPE

The following table shows the parameters for this structure.

Table 5–44 MODIFIERS_REC_TYPE Parameters

Parameter	Usage	Type	Req	Drv
arithmetic_operator	Null	Varchar2	No	No
attribute1	Null	Varchar2	No	No
attribute2	Null	Varchar2	No	No
attribute3	Null	Varchar2	No	No

Table 5–44 MODIFIERS_REC_TYPE Parameters

Parameter	Usage	Type	Req	Drv
attribute4	Null	Varchar2	No	No
attribute5	Null	Varchar2	No	No
attribute6	Null	Varchar2	No	No
attribute7	Null	Varchar2	No	No
attribute8	Null	Varchar2	No	No
attribute9	Null	Varchar2	No	No
attribute10	Null	Varchar2	No	No
attribute11	Null	Varchar2	No	No
attribute12	Null	Varchar2	No	No
attribute13	Null	Varchar2	No	No
attribute14	Null	Varchar2	No	No
attribute15	Null	Varchar2	No	No
automatic_flag	Null	Varchar2	Yes	No
comments	Null	Varchar2	No	No
context	Null	Varchar2	No	No
created_by	Null	Number	No	No
creation_date	Null	Date	No	No
effective_period_uom	Null	Varchar2	No	No
end_date_active	Null	Date	Yes	No
estim_accrual_rate	Null	Number	No	No
generate_using_formula_id	Null	Number	No	No
inventory_item_id	Null	Number	No	No
last_updated_by	Null	Number	No	No
last_update_date	Null	Date	No	No
last_update_login	Null	Number	No	No
list_header_id	Null	Number	No	No
list_line_id	Null	Number	No	No

Table 5–44 MODIFIERS_REC_TYPE Parameters

Parameter	Usage	Type	Req	Drv
list_line_type_code	Null	Varchar2	Yes	No
list_price	Null	Number	No	No
modifier_level_code	Null	Varchar2	Yes	No
number_effective_periods	Null	Number	No	No
operand	Null	Number	No	No
organization_id	Null	Number	No	No
override_flag	Null	Varchar2	No	No
percent_price	Null	Number	No	No
price_break_type_code	Null	Varchar2	Yes	No
price_by_formula_id	Null	Number	No	No
primary_uom_flag	Null	Varchar2	No	No
print_on_invoice_flag	Null	Varchar2	No	No
program_application_id	Null	Number	No	No
program_id	Null	Number	No	No
program_update_date	Null	Date	No	No
rebate_trxn_type_code	Null	Varchar2	No	No
related_item_id	Null	Number	No	No
relationship_type_id	Null	Number	No	No
reprice_flag	Null	Varchar2	No	No
request_id	Null	Number	No	No
revision	Null	Varchar2	No	No
revision_date	Null	Date	No	No
revision_reason_code	Null	Varchar2	No	No
start_date_active	Null	Date	Yes	No
substitution_attribute	Null	Varchar2	No	No
substitution_context	Null	Varchar2	No	No
substitution_value	Null	Varchar2	No	No

Table 5–44 MODIFIERS_REC_TYPE Parameters

Parameter	Usage	Type	Req	Drv
accrual_flag	Null	Varchar2	Yes	No
pricing_group_sequence	Null	Number	Yes	No
incompatibility_grp_code	Null	Varchar2	No	No
list_line_no	Null	Varchar2	No	No
from_rltd_modifier_id	Null	Number	No	No
to_rltd_modifier_id	Null	Number	No	No
rltd_modifier_grp_no	Null	Number	No	No
rltd_modifier_grp_type	Null	Varchar2	No	No
pricing_phase_id	Null	Number	Yes	No
product_precedence	Null	Number	Yes	No
expiration_period_start_date	Null	Date	No	No
number_expiration_periods	Null	Number	No	No
expiration_period_uom	Null	Varchar2	No	No
expiration_date	Null	Date	No	No
estim_gl_value	Null	Number	No	No
benefit_price_list_line_id	Null	Number	No	No
benefit_limit	Null	Number	No	No
charge_type_code	Null	Varchar2	No	No
charge_subtype_code	Null	Varchar2	No	No
benefit_qty	Null	Number	No	No
benefit_uom_code	Null	Varchar2	No	No
accrual_conversion_rate	Null	Number	No	No
proration_type_code	Null	Varchar2	No	No
return_status	Null	Varchar2	No	No
db_flag	Null	Varchar2	No	No
operation	Null	Varchar2	Yes	No

arithmetic_operator

Default Value: FND_API.G_MISS_CHAR

attribute1-15

Default Value: FND_API.G_MISS_CHAR

automatic_flag

Default Value: FND_API.G_MISS_CHAR

comments

Default Value: FND_API.G_MISS_CHAR

context

Default Value: FND_API.G_MISS_CHAR

created_by

Default Value: FND_API.G_MISS_NUM

creation_date

Default Value: FND_API.G_MISS_DATE

effective_period_uom

Default Value: FND_API.G_MISS_CHAR

end_date_active

Default Value: FND_API.G_MISS_DATE

estim_accrual_rate

Default Value: FND_API.G_MISS_NUM

generate_using_formula_id

Default Value: FND_API.G_MISS_NUM

inventory_item_id

Default Value: FND_API.G_MISS_NUM

last_updated_by

Default Value: FND_API.G_MISS_NUM

last_update_date

Default Value: FND_API.G_MISS_DATE

last_update_login

Default Value: FND_API.G_MISS_NUM

list_header_id

Default Value: FND_API.G_MISS_NUM

list_line_id

Default Value: FND_API.G_MISS_NUM

list_line_type_code

Default Value: FND_API.G_MISS_CHAR

list_price

Default Value: FND_API.G_MISS_NUM

modifier_level_code

Default Value: FND_API.G_MISS_CHAR

number_effective_periods

Default Value: FND_API.G_MISS_NUM

operand

Default Value: FND_API.G_MISS_NUM

organization_id

Default Value: FND_API.G_MISS_NUM

override_flag

Default Value: FND_API.G_MISS_CHAR

percent_price

Default Value: FND_API.G_MISS_NUM

price_break_type_code

Default Value: FND_API.G_MISS_CHAR

price_by_formula_id

Default Value: FND_API.G_MISS_NUM

primary_uom_flag

Default Value: FND_API.G_MISS_CHAR

print_on_invoice_flag

Default Value: FND_API.G_MISS_CHAR

program_application_id

Default Value: FND_API.G_MISS_NUM

program_id

Default Value: FND_API.G_MISS_NUM

program_update_date

Default Value: FND_API.G_MISS_DATE

rebate_trxn_type_code

Default Value: FND_API.G_MISS_CHAR

related_item_id

Default Value: FND_API.G_MISS_NUM

relationship_type_id

Default Value: FND_API.G_MISS_NUM

reprice_flag

Default Value: FND_API.G_MISS_CHAR

request_id

Default Value: FND_API.G_MISS_NUM

revision

Default Value: FND_API.G_MISS_CHAR

revision_date

Default Value: FND_API.G_MISS_DATE

revision_reason_code

Default Value: FND_API.G_MISS_CHAR

start_date_active

Default Value: FND_API.G_MISS_DATE

substitution_attribute

Default Value: FND_API.G_MISS_CHAR

substitution_context

Default Value: FND_API.G_MISS_CHAR

substitution_value

Default Value: FND_API.G_MISS_CHAR

accrual_flag

Default Value: FND_API.G_MISS_CHAR

pricing_group_sequence

Default Value: FND_API.G_MISS_NUM

incompatibility_grp_code

Default Value: FND_API.G_MISS_CHAR

list_line_no

Default Value: FND_API.G_MISS_CHAR

from_rltd_modifier_id

Default Value: FND_API.G_MISS_NUM

to_rltd_modifier_id

Default Value: FND_API.G_MISS_NUM

rltd_modifier_grp_no

Default Value: FND_API.G_MISS_NUM

rltd_modifier_grp_type

Default Value: FND_API.G_MISS_CHAR

pricing_phase_id

Default Value: FND_API.G_MISS_NUM

product_precedence

Default Value: FND_API.G_MISS_NUM

expiration_period_start_date

Default Value: FND_API.G_MISS_DATE

number_expiration_periods

Default Value: FND_API.G_MISS_NUM

expiration_period_uom

Default Value: FND_API.G_MISS_CHAR

expiration_date

Default Value: FND_API.G_MISS_DATE

estim_gl_value

Default Value: FND_API.G_MISS_NUM

benefit_price_list_line_id

Default Value: FND_API.G_MISS_NUM

benefit_limit

Default Value: FND_API.G_MISS_NUM

charge_type_code

Default Value: FND_API.G_MISS_CHAR

charge_subtype_code

Default Value: FND_API.G_MISS_CHAR

benefit_qty

Default Value: FND_API.G_MISS_NUM

benefit_uom_code

Default Value: FND_API.G_MISS_CHAR

accrual_conversion_rate

Default Value: FND_API.G_MISS_NUM

proration_type_code

Default Value: FND_API.G_MISS_CHAR

return_status

Default Value: FND_API.G_MISS_CHAR

db_flag

Default Value: FND_API.G_MISS_CHAR

operation

Default Value: FND_API.G_MISS_CHAR

MODIFIERS_TBL_TYPE

The following table shows the parameters for this structure.

Table 5–45 MODIFIERS_TBL_TYPE Parameters

Parameter	Usage	Type	Req	Drv
Modifiers_Rec_Type	Null	Record	No	No

MODIFIERS_VAL_REC_TYPE

The following table shows the parameters for this structure.

Table 5–46 MODIFIERS_VAL_REC_TYPE Parameters

Parameter	Usage	Type	Req	Drv
accrual_type	Null	Varchar2	No	No
accrual_uom	Null	Varchar2	No	No
automatic	Null	Varchar2	No	No
generate_using_formula	Null	Varchar2	No	No
gl_class	Null	Varchar2	No	No
inventory_item	Null	Varchar2	No	No
list_header	Null	Varchar2	No	No
list_line	Null	Varchar2	No	No
list_line_type	Null	Varchar2	No	No
list_price_uom	Null	Varchar2	No	No
modifier_level	Null	Varchar2	No	No
organization	Null	Varchar2	No	No
override	Null	Varchar2	No	No
price_break_type	Null	Varchar2	No	No
price_by_formula	Null	Varchar2	No	No
primary_uom	Null	Varchar2	No	No
print_on_invoice	Null	Varchar2	No	No
rebate_subtype	Null	Varchar2	No	No
rebate_transaction_type	Null	Varchar2	No	No
related_item	Null	Varchar2	No	No
relationship_type	Null	Varchar2	No	No
reprice	Null	Varchar2	No	No
revision_reason	Null	Varchar2	No	No

accrual_type

Default Value: FND_API.G_MISS_CHAR

accrual_uom

Default Value: FND_API.G_MISS_CHAR

automatic

Default Value: FND_API.G_MISS_CHAR

generate_using_formula

Default Value: FND_API.G_MISS_CHAR

gl_class

Default Value: FND_API.G_MISS_CHAR

inventory_item

Default Value: FND_API.G_MISS_CHAR

list_header

Default Value: FND_API.G_MISS_CHAR

list_line

Default Value: FND_API.G_MISS_CHAR

list_line_type

Default Value: FND_API.G_MISS_CHAR

list_price_uom

Default Value: FND_API.G_MISS_CHAR

modifier_level

Default Value: FND_API.G_MISS_CHAR

organization

Default Value: FND_API.G_MISS_CHAR

override

Default Value: FND_API.G_MISS_CHAR

price_break_type

Default Value: FND_API.G_MISS_CHAR

price_by_formula

Default Value: FND_API.G_MISS_CHAR

primary_uom

Default Value: FND_API.G_MISS_CHAR

print_on_invoice

Default Value: FND_API.G_MISS_CHAR

rebate_subtype

Default Value: FND_API.G_MISS_CHAR

rebate_transaction_type

Default Value: FND_API.G_MISS_CHAR

related_item

Default Value: FND_API.G_MISS_CHAR

relationship_type

Default Value: FND_API.G_MISS_CHAR

reprice

Default Value: FND_API.G_MISS_CHAR

revision_reason

Default Value: FND_API.G_MISS_CHAR

MODIFIERS_VAL_TBL_TYPE

The following table shows the parameters for this structure.

Table 5–47 MODIFIERS_VAL_TBL_TYPE Parameters

Parameter	Usage	Type	Req	Drv
Modifiers_Val_Rec_Type	Null	Record	No	No

QUALIFIERS_REC_TYPE

The following table shows the parameters for this structure.

Table 5–48 QUALIFIERS_REC_TYPE Parameters

Parameter	Usage	Type	Req	Drv
attribute1	Null	Varchar2	No	No
attribute2	Null	Varchar2	No	No
attribute3	Null	Varchar2	No	No
attribute4	Null	Varchar2	No	No
attribute5	Null	Varchar2	No	No
attribute6	Null	Varchar2	No	No
attribute7	Null	Varchar2	No	No
attribute8	Null	Varchar2	No	No
attribute9	Null	Varchar2	No	No
attribute10	Null	Varchar2	No	No
attribute11	Null	Varchar2	No	No
attribute12	Null	Varchar2	No	No
attribute13	Null	Varchar2	No	No
attribute14	Null	Varchar2	No	No
attribute15	Null	Varchar2	No	No
comparison_operator_code	Null	Varchar2	Yes	No
context	Null	Varchar2	No	No
created_by	Null	Number	No	No
created_from_rule_id	Null	Number	No	No
creation_date	Null	Date	No	No
end_date_active	Null	Date	No	No

Table 5–48 QUALIFIERS_REC_TYPE Parameters

Parameter	Usage	Type	Req	Drv
excluder_flag	Null	Varchar2	No	No
last_updated_by	Null	Number	No	No
last_update_date	Null	Date	No	No
last_update_login	Null	Number	No	No
list_header_id	Null	Number	No	No
list_line_id	Null	Number	No	No
program_application_id	Null	Number	No	No
program_id	Null	Number	No	No
program_update_date	Null	Date	No	No
qualifier_attribute	Null	Varchar2	Yes	No
qualifier_attr_value	Null	Varchar2	Yes	No
qualifier_context	Null	Varchar2	Yes	No
qualifier_grouping_no	Null	Number	Yes	No
qualifier_precedence	Null	Number	Yes	No
qualifier_id	Null	Number	No	No
qualifier_rule_id	Null	Number	No	No
request_id	Null	Number	No	No
start_date_active	Null	Date	Yes	No
return_status	Null	Varchar2	No	No
db_flag	Null	Varchar2	No	No
operation	Null	Varchar2	Yes	No

attribute1-15

Default Value: FND_API.G_MISS_CHAR

comparison_operator_code

Default Value: FND_API.G_MISS_CHAR

context

Default Value: FND_API.G_MISS_CHAR

created_by

Default Value: FND_API.G_MISS_NUM

created_from_rule_id

Default Value: FND_API.G_MISS_NUM

creation_date

Default Value: FND_API.G_MISS_DATE

end_date_active

Default Value: FND_API.G_MISS_DATE

excluder_flag

Default Value: FND_API.G_MISS_CHAR

last_updated_by

Default Value: FND_API.G_MISS_NUM

last_update_date

Default Value: FND_API.G_MISS_DATE

last_update_login

Default Value: FND_API.G_MISS_NUM

list_header_id

Default Value: FND_API.G_MISS_NUM

list_line_id

Default Value: FND_API.G_MISS_NUM

program_application_id

Default Value: FND_API.G_MISS_NUM

program_id

Default Value: FND_API.G_MISS_NUM

program_update_date

Default Value: FND_API.G_MISS_DATE

qualifier_attribute

Default Value: FND_API.G_MISS_CHAR

qualifier_attr_value

Default Value: FND_API.G_MISS_CHAR

qualifier_context

Default Value: FND_API.G_MISS_CHAR

qualifier_grouping_no

Default Value: FND_API.G_MISS_NUM

qualifier_id

Default Value: FND_API.G_MISS_NUM

qualifier_rule_id

Default Value: FND_API.G_MISS_NUM

request_id

Default Value: FND_API.G_MISS_NUM

start_date_active

Default Value: FND_API.G_MISS_DATE

return_status

Default Value: FND_API.G_MISS_CHAR

db_flag

Default Value: FND_API.G_MISS_CHAR

operation

Default Value: FND_API.G_MISS_CHAR

QUALIFIERS_TBL_TYPE

The following table shows the parameters for this structure.

Table 5–49 QUALIFIERS_TBL_TYPE Parameters

Parameter	Usage	Type	Req	Drv
Qualifiers_Rec_Type	Null	Record	No	No

QUALIFIERS_VAL_REC_TYPE

The following table shows the parameters for this structure.

Table 5–50 QUALIFIERS_VAL_REC_TYPE Parameters

Parameter	Usage	Type	Req	Drv
comparison_operator	Null	Varchar2	No	No
created_from_rule	Null	Varchar2	No	No
excluder	Null	Varchar2	No	No
list_header	Null	Varchar2	No	No
list_line	Null	Varchar2	No	No
qualifier	Null	Varchar2	No	No
qualifier_rule	Null	Varchar2	No	No

comparison_operator

Default Value: FND_API.G_MISS_CHAR

created_from_rule

Default Value: FND_API.G_MISS_CHAR

excluder

Default Value: FND_API.G_MISS_CHAR

list_header

Default Value: FND_API.G_MISS_CHAR

list_line

Default Value: FND_API.G_MISS_CHAR

qualifier

Default Value: FND_API.G_MISS_CHAR

qualifier_rule

Default Value: FND_API.G_MISS_CHAR

QUALIFIERS_VAL_TBL_TYPE

The following table shows the parameters for this structure.

Table 5–51 QUALIFIERS_VAL_TBL_TYPE Parameters

Parameter	Usage	Type	Req	Drv
Qualifiers_Val_Rec_Type	Null	Record	No	No

PRICING_ATTR_REC_TYPE

The following table shows the parameters for this structure.

Table 5–52 PRICING_ATTR_REC_TYPE Parameters

Parameter	Usage	Type	Req	Drv
accumulate_flag	Null	Varchar2	No	No
attribute1	Null	Varchar2	No	No
attribute2	Null	Varchar2	No	No
attribute3	Null	Varchar2	No	No
attribute4	Null	Varchar2	No	No
attribute5	Null	Varchar2	No	No
attribute6	Null	Varchar2	No	No
attribute7	Null	Varchar2	No	No
attribute8	Null	Varchar2	No	No
attribute9	Null	Varchar2	No	No
attribute10	Null	Varchar2	No	No
attribute11	Null	Varchar2	No	No

Table 5–52 PRICING_ATTR_REC_TYPE Parameters

Parameter	Usage	Type	Req	Drv
attribute12	Null	Varchar2	No	No
attribute13	Null	Varchar2	No	No
attribute14	Null	Varchar2	No	No
attribute15	Null	Varchar2	No	No
attribute_grouping_no	Null	Number	No	No
context	Null	Varchar2	No	No
created_by	Null	Number	No	No
creation_date	Null	Date	No	No
excluder_flag	Null	Varchar2	No	No
last_updated_by	Null	Number	No	No
last_update_date	Null	Date	No	No
last_update_login	Null	Number	No	No
list_line_id	Null	Number	No	No
pricing_attribute	Null	Varchar2	No	No
pricing_attribute_context	Null	Varchar2	No	No
pricing_attribute_id	Null	Number	No	No
pricing_attr_value_from	Null	Varchar2	No	No
pricing_attr_value_to	Null	Varchar2	No	No
product_attribute	Null	Varchar2	Yes	No
product_attribute_context	Null	Varchar2	Yes	No
product_attr_value	Null	Varchar2	Yes	No
product_uom_code	Null	Varchar2	Yes	No
program_application_id	Null	Number	No	No
program_id	Null	Number	No	No
program_update_date	Null	Date	No	No
product_attribute_datatype	Null	Varchar2	No	No
pricing_attribute_datatype	Null	Varchar2	No	No

Table 5–52 PRICING_ATTR_REC_TYPE Parameters

Parameter	Usage	Type	Req	Drv
comparison_operator_code	Null	Varchar2	Yes	No
request_id	Null	Number	No	No
return_status	Null	Varchar2	No	No
db_flag	Null	Varchar2	No	No
operation	Null	Varchar2	Yes	No
MODIFIERS_index	Null	Number	Yes	No

accumulate_flag

Default Value: FND_API.G_MISS_CHAR

attribute1-15

Default Value: FND_API.G_MISS_CHAR

attribute_grouping_no

Default Value: FND_API.G_MISS_NUM

context

Default Value: FND_API.G_MISS_CHAR

created_by

Default Value: FND_API.G_MISS_NUM

creation_date

Default Value: FND_API.G_MISS_DATE

excluder_flag

Default Value: FND_API.G_MISS_CHAR

last_updated_by

Default Value: FND_API.G_MISS_NUM

last_update_date

Default Value: FND_API.G_MISS_DATE

last_update_login

Default Value: FND_API.G_MISS_NUM

list_line_id

Default Value: FND_API.G_MISS_NUM

pricing_attribute

Default Value: FND_API.G_MISS_CHAR

pricing_attribute_context

Default Value: FND_API.G_MISS_CHAR

pricing_attribute_id

Default Value: FND_API.G_MISS_NUM

pricing_attr_value_from

Default Value: FND_API.G_MISS_CHAR

pricing_attr_value_to

Default Value: FND_API.G_MISS_CHAR

product_attribute

Default Value: FND_API.G_MISS_CHAR

product_attribute_context

Default Value: FND_API.G_MISS_CHAR

product_attr_value

Default Value: FND_API.G_MISS_CHAR

product_uom_code

Default Value: FND_API.G_MISS_CHAR

program_application_id

Default Value: FND_API.G_MISS_NUM

program_id
Default Value: FND_API.G_MISS_NUM

program_update_date
Default Value: FND_API.G_MISS_DATE

product_attribute_datatype
Default Value: FND_API.G_MISS_CHAR

pricing_attribute_datatype
Default Value: FND_API.G_MISS_CHAR

comparison_operator_code
Default Value: FND_API.G_MISS_CHAR

request_id
Default Value: FND_API.G_MISS_NUM

return_status
Default Value: FND_API.G_MISS_CHAR

db_flag
Default Value: FND_API.G_MISS_CHAR

operation
Default Value: FND_API.G_MISS_CHAR

MODIFIERS_index
Default Value: FND_API.G_MISS_NUM

PRICING_ATTR_TBL_TYPE

The following table shows the parameters for this structure.

Table 5–53 PRICING_ATTR_TBL_TYPE Parameters

Parameter	Usage	Type	Req	Drv
Pricing_Attr_Rec_Type	Null	Record	No	No

PRICING_ATTR_VAL_REC_TYPE

The following table shows the parameters for this structure.

Table 5–54 PRICING_ATTR_VAL_REC_TYPE Parameters

Parameter	Usage	Type	Req	Drv
accumulate	Null	Varchar2	No	No
excluder	Null	Varchar2	No	No
list_line	Null	Varchar2	No	No
pricing_attribute	Null	Varchar2	No	No
product_uom	Null	Varchar2	No	No

accumulate

Default Value: FND_API.G_MISS_CHAR

excluder

Default Value: FND_API.G_MISS_CHAR

list_line

Default Value: FND_API.G_MISS_CHAR

pricing_attribute

Default Value: FND_API.G_MISS_CHAR

product_uom

Default Value: FND_API.G_MISS_CHAR

PRICING_ATTR_VAL_TBL_TYPE

The following table shows the parameters for this structure.

Table 5–55 PRICING_ATTR_VAL_TBL_TYPE Parameters

Parameter	Usage	Type	Req	Drv
Pricing_Attr_Val_Rec_Type	Null	Record	No	No

Validation of Business Object for Modifier Setup API

Standard Validation

Oracle Advanced Pricing validates all required columns in the Business Object for Modifier Setup API. For more information, see: *Oracle Pricing Technical Reference Manual*.

Other Validation

None

Error Handling

If any validation fails, the API will return error status to the calling module. The Business Object for Modifier Setup API processes the rows and reports the values in the following table for every record.

Table 5–56 Error Handling

Condition	PROCESS_STATUS	ERROR_MESSAGE
Success	5	null
Failure	4	actual error message

See

Oracle Pricing Technical Reference Manual

Example of Modifier Setup Application Program Interface

Example 1: Line level discount of 8% discount on all products

```
File Path : $QP_TOP/patch/115/sql/QPXEKDS1.sql
/* Discount Creation - Get 8% discount on all products */
REM FILETYPE : NOEXEC

REM Added for ARU db drv auto generation
REM dbdrv: none

SET VERIFY OFF
WHenever SQLERROR EXIT FAILURE ROLLBACK;
--set serveroutput on
declare
/* $Header: QPXEKDS1.sql 115.4 2002/05/31 22:04:06 mkarya noship $ */
```

```

l_control_recQP_GLOBALS.Control_Rec_Type;
l_return_statusVARCHAR2(1);
x_msg_countnumber;
x_msg_dataVarchar2(2000);
x_msg_indexnumber;

l_MODIFIER_LIST_recQP_Modifiers_PUB.Modifier_List_Rec_Type;
l_MODIFIER_LIST_val_recQP_Modifiers_PUB.Modifier_List_Val_Rec_Type;
l_MODIFIERS_tblQP_Modifiers_PUB.Modifiers_Tbl_Type;
l_MODIFIERS_val_tblQP_Modifiers_PUB.Modifiers_Val_Tbl_Type;
l_QUALIFIERS_tblQP_Qualifier_Rules_PUB.Qualifiers_Tbl_Type;
l_QUALIFIERS_val_tblQP_Qualifier_Rules_PUB.Qualifiers_Val_Tbl_Type;
l_PRICING_ATTR_tblQP_Modifiers_PUB.Pricing_Attr_Tbl_Type;
l_PRICING_ATTR_val_tblQP_Modifiers_PUB.Pricing_Attr_Val_Tbl_Type;

l_x_MODIFIER_LIST_recQP_Modifiers_PUB.Modifier_List_Rec_Type;
l_x_MODIFIER_LIST_val_recQP_Modifiers_PUB.Modifier_List_Val_Rec_Type;
l_x_MODIFIERS_tblQP_Modifiers_PUB.Modifiers_Tbl_Type;
l_x_MODIFIERS_val_tblQP_Modifiers_PUB.Modifiers_Val_Tbl_Type;
l_x_QUALIFIERS_tblQP_Qualifier_Rules_PUB.Qualifiers_Tbl_Type;
l_x_QUALIFIERS_val_tblQP_Qualifier_Rules_PUB.Qualifiers_Val_Tbl_Type;
l_x_PRICING_ATTR_tblQP_Modifiers_PUB.Pricing_Attr_Tbl_Type;
l_x_PRICING_ATTR_val_tblQP_Modifiers_PUB.Pricing_Attr_Val_Tbl_Type;

mll_rec qp_list_lines%ROWTYPE;
pra_rec qp_pricing_attributes%ROWTYPE;

Begin

/* Create a Modifier header of type 'PRO' (Promotion) */

    l_MODIFIER_LIST_rec.currency_code:= 'USD';
    l_MODIFIER_LIST_rec.list_type_code:= 'PRO';
    l_MODIFIER_LIST_rec.start_date_active      := sysdate;
    l_MODIFIER_LIST_rec.end_date_active        := sysdate+10;
    l_MODIFIER_LIST_rec.source_system_code:= 'QP';
    l_MODIFIER_LIST_rec.active_flag:= 'Y';
    l_MODIFIER_LIST_rec.name:= 'New HALLOWEAN 2000 Deal';
    l_MODIFIER_LIST_rec.description:= ' Latest New Description of HALLOWEAN
    2000';
    l_MODIFIER_LIST_rec.version_no:= '9.4';
    l_MODIFIER_LIST_rec.pte_code                := 'ORDFUL';
    l_MODIFIER_LIST_rec.operation:= QP_GLOBALS.G_OPR_CREATE;

/* Create a Modifier line to specify 'Get 8% discount' condition */

```

```

l_MODIFIERS_tbl(1).list_line_type_code := 'DIS';
l_MODIFIERS_tbl(1).automatic_flag:= 'Y';
l_MODIFIERS_tbl(1).modifier_level_code := 'LINE';
l_MODIFIERS_tbl(1).accrual_flag := 'N';
l_MODIFIERS_tbl(1).start_date_active := sysdate;
l_MODIFIERS_tbl(1).end_date_active := sysdate+10;l_MODIFIERS_
tbl(1).arithmetic_operator := '%';
l_MODIFIERS_tbl(1).pricing_group_sequence := 1;
l_MODIFIERS_tbl(1).pricing_phase_id := 2;
l_MODIFIERS_tbl(1).operand := 8;
l_MODIFIERS_tbl(1).operation := QP_GLOBALS.G_OPR_CREATE;

/* Call the Modifiers Public API to create the modifier header and a modifier
line */

```

QP_Modifiers_PUB.Process_Modifiers

```

p_api_version_number=> 1.0
, p_init_msg_list=> FND_API.G_FALSE
, p_return_values=> FND_API.G_FALSE
, p_commit=> FND_API.G_FALSE
, x_return_status=> l_return_status
, x_msg_count=>x_msg_count
, x_msg_data=>x_msg_data
,p_MODIFIER_LIST_rec=> l_MODIFIER_LIST_rec
,p_MODIFIERS_tbl=> l_MODIFIERS_tbl
--,p_QUALIFIERS_tbl=> l_QUALIFIERS_tbl
--,p_PRICING_ATTR_tbl=> l_PRICING_ATTR_tbl
,x_MODIFIER_LIST_rec=> l_MODIFIER_LIST_rec
,x_MODIFIER_LIST_val_rec=> l_MODIFIER_LIST_val_rec
,x_MODIFIERS_tbl=> l_MODIFIERS_tbl
,x_MODIFIERS_val_tbl=> l_MODIFIERS_val_tbl
,x_QUALIFIERS_tbl=> l_QUALIFIERS_tbl
,x_QUALIFIERS_val_tbl=> l_QUALIFIERS_val_tbl
,x_PRICING_ATTR_tbl=> l_PRICING_ATTR_tbl
,x_PRICING_ATTR_val_tbl=> l_PRICING_ATTR_val_tbl
);

IF l_return_status <> FND_API.G_RET_STS_SUCCESS THEN

    RAISE FND_API.G_EXC_UNEXPECTED_ERROR;

END IF;

```



```
EXCEPTION

    WHEN FND_API.G_EXC_ERROR THEN

        l_return_status := FND_API.G_RET_STS_ERROR;

        --Get message count and data

        dbms_output.put_line('err msg 1 is : ' || x_msg_data);

    WHEN FND_API.G_EXC_UNEXPECTED_ERROR THEN

        l_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;
        dbms_output.put_line(' msg count 2 is : ' || x_msg_count);

        for k in 1 .. x_msg_count loop

            x_msg_data := oe_msg_pub.get( p_msg_index => k,

            p_encoded => 'F'

            );

        --Get message count and data

        dbms_output.put_line('err msg ' || k || 'is: ' || x_msg_data);

        end loop;

    WHEN OTHERS THEN

        l_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;

        dbms_output.put_line(' msg count 2 is : ' || x_msg_count);

        for k in 1 .. x_msg_count loop

            x_msg_data := oe_msg_pub.get( p_msg_index => k,

            p_encoded => 'F'

            );

        --Get message count and data

        dbms_output.put_line('err msg ' || k || 'is: ' || x_msg_data);
```

```
end loop;
```

```
END;
```

```
/
```

Example 2: Buy more than 5 quantities of item 62081 , Get 8% discount

File Path : \$QP_TOP/patch/115/sql/QPXEXDS2.sql

/* Discount Creation - Buy 5 of item 62081, Get 8% discount */

REM FILETYPE : NOEXEC

REM Added for ARU db drv auto generation

REM dbdrv: none

SET VERIFY OFF

WHENEVER SQLERROR EXIT FAILURE ROLLBACK;

--set serveroutput on

declare

/* \$Header: QPXEXDS2.sql 115.3 2002/05/31 22:15:07 mkarya noship \$ */

l_control_recQP_GLOBALS.Control_Rec_Type;

l_return_statusVARCHAR2(1);

x_msg_countnumber;

x_msg_dataVarchar2(2000);

x_msg_indexnumber;

l_MODIFIER_LIST_recQP_Modifiers_PUB.Modifier_List_Rec_Type;

l_MODIFIER_LIST_val_recQP_Modifiers_PUB.Modifier_List_Val_Rec_Type;

l_MODIFIERS_tblQP_Modifiers_PUB.Modifiers_Tbl_Type;

l_MODIFIERS_val_tblQP_Modifiers_PUB.Modifiers_Val_Tbl_Type;

l_QUALIFIERS_tblQP_Qualifier_Rules_PUB.Qualifiers_Tbl_Type;

l_QUALIFIERS_val_tblQP_Qualifier_Rules_PUB.Qualifiers_Val_Tbl_Type;

l_PRICING_ATTR_tblQP_Modifiers_PUB.Pricing_Attr_Tbl_Type;

l_PRICING_ATTR_val_tblQP_Modifiers_PUB.Pricing_Attr_Val_Tbl_Type;

l_x_MODIFIER_LIST_recQP_Modifiers_PUB.Modifier_List_Rec_Type;

l_x_MODIFIER_LIST_val_recQP_Modifiers_PUB.Modifier_List_Val_Rec_Type;

l_x_MODIFIERS_tblQP_Modifiers_PUB.Modifiers_Tbl_Type;

l_x_MODIFIERS_val_tblQP_Modifiers_PUB.Modifiers_Val_Tbl_Type;

l_x_QUALIFIERS_tblQP_Qualifier_Rules_PUB.Qualifiers_Tbl_Type;

l_x_QUALIFIERS_val_tblQP_Qualifier_Rules_PUB.Qualifiers_Val_Tbl_Type;

l_x_PRICING_ATTR_tblQP_Modifiers_PUB.Pricing_Attr_Tbl_Type;

l_x_PRICING_ATTR_val_tblQP_Modifiers_PUB.Pricing_Attr_Val_Tbl_Type;

mll_rec qp_list_lines%ROWTYPE;

```

pra_rec qp_pricing_attributes%ROWTYPE;

Begin

/* Create a Modifier header of type 'PRO' (Promotion) */

    l_MODIFIER_LIST_rec.currency_code:= 'USD';
    l_MODIFIER_LIST_rec.list_type_code:= 'PRO';
    l_MODIFIER_LIST_rec.start_date_active      := sysdate;
    l_MODIFIER_LIST_rec.end_date_active        := sysdate+10;l_MODIFIER_
LIST_rec.source_system_code:= 'QP';
    l_MODIFIER_LIST_rec.active_flag:= 'Y';
    l_MODIFIER_LIST_rec.name:= 'New HALLOWEAN 2000 Deal';
    l_MODIFIER_LIST_rec.description:= 'New Description of HALLOWEAN 2000';
    l_MODIFIER_LIST_rec.version_no:= '5.7';
    l_MODIFIER_LIST_rec.pte_code                := 'ORDFUL';
    l_MODIFIER_LIST_rec.operation:= QP_GLOBALS.G_OPR_CREATE;

/* Create a Modifier line to specify 'Get 8% discount' condition */

    l_MODIFIERS_tbl(1).list_line_type_code := 'DIS';
    l_MODIFIERS_tbl(1).automatic_flag:= 'Y';
    l_MODIFIERS_tbl(1).modifier_level_code := 'LINE';
    l_MODIFIERS_tbl(1).accrual_flag := 'N';
    l_MODIFIERS_tbl(1).start_date_active := sysdate;
    l_MODIFIERS_tbl(1).end_date_active := sysdate+10;
    l_MODIFIERS_tbl(1).operand := 8;
    l_MODIFIERS_tbl(1).arithmetic_operator := '%';
    l_MODIFIERS_tbl(1).pricing_group_sequence := 1;
    l_MODIFIERS_tbl(1).pricing_phase_id := 3;
    l_MODIFIERS_tbl(1).price_break_type_code := 'POINT';
    l_MODIFIERS_tbl(1).operation := QP_GLOBALS.G_OPR_CREATE;

/* Create a Pricing Attribute record to specify the 'Buy 5 of item 62081'
condition */

    l_PRICING_ATTR_tbl(1).product_attribute_context:= 'ITEM';
    l_PRICING_ATTR_tbl(1).product_attribute:= 'PRICING_ATTRIBUTE1';
    l_PRICING_ATTR_tbl(1).product_attr_value:= '62081';
    l_PRICING_ATTR_tbl(1).pricing_attribute_context:= 'VOLUME';
    l_PRICING_ATTR_tbl(1).pricing_attribute:= 'PRICING_ATTRIBUTE3';
    l_PRICING_ATTR_tbl(1).pricing_attr_value_from:= '5';
    l_PRICING_ATTR_tbl(1).comparison_operator_code:= 'BETWEEN';
    l_PRICING_ATTR_tbl(1).product_uom_code:= 'Ea';
    l_PRICING_ATTR_tbl(1).excluder_flag:= 'N';

```

```

l_PRICING_ATTR_tbl(1).MODIFIERS_index:=1;
l_PRICING_ATTR_tbl(1).operation          := QP_GLOBALS.G_OPR_CREATE;

/* Call the Modifiers Public API to create the modifier, modifier line and a
pricing attribute record */

QP_Modifiers_PUB.Process_Modifiers

( p_api_version_number=> 1.0
, p_init_msg_list=> FND_API.G_FALSE
, p_return_values=> FND_API.G_FALSE
, p_commit=> FND_API.G_FALSE
, x_return_status=> l_return_status
, x_msg_count=> x_msg_count
, x_msg_data=> x_msg_data

,p_MODIFIER_LIST_rec=> l_MODIFIER_LIST_rec
,p_MODIFIERS_tbl=> l_MODIFIERS_tbl
--,p_QUALIFIERS_tbl=> l_QUALIFIERS_tbl
,p_PRICING_ATTR_tbl=> l_PRICING_ATTR_tbl
,x_MODIFIER_LIST_rec=> l_MODIFIER_LIST_rec
,x_MODIFIER_LIST_val_rec=> l_MODIFIER_LIST_val_rec
,x_MODIFIERS_tbl=> l_MODIFIERS_tbl
,x_MODIFIERS_val_tbl=> l_MODIFIERS_val_tbl
,x_QUALIFIERS_tbl=> l_QUALIFIERS_tbl
,x_QUALIFIERS_val_tbl=> l_QUALIFIERS_val_tbl
,x_PRICING_ATTR_tbl=> l_PRICING_ATTR_tbl
,x_PRICING_ATTR_val_tbl=> l_PRICING_ATTR_val_tbl
);

IF l_return_status <> FND_API.G_RET_STS_SUCCESS THEN

    RAISE FND_API.G_EXC_UNEXPECTED_ERROR;

END IF;

EXCEPTION

    WHEN FND_API.G_EXC_ERROR THEN

        l_return_status := FND_API.G_RET_STS_ERROR;

--Get message count and data

        dbms_output.put_line('err msg 1 is : ' || x_msg_data);

```

```

        WHEN FND_API.G_EXC_UNEXPECTED_ERROR THEN

            l_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;

            dbms_output.put_line(' msg count 2 is : ' || x_msg_count);

        for k in 1 .. x_msg_count loop

            x_msg_data := oe_msg_pub.get( p_msg_index => k,

                p_encoded => 'F'

            );

            --Get message count and data

            dbms_output.put_line('err msg ' || k || 'is: ' || x_msg_data);

        end loop;
    WHEN OTHERS THEN

        l_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;
        dbms_output.put_line(' msg count 2 is : ' || x_msg_count);
        for k in 1 .. x_msg_count loop

            x_msg_data := oe_msg_pub.get( p_msg_index => k,

                p_encoded => 'F'

            );

            --Get message count and data

            dbms_output.put_line('err msg ' || k || 'is: ' || x_msg_data);

        end loop;

    END;
/

```

Example 3: For customer 1000, Buy more than 2 Units of item 62081, Get 10% discount

```

File Path : $QP_TOP/patch/115/sql/QPXEXDS3.sql
/* Discount Creation - For customer 1000, Buy 2 of item 62081, Get 10% discount
*/
REM FILETYPE : NOEXEC

```

```

REM Added for ARU db drv auto generation
REM dbdrv: none

SET VERIFY OFF
WHenever SQLERROR EXIT FAILURE ROLLBACK;
--set serveroutput on
declare
/* $Header: QPXEXDS3.sql 115.3 2002/05/31 22:24:21 mkarya noship $ */
l_control_recQP_GLOBALS.Control_Rec_Type;
l_return_statusVARCHAR2(1);
x_msg_countnumber;
x_msg_dataVarchar2(2000);
x_msg_indexnumber;

l_MODIFIER_LIST_recQP_Modifiers_PUB.Modifier_List_Rec_Type;
l_MODIFIER_LIST_val_recQP_Modifiers_PUB.Modifier_List_Val_Rec_Type;
l_MODIFIERS_tblQP_Modifiers_PUB.Modifiers_Tbl_Type;
l_MODIFIERS_val_tblQP_Modifiers_PUB.Modifiers_Val_Tbl_Type;
l_QUALIFIERS_tblQP_Qualifier_Rules_PUB.Qualifiers_Tbl_Type;
l_QUALIFIERS_val_tblQP_Qualifier_Rules_PUB.Qualifiers_Val_Tbl_Type;
l_PRICING_ATTR_tblQP_Modifiers_PUB.Pricing_Attr_Tbl_Type;
l_PRICING_ATTR_val_tblQP_Modifiers_PUB.Pricing_Attr_Val_Tbl_Type;

l_x_MODIFIER_LIST_recQP_Modifiers_PUB.Modifier_List_Rec_Type;
l_x_MODIFIER_LIST_val_recQP_Modifiers_PUB.Modifier_List_Val_Rec_Type;
l_x_MODIFIERS_tblQP_Modifiers_PUB.Modifiers_Tbl_Type;
l_x_MODIFIERS_val_tblQP_Modifiers_PUB.Modifiers_Val_Tbl_Type;
l_x_QUALIFIERS_tblQP_Qualifier_Rules_PUB.Qualifiers_Tbl_Type;
l_x_QUALIFIERS_val_tblQP_Qualifier_Rules_PUB.Qualifiers_Val_Tbl_Type;
l_x_PRICING_ATTR_tblQP_Modifiers_PUB.Pricing_Attr_Tbl_Type;
l_x_PRICING_ATTR_val_tblQP_Modifiers_PUB.Pricing_Attr_Val_Tbl_Type;

mll_rec qp_list_lines%ROWTYPE;
pra_rec qp_pricing_attributes%ROWTYPE;

Begin

/* Create a Modifier header of type 'PRO' (Promotion) */
    l_MODIFIER_LIST_rec.currency_code:= 'USD';
    l_MODIFIER_LIST_rec.list_type_code:= 'PRO';
    l_MODIFIER_LIST_rec.start_date_active      := sysdate;
    l_MODIFIER_LIST_rec.end_date_active        := sysdate+10;
    l_MODIFIER_LIST_rec.source_system_code:= 'QP';
    l_MODIFIER_LIST_rec.active_flag:= 'Y';

```

```
l_MODIFIER_LIST_rec.name:= 'New Year 2001 Promotion';
l_MODIFIER_LIST_rec.description:= 'New Year 2001 Promotion';
  l_MODIFIER_LIST_rec.version_no:= '6.7';
  l_MODIFIER_LIST_rec.pte_code           := 'ORDFUL';
  l_MODIFIER_LIST_rec.operation:= QP_GLOBALS.G_OPR_CREATE;

/* Create a Modifier line to specify 'Get 10% discount' condition */

l_MODIFIERS_tbl(1).list_line_type_code := 'DIS';
l_MODIFIERS_tbl(1).automatic_flag := 'Y';
l_MODIFIERS_tbl(1).modifier_level_code := 'LINE';
l_MODIFIERS_tbl(1).accrual_flag := 'N';
l_MODIFIERS_tbl(1).start_date_active := sysdate;
l_MODIFIERS_tbl(1).end_date_active := sysdate+10;
l_MODIFIERS_tbl(1).operand := 10;
l_MODIFIERS_tbl(1).arithmetic_operator := '%';
l_MODIFIERS_tbl(1).pricing_group_sequence := 1;
l_MODIFIERS_tbl(1).pricing_phase_id := 3;
l_MODIFIERS_tbl(1).price_break_type_code := 'POINT';
l_MODIFIERS_tbl(1).operation := QP_GLOBALS.G_OPR_CREATE;

/* Create a Pricing Attribute record to specify the 'Buy 2 of item 62081'
condition */

l_PRICING_ATTR_tbl(1).product_attribute_context:= 'ITEM';
l_PRICING_ATTR_tbl(1).product_attribute:= 'PRICING_ATTRIBUTE1';
l_PRICING_ATTR_tbl(1).product_attr_value:= '62081';
l_PRICING_ATTR_tbl(1).pricing_attribute_context:= 'VOLUME';
l_PRICING_ATTR_tbl(1).pricing_attribute:= 'PRICING_ATTRIBUTE3';
l_PRICING_ATTR_tbl(1).pricing_attr_value_from:= '2';
l_PRICING_ATTR_tbl(1).comparison_operator_code:= 'BETWEEN';
l_PRICING_ATTR_tbl(1).product_uom_code:= 'Ea';
l_PRICING_ATTR_tbl(1).excluder_flag:= 'N';
l_PRICING_ATTR_tbl(1).MODIFIERS_index:=1;
l_PRICING_ATTR_tbl(1).operation           := QP_GLOBALS.G_OPR_CREATE;

/* Create a Qualifier Record to specify 'For Customer 1000' condition */

l_QUALIFIERS_tbl(1).excluder_flag := 'N';
l_QUALIFIERS_tbl(1).comparison_operator_code := '=';
l_QUALIFIERS_tbl(1).qualifier_context := 'CUSTOMER';
l_QUALIFIERS_tbl(1).qualifier_attribute := 'QUALIFIER_ATTRIBUTE2';
l_QUALIFIERS_tbl(1).qualifier_attr_value := '1000';
l_QUALIFIERS_tbl(1).qualifier_grouping_no := 5467;
l_QUALIFIERS_tbl(1).qualifier_precedence := 1;
```

```

l_QUALIFIERS_tbl(1).start_date_active := '06-OCT-00';
l_QUALIFIERS_tbl(1).end_date_active := '11-OCT-00';
l_QUALIFIERS_tbl(1).operation := QP_GLOBALS.G_OPR_CREATE;

```

```

/* Call the Modifiers Public API to create the modifier header, modifier line,
qualifier and a pricing attributes record */

```

QP_Modifiers_PUB.Process_Modifiers

```

( p_api_version_number=> 1.0
, p_init_msg_list=> FND_API.G_FALSE
, p_return_values=> FND_API.G_FALSE
, p_commit=> FND_API.G_FALSE
, x_return_status=> l_return_status
, x_msg_count=>x_msg_count
, x_msg_data=>x_msg_data

,p_MODIFIER_LIST_rec=> l_MODIFIER_LIST_rec
,p_MODIFIERS_tbl=> l_MODIFIERS_tbl
,p_QUALIFIERS_tbl=> l_QUALIFIERS_tbl
,p_PRICING_ATTR_tbl=> l_PRICING_ATTR_tbl
,x_MODIFIER_LIST_rec=> l_MODIFIER_LIST_rec
,x_MODIFIER_LIST_val_rec=> l_MODIFIER_LIST_val_rec
,x_MODIFIERS_tbl=> l_MODIFIERS_tbl
,x_MODIFIERS_val_tbl=> l_MODIFIERS_val_tbl
,x_QUALIFIERS_tbl=> l_QUALIFIERS_tbl
,x_QUALIFIERS_val_tbl=> l_QUALIFIERS_val_tbl
,x_PRICING_ATTR_tbl=> l_PRICING_ATTR_tbl
,x_PRICING_ATTR_val_tbl=> l_PRICING_ATTR_val_tbl
);

IF l_return_status <> FND_API.G_RET_STS_SUCCESS THEN

    RAISE FND_API.G_EXC_UNEXPECTED_ERROR;

END IF;

EXCEPTION

    WHEN FND_API.G_EXC_ERROR THEN

l_return_status := FND_API.G_RET_STS_ERROR;

--Get message count and data

```



```
        dbms_output.put_line('err msg 1 is : ' || x_msg_data);

        WHEN FND_API.G_EXC_UNEXPECTED_ERROR THEN

l_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;

        dbms_output.put_line(' msg count 2 is : ' || x_msg_count);

        for k in 1 .. x_msg_count loop

x_msg_data := oe_msg_pub.get( p_msg_index => k,

p_encoded => 'F'

);

--Get message count and data

        dbms_output.put_line('err msg ' || k || 'is: ' || x_msg_data);

end loop;

        WHEN OTHERS THEN

l_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;

        dbms_output.put_line(' msg count 2 is : ' || x_msg_count);

        for k in 1 .. x_msg_count loop

x_msg_data := oe_msg_pub.get( p_msg_index => k,

p_encoded => 'F'

);

-- Get message count and data

        dbms_output.put_line('err msg ' || k || 'is: ' || x_msg_data);

end loop;

END;
/
```

Example 4: Charge 2% surcharge on all products

```
File Path : /nfs/group/qpdev/qp/11.5/patch/115/sql/QPXEXSUR.sql
/* Surcharge Creation - Charge 2% surcharge on all products */
REM FILETYPE : NOEXEC

REM Added for ARU db drv auto generation
REM dbdrv: none

SET VERIFY OFF
WHENEVER SQLERROR EXIT FAILURE ROLLBACK;
--set serveroutput on
declare
/* $Header: QPXEXSUR.sql 115.4 2002/05/31 22:48:04 mkarya noship $ */
l_control_recQP_GLOBALS.Control_Rec_Type;
l_return_statusVARCHAR2(1);
x_msg_countnumber;
x_msg_dataVarchar2(2000);
x_msg_indexnumber;

l_MODIFIER_LIST_recQP_Modifiers_PUB.Modifier_List_Rec_Type;
l_MODIFIER_LIST_val_recQP_Modifiers_PUB.Modifier_List_Val_Rec_Type;
l_MODIFIERS_tblQP_Modifiers_PUB.Modifiers_Tbl_Type;
l_MODIFIERS_val_tblQP_Modifiers_PUB.Modifiers_Val_Tbl_Type;
l_QUALIFIERS_tblQP_Qualifier_Rules_PUB.Qualifiers_Tbl_Type;
l_QUALIFIERS_val_tblQP_Qualifier_Rules_PUB.Qualifiers_Val_Tbl_Type;
l_PRICING_ATTR_tblQP_Modifiers_PUB.Pricing_Attr_Tbl_Type;
l_PRICING_ATTR_val_tblQP_Modifiers_PUB.Pricing_Attr_Val_Tbl_Type;

l_x_MODIFIER_LIST_recQP_Modifiers_PUB.Modifier_List_Rec_Type;
l_x_MODIFIER_LIST_val_recQP_Modifiers_PUB.Modifier_List_Val_Rec_Type;
l_x_MODIFIERS_tblQP_Modifiers_PUB.Modifiers_Tbl_Type;
l_x_MODIFIERS_val_tblQP_Modifiers_PUB.Modifiers_Val_Tbl_Type;
l_x_QUALIFIERS_tblQP_Qualifier_Rules_PUB.Qualifiers_Tbl_Type;
l_x_QUALIFIERS_val_tblQP_Qualifier_Rules_PUB.Qualifiers_Val_Tbl_Type;
l_x_PRICING_ATTR_tblQP_Modifiers_PUB.Pricing_Attr_Tbl_Type;
l_x_PRICING_ATTR_val_tblQP_Modifiers_PUB.Pricing_Attr_Val_Tbl_Type;

mll_rec qp_list_lines%ROWTYPE;
pra_rec qp_pricing_attributes%ROWTYPE;

Begin

/* Create a Modifier header of type 'SLT' (Surcharge List) */

l_MODIFIER_LIST_rec.currency_code:= 'USD';
```

```

l_MODIFIER_LIST_rec.list_type_code:= 'SLT';
l_MODIFIER_LIST_rec.start_date_active      := sysdate;
l_MODIFIER_LIST_rec.end_date_active        := sysdate+10;
l_MODIFIER_LIST_rec.source_system_code:= 'QP';
l_MODIFIER_LIST_rec.active_flag:= 'Y';
l_MODIFIER_LIST_rec.name:= 'Surcharge';
l_MODIFIER_LIST_rec.description:= '2% Surcharge';
l_MODIFIER_LIST_rec.version_no:= '1.9';
l_MODIFIER_LIST_rec.pte_code                := 'ORDFUL';
l_MODIFIER_LIST_rec.operation:= QP_GLOBALS.G_OPR_CREATE;

/* Create a Modifier line to specify 'Charge 2% Surcharge' condition */

l_MODIFIERS_tbl(1).list_line_type_code := 'SUR';
l_MODIFIERS_tbl(1).automatic_flag:= 'Y';
l_MODIFIERS_tbl(1).modifier_level_code := 'LINE';
l_MODIFIERS_tbl(1).accrual_flag := 'N';
l_MODIFIERS_tbl(1).start_date_active := sysdate;
l_MODIFIERS_tbl(1).end_date_active := sysdate+10;
l_MODIFIERS_tbl(1).operand := 2;
l_MODIFIERS_tbl(1).arithmetic_operator := '%';
l_MODIFIERS_tbl(1).pricing_group_sequence := 1;
l_MODIFIERS_tbl(1).pricing_phase_id := 2;
l_MODIFIERS_tbl(1).operation := QP_GLOBALS.G_OPR_CREATE;

/* Call the Modifiers Public API to create the modifier header and a modifier
line */

```

QP_Modifiers_PUB.Process_Modifiers

```

( p_api_version_number=> 1.0
, p_init_msg_list=> FND_API.G_FALSE
, p_return_values=> FND_API.G_FALSE
, p_commit=> FND_API.G_FALSE
, x_return_status=> l_return_status
, x_msg_count=>x_msg_count
, x_msg_data=>x_msg_data
,p_MODIFIER_LIST_rec=> l_MODIFIER_LIST_rec
,p_MODIFIERS_tbl=> l_MODIFIERS_tbl
--,p_QUALIFIERS_tbl=> l_QUALIFIERS_tbl
--,p_PRICING_ATTR_tbl=> l_PRICING_ATTR_tbl
,x_MODIFIER_LIST_rec=> l_MODIFIER_LIST_rec
,x_MODIFIER_LIST_val_rec=> l_MODIFIER_LIST_val_rec
,x_MODIFIERS_tbl=> l_MODIFIERS_tbl
,x_MODIFIERS_val_tbl=> l_MODIFIERS_val_tbl

```

```

,x_QUALIFIERS_tbl=> l_QUALIFIERS_tbl
,x_QUALIFIERS_val_tbl=> l_QUALIFIERS_val_tbl
,x_PRICING_ATTR_tbl=> l_PRICING_ATTR_tbl
,x_PRICING_ATTR_val_tbl=> l_PRICING_ATTR_val_tbl
);
    IF l_return_status <> FND_API.G_RET_STS_SUCCESS THEN

        RAISE FND_API.G_EXC_UNEXPECTED_ERROR;

    END IF;

EXCEPTION

    WHEN FND_API.G_EXC_ERROR THEN

        l_return_status := FND_API.G_RET_STS_ERROR;

--Get message count and data

        dbms_output.put_line('err msg 1 is : ' || x_msg_data);

    WHEN FND_API.G_EXC_UNEXPECTED_ERROR THEN

        l_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;
        dbms_output.put_line(' msg count 2 is : ' || x_msg_count);

    for k in 1 .. x_msg_count loop

        x_msg_data := oe_msg_pub.get( p_msg_index => k,
        p_encoded => 'F'
        );

        --Get message count and data
        dbms_output.put_line('err msg ' || k || 'is: ' || x_msg_data);
    end loop;

    WHEN OTHERS THEN

        l_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;

        dbms_output.put_line(' msg count 2 is : ' || x_msg_count);

        for k in 1 .. x_msg_count loop
            x_msg_data := oe_msg_pub.get( p_msg_index => k,

```

```

p_encoded => 'F'

);
-- Get message count and data
  dbms_output.put_line('err msg ' || k || 'is: ' || x_msg_data);

end loop;
END;
/

```

Example 5: For customer 1000, Buy more than 2 units of item 62081, Get a Payment Term 2/10 NET 30

```

File Path : $QP_TOP/patch/115/sql/QPXEXTSN.sql
/* Terms Substitution Creation - For customer 1000, Buy 2 of item 62081,
   Get a Payment Term 2/10 NET 30 */
REM FILETYPE : NOEXEC

```

```

REM Added for ARU db drv auto generation
REM dbdrv: none

```

```

SET VERIFY OFF
WHENEVER SQLERROR EXIT FAILURE ROLLBACK;
--set serveroutput on
declare
/* $Header: QPXEXTSN.sql 115.3 2002/05/31 22:45:46 mkarya noship $ */
l_control_recQP_GLOBALS.Control_Rec_Type;
l_return_statusVARCHAR2(1);
x_msg_countnumber;
x_msg_dataVarchar2(2000);
x_msg_indexnumber;

l_MODIFIER_LIST_recQP_Modifiers_PUB.Modifier_List_Rec_Type;
l_MODIFIER_LIST_val_recQP_Modifiers_PUB.Modifier_List_Val_Rec_Type;
l_MODIFIERS_tblQP_Modifiers_PUB.Modifiers_Tbl_Type;
l_MODIFIERS_val_tblQP_Modifiers_PUB.Modifiers_Val_Tbl_Type;
l_QUALIFIERS_tblQP_Qualifier_Rules_PUB.Qualifiers_Tbl_Type;
l_QUALIFIERS_val_tblQP_Qualifier_Rules_PUB.Qualifiers_Val_Tbl_Type;
l_PRICING_ATTR_tblQP_Modifiers_PUB.Pricing_Attr_Tbl_Type;
l_PRICING_ATTR_val_tblQP_Modifiers_PUB.Pricing_Attr_Val_Tbl_Type;

l_x_MODIFIER_LIST_recQP_Modifiers_PUB.Modifier_List_Rec_Type;
l_x_MODIFIER_LIST_val_recQP_Modifiers_PUB.Modifier_List_Val_Rec_Type;
l_x_MODIFIERS_tblQP_Modifiers_PUB.Modifiers_Tbl_Type;
l_x_MODIFIERS_val_tblQP_Modifiers_PUB.Modifiers_Val_Tbl_Type;

```

```

l_x_QUALIFIERS_tblQP_Qualifier_Rules_PUB.Qualifiers_Tbl_Type;
l_x_QUALIFIERS_val_tblQP_Qualifier_Rules_PUB.Qualifiers_Val_Tbl_Type;
l_x_PRICING_ATTR_tblQP_Modifiers_PUB.Pricing_Attr_Tbl_Type;
l_x_PRICING_ATTR_val_tblQP_Modifiers_PUB.Pricing_Attr_Val_Tbl_Type;

mll_rec qp_list_lines%ROWTYPE;
pra_rec qp_pricing_attributes%ROWTYPE;

Begin

/* Create a Modifier header of type 'PRO' (Promotion) */

    l_MODIFIER_LIST_rec.currency_code:= 'USD';
    l_MODIFIER_LIST_rec.list_type_code:= 'PRO';
    l_MODIFIER_LIST_rec.start_date_active      := sysdate;
    l_MODIFIER_LIST_rec.end_date_active        := sysdate+10;
    l_MODIFIER_LIST_rec.source_system_code:= 'QP';
    l_MODIFIER_LIST_rec.active_flag:= 'Y';
    l_MODIFIER_LIST_rec.name:= 'Terms Substitution';
    l_MODIFIER_LIST_rec.description:= 'New Year 2001 Promotion';
    l_MODIFIER_LIST_rec.version_no:= '1.7';
    l_MODIFIER_LIST_rec.pte_code               := 'ORDFUL';
    l_MODIFIER_LIST_rec.operation:= QP_GLOBALS.G_OPR_CREATE;

/* Create a Modifier line to specify 'Get Payment Term 2/10 NET 30 ( Terms
id=1000) ' condition */

    l_MODIFIERS_tbl(1).list_line_type_code := 'TSN';
    l_MODIFIERS_tbl(1).automatic_flag:= 'Y';
    l_MODIFIERS_tbl(1).modifier_level_code := 'LINE';
    l_MODIFIERS_tbl(1).accrual_flag := 'N';
    l_MODIFIERS_tbl(1).start_date_active := sysdate;
    l_MODIFIERS_tbl(1).end_date_active := sysdate+10;--
    l_MODIFIERS_tbl(1).pricing_group_sequence := 1;
    l_MODIFIERS_tbl(1).pricing_phase_id := 3;
    l_MODIFIERS_tbl(1).price_break_type_code := 'POINT';
    l_MODIFIERS_tbl(1).substitution_context := 'TERMS';
    l_MODIFIERS_tbl(1).substitution_attribute := 'QUALIFIER_ATTRIBUTE1';
    l_MODIFIERS_tbl(1).substitution_value := '1000';
    l_MODIFIERS_tbl(1).operation := QP_GLOBALS.G_OPR_CREATE;

/* Create a Pricing Attribute record to specify the 'Buy 2 of item 62081'
condition */

    l_PRICING_ATTR_tbl(1).product_attribute_context:= 'ITEM';

```

```

l_PRICING_ATTR_tbl(1).product_attribute:= 'PRICING_ATTRIBUTE1';
l_PRICING_ATTR_tbl(1).product_attr_value:= '62081';
l_PRICING_ATTR_tbl(1).pricing_attribute_context:= 'VOLUME';
l_PRICING_ATTR_tbl(1).pricing_attribute:= 'PRICING_ATTRIBUTE3';
l_PRICING_ATTR_tbl(1).pricing_attr_value_from:= '2';
l_PRICING_ATTR_tbl(1).comparison_operator_code:= 'BETWEEN';
l_PRICING_ATTR_tbl(1).product_uom_code:= 'Ea';
l_PRICING_ATTR_tbl(1).excluder_flag:= 'N';
l_PRICING_ATTR_tbl(1).MODIFIERS_index:=1;
l_PRICING_ATTR_tbl(1).operation                               := QP_GLOBALS.G_OPR_CREATE;

/* Create a Qualifier Record to specify 'For Customer 1000' condition */

l_QUALIFIERS_tbl(1).excluder_flag := 'N';
l_QUALIFIERS_tbl(1).comparison_operator_code := '=';
l_QUALIFIERS_tbl(1).qualifier_context := 'CUSTOMER';
l_QUALIFIERS_tbl(1).qualifier_attribute := 'QUALIFIER_ATTRIBUTE2';
l_QUALIFIERS_tbl(1).qualifier_attr_value := '1000';
l_QUALIFIERS_tbl(1).qualifier_grouping_no := 5467;
l_QUALIFIERS_tbl(1).qualifier_precedence := 1;
l_QUALIFIERS_tbl(1).start_date_active := sysdate;
l_QUALIFIERS_tbl(1).end_date_active := sysdate+10;
l_QUALIFIERS_tbl(1).operation := QP_GLOBALS.G_OPR_CREATE;

/* Call the Modifiers Public API to create the modifier header, modifier line,
qualifier and a pricing attributes record */

QP_Modifiers_PUB.Process_Modifiers

( p_api_version_number=> 1.0
, p_init_msg_list=> FND_API.G_FALSE
, p_return_values=> FND_API.G_FALSE
, p_commit=> FND_API.G_FALSE
, x_return_status=> l_return_status
, x_msg_count=>x_msg_count
, x_msg_data=>x_msg_data

,p_MODIFIER_LIST_rec=> l_MODIFIER_LIST_rec
,p_MODIFIERS_tbl=> l_MODIFIERS_tbl
,p_QUALIFIERS_tbl=> l_QUALIFIERS_tbl
,p_PRICING_ATTR_tbl=> l_PRICING_ATTR_tbl
,x_MODIFIER_LIST_rec=> l_MODIFIER_LIST_rec
,x_MODIFIER_LIST_val_rec=> l_MODIFIER_LIST_val_rec
,x_MODIFIERS_tbl=> l_MODIFIERS_tbl
,x_MODIFIERS_val_tbl=> l_MODIFIERS_val_tbl

```

```
,x_QUALIFIERS_tbl=> l_QUALIFIERS_tbl
,x_QUALIFIERS_val_tbl=> l_QUALIFIERS_val_tbl
,x_PRICING_ATTR_tbl=> l_PRICING_ATTR_tbl
,x_PRICING_ATTR_val_tbl=> l_PRICING_ATTR_val_tbl
);

IF l_return_status <> FND_API.G_RET_STS_SUCCESS THEN

    RAISE FND_API.G_EXC_UNEXPECTED_ERROR;

END IF;

EXCEPTION

    WHEN FND_API.G_EXC_ERROR THEN

l_return_status := FND_API.G_RET_STS_ERROR;

--Get message count and data

    dbms_output.put_line('err msg 1 is : ' || x_msg_data);

    WHEN FND_API.G_EXC_UNEXPECTED_ERROR THEN

        l_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;

        dbms_output.put_line(' msg count 2 is : ' || x_msg_count);

    for k in 1 .. x_msg_count loop

x_msg_data := oe_msg_pub.get( p_msg_index => k,

p_encoded => 'F'

    );

        --Get message count and data

        dbms_output.put_line('err msg ' || k || 'is: ' || x_msg_data);

    end loop;

    WHEN OTHERS THEN

        l_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;
```



```

        dbms_output.put_line(' msg count 2 is : ' || x_msg_count);

    for k in 1 .. x_msg_count loop

        x_msg_data := oe_msg_pub.get( p_msg_index => k,
        p_encoded => 'F'

    );
        --Get message count and data
        dbms_output.put_line('err msg ' || k || 'is: ' || x_msg_data);

    end loop;

END;
/

```

Example 6: Other Item Discount Creation - Buy 1 of item 62081, Get 1 free

File Path: \$QP_TOP/patch/115/sql/QPXEX0ID.sql

```

/* Other Item Discount Creation - Buy 1 of item 62081, Get 1 free */
REM FILETYPE : NOEXEC

REM Added for ARU db drv auto generation
REM dbdrv: none

SET VERIFY OFF
WHenever SQLERROR EXIT FAILURE ROLLBACK;
--set serveroutput on
declare
/* $Header: QPXEX0ID.sql 115.2 2002/05/31 22:28:05 mkarya noship $ */

l_control_recQP_GLOBALS.Control_Rec_Type;
l_return_statusVARCHAR2(1);
x_msg_countnumber;
x_msg_dataVarchar2(2000);
x_msg_indexnumber;

l_MODIFIER_LIST_recQP_Modifiers_PUB.Modifier_List_Rec_Type;
l_MODIFIER_LIST_val_recQP_Modifiers_PUB.Modifier_List_Val_Rec_Type;
l_MODIFIERS_tblQP_Modifiers_PUB.Modifiers_Tbl_Type;
l_MODIFIERS_val_tblQP_Modifiers_PUB.Modifiers_Val_Tbl_Type;
l_QUALIFIERS_tblQP_Qualifier_Rules_PUB.Qualifiers_Tbl_Type;
l_QUALIFIERS_val_tblQP_Qualifier_Rules_PUB.Qualifiers_Val_Tbl_Type;

```

```

l_PRICING_ATTR_tblQP_Modifiers_PUB.Pricing_Attr_Tbl_Type;
l_PRICING_ATTR_val_tblQP_Modifiers_PUB.Pricing_Attr_Val_Tbl_Type;

l_x_MODIFIER_LIST_recQP_Modifiers_PUB.Modifier_List_Rec_Type;
l_x_MODIFIER_LIST_val_recQP_Modifiers_PUB.Modifier_List_Val_Rec_Type;
l_x_MODIFIERS_tblQP_Modifiers_PUB.Modifiers_Tbl_Type;
l_x_MODIFIERS_val_tblQP_Modifiers_PUB.Modifiers_Val_Tbl_Type;
l_x_QUALIFIERS_tblQP_Qualifier_Rules_PUB.Qualifiers_Tbl_Type;
l_x_QUALIFIERS_val_tblQP_Qualifier_Rules_PUB.Qualifiers_Val_Tbl_Type;
l_x_PRICING_ATTR_tblQP_Modifiers_PUB.Pricing_Attr_Tbl_Type;
l_x_PRICING_ATTR_val_tblQP_Modifiers_PUB.Pricing_Attr_Val_Tbl_Type;

mll_rec qp_list_lines%ROWTYPE;
pra_rec qp_pricing_attributes%ROWTYPE;

Begin

/* Create a Modifier header of type 'PRO' (Promotion) */
  l_MODIFIER_LIST_rec.currency_code:= 'USD';
  l_MODIFIER_LIST_rec.list_type_code:= 'PRO';
  l_MODIFIER_LIST_rec.start_date_active      := sysdate;
  l_MODIFIER_LIST_rec.end_date_active        := sysdate+10;
  l_MODIFIER_LIST_rec.source_system_code:= 'QP';
  l_MODIFIER_LIST_rec.active_flag:= 'Y';
  l_MODIFIER_LIST_rec.name:= 'latest 3.7  OID 2001 Promotion';
  l_MODIFIER_LIST_rec.description:= 'latest OID 2001 Promotion';
  l_MODIFIER_LIST_rec.version_no:= '3.7';
  l_MODIFIER_LIST_rec.pte_code               := 'ORDFUL';
  l_MODIFIER_LIST_rec.operation:= QP_GLOBALS.G_OPR_CREATE;

/* Create a Modifier line to specify Other Item Discount (OID) */

l_MODIFIERS_tbl(1).list_line_type_code := 'OID';
l_MODIFIERS_tbl(1).automatic_flag:= 'Y';
l_MODIFIERS_tbl(1).modifier_level_code := 'LINE';
l_MODIFIERS_tbl(1).accrual_flag := 'N';
l_MODIFIERS_tbl(1).start_date_active := sysdate;
l_MODIFIERS_tbl(1).end_date_active := sysdate+10;
l_MODIFIERS_tbl(1).pricing_group_sequence := 1;
l_MODIFIERS_tbl(1).pricing_phase_id := 3;
l_MODIFIERS_tbl(1).product_precedence := 12;
l_MODIFIERS_tbl(1).price_break_type_code := 'POINT';
l_MODIFIERS_tbl(1).modifier_parent_index := 1;
l_MODIFIERS_tbl(1).operation := QP_GLOBALS.G_OPR_CREATE;

```

```
/* Create a Modifier line to specify the free (100% discount) condition */
```

```
l_MODIFIERS_tbl(2).list_line_type_code := 'DIS';
l_MODIFIERS_tbl(2).automatic_flag:= 'Y';
l_MODIFIERS_tbl(2).modifier_level_code := 'LINE';
l_MODIFIERS_tbl(2).accrual_flag := 'N';
l_MODIFIERS_tbl(2).start_date_active := sysdate;
l_MODIFIERS_tbl(2).end_date_active := sysdate+10;
l_MODIFIERS_tbl(2).operand := 100;
l_MODIFIERS_tbl(2).arithmetic_operator := '%';
l_MODIFIERS_tbl(2).pricing_group_sequence := 1;
l_MODIFIERS_tbl(2).pricing_phase_id := 3;
l_MODIFIERS_tbl(2).product_precedence := 12;
l_MODIFIERS_tbl(2).price_break_type_code := 'POINT';
l_MODIFIERS_tbl(2).modifier_parent_index := 1;
l_MODIFIERS_tbl(2).rltd_modifier_grp_no := 12;
l_MODIFIERS_tbl(2).rltd_modifier_grp_type := 'BENEFIT';
l_MODIFIERS_tbl(2).operation := QP_GLOBALS.G_OPR_CREATE;
```

```
/* Create a Pricing Attribute record to specify the 'Buy 1 of item 62081'
condition. It is linked to the OID modifier record by specifying modifiers_
index=1 */
```

```
l_PRICING_ATTR_tbl(1).product_attribute_context:= 'ITEM';
l_PRICING_ATTR_tbl(1).product_attribute:= 'PRICING_ATTRIBUTE1';
l_PRICING_ATTR_tbl(1).product_attr_value:= '62081';
l_PRICING_ATTR_tbl(1).pricing_attribute_context:= 'VOLUME';
l_PRICING_ATTR_tbl(1).pricing_attribute:= 'PRICING_ATTRIBUTE3';
l_PRICING_ATTR_tbl(1).pricing_attr_value_from:= '1';
l_PRICING_ATTR_tbl(1).comparison_operator_code:= 'BETWEEN';
l_PRICING_ATTR_tbl(1).product_uom_code:= 'Ea';
l_PRICING_ATTR_tbl(1).excluder_flag:= 'N';
l_PRICING_ATTR_tbl(1).MODIFIERS_index:=1;
l_PRICING_ATTR_tbl(1).operation:= QP_GLOBALS.G_OPR_CREATE;
```

```
/* Create a Pricing Attribute record to specify the 'Get 1 of item 62081'
condition. It is linked to the DIS modifier record by specifying modifiers_
index=2 */
```

```
l_PRICING_ATTR_tbl(2).product_attribute_context:= 'ITEM';
l_PRICING_ATTR_tbl(2).product_attribute:= 'PRICING_ATTRIBUTE1';
l_PRICING_ATTR_tbl(2).product_attr_value:= '62081';
l_PRICING_ATTR_tbl(2).pricing_attribute_context:= 'VOLUME';
l_PRICING_ATTR_tbl(2).pricing_attribute:= 'PRICING_ATTRIBUTE3';
l_PRICING_ATTR_tbl(2).pricing_attr_value_from:= '1';
```

```
l_PRICING_ATTR_tbl(2).comparison_operator_code:= 'BETWEEN';
l_PRICING_ATTR_tbl(2).product_uom_code:= 'Ea';
l_PRICING_ATTR_tbl(2).excluder_flag:= 'N';
l_PRICING_ATTR_tbl(2).MODIFIERS_index:=2;
l_PRICING_ATTR_tbl(2).operation                := QP_GLOBALS.G_OPR_CREATE;
```

QP_Modifiers_PUB.Process_Modifiers

```
( p_api_version_number => 1
, p_init_msg_list => FND_API.G_FALSE
, p_return_values => FND_API.G_FALSE
, p_commit => FND_API.G_FALSE
, x_return_status => l_return_status
, x_msg_count => x_msg_count
, x_msg_data => x_msg_data

,p_MODIFIER_LIST_rec=> l_MODIFIER_LIST_rec
,p_MODIFIERS_tbl=> l_MODIFIERS_tbl
,p_PRICING_ATTR_tbl=> l_PRICING_ATTR_tbl
,x_MODIFIER_LIST_rec=> l_x_MODIFIER_LIST_rec
,x_MODIFIER_LIST_val_rec=> l_x_MODIFIER_LIST_val_rec
,x_MODIFIERS_tbl=> l_x_MODIFIERS_tbl
,x_MODIFIERS_val_tbl=> l_x_MODIFIERS_val_tbl
,x_QUALIFIERS_tbl=> l_x_QUALIFIERS_tbl
,x_QUALIFIERS_val_tbl=> l_x_QUALIFIERS_val_tbl
,x_PRICING_ATTR_tbl=> l_x_PRICING_ATTR_tbl
,x_PRICING_ATTR_val_tbl=> l_x_PRICING_ATTR_val_tbl
);

IF gpr_return_status <> FND_API.G_RET_STS_SUCCESS THEN

    RAISE FND_API.G_EXC_UNEXPECTED_ERROR;

END IF;

EXCEPTION

    WHEN FND_API.G_EXC_ERROR THEN

        l_return_status := FND_API.G_RET_STS_ERROR;

        -- Get message count and data

        --dbms_output.put_line('err msg 1 is : ' || x_msg_data);
```

```

Rollback

WHEN FND_API.G_EXC_UNEXPECTED_ERROR THEN

    l_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;
    --dbms_output.put_line(' msg count 2 is : ' || x_msg_count);

    for k in 1 .. x_msg_                                count loop
    x_msg_data := oe_msg_pub.get( p_msg_index => k,

        p_encoded => 'F'
    );
        /*
            oe_msg_pub.Count_And_Get
        (   p_count  => gpr_msg_count
        ,   p_data   => gpr_msg_data
        */

    --Get message count and data
    --dbms_output.put_line('err msg ' || k || 'is: ' || x_msg_data);
    null;
    end loop;
    Rollback;

    WHEN OTHERS THEN

        l_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;

    --dbms_output.put_line(' msg count 5 is : ' || x_msg_count);

        for k in 1 .. x_msg_count loop

            x_msg_data := oe_msg_pub.get( p_msg_index => k,

p_encoded => 'F'

        );
            --Get message count and data
            --dbms_output.put_line('err msg ' || k || 'is: ' || x_msg_data);

        end loop;
    END;
/

```

Example 7: Promotional Goods - Buy 1 of item 45 and 1 item 63, Get 1 item 62081 at 20% discount

File Path : \$QP_TOP/patch/115/sql/QPXEXPRG.sql

```
/* Promotional Goods - Buy 1 of item 45 and 1 item 63, Get 1 item 62081 at 20%
discount */
REM FILETYPE : NOEXEC
```

```
REM Added for ARU db drv auto generation
REM dbdrv: none
```

```
SET VERIFY OFF
WHENEVER SQLERROR EXIT FAILURE ROLLBACK;
--set serveroutput on
declare
/* $Header: QPXEXPRG.sql 115.2 2002/05/31 22:36:47 mkarya noship $ */
```

```
l_control_recQP_GLOBALS.Control_Rec_Type;
l_return_statusVARCHAR2(1);
x_msg_countnumber;
x_msg_dataVarchar2(2000);
x_msg_indexnumber;
```

```
l_MODIFIER_LIST_recQP_Modifiers_PUB.Modifier_List_Rec_Type;
l_MODIFIER_LIST_val_recQP_Modifiers_PUB.Modifier_List_Val_Rec_Type;
l_MODIFIERS_tblQP_Modifiers_PUB.Modifiers_Tbl_Type;
l_MODIFIERS_val_tblQP_Modifiers_PUB.Modifiers_Val_Tbl_Type;
l_QUALIFIERS_tblQP_Qualifier_Rules_PUB.Qualifiers_Tbl_Type;
l_QUALIFIERS_val_tblQP_Qualifier_Rules_PUB.Qualifiers_Val_Tbl_Type;
l_PRICING_ATTR_tblQP_Modifiers_PUB.Pricing_Attr_Tbl_Type;
l_PRICING_ATTR_val_tblQP_Modifiers_PUB.Pricing_Attr_Val_Tbl_Type;
```

```
l_x_MODIFIER_LIST_recQP_Modifiers_PUB.Modifier_List_Rec_Type;
l_x_MODIFIER_LIST_val_recQP_Modifiers_PUB.Modifier_List_Val_Rec_Type;
l_x_MODIFIERS_tblQP_Modifiers_PUB.Modifiers_Tbl_Type;
l_x_MODIFIERS_val_tblQP_Modifiers_PUB.Modifiers_Val_Tbl_Type;
l_x_QUALIFIERS_tblQP_Qualifier_Rules_PUB.Qualifiers_Tbl_Type;
l_x_QUALIFIERS_val_tblQP_Qualifier_Rules_PUB.Qualifiers_Val_Tbl_Type;
l_x_PRICING_ATTR_tblQP_Modifiers_PUB.Pricing_Attr_Tbl_Type;
l_x_PRICING_ATTR_val_tblQP_Modifiers_PUB.Pricing_Attr_Val_Tbl_Type;
```

```
mll_rec qp_list_lines%ROWTYPE;
pra_rec qp_pricing_attributes%ROWTYPE;
```

Begin

```

/* Create a Modifier header of type 'PRO' (Promotion)- linked to item 45 */

    l_MODIFIER_LIST_rec.currency_code:= 'USD';
    l_MODIFIER_LIST_rec.list_type_code:= 'PRO';
    l_MODIFIER_LIST_rec.start_date_active      := sysdate;
    l_MODIFIER_LIST_rec.end_date_active        := sysdate+10;
    l_MODIFIER_LIST_rec.source_system_code:= 'QP';
    l_MODIFIER_LIST_rec.active_flag:= 'Y';
    l_MODIFIER_LIST_rec.name := 'latest 1.8 PRG 2001 Promotion';
    l_MODIFIER_LIST_rec.description:= 'latest PRG 2001 Promotion';
    l_MODIFIER_LIST_rec.version_no:= '1.8';
    l_MODIFIER_LIST_rec.pte_code                := 'ORDFUL';
    l_MODIFIER_LIST_rec.operation:= QP_GLOBALS.G_OPR_CREATE;

/* Create a Modifier line to specify Promotional Goods (PRG) - linked to item 45
*/

    l_MODIFIERS_tbl(1).list_line_type_code := 'PRG';
    l_MODIFIERS_tbl(1).automatic_flag:= 'Y';
    l_MODIFIERS_tbl(1).modifier_level_code := 'LINE';
    l_MODIFIERS_tbl(1).accrual_flag := 'N';
    l_MODIFIERS_tbl(1).start_date_active := sysdate;
    l_MODIFIERS_tbl(1).end_date_active := sysdate+10;
    l_MODIFIERS_tbl(1).pricing_group_sequence := 1;
    l_MODIFIERS_tbl(1).pricing_phase_id := 3;
    l_MODIFIERS_tbl(1).product_precedence := 12;
    l_MODIFIERS_tbl(1).price_break_type_code := 'POINT';
    l_MODIFIERS_tbl(1).modifier_parent_index := 1;
    l_MODIFIERS_tbl(1).operation := QP_GLOBALS.G_OPR_CREATE;

/* Create a Modifier line to specify related item (RLTD) - linked to item 63 */

    l_MODIFIERS_tbl(2).list_line_type_code := 'RLTD';
    l_MODIFIERS_tbl(2).automatic_flag:= 'Y';
    l_MODIFIERS_tbl(2).modifier_level_code := 'LINE';
    l_MODIFIERS_tbl(2).accrual_flag := 'N';
    l_MODIFIERS_tbl(2).start_date_active := sysdate;
    l_MODIFIERS_tbl(2).end_date_active := sysdate+10;
    l_MODIFIERS_tbl(2).pricing_group_sequence := 1;
    l_MODIFIERS_tbl(2).pricing_phase_id := 3;
    l_MODIFIERS_tbl(2).product_precedence := 12;
    l_MODIFIERS_tbl(2).price_break_type_code := 'POINT';
    l_MODIFIERS_tbl(2).modifier_parent_index := 1;

```

```

l_MODIFIERS_tbl(2).rltd_modifier_grp_no := 12;
l_MODIFIERS_tbl(2).rltd_modifier_grp_type := 'QUALIFIER';
l_MODIFIERS_tbl(2).operation := QP_GLOBALS.G_OPR_CREATE;

/* Create a Modifier line to specify 20% discount on item 62081 */

l_MODIFIERS_tbl(3).list_line_type_code := 'DIS';
l_MODIFIERS_tbl(3).automatic_flag:= 'Y';
l_MODIFIERS_tbl(3).modifier_level_code := 'LINE';
l_MODIFIERS_tbl(3).accrual_flag := 'N';
l_MODIFIERS_tbl(3).start_date_active := sysdate;
l_MODIFIERS_tbl(3).end_date_active := sysdate+10;
l_MODIFIERS_tbl(3).operand := 20;
l_MODIFIERS_tbl(3).arithmetic_operator := '%';
l_MODIFIERS_tbl(3).pricing_group_sequence := 1;
l_MODIFIERS_tbl(3).pricing_phase_id := 3;
l_MODIFIERS_tbl(3).product_precedence := 12;
l_MODIFIERS_tbl(3).price_break_type_code := 'POINT';
l_MODIFIERS_tbl(3).modifier_parent_index := 1;
l_MODIFIERS_tbl(3).rltd_modifier_grp_no := 10;
l_MODIFIERS_tbl(3).rltd_modifier_grp_type := 'BENEFIT';
l_MODIFIERS_tbl(3).benefit_price_list_line_id := 41187;
l_MODIFIERS_tbl(3).benefit_qty := 1;
l_MODIFIERS_tbl(3).benefit_uom_code := 'Ea';
l_MODIFIERS_tbl(3).operation := QP_GLOBALS.G_OPR_CREATE;

/* Create a Pricing Attribute record to specify the 'Buy 1 of item 45'
condition. It is linked to the PRG modifier record by specifying modifiers_
index=1 */

l_PRICING_ATTR_tbl(1).product_attribute_context:= 'ITEM';
l_PRICING_ATTR_tbl(1).product_attribute:= 'PRICING_ATTRIBUTE1';
l_PRICING_ATTR_tbl(1).product_attr_value:= '45';
l_PRICING_ATTR_tbl(1).pricing_attribute_context:= 'VOLUME';
l_PRICING_ATTR_tbl(1).pricing_attribute:= 'PRICING_ATTRIBUTE3';
l_PRICING_ATTR_tbl(1).pricing_attr_value_from:= '1';
l_PRICING_ATTR_tbl(1).comparison_operator_code:= 'BETWEEN';
l_PRICING_ATTR_tbl(1).product_uom_code:= 'Ea';
l_PRICING_ATTR_tbl(1).excluder_flag:= 'N';
l_PRICING_ATTR_tbl(1).MODIFIERS_index:=1;
l_PRICING_ATTR_tbl(1).operation := QP_GLOBALS.G_OPR_CREATE;

/* Create a Pricing Attribute record to specify the 'Buy 1 of item 63'
condition. It is linked to the RLTD modifier record by specifying modifiers_
index=2 */

```



```

l_PRICING_ATTR_tbl(2).product_attribute_context:= 'ITEM';
l_PRICING_ATTR_tbl(2).product_attribute:= 'PRICING_ATTRIBUTE1';
l_PRICING_ATTR_tbl(2).product_attr_value:= '63';
l_PRICING_ATTR_tbl(2).pricing_attribute_context:= 'VOLUME';
l_PRICING_ATTR_tbl(2).pricing_attribute:= 'PRICING_ATTRIBUTE3';
l_PRICING_ATTR_tbl(2).pricing_attr_value_from:= '1';
l_PRICING_ATTR_tbl(2).comparison_operator_code:= 'BETWEEN';
l_PRICING_ATTR_tbl(2).product_uom_code:= 'Ea';
l_PRICING_ATTR_tbl(2).excluder_flag:= 'N';
l_PRICING_ATTR_tbl(2).MODIFIERS_index:=2;
l_PRICING_ATTR_tbl(2).operation                               := QP_GLOBALS.G_OPR_CREATE;

```

```

/* Create a Pricing Attribute record to specify the 'Get 1 of item 62081'
condition. It is linked to the DIS modifier record by specifying modifiers_
index=3 */

```

```

l_PRICING_ATTR_tbl(3).product_attribute_context:= 'ITEM';
l_PRICING_ATTR_tbl(3).product_attribute:= 'PRICING_ATTRIBUTE1';
l_PRICING_ATTR_tbl(3).product_attr_value:= '62081';
l_PRICING_ATTR_tbl(3).pricing_attribute_context:= 'VOLUME';
l_PRICING_ATTR_tbl(3).pricing_attribute:= 'PRICING_ATTRIBUTE3';
l_PRICING_ATTR_tbl(3).pricing_attr_value_from:= '1';
l_PRICING_ATTR_tbl(3).comparison_operator_code:= 'BETWEEN';
l_PRICING_ATTR_tbl(3).product_uom_code:= 'Ea';
l_PRICING_ATTR_tbl(3).excluder_flag:= 'N';
l_PRICING_ATTR_tbl(3).MODIFIERS_index:=3;
l_PRICING_ATTR_tbl(3).operation                               := QP_GLOBALS.G_OPR_CREATE;

```

QP_Modifiers_PUB.Process_Modifiers

```

( p_api_version_number=> 1.0
, p_init_msg_list=> FND_API.G_FALSE
, p_return_values=> FND_API.G_FALSE
, p_commit=> FND_API.G_FALSE
, x_return_status=> l_return_status
, x_msg_count=>x_msg_count
, x_msg_data=>x_msg_data

,p_MODIFIER_LIST_rec=> l_MODIFIER_LIST_rec
,p_MODIFIERS_tbl=> l_MODIFIERS_tbl
,p_PRICING_ATTR_tbl=> l_PRICING_ATTR_tbl
,x_MODIFIER_LIST_rec=> l_x_MODIFIER_LIST_rec
,x_MODIFIER_LIST_val_rec=> l_x_MODIFIER_LIST_val_rec
,x_MODIFIERS_tbl=> l_x_MODIFIERS_tbl

```

```
,x_MODIFIERS_val_tbl=> l_x_MODIFIERS_val_tbl
,x_QUALIFIERS_tbl=> l_x_QUALIFIERS_tbl
,x_QUALIFIERS_val_tbl=> l_x_QUALIFIERS_val_tbl
,x_PRICING_ATTR_tbl=> l_x_PRICING_ATTR_tbl
,x_PRICING_ATTR_val_tbl=> l_x_PRICING_ATTR_val_tbl
);

IF l_return_status <> FND_API.G_RET_STS_SUCCESS THEN

    RAISE FND_API.G_EXC_UNEXPECTED_ERROR;

END IF;

EXCEPTION

    WHEN FND_API.G_EXC_ERROR THEN

        l_return_status := FND_API.G_RET_STS_ERROR;

        --Get message count and data

        --dbms_output.put_line('err msg 1 is : ' || x_msg_data);

    WHEN FND_API.G_EXC_UNEXPECTED_ERROR THEN

        l_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;
        --dbms_output.put_line(' msg count 2 is : ' || x_msg_count);

        for k in 1 .. x_msg_count loop

x_msg_data := oe_msg_pub.get( p_msg_index => k,

p_encoded => 'F'

);

        --Get message count and data
        --dbms_output.put_line('err msg ' || k || 'is: ' || x_msg_data);

        end loop;

    WHEN OTHERS THEN

        l_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;
```

```

--dbms_output.put_line(' msg count 5 is : ' || x_msg_count);

    for k in 1 .. x_msg_count loop

x_msg_data := oe_msg_pub.get( p_msg_index => k,
p_encoded => 'F'
);
    --Get message count and data

    --dbms_output.put_line('err msg ' || k || 'is: ' || x_msg_data);

end loop;
END;
/

```

Example 8: Coupon Issue - Buy 2 of item 45, Get Coupon for 20% discount

```

File Path : $QP_TOP/patch/115/sql/QPXEXCIE.sql
-- Coupon Issue - Buy 2 of item 45, Get Coupon for 20% discount
REM FILETYPE : NOEXEC

REM Added for ARU db drv auto generation
REM dbdrv: none

SET VERIFY OFF
WHenever SQLERROR EXIT FAILURE ROLLBACK;
--set serveroutput on
declare
/* $Header: QPXEXCIE.sql 115.2 2002/05/31 22:19:14 mkarya noship $ */

l_control_recQP_GLOBALS.Control_Rec_Type;
l_return_statusVARCHAR2(1);
x_msg_countnumber;
x_msg_dataVarchar2(2000);
x_msg_indexnumber;

l_MODIFIER_LIST_recQP_Modifiers_PUB.Modifier_List_Rec_Type;
l_MODIFIER_LIST_val_recQP_Modifiers_PUB.Modifier_List_Val_Rec_Type;
l_MODIFIERS_tblQP_Modifiers_PUB.Modifiers_Tbl_Type;
l_MODIFIERS_val_tblQP_Modifiers_PUB.Modifiers_Val_Tbl_Type;
l_QUALIFIERS_tblQP_Qualifier_Rules_PUB.Qualifiers_Tbl_Type;
l_QUALIFIERS_val_tblQP_Qualifier_Rules_PUB.Qualifiers_Val_Tbl_Type;
l_PRICING_ATTR_tblQP_Modifiers_PUB.Pricing_Attr_Tbl_Type;
l_PRICING_ATTR_val_tblQP_Modifiers_PUB.Pricing_Attr_Val_Tbl_Type;

```

```

l_x_MODIFIER_LIST_recQP_Modifiers_PUB.Modifier_List_Rec_Type;
l_x_MODIFIER_LIST_val_recQP_Modifiers_PUB.Modifier_List_Val_Rec_Type;
l_x_MODIFIERS_tblQP_Modifiers_PUB.Modifiers_Tbl_Type;
l_x_MODIFIERS_val_tblQP_Modifiers_PUB.Modifiers_Val_Tbl_Type;
l_x_QUALIFIERS_tblQP_Qualifier_Rules_PUB.Qualifiers_Tbl_Type;
l_x_QUALIFIERS_val_tblQP_Qualifier_Rules_PUB.Qualifiers_Val_Tbl_Type;
l_x_PRICING_ATTR_tblQP_Modifiers_PUB.Pricing_Attr_Tbl_Type;
l_x_PRICING_ATTR_val_tblQP_Modifiers_PUB.Pricing_Attr_Val_Tbl_Type;

mll_rec qp_list_lines%ROWTYPE;
pra_rec qp_pricing_attributes%ROWTYPE;

Begin

/* Create a Modifier header of type 'PRO' (Promotion) */

    l_MODIFIER_LIST_rec.currency_code:= 'USD';
    l_MODIFIER_LIST_rec.list_type_code:= 'PRO';
    l_MODIFIER_LIST_rec.start_date_active      := sysdate;
    l_MODIFIER_LIST_rec.end_date_active        := sysdate+10;
    l_MODIFIER_LIST_rec.source_system_code:= 'QP';
    l_MODIFIER_LIST_rec.active_flag:= 'Y';
    l_MODIFIER_LIST_rec.name:= 'latest 2.1 CIE 2001 Promotion';
    l_MODIFIER_LIST_rec.description:= 'latest CIE 2001 Promotion';
    l_MODIFIER_LIST_rec.version_no:= '2.1';
    l_MODIFIER_LIST_rec.pte_code                := 'ORDFUL';
    l_MODIFIER_LIST_rec.operation:= QP_GLOBALS.G_OPR_CREATE;

/* Create a modifier of type Coupon Issue (CIE). The from_rltd_modifier_id is
the list_line_id of another modifier which specifies the 'Get 20% discount'
condition of the Coupon. So, this modifier needs to be created prior to creating
this Coupun Issue */

    l_MODIFIERS_tbl(1).list_line_type_code := 'CIE';
    l_MODIFIERS_tbl(1).automatic_flag:= 'Y';
    l_MODIFIERS_tbl(1).modifier_level_code := 'LINE';
    l_MODIFIERS_tbl(1).accrual_flag := 'Y';
    l_MODIFIERS_tbl(1).start_date_active := sysdate;
    l_MODIFIERS_tbl(1).end_date_active := sysdate+10;
    l_MODIFIERS_tbl(1).pricing_group_sequence := 1;
    l_MODIFIERS_tbl(1).pricing_phase_id := 3;
    l_MODIFIERS_tbl(1).product_precedence := 12;
    l_MODIFIERS_tbl(1).price_break_type_code := 'POINT';
    l_MODIFIERS_tbl(1).modifier_parent_index := 1;
    l_MODIFIERS_tbl(1).to_rltd_modifier_id := 306693;

```

```

l_MODIFIERS_tbl(1).rltd_modifier_grp_no := 10;
l_MODIFIERS_tbl(1).rltd_modifier_grp_type := 'COUPON';
l_MODIFIERS_tbl(1).operation := QP_GLOBALS.G_OPR_CREATE;

/* Create a Pricing Attribute record to specify the 'Buy 2 of item 45'
condition. It is linked to the CIE modifier record by specifying modifiers_
index=1 */

l_PRICING_ATTR_tbl(1).product_attribute_context:= 'ITEM';
l_PRICING_ATTR_tbl(1).product_attribute:= 'PRICING_ATTRIBUTE1';
l_PRICING_ATTR_tbl(1).product_attr_value:= '45';
l_PRICING_ATTR_tbl(1).pricing_attribute_context:= 'VOLUME';
l_PRICING_ATTR_tbl(1).pricing_attribute:= 'PRICING_ATTRIBUTE10';
l_PRICING_ATTR_tbl(1).pricing_attr_value_from := 2;
l_PRICING_ATTR_tbl(1).comparison_operator_code:= 'BETWEEN';
l_PRICING_ATTR_tbl(1).product_uom_code:= 'Ea';
l_PRICING_ATTR_tbl(1).excluder_flag:= 'N';
l_PRICING_ATTR_tbl(1).MODIFIERS_index:=1;
l_PRICING_ATTR_tbl(1).operation := QP_GLOBALS.G_

    , p_init_msg_list=> FND_API.G_FALSE
    , p_return_values=> FND_API.G_FALSE
    , p_commit=> FND_API.G_FALSE
    , x_return_status=> l_return_status
    , x_msg_count=>x_msg_count
    , x_msg_data=>x_msg_data

,p_MODIFIER_LIST_rec=> l_MODIFIER_LIST_rec
,p_MODIFIERS_tbl=> l_MODIFIERS_tbl
,p_QUALIFIERS_tbl=> l_QUALIFIERS_tbl
,p_PRICING_ATTR_tbl=> l_PRICING_ATTR_tbl
,x_MODIFIER_LIST_rec=> l_x_MODIFIER_LIST_rec
,x_MODIFIER_LIST_val_rec=> l_x_MODIFIER_LIST_val_rec
,x_MODIFIERS_tbl=> l_x_MODIFIERS_tbl
,x_MODIFIERS_val_tbl=> l_x_MODIFIERS_val_tbl
,x_QUALIFIERS_tbl=> l_x_QUALIFIERS_tbl
,x_QUALIFIERS_val_tbl=> l_x_QUALIFIERS_val_tbl
,x_PRICING_ATTR_tbl=> l_x_PRICING_ATTR_tbl
,x_PRICING_ATTR_val_tbl=> l_x_PRICING_ATTR_val_tbl
);

IF l_return_status <> FND_API.G_RET_STS_SUCCESS THEN

    RAISE FND_API.G_EXC_UNEXPECTED_ERROR;

```

```

        END IF;

    EXCEPTION

        WHEN FND_API.G_EXC_ERROR THEN

            l_return_status := FND_API.G_RET_STS_ERROR;

            --Get message count and data

            --dbms_output.put_line('err msg 1 is : ' || x_msg_data);

        WHEN FND_API.G_EXC_UNEXPECTED_ERROR THEN

            l_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;
            --dbms_output.put_line(' msg count 2 is : ' || x_msg_count);

            for k in 1 .. x_msg_count loop

x_msg_data := oe_msg_pub.get( p_msg_index => k,
p_encoded => 'F'

);
            --Get message count and data
            --dbms_output.put_line('err msg ' || k || 'is: ' || x_msg_data);

            end loop;

        WHEN OTHERS THEN

            l_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;

            --dbms_output.put_line(' msg count 5 is : ' || x_msg_count);

            for k in 1 .. x_msg_count loop

x_msg_data := oe_msg_pub.get( p_msg_index => k,
p_encoded => 'F'

);
            --Get message count and data
            --dbms_output.put_line('err msg ' || k || 'is: ' || x_msg_data);

            end loop;

```

```
END;
/
```

Example 9: Price Break - Buy 1-100 of item 45, Get 20% discount

Buy 101-200 of item 45, Get 25% discount

File Path: \$QP_TOP/patch/115/sql/QPXEXPBH.sql

```
/* Price Break - Buy 1-100 of item 45, Get 20% discount
Buy 101-200 of item 45, Get 25% discount */
REM FILETYPE : NOEXEC

REM Added for ARU db drv auto generation
REM dbdrv: none

SET VERIFY OFF
WHenever SQLERROR EXIT FAILURE ROLLBACK;
--set serveroutput on
declare
/* $Header: QPXEXPBH.sql 115.2 2002/05/31 22:32:45 mkarya noship $ */

l_control_recQP_GLOBALS.Control_Rec_Type;
l_return_statusVARCHAR2(1);
x_msg_countnumber;
x_msg_dataVarchar2(2000);
x_msg_indexnumber;

l_MODIFIER_LIST_recQP_Modifiers_PUB.Modifier_List_Rec_Type;
l_MODIFIER_LIST_val_recQP_Modifiers_PUB.Modifier_List_Val_Rec_Type;
l_MODIFIERS_tblQP_Modifiers_PUB.Modifiers_Tbl_Type;
l_MODIFIERS_val_tblQP_Modifiers_PUB.Modifiers_Val_Tbl_Type;
l_QUALIFIERS_tblQP_Qualifier_Rules_PUB.Qualifiers_Tbl_Type;
l_QUALIFIERS_val_tblQP_Qualifier_Rules_PUB.Qualifiers_Val_Tbl_Type;
l_PRICING_ATTR_tblQP_Modifiers_PUB.Pricing_Attr_Tbl_Type;
l_PRICING_ATTR_val_tblQP_Modifiers_PUB.Pricing_Attr_Val_Tbl_Type;

l_x_MODIFIER_LIST_recQP_Modifiers_PUB.Modifier_List_Rec_Type;
l_x_MODIFIER_LIST_val_recQP_Modifiers_PUB.Modifier_List_Val_Rec_Type;
l_x_MODIFIERS_tblQP_Modifiers_PUB.Modifiers_Tbl_Type;
l_x_MODIFIERS_val_tblQP_Modifiers_PUB.Modifiers_Val_Tbl_Type;
l_x_QUALIFIERS_tblQP_Qualifier_Rules_PUB.Qualifiers_Tbl_Type;
l_x_QUALIFIERS_val_tblQP_Qualifier_Rules_PUB.Qualifiers_Val_Tbl_Type;
l_x_PRICING_ATTR_tblQP_Modifiers_PUB.Pricing_Attr_Tbl_Type;
l_x_PRICING_ATTR_val_tblQP_Modifiers_PUB.Pricing_Attr_Val_Tbl_Type;
```

```
mll_rec qp_list_lines%ROWTYPE;
pra_rec qp_pricing_attributes%ROWTYPE;

Begin

/* Create a Modifier header of type 'PRO' (Promotion) */

    l_MODIFIER_LIST_rec.currency_code:= 'USD';
    l_MODIFIER_LIST_rec.list_type_code:= 'PRO';
    l_MODIFIER_LIST_rec.start_date_active      := sysdate;
    l_MODIFIER_LIST_rec.end_date_active        := sysdate+10;
    l_MODIFIER_LIST_rec.source_system_code:= 'QP';
    l_MODIFIER_LIST_rec.active_flag:= 'Y';
    l_MODIFIER_LIST_rec.name:= 'latest 1.9 PBH 2001 Promotion';
    l_MODIFIER_LIST_rec.description:= 'latest PBH 2001 Promotion';
    l_MODIFIER_LIST_rec.version_no:= '1.9';
    l_MODIFIER_LIST_rec.pte_code               := 'ORDFUL';
    l_MODIFIER_LIST_rec.operation:= QP_GLOBALS.G_OPR_CREATE;

/* Create a Modifier line to specify Price Break (PBH) - linked to item 45 */

    l_MODIFIERS_tbl(1).list_line_type_code := 'PBH';
    l_MODIFIERS_tbl(1).automatic_flag:= 'Y';
    l_MODIFIERS_tbl(1).modifier_level_code := 'LINE';
    l_MODIFIERS_tbl(1).accrual_flag := 'N';
    l_MODIFIERS_tbl(1).start_date_active := sysdate;
    l_MODIFIERS_tbl(1).end_date_active := sysdate+10;
    l_MODIFIERS_tbl(1).pricing_group_sequence := 1;
    l_MODIFIERS_tbl(1).pricing_phase_id := 3;
    l_MODIFIERS_tbl(1).product_precedence := 12;
    l_MODIFIERS_tbl(1).price_break_type_code := 'POINT';
    l_MODIFIERS_tbl(1).modifier_parent_index := 1;
    l_MODIFIERS_tbl(1).operation := QP_GLOBALS.G_OPR_CREATE;

/* Create a Modifier line to specify 20% discount - linked to item 45 */

    l_MODIFIERS_tbl(2).list_line_type_code := 'DIS';
    l_MODIFIERS_tbl(2).automatic_flag:= 'Y';
    l_MODIFIERS_tbl(2).modifier_level_code := 'LINE';
    l_MODIFIERS_tbl(2).accrual_flag := 'N';
    l_MODIFIERS_tbl(2).start_date_active := sysdate;
    l_MODIFIERS_tbl(2).end_date_active := sysdate+10;
    l_MODIFIERS_tbl(2).operand := 20;
    l_MODIFIERS_tbl(2).arithmetic_operator := '%';
```



```
l_MODIFIERS_tbl(2).pricing_group_sequence := 1;
l_MODIFIERS_tbl(2).pricing_phase_id := 3;
l_MODIFIERS_tbl(2).product_precedence := 12;
l_MODIFIERS_tbl(2).price_break_type_code := 'POINT';
l_MODIFIERS_tbl(2).modifier_parent_index := 1;
l_MODIFIERS_tbl(2).rltd_modifier_grp_no := 10;
l_MODIFIERS_tbl(2).rltd_modifier_grp_type := 'PRICE BREAK';
l_MODIFIERS_tbl(2).operation := QP_GLOBALS.G_OPR_CREATE;

/* Create a Modifier line to specify 25% discount - linked to item 45 */

l_MODIFIERS_tbl(3).list_line_type_code := 'DIS';
l_MODIFIERS_tbl(3).automatic_flag:= 'Y';
l_MODIFIERS_tbl(3).modifier_level_code := 'LINE';
l_MODIFIERS_tbl(3).accrual_flag := 'N';
l_MODIFIERS_tbl(3).start_date_active := sysdate;
l_MODIFIERS_tbl(3).end_date_active := sysdate+10;
l_MODIFIERS_tbl(3).operand := 25;
l_MODIFIERS_tbl(3).arithmetic_operator := '%';
l_MODIFIERS_tbl(3).pricing_group_sequence := 1;
l_MODIFIERS_tbl(3).pricing_phase_id := 3;
l_MODIFIERS_tbl(3).product_precedence := 12;
l_MODIFIERS_tbl(3).price_break_type_code := 'POINT';
l_MODIFIERS_tbl(3).modifier_parent_index := 1;
l_MODIFIERS_tbl(3).rltd_modifier_grp_no := 10;
l_MODIFIERS_tbl(3).rltd_modifier_grp_type := 'PRICE BREAK';
l_MODIFIERS_tbl(3).operation := QP_GLOBALS.G_OPR_CREATE;

/* Create a Pricing Attribute record to specify the 'Buy item 45' condition. It
is linked to the PBH modifier record by specifying modifiers_index=1 */

l_PRICING_ATTR_tbl(1).product_attribute_context:= 'ITEM';
l_PRICING_ATTR_tbl(1).product_attribute:= 'PRICING_ATTRIBUTE1';
l_PRICING_ATTR_tbl(1).product_attr_value:= '45';
l_PRICING_ATTR_tbl(1).pricing_attribute_context:= 'VOLUME';
l_PRICING_ATTR_tbl(1).pricing_attribute:= 'PRICING_ATTRIBUTE10';
l_PRICING_ATTR_tbl(1).comparison_operator_code:= 'BETWEEN';
l_PRICING_ATTR_tbl(1).product_uom_code:= 'Ea';
l_PRICING_ATTR_tbl(1).excluder_flag:= 'N';
l_PRICING_ATTR_tbl(1).MODIFIERS_index:=1;
l_PRICING_ATTR_tbl(1).operation := QP_GLOBALS.G_OPR_CREATE;

/* Create a Pricing Attribute record to specify the 'Buy 1-100 of item 45'
condition. It is linked to the DIS modifier record by specifying modifiers_
index=2 */
```

```
l_PRICING_ATTR_tbl(2).product_attribute_context:= 'ITEM';
l_PRICING_ATTR_tbl(2).product_attribute:= 'PRICING_ATTRIBUTE1';
l_PRICING_ATTR_tbl(2).product_attr_value:= '45';
l_PRICING_ATTR_tbl(2).pricing_attribute_context:= 'VOLUME';
l_PRICING_ATTR_tbl(2).pricing_attribute:= 'PRICING_ATTRIBUTE10';
l_PRICING_ATTR_tbl(2).pricing_attr_value_from:= '1';
l_PRICING_ATTR_tbl(2).pricing_attr_value_to:= '100';
l_PRICING_ATTR_tbl(2).comparison_operator_code:= 'BETWEEN';
l_PRICING_ATTR_tbl(2).product_uom_code:= 'Ea';
l_PRICING_ATTR_tbl(2).excluder_flag:= 'N';
l_PRICING_ATTR_tbl(2).MODIFIERS_index:=2;
l_PRICING_ATTR_tbl(2).operation                := QP_GLOBALS.G_OPR_CREATE;
```

```
/* Create a Pricing Attribute record to specify the 'Buy 101-200 of item 45'
condition. It is linked to the DIS modifier record by specifying modifiers_
index=3 */
```

```
l_PRICING_ATTR_tbl(3).product_attribute_context:= 'ITEM';
l_PRICING_ATTR_tbl(3).product_attribute:= 'PRICING_ATTRIBUTE1';
l_PRICING_ATTR_tbl(3).product_attr_value:= '45';
l_PRICING_ATTR_tbl(3).pricing_attribute_context:= 'VOLUME';
l_PRICING_ATTR_tbl(3).pricing_attribute:= 'PRICING_ATTRIBUTE10';
l_PRICING_ATTR_tbl(3).pricing_attr_value_from:= '101';
l_PRICING_ATTR_tbl(3).pricing_attr_value_to:= '200';
l_PRICING_ATTR_tbl(3).comparison_operator_code:= 'BETWEEN';
l_PRICING_ATTR_tbl(3).product_uom_code:= 'Ea';
l_PRICING_ATTR_tbl(3).excluder_flag:= 'N';
l_PRICING_ATTR_tbl(3).MODIFIERS_index:=3;
l_PRICING_ATTR_tbl(3).operation                := QP_GLOBALS.G_OPR_CREATE;
```

QP_Modifiers_PUB.Process_Modifiers

```
( p_api_version_number=> 1.0
, p_init_msg_list=> FND_API.G_FALSE
, p_return_values=> FND_API.G_FALSE
, p_commit=> FND_API.G_FALSE
, x_return_status=> l_return_status
, x_msg_count=> x_msg_count
, x_msg_data=> x_msg_data
,p_MODIFIER_LIST_rec=> l_MODIFIER_LIST_rec
,p_MODIFIERS_tbl=> l_MODIFIERS_tbl
,p_PRICING_ATTR_tbl=> l_PRICING_ATTR_tbl
,x_MODIFIER_LIST_rec=> l_x_MODIFIER_LIST_rec
,x_MODIFIER_LIST_val_rec=> l_x_MODIFIER_LIST_val_rec
```

```
,x_MODIFIERS_tbl=> l_x_MODIFIERS_tbl
,x_MODIFIERS_val_tbl=> l_x_MODIFIERS_val_tbl
,x_QUALIFIERS_tbl=> l_x_QUALIFIERS_tbl
,x_QUALIFIERS_val_tbl=> l_x_QUALIFIERS_val_tbl
,x_PRICING_ATTR_tbl=> l_x_PRICING_ATTR_tbl
,x_PRICING_ATTR_val_tbl => l_x_PRICING_ATTR_val_tbl
);

IF l_return_status <> FND_API.G_RET_STS_SUCCESS THEN

    RAISE FND_API.G_EXC_UNEXPECTED_ERROR;

END IF;

EXCEPTION

    WHEN FND_API.G_EXC_ERROR THEN

        l_return_status := FND_API.G_RET_STS_ERROR;

        -- Get message count and data

        --dbms_output.put_line('err msg 1 is : ' || x_msg_data);

    WHEN FND_API.G_EXC_UNEXPECTED_ERROR THEN

        l_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;

        --dbms_output.put_line(' msg count 2 is : ' || x_msg_count);

    for k in 1 .. x_msg_count loop

        x_msg_data := oe_msg_pub.get( p_msg_index => k,

        p_encoded => 'F'

        );

        --Get message count and data
        --dbms_output.put_line('err msg ' || k || 'is: ' || x_msg_data);

    end loop;
```

```
        WHEN OTHERS THEN

            l_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;

            --dbms_output.put_line(' msg count 5 is : ' || x_msg_count);

        for k in 1 .. x_msg_count loop

            x_msg_data := oe_msg_pub.get( p_msg_index => k,

                p_encoded => 'F'

            );
            --Get message count and data
            --dbms_output.put_line('err msg ' || k || 'is: ' || x_msg_data);

        end loop;

    END;
/
```

Business Object for Pricing Formulas Application Program Interface

This section explains how to use the Business Object for Pricing Formulas API and how it functions in Oracle Advanced Pricing. The Business Object for Pricing Formulas package consists of entities to support the Formulas window.

Functional Overview

The Formulas window is based on the following APIs. However, the modal Formula Factors window is based on the Modifiers public API QP_Modifiers_PUB.Process_Modifiers.

The package QP_Price_Formula_PUB contains the following public record type and table of records definitions:

- Formula_Rec_Type: A record type corresponding to the columns in the Formula Headers view (QP_PRICE_FORMULAS_VL).
- Formula_Tbl_Type
- Formula_Val_Rec_Type: A record type used to store values corresponding to IDs in the formula record.
- Formula_Val_Tbl_Type
- Formula_Lines_Rec_Type: A record type corresponding to the columns in the Formula Lines table (QP_PRICE_FORMULA_LINES).
- Formula_Lines_Tbl_Type
- Formula_Lines_Val_Rec_Type: A record type used to store values corresponding to IDs in the Formula Lines record.
- Formula_Lines_Val_Tbl_Type
- QP_Price_formula_PUB.Process_Price_Formula: Performs the insert, update, and delete of price formula header and price formula lines.
- QP_Price_formula_PUB.Lock_Price_Formula: Locks price formula header and price formula lines records prior to updates.
- QP_Price_formula_PUB.Get_Price_Formula: Retrieves the price formula header and lines for a given formula.

Setting Up and Parameter Descriptions

The following chart describes all parameters and the inbound and outbound parameters. Additional information on these parameters follows.

QP_PRICE_FORMULA.PROCESS_PRICE_FORMULA

The following table shows the parameters for this structure.

Table 5–57 QP_PRICE_FORMULA.PROCESS_PRICE_FORMULA Parameters

Parameter	Usage	Type	Req	Drv
p_api_version_number	In	Number	No	No
p_init_msg_list	In	Varchar2	No	No
p_return_values	In	Varchar2	No	No
p_commit	In	Varchar2	No	No
x_return_status	Out	Varchar2	No	No
x_msg_count	Out	Number	No	No
x_msg_data	Out	Varchar2	No	No
p_FORMULA_rec	In	Formula_Rec_Type	No	No
p_FORMULA_val_rec	In	Formula_Val_Rec_Type	No	No
p_FORMULA_LINES_tbl	In	Formula_Lines_Tbl_Type	No	No
p_FORMULA_LINES_val_tbl	In	Formula_Lines_Val_Tbl_Type	No	No
x_FORMULA_rec	Out	Formula_Rec_Type	No	No
x_FORMULA_val_rec	Out	Formula_Val_Rec_Type	No	No
x_FORMULA_LINES_tbl	Out	Formula_Lines_Tbl_Type	No	No
x_FORMULA_LINES_val_tbl	Out	Formula_Lines_Val_Tbl_Type	No	No

p_init_msg_list

Default Value: FND_API.G_FALSE

p_return_values

Default Value: FND_API.G_FALSE

p_commit

Default Value: FND_API.G_FALSE

p_FORMULA_rec

Default Value: G_MISS_FORMULA_REC

p_FORMULA_val_rec

Default Value: G_MISS_FORMULA_VAL_REC

p_FORMULA_LINES_tbl

Default Value: G_MISS_FORMULA_LINES_TBL

p_FORMULA_LINES_val_tbl

Default Value: G_MISS_FORMULA_LINES_VAL_TBL

FORMULA_REC_TYPE The following table shows the parameters for this structure:**Table 5–58 FORMULA_REC_TYPE Parameters**

Parameter	Usage	Type	Req	Drv
attribute1	Null	Varchar2	No	No
attribute2	Null	Varchar2	No	No
attribute3	Null	Varchar2	No	No
attribute4	Null	Varchar2	No	No
attribute5	Null	Varchar2	No	No
attribute6	Null	Varchar2	No	No
attribute7	Null	Varchar2	No	No
attribute8	Null	Varchar2	No	No
attribute9	Null	Varchar2	No	No
attribute10	Null	Varchar2	No	No
attribute11	Null	Varchar2	No	No
attribute12	Null	Varchar2	No	No
attribute13	Null	Varchar2	No	No
attribute14	Null	Varchar2	No	No

Table 5–58 FORMULA_REC_TYPE Parameters

Parameter	Usage	Type	Req	Drv
attribute15	Null	Varchar2	No	No
context	Null	Varchar2	No	No
created_by	Null	Number	No	No
creation_date	Null	Date	No	No
description	Null	Varchar2	No	No
end_date_active	Null	Date	No	No
formula	Null	Varchar2	Yes	No
last_updated_by	Null	Number	No	No
last_update_date	Null	Date	No	No
last_update_login	Null	Number	No	No
name	Null	Varchar2	Yes	No
price_formula_id	Null	Number	No	No
start_date_active	Null	Date	No	No
return_status	Null	Varchar2	No	No
db_flag	Null	Varchar2	No	No
operation	Null	Varchar2	Yes	No

attribute1-15

Default Value: FND_API.G_MISS_CHAR

context

Default Value: FND_API.G_MISS_CHAR

created_by

Default Value: FND_API.G_MISS_NUM

creation_date

Default Value: FND_API.G_MISS_DATE

description

Default Value: FND_API.G_MISS_CHAR

end_date_active

Default Value: FND_API.G_MISS_DATE

formula

Default Value: FND_API.G_MISS_CHAR

last_updated_by

Default Value: FND_API.G_MISS_NUM

last_update_date

Default Value: FND_API.G_MISS_DATE

last_update_login

Default Value: FND_API.G_MISS_NUM

name

Default Value: FND_API.G_MISS_CHAR

price_formula_id

Default Value: Comes from the sequence QP_PRICE_FORMULAS_B_S

start_date_active

Default Value: FND_API.G_MISS_DATE

step_number

Default Value: FND_API.G_MISS_NUM

retun_status

Default Value: FND_API.G_MISS_CHAR

db_flag

Default Value: FND_API.G_MISS_CHAR

operation

Default Value: FND_API.G_MISS_CHAR

FORMULA_TBL_TYPE

The following table shows the parameters for this structure.

Table 5–59 FORMULA_TBL_TYPE Parameters

Parameter	Usage	Type	Req	Drv
Formula_Rec_Type	Null	Record	No	No

FORMULA_VAL_REC_TYPE

The following table shows the parameters for this structure.

Table 5–60 FORMULA_VAL_REC_TYPE Parameters

Parameter	Usage	Type	Req	Drv
price_formula	Null	Number	No	No

price formula

Default Value: FND_API.G_MISS_CHAR

FORMULA_VAL_TBL_TYPE

The following table shows the parameters for this structure.

Table 5–61 FORMULA_LINES_REC_TYPE Parameters

Parameter	Usage	Type	Req	Drv
Formula_Val_Rec_Type	Null	Record	No	No

FORMULA_LINES_REC_TYPE

The following table shows the parameters for this structure.

Table 5–62 FORMULA_LINES_REC_TYPE Parameters

Parameter	Usage	Type	Req	Drv
attribute1	Null	Varchar2	No	No
attribute2	Null	Varchar2	No	No

Table 5–62 FORMULA_LINES_REC_TYPE Parameters

Parameter	Usage	Type	Req	Drv
attribute3	Null	Varchar2	No	No
attribute4	Null	Varchar2	No	No
attribute5	Null	Varchar2	No	No
attribute6	Null	Varchar2	No	No
attribute7	Null	Varchar2	No	No
attribute8	Null	Varchar2	No	No
attribute9	Null	Varchar2	No	No
attribute10	Null	Varchar2	No	No
attribute11	Null	Varchar2	No	No
attribute12	Null	Varchar2	No	No
attribute13	Null	Varchar2	No	No
attribute14	Null	Varchar2	No	No
attribute15	Null	Varchar2	No	No
context	Null	Varchar2	No	No
created_by	Null	Number	No	No
creation_date	Null	Date	No	No
end_date_active	Null	Date	No	No
last_updated_by	Null	Number	No	No
last_update_date	Null	Date	No	No
last_update_login	Null	Number	No	No
numeric_constant	Null	Number	Yes ¹	No
price_formula_id	Null	Number	No	No
price_formula_line_id	Null	Number	No	No
formula_line_type_code	Null	Varchar2	Yes	No
price_list_line_id	Null	Number	Yes ²	No
price_modifier_list_id	Null	Number	Yes ³	No
pricing_attribute	Null	Varchar2	Yes ⁴	No

Table 5–62 FORMULA_LINES_REC_TYPE Parameters

Parameter	Usage	Type	Req	Drv
pricing_attribute_context	Null	Varchar2	Yes ⁵	No
start_date_active	Null	Date	No	No
step_number	Null	number	Yes	No
return_status	Null	Varchar2	No	No
reqd_flag	Null	Varchar2	No	No
db_flag	Null	Varchar2	No	No
operation	Null	Varchar2	Yes	No

The following table describes notations listed in the preceding table:

Table 5–63 Notations

Note	Description
1	Conditionally Required when formula_line_type_code is 'NUM'
2	Conditionally Required when formula_line_type_code is 'PLL'
3	Conditionally Required when formula_line_type_code is 'ML'
4	Conditionally Required when formula_line_type_code is 'PRA'
5	Conditionally Required when formula_line_type_code is 'PRA'

attribute1-15

Default Value: FND_API.G_MISS_CHAR

context

Default Value: FND_API.G_MISS_CHAR

created_by

Default Value: FND_API.G_MISS_NUM

creation_date

Default Value: FND_API.G_MISS_DATE

end_date_active

Default Value: FND_API.G_MISS_DATE

last_updated_by

Default Value: FND_API.G_MISS_NUM

last_update_date

Default Value: FND_API.G_MISS_DATE

last_update_login

Default Value: FND_API.G_MISS_NUM

numeric_constant

Default Value: FND_API.G_MISS_NUM

price_formula_id

Default Value: FND_API.G_MISS_NUM

price_formula_line_id

Default Value: Comes for the sequence QP_PRICE_FORMULA_LINES_S

formula_line_type_code

Default Value: FND_API.G_MISS_CHAR

price_list_line_id

Default Value: FND_API.G_MISS_NUM

price_modifier_list_id

Default Value: FND_API.G_MISS_NUM

pricing_attribute

Default Value: FND_API.G_MISS_CHAR

pricing_attribute_context

Default Value: FND_API.G_MISS_CHAR

start_date_active

Default Value: FND_API.G_MISS_DATE

step_number

Default Value: FND_API.G_MISS_NUM

reqd_flag

Default Value: FND_API.G_MISS_CHAR

retun_status

Default Value: FND_API.G_MISS_CHAR

db_flag

Default Value: FND_API.G_MISS_CHAR

operation

Default Value: FND_API.G_MISS_CHAR

FORMULA_LINES_TBL_TYPE

The following table shows the parameters for this structure.

Table 5–64 FORMULA_LINES_TBL_TYPE Parameters

Parameter	Usage	Type	Req	Drv
Formula_Lines_Rec_Type	Null	Record	No	No

FORMULA_LINES_VAL_REC_TYPE

The following table shows the parameters for this structure.

Table 5–65 FORMULA_LINES_VAL_REC_TYPE Parameters

Parameter	Usage	Type	Req	Drv
price_formula	Null	Varchar2	No	No
price_formula_line	Null	Varchar2	No	No
price_formula_line_type	Null	Varchar2	No	No
price_list_line	Null	Varchar2	No	No

Table 5–65 FORMULA_LINES_VAL_REC_TYPE Parameters

Parameter	Usage	Type	Req	Drv
price_modifier_list	Null	Varchar2	No	No

price_formula

Default Value: FND_API.G_MISS_CHAR

price_formula_line

Default Value: FND_API.G_MISS_CHAR

price_formula_line_type

Default Value: FND_API.G_MISS_CHAR

price_list_line

Default Value: FND_API.G_MISS_CHAR

price_modifier_list

Default Value: FND_API.G_MISS_CHAR

FORMULA_LINES_VAL_TBL_TYPE

The following table shows the parameters for this structure.

Table 5–66 FORMULA_LINES_VAL_TBL_TYPE Parameters

Parameter	Usage	Type	Req	Drv
Formula_Lines_Val_Rec_Type	Null	Record	No	No

Validation of Business Object for Pricing Formulas API

Standard Validation

Oracle Advanced Pricing validates all required columns in the Business Object for Pricing Formulas API. For more information, see: *Oracle Pricing Technical Reference Manual*.

Other Validation

None

Error Handling

If any validation fails, the API will return error status to the calling module. The Business Object for Pricing Formulas API processes the rows and reports the values in the following table for every record.

Table 5–67 Error Handling

Condition	PROCESS_STATUS	ERROR_MESSAGE
Success	5	null
Failure	4	actual error message

Example of Pricing Formulas API

The following code can also be found in file QPPFXMP1.sql in \$QP_TOP/patch/115/sql directory:

```
--set serveroutput on
/*****
This sample script inserts a Pricing Formula with seven different pricing
formula lines to demonstrate the six formula line types supported in Advanced
Pricing formulas:

* Price List Line (PLL)
* Function (FUNC)
* List Price(LP)
* Numeric Constant (NUM)
* Pricing Attribute (PRA)
* Factor List (ML)
* Modifier Value (MV)
*****/
```

Oracle Basic Pricing supports three formula line types:

* Pricing Attribute (PRA)
 * Numeric Constant (NUM)
 * Factor List (ML)

1. The following code can also be found in file QPPFXMP1.sql in \$QP_TOP/patch/115/sql directory

```
--set serveroutput on
/*****
```

A pricing formula header record and seven price formula lines are created. For the formula line of type *Factor List*, the list_header_id of an existing Factor List is used in this sample script.

This script must be modified by the user so that the following column is populated with a valid list_header_id of an existing Factor List:

```
gpr_formula_lines_tbl(K).price_modifier_list_id
```

and the following column is populated with a valid list_line_id of an existing Price List Line from the instance where this script is run:

```
gpr_formula_lines_tbl(K).price_list_line_id
```

For more information, see: *Oracle Advanced Pricing User's Guide, Seed Data*.

```
*****/
declare
```

```
gpr_return_status varchar2(1) := NULL;
gpr_msg_count number := 0;
gpr_msg_data varchar2(2000);
gpr_formula_rec      QP_PRICE_FORMULA_PUB.Formula_Rec_Type;
gpr_formula_val_rec  QP_PRICE_FORMULA_PUB.Formula_Val_Rec_Type;
gpr_formula_lines_tbl QP_PRICE_FORMULA_PUB.Formula_Lines_Tbl_Type;
gpr_formula_lines_val_tbl QP_PRICE_FORMULA_PUB.Formula_Lines_Val_Tbl_Type;
ppr_formula_rec      QP_PRICE_FORMULA_PUB.Formula_Rec_Type;
ppr_formula_val_rec  QP_PRICE_FORMULA_PUB.Formula_Val_Rec_Type;
ppr_formula_lines_tbl QP_PRICE_FORMULA_PUB.Formula_Lines_Tbl_Type;
ppr_formula_lines_val_tbl QP_PRICE_FORMULA_PUB.Formula_Lines_Val_Tbl_Type;
```

```
K number := 1;
```

```
begin
```

```
/* Set the price_formula_id to g_miss_num to
```

```

Create the Price Formula Record(Header)*/
    gpr_formula_rec.price_formula_id := FND_API.G_MISS_NUM;
    gpr_formula_rec.name := 'Sample1-PF 1025-1';
    gpr_formula_rec.description := 'Sample Pricing Formula';
    gpr_formula_rec.formula := 'SQRT(1)*2-NVL(3,4)/5+6';

--Any valid Mathematical Expression including built-in database functions.
--Every operand must correspond to a step_number in a price formula line.

    gpr_formula_rec.operation := QP_GLOBALS.G_OPR_CREATE;

    /* Create price formula line 1 of type 'List Price'(LP) */
    K := 1;

    gpr_formula_lines_tbl(K).price_formula_id := FND_API.G_MISS_NUM;
    gpr_formula_lines_tbl(K).price_formula_line_id
:= FND_API.G_MISS_NUM;

    gpr_formula_lines_tbl(K).formula_line_type_code := 'LP'
    gpr_formula_lines_tbl(K).step_number := 1;
    gpr_formula_lines_tbl(K).operation := QP_GLOBALS.G_OPR_CREATE;

    /* Create price formula line 2 of type 'Price List Line'(PLL) */
    K := K + 1;
    gpr_formula_lines_tbl(K).price_formula_id := FND_API.G_MISS_NUM;
    gpr_formula_lines_tbl(K).price_formula_line_id

:= FND_API.G_MISS_NUM;

    gpr_formula_lines_tbl(K).formula_line_type_code := 'PLL';
    gpr_formula_lines_tbl(K).step_number := 2;
    gpr_formula_lines_tbl(K).price_list_line_id := 293195;

-- Corresponds to the list_line_id of the item 'dw01' on the PriceList
-- 'Testing 1023'.
    gpr_formula_lines_tbl(K).operation := QP_GLOBALS.G_OPR_CREATE;

    /* Create price formula line 3 of type 'Pricing Attribute'(PRA) */
    K := K + 1;
    gpr_formula_lines_tbl(K).price_formula_id := FND_API.G_MISS_NUM;
    gpr_formula_lines_tbl(K).price_formula_line_id

:= FND_API.G_MISS_NUM;

    gpr_formula_lines_tbl(K).formula_line_type_code := 'PRA';

```

```
gpr_formula_lines_tbl(K).step_number := 3;
gpr_formula_lines_tbl(K).pricing_attribute_context

:= 'PRICING ATTRIBUTE';

gpr_formula_lines_tbl(K).pricing_attribute := 'PRICING_ATTRIBUTE12';

-- Corresponds to the Pricing Attribute 'Insurance Cost'

gpr_formula_lines_tbl(K).operation := QP_GLOBALS.G_OPR_CREATE;
/* Create price formula line 4 of type 'Numeric Constant'(NUM) */
K := K + 1;

gpr_formula_lines_tbl(K).price_formula_id := FND_API.G_MISS_NUM;
gpr_formula_lines_tbl(K).price_formula_line_id

:= FND_API.G_MISS_NUM;

gpr_formula_lines_tbl(K).formula_line_type_code := 'NUM';
gpr_formula_lines_tbl(K).step_number := 4;
gpr_formula_lines_tbl(K).numeric_constant := 1000;
gpr_formula_lines_tbl(K).operation := QP_GLOBALS.G_OPR_CREATE;

/* Create price formula line 5 of type 'Function'(FUNC) */

-- User must customize the QP_CUSTOM.Get_Custom_Price function
-- to return a numeric value and also set the profile option
-- 'QP: Get Custom Price Customized' to 'Yes' at the Site Level to
-- successfully use this formula line type (FUNC) in their formulas.

K := K + 1;
gpr_formula_lines_tbl(K).price_formula_id := FND_API.G_MISS_NUM;
gpr_formula_lines_tbl(K).price_formula_line_id

:= FND_API.G_MISS_NUM;

gpr_formula_lines_tbl(K).formula_line_type_code := 'FUNC';
gpr_formula_lines_tbl(K).step_number := 5;
gpr_formula_lines_tbl(K).operation := QP_GLOBALS.G_OPR_CREATE;

/* Create price formula line 6 of type 'Factor List'(ML) */

K := K + 1;

gpr_formula_lines_tbl(K).price_formula_id := FND_API.G_MISS_NUM;
```

```

        gpr_formula_lines_tbl(K).price_formula_line_id

:= FND_API.G_MISS_NUM;

        gpr_formula_lines_tbl(K).formula_line_type_code := 'ML';
        gpr_formula_lines_tbl(K).step_number := 6;
        gpr_formula_lines_tbl(K).price_modifier_list_id := 50174;

-- Corresponds to the list_header_id of an existing Factor List
-- 'ABC'

        gpr_formula_lines_tbl(K).operation := QP_GLOBALS.G_OPR_CREATE;

/* Create price formula line 7 of type 'Modifier Value'(MV) */
K := K + 1;

        gpr_formula_lines_tbl(K).price_formula_id := FND_API.G_MISS_NUM;
        gpr_formula_lines_tbl(K).price_formula_line_id:= FND_API.G_MISS_NUM;
        gpr_formula_lines_tbl(K).formula_line_type_code := 'MV'
        gpr_formula_lines_tbl(K).step_number := 7;
        gpr_formula_lines_tbl(K).operation := QP_GLOBALS.G_OPR_CREATE;
--dbms_output.put_line('before process price formula ');

```

QP_PRICE_FORMULA_PUB.Process_Price_Formula

```

(   p_api_version_number => 1
,   p_init_msg_list    => FND_API.G_FALSE
,   p_return_values    => FND_API.G_FALSE
,   p_commit           => FND_API.G_FALSE
,   x_return_status    => gpr_return_status
,   x_msg_count        => gpr_msg_count
,   x_msg_data         => gpr_msg_data
,   p_FORMULA_rec      => gpr_formula_rec
,   p_FORMULA_LINES_tbl => gpr_formula_lines_tbl
,   x_FORMULA_rec      => ppr_formula_rec
,   x_FORMULA_val_rec  => ppr_formula_val_rec
,   x_FORMULA_LINES_tbl => ppr_formula_lines_tbl
,   x_FORMULA_LINES_val_tbl => ppr_formula_lines_val_tbl
);

        IF gpr_return_status <> FND_API.G_RET_STS_SUCCESS THEN

        RAISE FND_API.G_EXC_UNEXPECTED_ERROR;

END IF;

```

```
--dbms_output.put_line('after process price formula ');

EXCEPTION

    WHEN FND_API.G_EXC_ERROR THEN

        gpr_return_status := FND_API.G_RET_STS_ERROR;

        --Get message count and data

        --dbms_output.put_line('err msg 1 is : ' || gpr_msg_data);

Rollback;

    WHEN FND_API.G_EXC_UNEXPECTED_ERROR THEN

        gpr_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;

        --dbms_output.put_line(' msg count 2 is : ' || gpr_msg_count);

        for k in 1 .. gpr_msg_count loop

            gpr_msg_data := oe_msg_pub.get( p_msg_index => k,

p_encoded => 'F'

            );
        /*

            oe_msg_pub.Count_And_Get

        (   p_countgpr_msg_count

        ,   p_data => gpr_msg_data

        );

        */

        --Get message count and data
        --dbms_output.put_line('err msg ' || k || 'is: ' || gpr_msg_data);
        null;

    end loop;
```

```

Rollback;

WHEN OTHERS THEN

    gpr_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;

    --Get message count and data
    --dbms_output.put_line('err msg 3 is : ' || gpr_msg_data);

    Rollback;

end;
/
commit;
exit;

```

2. The following code can also be found in file QPPFXMP2.sql in \$QP_TOP/patch/115/sql directory

```

--set serveroutput on
/*****
Sample script which inserts a Pricing Formula and one price formula line of type
Factor List (ML) A new Factor List is created in this sample script.
Factor Lists can be created/updated only in the Pricing Formulas window in the
Factors window if using the application. A factor list originally created in one
formula can be used in other formulas as well. Any modification to a factor
list's factors is reflected in all formulas using the factor list. However,
while using APIs to create Factor Lists, we use the Modifier API. A factor list,
its factors and pricing attributes use the same tables as a Modifier List,
Modifiers and Pricing Attributes.
A factor list is a modifier list with a list_type_code of PML and a factor is a
Modifier with a list_line_type_code of PMR:
A pricing formula header record and one price formula line of type factor list
ML are created. Corresponding to the formula line of type factor list, 1 Factor
List record (Modifier List) and one factor record (Modifier) are created. In
this script, for the factor record, a base pricing attribute record and an
associated pricing attribute record are created. The Modifiers API is used to
create the factor list, factor and their pricing attributes.
For more information on flexfields and seed data, see Oracle Advanced Pricing
User's Guide.
*****/

declare

```

```
gpr_return_status varchar2(1) := NULL;
gpr_msg_count number := 0;
gpr_msg_data varchar2(2000);
gpr_formula_rec      QP_PRICE_FORMULA_PUB.Formula_Rec_Type;
gpr_formula_lines_tbl QP_PRICE_FORMULA_PUB.Formula_Lines_Tbl_Type;
ppr_formula_rec      QP_PRICE_FORMULA_PUB.Formula_Rec_Type;
ppr_formula_val_rec  QP_PRICE_FORMULA_PUB.Formula_Val_Rec_Type;
ppr_formula_lines_tbl QP_PRICE_FORMULA_PUB.Formula_Lines_Tbl_Type;
ppr_formula_lines_val_tbl QP_PRICE_FORMULA_PUB.Formula_Lines_Val_Tbl_Type;

gpr_modifier_list_rec QP_MODIFIERS_PUB.Modifier_List_Rec_Type;
gpr_modifiers_tbl     QP_MODIFIERS_PUB.Modifiers_Tbl_Type;
gpr_pricing_attr_tbl  QP_MODIFIERS_PUB.Pricing_Attr_Tbl_Type;

ppr_modifier_list_rec QP_MODIFIERS_PUB.Modifier_List_Rec_Type;
ppr_modifier_list_val_rec QP_MODIFIERS_PUB.Modifier_List_Val_Rec_Type;
ppr_modifiers_tbl     QP_MODIFIERS_PUB.Modifiers_Tbl_Type;
ppr_modifiers_val_tbl  QP_MODIFIERS_PUB.Modifiers_Val_Tbl_Type;
ppr_pricing_attr_tbl   QP_MODIFIERS_PUB.Pricing_Attr_Tbl_Type;
ppr_pricing_attr_val_tbl QP_MODIFIERS_PUB.Pricing_Attr_Val_Tbl_Type;
ppr_qualifiers_tbl     QP_QUALIFIER_RULES_PUB.Qualifiers_Tbl_Type;
ppr_qualifiers_val_tbl  QP_QUALIFIER_RULES_PUB.Qualifiers_Val_Tbl_Type;

K number := 1;
J number := 1;
I number := 1;

begin

    /* Set the price_formula_id to g_miss_num to

Create the Price Formula Record(Header)*/
gpr_formula_rec.price_formula_id := FND_API.G_MISS_NUM;
gpr_formula_rec.name := 'Sample2-PF 1025-6';
gpr_formula_rec.description := 'Sample Pricing Formula';
gpr_formula_rec.formula := '1';

    --Any valid Mathematical Expression including built-in database functions.
    --Every operand must correspond to a step_number in a price formula line.

gpr_formula_rec.operation := QP_GLOBALS.G_OPR_CREATE;

    /* Prior to creating the formula line of type 'Factor List', we first create a
Factor List, Factors and pricing attributes using the Modifiers API.
```

This is because the list_header_id of the Factor_List(Modifier_List) must be populated in the price_modifier_list_id column of the Formula Line Record which is a mandatory column when formula_line_type_code is 'ML'. */

```

/* Create Factor List (Modifier List) record */

gpr_modifier_list_rec.list_header_id := FND_API.G_MISS_NUM;
gpr_modifier_list_rec.name := 'SAMPLE FACTOR LIST 6';
gpr_modifier_list_rec.currency_code := 'USD';
gpr_modifier_list_rec.list_type_code := 'PML';

--For Factor Lists the Modifier List Type is 'PML'.

gpr_modifier_list_rec.operation := QP_GLOBALS.G_OPR_CREATE;

/* Create Factor (Modifier) record 1 */
J := 1;

gpr_modifiers_tbl(J).list_header_id := FND_API.G_MISS_NUM;
gpr_modifiers_tbl(J).list_line_id := FND_API.G_MISS_NUM;
gpr_modifiers_tbl(J).list_line_type_code := 'PMR';
--For Factors the Modifier Type is 'PMR'.
gpr_modifiers_tbl(J).operand := 0.8;
--Corresponds to the Adjustment Factor
gpr_modifiers_tbl(J).arithmetic_operator := 'UNIT_PRICE';
gpr_modifiers_tbl(J).modifier_level_code := 'NULL';
gpr_modifiers_tbl(J).operation := QP_GLOBALS.G_OPR_CREATE;

--Any number of Pricing Attributes may be created for a Factor. But
--only the first pricing attribute record is considered as the Base
--Pricing Attribute and all subsequent Pricing Attributes for the
--same Factor will be considered Associated Pricing Attributes. All
--pricing attributes for a factor are treated as 'AND' conditions.
--Factors are treated as 'OR' conditions. User does not have to make
--any special changes or considerations as far as setup goes.

/* Create Pricing Attribute 1 (Base Pricing Attribute) for Factor 1.*/

I := 1;

gpr_pricing_attr_tbl(I).list_line_id := FND_API.G_MISS_NUM;
gpr_pricing_attr_tbl(I).pricing_attribute_id := FND_API.G_MISS_NUM;
gpr_pricing_attr_tbl(I).modifiers_index := 1;
--Corresponds to the Factor Number. In this case it is 1.
gpr_pricing_attr_tbl(I).pricing_attribute_context := 'PRICING ATTRIBUTE';

```



```

gpr_pricing_attr_tbl(I).pricing_attribute := 'PRICING ATTRIBUTE12';
--Corresponds to the Pricing Attribute 'Insurance Cost'
gpr_pricing_attr_tbl(I).pricing_attr_value_from := '100';
gpr_pricing_attr_tbl(I).pricing_attr_value_to := '120';
gpr_pricing_attr_tbl(I).comparison_operator_code := 'BETWEEN' ;
gpr_pricing_attr_tbl(I).operation := QP_GLOBALS.G_OPR_CREATE;

      /* Create Pricing Attribute 2(Associated Pricing Attribute)for Factor 1.*/
      I := I + 1;

gpr_pricing_attr_tbl(I).list_line_id := FND_API.G_MISS_NUM;
gpr_pricing_attr_tbl(I).pricing_attribute_id := FND_API.G_MISS_NUM;
gpr_pricing_attr_tbl(I).modifiers_index := 1;
--Corresponds to the Factor Number. In this case it is 1.
gpr_pricing_attr_tbl(I).pricing_attribute_context := 'PRICING ATTRIBUTE';
gpr_pricing_attr_tbl(I).pricing_attribute := 'PRICING ATTRIBUTE16';
--Corresponds to the Pricing Attribute 'Freight Cost'
gpr_pricing_attr_tbl(I).pricing_attr_value_from := '11';
gpr_pricing_attr_tbl(I).comparison_operator_code := '=' ;
gpr_pricing_attr_tbl(I).operation := QP_GLOBALS.G_OPR_CREATE;

```

QP_MODIFIERS_PUB.Process_Modifiers

```

(   p_api_version_number      => 1
,   p_init_msg_list=> FND_API.G_FALSE
,   p_return_values=> FND_API.G_FALSE
,   p_commit=> FND_API.G_FALSE
,   x_return_status=> gpr_return_status
,   x_msg_count=> gpr_msg_count
,   x_msg_data=> gpr_msg_data
,   p_modifier_list_rec=> gpr_modifier_list_rec
,   p_modifiers_tbl=> gpr_modifiers_tbl
,   p_pricing_attr_tbl=> gpr_pricing_attr_tbl
,   x_MODIFIER_LIST_rec=> ppr_modifier_list_rec
,   x_MODIFIER_LIST_val_rec=> ppr_modifier_list_val_rec
,   x_MODIFIERS_tbl=> ppr_modifiers_tbl
,   x_MODIFIERS_val_tbl=> ppr_modifiers_val_tbl
,   x_QUALIFIERS_tbl=> ppr_qualifiers_tbl
,   x_QUALIFIERS_val_tbl=> ppr_qualifiers_val_tbl
,   x_PRICING_ATTR_tbl=> ppr_pricing_attr_tbl
,   x_PRICING_ATTR_val_tbl=> ppr_pricing_attr_val_tbl
);

IF gpr_return_status <> FND_API.G_RET_STS_SUCCESS THEN

```

```

        RAISE FND_API.G_EXC_UNEXPECTED_ERROR;

    END IF;

    /* Create price formula line 1 of type 'Factor List'(ML) */

    K := 1;

    gpr_formula_lines_tbl(K).price_formula_id := FND_API.G_MISS_NUM;
    gpr_formula_lines_tbl(K).price_formula_line_id
    gpr_formula_lines_tbl(K).formula_line_type_code := 'ML';
    gpr_formula_lines_tbl(K).step_number := 1;
    gpr_formula_lines_tbl(K).price_modifier_list_id
:= ppr_modifier_list_rec.list_header_id;
    --Corresponds to the list_header_id of the new Factor List
    --created above.
    gpr_formula_lines_tbl(K).operation := QP_GLOBALS.G_OPR_CREATE;

    QP_PRICE_FORMULA_PUB.Process_Price_Formula

(
    p_api_version_number=> 1
,
    p_init_msg_list=> FND_API.G_FALSE
,
    p_return_values=> FND_API.G_FALSE
,
    p_commit=> FND_API.G_FALSE
,
    x_return_status=> gpr_return_status
,
    x_msg_count=> gpr_msg_count
,
    x_msg_data=> gpr_msg_data
,
    p_FORMULA_rec=> gpr_formula_rec
,
    p_FORMULA_LINES_tbl=> gpr_formula_lines_tbl
,
    x_FORMULA_rec=> ppr_formula_rec
,
    x_FORMULA_val_rec=> ppr_formula_val_rec
,
    x_FORMULA_LINES_tbl=> ppr_formula_lines_tbl
,
    x_FORMULA_LINES_val_tbl=> ppr_formula_lines_val_tbl
);

    IF gpr_return_status <> FND_API.G_RET_STS_SUCCESS THEN

        RAISE FND_API.G_EXC_UNEXPECTED_ERROR;

    END IF;

EXCEPTION

    WHEN FND_API.G_EXC_ERROR THEN

```

```
gpr_return_status := FND_API.G_RET_STS_ERROR;
--Get message count and data
--dbms_output.put_line('err msg 1 is : ' || gpr_msg_data);

Rollback;

    WHEN FND_API.G_EXC_UNEXPECTED_ERROR THEN

gpr_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;
--dbms_output.put_line(' msg count 2 is : ' || gpr_msg_count);

        for k in 1 .. gpr_msg_count loop
            gpr_msg_data := oe_msg_pub.get( p_msg_index => k,
p_encoded => 'F'
);
            /*

oe_msg_pub.Count_And_Get

(   p_count=> gpr_msg_count
,   p_data=> gpr_msg_data
);

*/

            --Get message count and data
            --dbms_output.put_line('err msg ' || k || 'is: ' || gpr_msg_data);

null;
end loop;

Rollback;

    WHEN OTHERS THEN

        gpr_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;

--Get message count and data
--dbms_output.put_line('err msg 3 is : ' || gpr_msg_data);

Rollback;
end;
/
commit;
exit;
```

See

Oracle Pricing Technical Reference Manual

Business Object for Pricing Limits Application Program Interface

This section explains how to use the Business Object for Pricing Limits API and how it functions in Oracle Advanced Pricing. The Business Object for Pricing Limits package consists of entities to support the Limits window.

Functional Overview

The Limits window is based on the following APIs:

The package QP_Limits_PUB contains the following public record type and table of records definitions:

- **Limits_Rec_Type**: A record type corresponding to the columns in the Pricing Limits view (QP_LIMITS).
- **Limits_Tbl_Type**
- **Limits_Val_Rec_Type**: A record type used to store values corresponding to IDs in the Limits record.
- **Limits_Val_Tbl_Type**
- **Limit_Attrs_Rec_Type**: A record type corresponding to the columns in the Limit Attributes. View (QP_LIMIT_ATTRIBUTES)
- **Limit_Attrs_Tbl_Type**
- **Limit_Attrs_Val_Rec_Type**: A record type used to store values corresponding to IDs in the Limits Attributes record.
- **Limit_Attrs_Val_Tbl_Type**
- **Limit_Balances_Rec_Type** : A record type corresponding to the columns in the Limit Balances. View (QP_LIMIT_BALANCES)
- **Limit_Balances_Tbl_Type**
- **Limit_Balances_Val_Rec_Type** : A record type used to store values corresponding to IDs in the Limits Balances record.
- **Limit_Balances_Val_Tbl_Type**

Setting Up and Parameter Descriptions

The following chart describes all parameters used by the public API QP_LIMITS_PUB.PROCESS_LIMITS. All of the inbound and outbound parameters are listed. Additional information on these parameters follows.

PROCEDURE Process_Limits

Table 5–68 QP_LIMITS_PUB.PROCESS_LIMITS Parameters

Parameter	Usage	Type	Default
p_api_version_number	IN	Number	
p_init_msg_list	IN	Varchar2	FND_API.G_FALSE
p_return_values	IN	Varchar2	FND_API.G_FALSE
p_commit	IN	Varchar2	FND_API.G_FALSE
x_return_status	OUT	Varchar2	
x_msg_count	OUT	Number	
x_msg_data	OUT	Varchar2	
p_LIMITS_rec	IN	Limits_Rec_Type	G_MISS_LIMITS_REC
p_LIMITS_val_rec	IN	Limits_Val_Rec_Type	G_MISS_LIMITS_VAL_REC
p_LIMIT_ATTRS_tbl	IN	Limit_Attrs_Tbl_Type	G_MISS_LIMIT_ATTRS_TBL
p_LIMIT_ATTRS_val_tbl	IN	Limit_Attrs_Val_Tbl_Type	G_MISS_LIMIT_ATTRS_VAL_TBL
p_LIMIT_BALANCES_tbl	IN	Limit_Balances_Tbl_Type	G_MISS_LIMIT_BALANCES_TBL
p_LIMIT_BALANCES_val_tbl	IN	Limit_Balances_Val_Tbl_Type	G_MISS_LIMIT_BALANCES_VAL_TBL
x_LIMITS_rec	OUT	Limits_Rec_Type	
x_LIMITS_val_rec	OUT	Limits_Val_Rec_Type	
x_LIMIT_ATTRS_tbl	OUT	Limit_Attrs_Tbl_Type	
x_LIMIT_ATTRS_val_tbl	OUT	Limit_Attrs_Val_Tbl_Type	

Table 5–68 QP_LIMITS_PUB.PROCESS_LIMITS Parameters

Parameter	Usage	Type	Default
x_LIMIT_BALANCES_tbl	OUT	Limit_Balances_Tbl_Type	
x_LIMIT_BALANCES_val_tbl	OUT	Limit_Balances_Val_Tbl_Type	

PROCEDURE Lock_Limits

The following chart describes all parameters used by the public API QP_LIMITS_PUB.LOCK_LIMITS. All of the inbound and outbound parameters are listed. Additional information on these parameters follows.

Table 5–69 QP_LIMITS_PUB.LOCK_LIMITS Parameters

Parameter	Usage	Type	Default
p_api_version_number	IN	Number	
p_init_msg_list	IN	Varchar2	FND_API.G_FALSE
p_return_values	IN	Varchar2	FND_API.G_FALSE
x_return_status	OUT	Varchar2	
x_msg_count	OUT	Number	
x_msg_data	OUT	Varchar2	
p_LIMITS_rec	IN	Limits_Rec_Type	G_MISS_LIMITS_REC
p_LIMITS_val_rec	IN	Limits_Val_Rec_Type	G_MISS_LIMITS_VAL_REC
p_LIMIT_ATTRS_tbl	IN	Limit_Attrs_Tbl_Type	G_MISS_LIMIT_ATTRS_TBL
p_LIMIT_ATTRS_val_tbl	IN	Limit_Attrs_Val_Tbl_Type	G_MISS_LIMIT_ATTRS_VAL_TBL
p_LIMIT_BALANCES_tbl	IN	Limit_Balances_Tbl_Type	G_MISS_LIMIT_BALANCES_TBL
p_LIMIT_BALANCES_val_tbl	IN	Limit_Balances_Val_Tbl_Type	G_MISS_LIMIT_BALANCES_VAL_TBL
x_LIMITS_rec	OUT	Limits_Rec_Type	
x_LIMITS_val_rec	OUT	Limits_Val_Rec_Type	
x_LIMIT_ATTRS_tbl	OUT	Limit_Attrs_Tbl_Type	

Table 5–69 QP_LIMITS_PUB.LOCK_LIMITS Parameters

Parameter	Usage	Type	Default
x_LIMIT_ATTRS_val_tbl	OUT	Limit_Attrs_Val_Tbl_Type	
x_LIMIT_BALANCES_tbl	OUT	Limit_Balances_Tbl_Type	
x_LIMIT_BALANCES_val_tbl	OUT	Limit_Balances_Val_Tbl_Type	

PROCEDURE Get_Limits

The following chart describes all parameters used by the public API QP_LIMITS_PUB.GET_LIMITS. All of the inbound and outbound parameters are listed. Additional information on these parameters follows.

Table 5–70 QP_LIMITS_PUB.GET_LIMITS Parameters

Parameter	Usage	Type	Req	Default
p_api_version_number	IN	Number	Yes	
p_init_msg_list	IN	Varchar2	No	FND_API.G_FALSE
p_return_values	IN	Varchar2	No	FND_API.G_FALSE
x_return_status	OUT	Varchar2	No	
x_msg_count	OUT	Number	No	
x_msg_data	OUT	Varchar2	No	
p_limit_id	IN	Number	No	FND_API.G_MISS_NUM
p_limit	IN	Varchar2	No	FND_API.G_MISS_CHAR
x_LIMITS_rec	OUT	Limits_Rec_Type	No	
x_LIMITS_val_rec	OUT	Limits_Val_Rec_Type	No	
x_LIMIT_ATTRS_tbl	OUT	Limit_Attrs_Tbl_Type	No	
x_LIMIT_ATTRS_val_tbl	OUT	Limit_Attrs_Val_Tbl_Type	No	
x_LIMIT_BALANCES_tbl	OUT	Limit_Balances_Tbl_Type	No	

Table 5–70 QP_LIMITS_PUB.GET_LIMITS Parameters

Parameter	Usage	Type	Req	Default
x_LIMIT_BALANCES_val_tbl	OUT	Limit_Balances_Val_Tbl_Type	No	

PL/SQL Record Structures

For each column of the PL/SQL record structure, the following information has been documented:

- *Datatype*: data type for this field
- *Req*: X if required at entry, *Blank* if Optional
- *Drv*: For internal use, users cannot update these fields
- *Default*: Defaulted value for this field

Limits_Rec_Type

For column descriptions, please refer to the Oracle Pricing TRM for the table QP_LIMITS.

The Derived value for the following parameters is Null.

Table 5–71 QP_LIMITS

Parameter	Type	Req	Default
attribute1	Varchar2(240)	No	FND_API.G_MISS_CHAR
attribute2	Varchar2(240)	No	FND_API.G_MISS_CHAR
attribute3	Varchar2(240)	No	FND_API.G_MISS_CHAR
attribute4	Varchar2(240)	No	FND_API.G_MISS_CHAR
attribute5	Varchar2(240)	No	FND_API.G_MISS_CHAR
attribute6	Varchar2(240)	No	FND_API.G_MISS_CHAR
attribute7	Varchar2(240)	No	FND_API.G_MISS_CHAR
attribute8	Varchar2(240)	No	FND_API.G_MISS_CHAR
attribute9	Varchar2(240)	No	FND_API.G_MISS_CHAR
attribute10	Varchar2(240)	No	FND_API.G_MISS_CHAR
attribute11	Varchar2(240)	No	FND_API.G_MISS_CHAR

Table 5–71 QP_LIMITS

Parameter	Type	Req	Default
attribute12	Varchar2(240)	No	FND_API.G_MISS_CHAR
attribute13	Varchar2(240)	No	FND_API.G_MISS_CHAR
attribute14	Varchar2(240)	No	FND_API.G_MISS_CHAR
attribute15	Varchar2(240)	No	FND_API.G_MISS_CHAR
Amount	Number	Yes	FND_API.G_MISS_NUM
Basis	Varchar2(30)	Yes	FND_API.G_MISS_CHAR
Context	Varchar2(30)	No	FND_API.G_MISS_CHAR
created_by	Number	Yes	FND_API.G_MISS_CHAR
creation_date	Date	Yes	FND_API.G_MISS_DATE
last_updated_by	Number	Yes	FND_API.G_MISS_NUM
last_update_date	Date	Yes	FND_API.G_MISS_DATE
last_update_login	Number	No	FND_API.G_MISS_NUM
limit_exceed_action_code	Varchar2(30)	Yes	FND_API.G_MISS_CHAR
limit_id	Number	Yes	FND_API.G_MISS_NUM
limit_level_code	Varchar2(30)	Yes	FND_API.G_MISS_CHAR
limit_number	Number	Yes	FND_API.G_MISS_NUM
list_header_id	Number	Yes	FND_API.G_MISS_NUM
list_line_id	Number	No	FND_API.G_MISS_NUM
limit_hold_flag	Varchar2(1)	Yes	FND_API.G_MISS_CHAR
multival_attr1_type	Varchar2(30)	No	FND_API.G_MISS_CHAR
multival_attr1_context	Varchar2(30)	No	FND_API.G_MISS_CHAR
multival_attribute1	Varchar2(30)	No	FND_API.G_MISS_CHAR
multival_attr1_datatype	Varchar2(10)	No	FND_API.G_MISS_CHAR
multival_attr2_type	Varchar2(30)	No	FND_API.G_MISS_CHAR
multival_attr2_context	Varchar2(30)	No	FND_API.G_MISS_CHAR
multival_attribute2	Varchar2(30)	No	FND_API.G_MISS_CHAR
multival_attr2_datatype	Varchar2(10)	No	FND_API.G_MISS_CHAR

Table 5–71 QP_LIMITS

Parameter	Type	Req	Default
organization_flag	Varchar2(1)	Yes	FND_API.G_MISS_CHAR
program_application_id	Number	No	FND_API.G_MISS_NUM
program_id	Number	No	FND_API.G_MISS_NUM
program_update_date	Date	No	FND_API.G_MISS_DATE
request_id	Number	No	FND_API.G_MISS_NUM
return_status	Varchar2(1)	No	FND_API.G_MISS_CHAR
db_flag	Varchar2(1)	No	FND_API.G_MISS_CHAR
operation	Varchar2(30)	No	FND_API.G_MISS_CHAR

Limits_Tbl_Type**Table 5–72 Limits_Tbl_Type**

Parameter	Type	Req	Drv	Default
Limits_Rec_Type	Record			

Limits_Val_Rec_Type**Table 5–73 Limits_Val_Rec_Type**

Parameter	Type	Req	Drv	Default
limit_exceed_action	Varchar2(240)	No	No	FND_API.G_MISS_CHAR
limit	Varchar2(240)	No	No	FND_API.G_MISS_CHAR
limit_level	Varchar2(240)	No	No	FND_API.G_MISS_CHAR
list_header	Varchar2(240)	No	No	FND_API.G_MISS_CHAR
list_line	Varchar2(240)	No	No	FND_API.G_MISS_CHAR
organization	Varchar2(240)	No	No	FND_API.G_MISS_CHAR

Limits_Val_Tbl_Type**Table 5–74 Limits_Val_Tbl_Type**

Parameter	Type	Req	Drv	Default
Limits_Val_Rec_Type	Record			

Limit_Attrs_Rec_Type

For column descriptions, please refer to the Oracle Pricing TRM for the table QP_LIMIT_ATTRIBUTES.

Table 5–75 Limit_Attrs_Rec_Type

Parameter	Type	Req	Drv	Default
attribute1	Varchar2(240)	No	No	FND_API.G_MISS_CHAR
attribute2	Varchar2(240)	No	No	FND_API.G_MISS_CHAR
attribute3	Varchar2(240)	No	No	FND_API.G_MISS_CHAR
attribute4	Varchar2(240)	No	No	FND_API.G_MISS_CHAR
attribute5	Varchar2(240)	No	No	FND_API.G_MISS_CHAR
attribute6	Varchar2(240)	No	No	FND_API.G_MISS_CHAR
attribute7	Varchar2(240)	No	No	FND_API.G_MISS_CHAR
attribute8	Varchar2(240)	No	No	FND_API.G_MISS_CHAR
attribute9	Varchar2(240)	No	No	FND_API.G_MISS_CHAR
attribute10	Varchar2(240)	No	No	FND_API.G_MISS_CHAR
attribute11	Varchar2(240)	No	No	FND_API.G_MISS_CHAR
attribute12	Varchar2(240)	No	No	FND_API.G_MISS_CHAR
attribute13	Varchar2(240)	No	No	FND_API.G_MISS_CHAR
attribute14	Varchar2(240)	No	No	FND_API.G_MISS_CHAR
attribute15	Varchar2(240)	No	No	FND_API.G_MISS_CHAR
comparison_operator_code	Varchar2(30)	Yes	No	FND_API.G_MISS_CHAR
context	Varchar2(30)	No	No	FND_API.G_MISS_CHAR
created_by	Number	Yes	No	FND_API.G_MISS_CHAR

Table 5–75 Limit_Attrs_Rec_Type

Parameter	Type	Req	Drv	Default
creation_date	Date	Yes	No	FND_API.G_MISS_DATE
last_updated_by	Number	Yes	No	FND_API.G_MISS_NUM
last_update_date	Date	Yes	No	FND_API.G_MISS_DATE
last_update_login	Number	No	No	FND_API.G_MISS_NUM
limit_attribute	Varchar2(30)	Yes	No	FND_API.G_MISS_CHAR
limit_attribute_context	Varchar2(30)	Yes	No	FND_API.G_MISS_CHAR
limit_attribute_id	Number	Yes	No	FND_API.G_MISS_NUM
limit_attribute_type	Varchar2(30)	Yes	No	FND_API.G_MISS_CHAR
limit_attr_datatype	Varchar2(10)	Yes	No	FND_API.G_MISS_CHAR
limit_attr_value	Varchar2(240)	Yes	No	FND_API.G_MISS_CHAR
limit_id	Number	No	No	FND_API.G_MISS_NUM
program_application_id	Number	No	No	FND_API.G_MISS_NUM
program_id	Number	No	No	FND_API.G_MISS_NUM
program_update_date	Date	No	No	FND_API.G_MISS_DATE
request_id	Number	No	No	FND_API.G_MISS_NUM
return_status	Varchar2(1)	No	No	FND_API.G_MISS_CHAR
db_flag	Varchar2(1)	No	No	FND_API.G_MISS_CHAR
operation	Varchar2(30)	No	No	FND_API.G_MISS_CHAR

Limit_Attrs_Tbl_Type**Table 5–76 Limit_Attrs_Tbl_Type**

Parameter	Type	Req	Drv	Default
Limit_Attrs_Rec_Type	Record			

Limit_Attrs_Val_Rec_Type

Table 5–77 Limit_Attrs_Val_Rec_Type

Parameter	Type	Req	Default
comparison_operator	Varchar2(240)	No	FND_API.G_MISS_CHAR
limit_attribute	Varchar2(240)	No	FND_API.G_MISS_CHAR
limit	Varchar2(240)	No	FND_API.G_MISS_CHAR

Limit_Attrs_Val_Tbl_Type**Table 5–78 Limit_Attrs_Val_Tbl_Type**

Parameter	Type	Req	Drv	Default
Limit_Attrs_Val_Rec_Type	Record			

Limit_Balances_Rec_Type

For column descriptions, please refer to the Oracle Pricing TRM for the table QP_LIMIT_BALANCES.

Table 5–79 Limit_Balances_Rec_Type

Parameter	Type	Req	Drv	Default
attribute1	Varchar2(240)	No	No	FND_API.G_MISS_CHAR
attribute2	Varchar2(240)	No	No	FND_API.G_MISS_CHAR
attribute3	Varchar2(240)	No	No	FND_API.G_MISS_CHAR
attribute4	Varchar2(240)	No	No	FND_API.G_MISS_CHAR
attribute5	Varchar2(240)	No	No	FND_API.G_MISS_CHAR
attribute6	Varchar2(240)	No	No	FND_API.G_MISS_CHAR
attribute7	Varchar2(240)	No	No	FND_API.G_MISS_CHAR
attribute8	Varchar2(240)	No	No	FND_API.G_MISS_CHAR
attribute9	Varchar2(240)	No	No	FND_API.G_MISS_CHAR
attribute10	Varchar2(240)	No	No	FND_API.G_MISS_CHAR
attribute11	Varchar2(240)	No	No	FND_API.G_MISS_CHAR
attribute12	Varchar2(240)	No	No	FND_API.G_MISS_CHAR
attribute13	Varchar2(240)	No	No	FND_API.G_MISS_CHAR

Table 5–79 Limit_Balances_Rec_Type

Parameter	Type	Req	Drv	Default
attribute14	Varchar2(240)	No	No	FND_API.G_MISS_CHAR
attribute15	Varchar2(240)	No	No	FND_API.G_MISS_CHAR
available_amount	Number	Yes	No	FND_API.G_MISS_NUM
consumed_amount	Number	Yes	No	FND_API.G_MISS_NUM
context	Varchar2(30)	No	No	FND_API.G_MISS_CHAR
created_by		Yes	No	FND_API.G_MISS_CHAR
creation_date	Date	Yes	No	FND_API.G_MISS_DATE
last_updated_by	Number	Yes	No	FND_API.G_MISS_NUM
last_update_date	Date	Yes	No	FND_API.G_MISS_DATE
last_update_login	Number	No	No	FND_API.G_MISS_NUM
limit_balance_id	Number	Yes	No	FND_API.G_MISS_NUM
limit_id	Number	Yes	No	FND_API.G_MISS_NUM
organization_attr_context	Varchar2(30)	No	No	FND_API.G_MISS_CHAR
organization_attribute	Varchar2(30)	No	No	FND_API.G_MISS_CHAR
organization_attr_value	Varchar2(240)	No	No	FND_API.G_MISS_CHAR
multival_attr1_type	Varchar2(30)	No	No	FND_API.G_MISS_CHAR
multival_attr1_context	Varchar2(30)	No	No	FND_API.G_MISS_CHAR
multival_attribute1	Varchar2(30)	No	No	FND_API.G_MISS_CHAR
multival_attr1_value	Varchar2(240)	No	No	FND_API.G_MISS_CHAR
multival_attr1_datatype	Varchar2(10)	No	No	FND_API.G_MISS_CHAR
multival_attr2_type	Varchar2(30)	No	No	FND_API.G_MISS_CHAR
multival_attr2_context	Varchar2(30)	No	No	FND_API.G_MISS_CHAR
multival_attribute2	Varchar2(30)	No	No	FND_API.G_MISS_CHAR
multival_attr2_value	Varchar2(240)	No	No	FND_API.G_MISS_CHAR
multival_attr2_datatype	Varchar2(10)	No	No	FND_API.G_MISS_CHAR
program_application_id	Number	No	No	FND_API.G_MISS_NUM
program_id	Number	No	No	FND_API.G_MISS_NUM

Table 5–79 Limit_Balances_Rec_Type

Parameter	Type	Req	Drv	Default
program_update_date	Date	No	No	FND_API.G_MISS_DATE
request_id	Number	No	No	FND_API.G_MISS_NUM
reserved_amount	Number	Yes	No	FND_API.G_MISS_NUM
return_status	Varchar2(1)	No	No	FND_API.G_MISS_CHAR
db_flag	Varchar2(1)	No	No	FND_API.G_MISS_CHAR
operation	Varchar2(30)	No	No	FND_API.G_MISS_CHAR

Limit_Balances_Tbl_Type

Table 5–80 Limit_Balances_Tbl_Type

Parameter	Type	Req	Drv	Default
Limit_Balances_Rec_Type	Record			

Limit_Balances_Val_Rec_Type

Table 5–81 Limit_Balances_Val_Rec_Type

Parameter	Type	Req	Drv	Default
limit_balance	Varchar2(240)			FND_API.G_MISS_CHAR
limit	Varchar2(240)			FND_API.G_MISS_CHAR

Limit_Balances_Val_Tbl_Type

Table 5–82 Limit_Balances_Val_Tbl_Type

Parameter	Type	Req	Drv	Default
Limit_Balances_Val_Rec_Type	Record			

Validation of Limits Public API

Standard Validation

Oracle Advanced Pricing validates all required columns in the Limits Public API. For more information, see: *Oracle Pricing Technical Reference Manual*.

Other Validation

None

Error Handling

If any validation fails, the API will return error status to the calling module. The Limits Public API processes the rows and reports the values in the following table for every record.

Table 5-83 Error Handling

Condition	PROCESS_STATUS	ERROR_MESSAGE
Success	5	Null
Failure	4	actual error message

Example of Limits Public API

The following sample code demonstrates Limits Public API.

Example of inserting limits

```

set serverout on
declare
  glmt_return_status varchar2(30);
  glmt_msg_data varchar2(2000);
  glmt_msg_count number := 0;
  glmt_limits_recQP_LIMITS_PUB.Limits_Rec_Type;
  glmt_limits_val_recQP_LIMITS_PUB.Limits_Val_Rec_Type;
  glmt_limit_attrs_tblQP_LIMITS_PUB.Limit_Attrs_Tbl_Type;
  glmt_limit_attrs_val_tblQP_LIMITS_PUB.Limit_Attrs_Val_Tbl_Type;
  glmt_limit_balances_tblQP_LIMITS_PUB.Limit_Balances_Tbl_Type ;
  glmt_limit_balances_val_tblQP_LIMITS_PUB.Limit_Balances_Val_Tbl_Type ;
  plmt_limits_recQP_LIMITS_PUB.Limits_Rec_Type;
  plmt_limits_val_recQP_LIMITS_PUB.Limits_Val_Rec_Type;
  plmt_limit_attrs_tblQP_LIMITS_PUB.Limit_Attrs_Tbl_Type;
  plmt_limit_attrs_val_tblQP_LIMITS_PUB.Limit_Attrs_Val_Tbl_Type;
  plmt_limit_balances_tblQP_LIMITS_PUB.Limit_Balances_Tbl_Type ;
  plmt_limit_balances_val_tblQP_LIMITS_PUB.Limit_Balances_Val_Tbl_Type ;
  i number := 1;
  j number := 1;
begin
  /* Set the limit_id to g_miss_num to create the Limits Record(Header)*/

```

```
dbms_output.put_line('Initializing the values');

glmt_limits_rec.limit_id := FND_API.G_MISS_NUM;
glmt_limits_rec.created_by := FND_API.G_MISS_NUM;
glmt_limits_rec.last_updated_by := FND_API.G_MISS_NUM;
glmt_limits_rec.list_header_id := 7700;
--glmt_limits_rec.list_line_id := 1064;
glmt_limits_rec.limit_number := 1;
glmt_limits_rec.basis := 'COST';
glmt_limits_rec.organization_flag := 'N';
glmt_limits_rec.limit_level_code := 'TRANSACTION';
--glmt_limits_rec.limit_exceed_action_code := 'HARD';
glmt_limits_rec.amount := 1000;
glmt_limits_rec.LIMIT_HOLD_FLAG := 'Y';
glmt_limits_rec.MULTIVAL_ATTR1_CONTEXT := FND_API.G_MISS_CHAR;
glmt_limits_rec.MULTIVAL_ATTR2_CONTEXT := FND_API.G_MISS_CHAR;
glmt_limits_rec.operation := QP_GLOBALS.g_opr_create;

i := 1;
/* Set the limit_balance_id to g_miss_num to create the Limit Balance
Record(Header)*/
glmt_limit_balances_tbl(i).limit_balance_id := FND_API.G_MISS_NUM;
glmt_limit_balances_tbl(i).created_by := FND_API.G_MISS_NUM;
glmt_limit_balances_tbl(i).last_updated_by := FND_API.G_MISS_NUM;
glmt_limit_balances_tbl(i).available_amount := 10000;
glmt_limit_balances_tbl(i).reserved_amount := 0;
glmt_limit_balances_tbl(i).consumed_amount := 0;
glmt_limit_balances_tbl(i).operation := QP_GLOBALS.g_opr_create;
i := i + 1;

glmt_limit_balances_tbl(i).limit_balance_id := FND_API.G_MISS_NUM;
glmt_limit_balances_tbl(i).created_by := FND_API.G_MISS_NUM;
glmt_limit_balances_tbl(i).last_updated_by := FND_API.G_MISS_NUM;
glmt_limit_balances_tbl(i).available_amount := 10000;
glmt_limit_balances_tbl(i).reserved_amount := 0;
glmt_limit_balances_tbl(i).consumed_amount := 0;
glmt_limit_balances_tbl(i).operation := QP_GLOBALS.g_opr_create;
/* Create Limit Attribute of type 'Limit Attribute 2' */
glmt_limit_attrs_tbl(j).limit_attribute_id := FND_API.G_MISS_NUM;
glmt_limit_attrs_tbl(j).created_by := FND_API.G_MISS_NUM;
glmt_limit_attrs_tbl(j).last_updated_by := FND_API.G_MISS_NUM;
glmt_limit_attrs_tbl(j).limit_attribute_type := 'QUALIFIER';
glmt_limit_attrs_tbl(j).limit_attribute_context := 'CUSTOMER';
glmt_limit_attrs_tbl(j).limit_attribute := 'QUALIFIER_ATTRIBUTE7';
glmt_limit_attrs_tbl(j).limit_attr_value := '221';
```

```

glmt_limit_attrs_tbl(j).limit_attr_datatype := 'N';
glmt_limit_attrs_tbl(j).comparison_operator_code := '=';
glmt_limit_attrs_tbl(j).operation := QP_GLOBALS.g_opr_create;

```

QP_Limits_PUB.Process_Limits

```

(   p_api_version_number=> 1.0
,   x_return_status=> glmt_return_status
,   x_msg_count=> glmt_msg_count
,   x_msg_data=> glmt_msg_data
,   p_LIMITS_rec=> glmt_limits_rec
,   p_LIMITS_val_rec=> glmt_limits_val_rec
,   p_LIMIT_ATTRS_tbl=> glmt_limit_attrs_tbl
,   p_LIMIT_ATTRS_val_tbl=> glmt_limit_attrs_val_tbl
,   p_LIMIT_BALANCES_tbl=> glmt_limit_balances_tbl
,   p_LIMIT_BALANCES_val_tbl=> glmt_limit_balances_val_tbl
,   x_LIMITS_rec=> plmt_limits_rec
,   x_LIMITS_val_rec=> plmt_limits_val_rec
,   x_LIMIT_ATTRS_tbl=> plmt_limit_attrs_tbl
,   x_LIMIT_ATTRS_val_tbl=> plmt_limit_attrs_val_tbl
,   x_LIMIT_BALANCES_tbl=> plmt_limit_balances_tbl
,   x_LIMIT_BALANCES_val_tbl=> plmt_limit_balances_val_tbl
);
dbms_output.put_line('Return Status' || glmt_return_status);
IF glmt_return_status <> FND_API.G_RET_STS_SUCCESS
THEN
    dbms_output.put_line('Error');
    RAISE FND_API.G_EXC_UNEXPECTED_ERROR;
ELSE
    dbms_output.put_line('Success');
END IF;
EXCEPTION

    WHEN FND_API.G_EXC_ERROR THEN

        glmt_return_status := FND_API.G_RET_STS_ERROR;
        for k in 1 .. glmt_msg_count
        loop
            glmt_msg_data := oe_msg_pub.get( p_msg_index => k,p_encoded => 'F');
            dbms_output.put_line('err msg ' || k || 'is: ' || glmt_msg_data);
            oe_msg_pub.delete_msg(k);
        end loop;
        /*
        glmt_msg_data := oe_msg_pub.get_single_message(glmt_return_status);

```

```
*/
dbms_output.put_
line('*****');

dbms_output.put_line('err msg 1 is : ' || glmt_msg_data);
WHEN FND_API.G_EXC_UNEXPECTED_ERROR THEN
  glmt_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;

  dbms_output.put_line(' msg count is : ' || glmt_msg_count);
  dbms_output.put_line('err msg is : ' || glmt_msg_data);

  for k in 1 .. glmt_msg_count
  loop
    glmt_msg_data := oe_msg_pub.get( p_msg_index => k,p_encoded => 'F');
    dbms_output.put_line('err msg ' || k ||'is: ' || glmt_msg_data);
    oe_msg_pub.delete_msg(k);
  end loop;
  /*
  glmt_msg_data := oe_msg_pub.get_single_message(glmt_return_status);
  dbms_output.put_line('err msg is: ' || glmt_msg_data);
  */
WHEN OTHERS THEN
  glmt_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;

  dbms_output.put_line('err msg 3 is : ' || glmt_msg_data);

  for k in 1 .. glmt_msg_count
  loop
    glmt_msg_data := oe_msg_pub.get( p_msg_index => k,p_encoded => 'F');
    dbms_output.put_line('err msg ' || k ||'is: ' || glmt_msg_data);
    oe_msg_pub.delete_msg(k);
  end loop;
end ;
/
commit;
```

Example of how to update available amount on a limit balance

```
set serverout on
declare
  glmt_return_status varchar2(30);
  glmt_msg_data varchar2(2000);
  glmt_msg_count number := 0;
  glmt_limits_recQP_LIMITS_PUB.Limits_Rec_Type;
  glmt_limits_val_recQP_LIMITS_PUB.Limits_Val_Rec_Type;
```

```

glmt_limit_attrs_tblQP_LIMITS_PUB.Limit_Attrs_Tbl_Type;
glmt_limit_attrs_val_tblQP_LIMITS_PUB.Limit_Attrs_Val_Tbl_Type;
glmt_limit_balances_tblQP_LIMITS_PUB.Limit_Balances_Tbl_Type ;
glmt_limit_balances_val_tblQP_LIMITS_PUB.Limit_Balances_Val_Tbl_Type ;

plmt_limits_recQP_LIMITS_PUB.Limits_Rec_Type;
plmt_limits_val_recQP_LIMITS_PUB.Limits_Val_Rec_Type;
plmt_limit_attrs_tblQP_LIMITS_PUB.Limit_Attrs_Tbl_Type;
plmt_limit_attrs_val_tblQP_LIMITS_PUB.Limit_Attrs_Val_Tbl_Type;
plmt_limit_balances_tblQP_LIMITS_PUB.Limit_Balances_Tbl_Type ;
plmt_limit_balances_val_tblQP_LIMITS_PUB.Limit_Balances_Val_Tbl_Type ;

i number := 1;
begin
glmt_limits_rec.limit_id := 259;
glmt_limits_rec.organization_flag := 'N';
--glmt_limits_rec.amount := 700;
glmt_limits_rec.operation := QP_GLOBALS.G_OPR_UPDATE;
QP_Limits_PUB.Process_Limits
(
  p_api_version_number=> 1.0
  , x_return_status=> glmt_return_status
  , x_msg_count=> glmt_msg_count
  , x_msg_data=> glmt_msg_data
  , p_LIMITS_rec=> glmt_limits_rec
  , x_LIMITS_rec=> plmt_limits_rec
  , x_LIMITS_val_rec=> plmt_limits_val_rec
  , x_LIMIT_ATTRS_tbl=> plmt_limit_attrs_tbl
  , x_LIMIT_ATTRS_val_tbl=> plmt_limit_attrs_val_tbl
  , x_LIMIT_BALANCES_tbl=> plmt_limit_balances_tbl
  , x_LIMIT_BALANCES_val_tbl=> plmt_limit_balances_val_tbl
);
dbms_output.put_line('Return Status' || glmt_return_status);
IF glmt_return_status <> FND_API.G_RET_STS_SUCCESS
THEN
dbms_output.put_line('Error');
RAISE FND_API.G_EXC_UNEXPECTED_ERROR;
ELSE
dbms_output.put_line('Success');
END IF;

EXCEPTION
WHEN FND_API.G_EXC_ERROR THEN
glmt_return_status := FND_API.G_RET_STS_ERROR;
-- Get message count and data
dbms_output.put_line('err msg 1 is : ' || glmt_msg_data);

```

```
WHEN FND_API.G_EXC_UNEXPECTED_ERROR THEN
    glmt_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;
    dbms_output.put_line(' msg count 2 is : ' || glmt_msg_count);
    dbms_output.put_line('err msg 2 is : ' || glmt_msg_data);

    for k in 1 .. glmt_msg_count
    loop
        glmt_msg_data := oe_msg_pub.get( p_msg_index => k,p_encoded => 'F');
        /*
        oe_msg_pub.Count_And_Get(p_count => glmt_msg_count ,p_data => glmt_
        msg_data);
        */
        -- Get message count and data
        dbms_output.put_line('err msg ' || k || 'is: ' || glmt_msg_data);
    null;
    end loop;
WHEN OTHERS THEN
    glmt_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;
    -- Get message count and data
    dbms_output.put_line('err msg 3 is : ' || glmt_msg_data);
end ;
/
commit;
```

Example of deleting a limits record

```
set serverout on
declare
    glmt_return_status varchar2(30);
    glmt_msg_data varchar2(2000);
    glmt_msg_count number := 0;
    glmt_limits_recQP_LIMITS_PUB.Limits_Rec_Type;
    glmt_limits_val_recQP_LIMITS_PUB.Limits_Val_Rec_Type;
    glmt_limit_attrs_tblQP_LIMITS_PUB.Limit_Attrs_Tbl_Type;
    glmt_limit_attrs_val_tblQP_LIMITS_PUB.Limit_Attrs_Val_Tbl_Type;
    glmt_limit_balances_tblQP_LIMITS_PUB.Limit_Balances_Tbl_Type ;
    glmt_limit_balances_val_tblQP_LIMITS_PUB.Limit_Balances_Val_Tbl_Type ;

    plmt_limits_recQP_LIMITS_PUB.Limits_Rec_Type;
    plmt_limits_val_recQP_LIMITS_PUB.Limits_Val_Rec_Type;
    plmt_limit_attrs_tblQP_LIMITS_PUB.Limit_Attrs_Tbl_Type;
    plmt_limit_attrs_val_tblQP_LIMITS_PUB.Limit_Attrs_Val_Tbl_Type;
    plmt_limit_balances_tblQP_LIMITS_PUB.Limit_Balances_Tbl_Type ;
    plmt_limit_balances_val_tblQP_LIMITS_PUB.Limit_Balances_Val_Tbl_Type ;
```

```

i number := 1;
j number := 1;

begin

glmt_limits_rec.limit_id := 258;
QP_Limits_PUB.Process_Limits
(
  p_api_version_number=> 1.0
,
  x_return_status=> glmt_return_status
,
  x_msg_count=> glmt_msg_count
,
  x_msg_data=> glmt_msg_data
,
  p_LIMITS_rec=> glmt_limits_rec
,
  p_LIMITS_val_rec=> glmt_limits_val_rec
,
  p_LIMIT_ATTRS_tbl=> glmt_limit_attr_tbl
,
  p_LIMIT_ATTRS_val_tbl=> glmt_limit_attr_val_tbl
,
  p_LIMIT_BALANCES_tbl=> glmt_limit_balances_tbl
,
  p_LIMIT_BALANCES_val_tbl=> glmt_limit_balances_val_tbl
,
  x_LIMITS_rec=> plmt_limits_rec
,
  x_LIMITS_val_rec=> plmt_limits_val_rec
,
  x_LIMIT_ATTRS_tbl=> plmt_limit_attr_tbl
,
  x_LIMIT_ATTRS_val_tbl=> plmt_limit_attr_val_tbl
,
  x_LIMIT_BALANCES_tbl=> plmt_limit_balances_tbl
,
  x_LIMIT_BALANCES_val_tbl=> plmt_limit_balances_val_tbl
);

dbms_output.put_line('Return Status' || glmt_return_status);
IF glmt_return_status <> FND_API.G_RET_STS_SUCCESS THEN
  dbms_output.put_line('Error');
  RAISE FND_API.G_EXC_UNEXPECTED_ERROR;
ELSE
  dbms_output.put_line('Success');
END IF;
EXCEPTION
  WHEN FND_API.G_EXC_ERROR THEN
    glmt_return_status := FND_API.G_RET_STS_ERROR;
    -- Get message count and data
    dbms_output.put_line('err msg 1 is : ' || glmt_msg_data);
  WHEN FND_API.G_EXC_UNEXPECTED_ERROR THEN
    glmt_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;
    dbms_output.put_line(' msg count 2 is : ' || glmt_msg_count);
    dbms_output.put_line('err msg 2 is : ' || glmt_msg_data);
    for k in 1 .. glmt_msg_count
      loop
        glmt_msg_data := oe_msg_pub.get( p_msg_index => k,p_encoded => 'F');
        -- Get message count and data

```

```
        dbms_output.put_line('err msg ' || k || 'is: ' || glmt_msg_data);
        null;
    end loop;
    WHEN OTHERS THEN
        glmt_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;
        -- Get message count and data
        dbms_output.put_line('err msg 3 is : ' || glmt_msg_data);
    commit;
end ;
/
commit;
```


Get Currency Application Program Interface

This section explains how to use the Get_Currency API and how it functions in Oracle Advanced Pricing.

Get Currency API Features

The Get_Currency API is used to get all the currency codes for a given price list. The package QP_UTIL_PUB contains the procedure Get_Currency.

Functional Overview

While processing an order using a particular price list, the OM displays a valid set of currency codes when user activates the LOV. The OM calls this API to get the valid currency codes for the given price list and pricing effective date. If pricing effective date is not passed by calling application, the current date is defaulted. If profile QP: Multi Currency Installed is 'Y' then currency codes are retrieved by joining the view fnd_currencies_vl, table qp_list_headers_b and table qp_currency_details for the passed price list id else all the effective currencies are retrieved from view fnd_currencies_vl.

Setting Up and Parameter Descriptions

The following chart describes all parameters used by the Get_Currency API. All of the inbound and outbound parameters are listed.

Table 5–84 Get_Currency

Parameter	Usage	Type	Req	Drv
l_price_list_id	IN	Number	Yes	No
l_pricing_effective_date	IN	Date	No	No
l_currency_code_tbl	OUT	currency_code_tbl	No	No

CURRENCY_REC

The following table shows the parameters for this structure.

Table 5–85 CURRENCY_REC

Parameter	Usage	Type	Req	Drv
currency_code	Null	Varchar2	Yes	No
currency_name	Null	Varchar2	Yes	No
currency_precision	Null	Number	Yes	No

CURRENCY_CODE_TBL

The following table shows the parameters for this structure.

Table 5–86 CURRENCY_CODE_TBL

Parameter	Usage	Type	Req	Drv
Currency_rec	Null	Record	No	No

Validation of Get_Currency API

Standard Validation

The caller is responsible for passing the right parameters to Get_Currency.

For specific information on the data implied by these columns, see your *Oracle Pricing Technical Reference Manual* for details.

Other Validation

None.

Error Handling

If any exception occurs in the Get_Currency API, it does not return any Currency Codes

NOTE: The Package Specification and Body files are QPXRTCNS.pls and QPXRTCNB.pls and are available under the source control directory \$QP_TOP/patch/115/sql.

Get Custom Price Application Program Interface

This section explains how to use the Get_Custom Price API (used in Formulas Setup) and how it functions in Oracle Advanced Pricing.

Get Custom Price API Features

The Get Custom Price API is a customizable function to which the user may add custom code. The Get_Custom_Price is called by the pricing engine while evaluating a formula that contains a formula line (step) of type Function. One or more formulas may be set up to contain a formula line of type Function and the same Get_Custom_price API is called each time. So the user must code the logic in the API based on the price_formula_id that is passed as an input parameter to the API.

Functional Overview

The package specification QP_CUSTOM contains the specification for the Get_Custom_Price API. The package body/function body is not shipped with Oracle Advanced Pricing. The user must create the Package Body for QP_CUSTOM containing the function body for Get_Custom_Price which must adhere to the Function specification (in terms of parameters and return value) provided in the QP_CUSTOM package specification. For the engine to make a call to Get_Custom_Price API, set the profile QP: Get Custom Price Customized at the site level to Yes. The Pricing engine displays an error if this profile is set and QP_CUSTOM. Get_Custom_Price is not created in the database in the applications schema:

- API to all procedures within and outside of this package
- Get_Custom_Price: A customizable function.

Setting Up and Parameter Descriptions

The following chart describes all parameters used by the public Get Custom Price API. All of the inbound and outbound parameters are listed. Additional information on these parameters may follow. These parameters are input parameters to the Get_Custom_Price API and are passed by the Engine.

The following table shows the parameters for this structure.

Table 5–87 *Get_Custom_Price*

Parameter	Usage	Type	Req	Drv
p_price_formula_id	In	Number	Yes	No
p_list_price	In	Number	No	No
p_price_effective_date	In	Date	Yes	No
p_req_line_attrs_tbl	In	Table (REQ_LINE_ATTRS_TBL)	Yes	No
	Return	Number	Yes	No

REQ_LINE_ATTRS_TBL

The following table shows the parameters for this structure.

Table 5–88 *REQ_LINE_ATTRS_TBL*

Parameter	Usage	Type	Req	Drv
rec_line_attrs_rec	Null	Record (REQ_LINE_ATTRS_REC)	No	No

REQ_LINE_ATTRS_REC

The following table shows the parameters for this structure.

Table 5–89 *REQ_LINE_ATTRS_REC*

Parameter	Usage	Type	Req	Drv
line_index	Null	Number	Yes	No
attribute_type	Null	Varchar2	Yes	No
context	Null	Varchar2	Yes	No
attribute	Null	Varchar2	Yes	No
value	Null	Varchar2	Yes	No

Validation of Get Custom Price API

Standard Validation

The user is responsible for all code and validations in the Get Custom Price API. For more information, see: Oracle Pricing Technical Reference Manual.

Other Validation

None

Error Handling

The user is responsible for all error handling in the Get Custom Price API. If any user-coded validation fails, the API should raise exceptions to propagate it to the calling module.

Sample Code1: The following is a sample code showing how the body of the Get_Custom_Price function is coded in the file QPXCUSTB.pls.

The user must use the function specification of Get_Custom_Price as well as any type definitions from QPXCUSTS.pls.

The parameters to Get_Custom_Price are always fixed and not customizable. But the user can use the input parameters passed by the pricing engine in their custom code. The function returns a number. The user can code the function to return the desired value which must be a number. The return value is used in the evaluation of the formula.

For example, consider a formula having an expression 1*2 where 1 and 2 are step numbers. Each step number corresponds to a formula line. Each formula line has a type:

- Step 1 corresponds to a formula line of type Numeric Constant with a component of '200,' and
- Step 2 corresponds to a formula line of type Function. This means the value returned by the QP_CUSTOM.Get_Custom_Price function will be used as the value for this step.

To evaluate the Formula, the pricing engine first obtains the value of each step and substitutes the step with its value in the expression. So step 1 is substituted by the value which is 200. Step 2 is substituted with the value returned by Get_Custom_Price which must be customized by the user (the Profile Option mentioned earlier must also be set to Yes to use this Get_Custom_Price functionality).

If Get_Custom_Price is customized as below:

```

PACKAGE BODY QP_CUSTOM AS
/*****
The Get_Custom_Price Function name and parameters are not customizable but the
body can be been customized. The parameters are:

p_price_formula_id: Primary key of the formula that uses the Get_Custom_Price
function
p_list_price: List price of the price list line to which the formula using Get_
Custom_Price is attached. May have null value.
p_price_effective_date: Current date when Formula is being evaluated by the
pricing engine.
p_req_line_attrs_tbl: PL/SQL table of records containing Context, Attribute,
Attribute Value records for Product and Pricing Attributes and a column
indicating the type - whether Product Attribute or Pricing Attribute.
Also the engine passes the Pricing Attributes and Product Attributes of only the
current line to which the formula is attached.
The parameters are passed to the function by the pricing engine and can be used
in the function body.

*****/
FUNCTION Get_Custom_Price
    p_price_formula_id    IN NUMBER,
    p_list_price           IN NUMBER,
    p_price_effective_date IN DATE,
    p_req_line_attrs_tbl IN QP_FORMULA_PRICE_CALC_PVT.REQ_LINE_ATTRS_TBL)

RETURN NUMBER IS
v_requested_item VARCHAR2(240);
v_weight         NUMBER;

BEGIN
IF p_price_formula_id = 1726    -- Assume this is the internal Id/primary key for
the sample Formula 1*2

THEN
    --Loop through the PL/SQL table of records passed by the Engine as an input
    parameter and
    --containing Pricing Attributes and Product Attributes of the Price List
    Line or Modifier Line to which
    -- the current formula is attached.

    FOR i IN 1..p_req_line_attrs_tbl.count LOOP

```

```
        IF p_req_line_attrs_tbl(i).attribute_type = 'PRODUCT'
AND      /*Attribute Type is Product*/

        p_req_line_attrs_tbl(i).context = 'ITEM'
AND

        p_req_line_attrs_tbl(i).attribute = 'PRICING_ATTRIBUTE1'
THEN
        -- For this combination of Product Context and Attribute, the Attribute
        Value is the Inventory Item Id

        v_requested_item := p_req_line_attrs_tbl(i).value;

        END IF;

        IF p_req_line_attrs_tbl(i).attribute_type = 'PRICING'
AND      /*Attribute Type is Pricing*/

        p_req_line_attrs_tbl(i).context = 'MIXED'
AND

        p_req_line_attrs_tbl(i).attribute = 'PRICING_ATTRIBUTE4'
THEN
        --For this combination of Pricing Context and Attribute, let's say, the
        Attribute Value is the Weight of

        --the item to which the formula is attached.
        v_weight := p_req_line_attrs_tbl(i).value;

        END IF;

    END LOOP; /*For Loop*/

    RETURN v_weight;

EXCEPTION

    WHEN OTHERS THEN

    RETURN NULL;

END Get_Custom_Price;

END QP_CUSTOM;
```

Then if v_weight has a value '1.2' then Get_Custom_Price returns a value of

`'1.2'`

Therefore, the pricing engine evaluates the formula as $200 * 1.2 = 240$.

See

Oracle Pricing Technical Reference Manual

Get Price List Application Program Interface

This section explains how to use the `Get_Price_List` API and how it functions in Oracle Advanced Pricing.

Get Price List API Features

The Get Price List API is used to get all the price lists for a given currency code and/or agreement id. The package `QP_UTIL_PUB` contains the procedure `Get_Price_List`.

Functional Overview

While processing an order using a particular currency code and/or agreement id, the OM displays a valid set of price lists when user activates the LOV. The OM calls this API to get the valid price lists for the given currency code, and/or agreement id and pricing effective date. If pricing effective date is not passed by calling application, the current date is defaulted. If profile QP: Multi Currency Installed is 'Y' then price lists are retrieved by joining the view `qp_list_headers_vl` and table `qp_currency_details` else only view `qp_list_headers_vl` is used.

Setting Up and Parameter Descriptions

The following chart describes all parameters used by the `Get_Price_List` API. All of the inbound and outbound parameters are listed.

Table 5–90 *Get_Price_List*

Parameter	Usage	Type	Req	Drv
<code>l_currency_code</code>	IN	Varchar2	Yes	No
<code>l_pricing_effective_date</code>	IN	Date	No	No
<code>l_agreement_id</code>	IN	Number	No	No
<code>l_price_list_tbl</code>	OUT	<code>price_list_tbl</code>	No	No

PRICE_LIST_REC

The following table shows the parameters for this structure.

Table 5–91 PRICE_LIST_REC

Parameter	Usage	Type	Req	Drv
price_list_id	Null	Number	Yes	No
name	Null	Varchar2	Yes	No
Description	Null	Varchar2	Yes	No
start_date_active	Null	Date	No	No
end_date_active	Null	Date	No	No

PRICE_LIST_TBL

The following table shows the parameters for this structure.

Table 5–92 PRICE_LIST_TBL

Parameter	Usage	Type	Req	Drv
price_list_rec	Null	Record	No	No

Validation of Get_Price_List API**Standard Validation**

The caller is responsible for passing the right parameters to Get_Price_List.

For specific information on the data implied by these columns, see: Oracle Pricing Technical Reference Manual for details.

Other Validation

None.

Error Handling

If any exception occurs in the Get_Price_List API, it does not return any price lists.

NOTE: The Package Specification and Body files are QPXRTCNS.pls and QPXRTCNB.pls and are available under the source control directory \$QP_TOP/patch/115/sql.

Multi-Currency Conversion Setup Application Program Interface

This section explains how to use the Multi-Currency Conversion Setup API and how it functions in Oracle Advanced Pricing. The Multi-Currency Conversion Setup package consists of entities to set up Multi-Currency Conversion.

Functional Overview

The Multi-Currency Conversion Setup package QP_Currency_PUB.Process_Currency contains the following public record type and table of records entities:

- **Process_Currency:** QP_Currency_PUB.Process_Currency: Takes two record types and two table types as input parameters. Use this API to insert, and update Multi-Currency Conversion and to set up a Multi-Currency Conversion for a given p_CURR_LISTS_rec record structure. The Multi-Currency Conversion can not be deleted but it can be inactivated by setting the effective dates.

You can:

- Set up multiple multi-currency conversion lines by giving multiple multi-currency conversion line definitions in the p_CURR_DETAILS_tbl table structure.
- **Curr_Lists_Rec_Type:** Corresponds to the columns in the multi-currency header tables QP_CURRENCY_LISTS_B and QP_CURRENCY_LISTS_TL.
- **Curr_Lists_Val_Rec_Type:** Attributes that store the meaning of id or code columns in the multi-currency header table QP_CURRENCY_LISTS_B, for example, Base Currency.
- **Curr_Details_Rec_Type:** Corresponds to columns in the multi-currency conversion line table QP_CURRENCY_DETAILS.
- **Curr_Details_Tbl_Type:** Table of Curr_Details_Rec_Type.
- **Curr_Details_Val_Rec_Type:** Attributes that store the meaning of id or code columns in the multi-currency conversion line table QP_CURRENCY_DETAILS, for example, Markup_Formula.
- **Curr_Details_Val_Tbl_Type:** Table of Curr_Details_Val_Rec_Type.

Setting Up and Parameter Descriptions

The following chart describes all parameters used by the public Multi-Currency Conversion Setup. All of the inbound and outbound parameters are listed. Additional information on these parameters may follow.

PROCESS_CURRENCY

The following table shows the parameters for this structure.

Table 5–93 *PROCESS_CURRENCY*

Parameter	Usage	Type	Req	Drv
p_api_version_number	In	Number	No	No
p_init_msg_list	In	Varchar2	No	No
p_return_values	In	Varchar2	No	No
p_commit	In	Varchar2	No	No
x_return_status	Out	Varchar2	No	No
x_msg_count	Out	Number	No	No
x_msg_data	Out	Varchar2	No	No
p_CURR_LISTS_rec	In	Curr_Lists_Rec_Type	No	No
p_CURR_LISTS_val_rec	In	Curr_Lists_Val_Rec_Type	No	No
p_CURR_DETAILS_tbl	In	Curr_Details_Tbl_Type	No	No
p_CURR_DETAILS_val_tbl	In	Curr_Details_Val_Tbl_Type	No	No
x_CURR_LISTS_rec	Out	Curr_Lists_Rec_Type	No	No
x_CURR_LISTS_val_rec	Out	Curr_Lists_Val_Rec_Type	No	No
x_CURR_DETAILS_tbl	Out	Curr_Details_Tbl_Type	No	No
x_CURR_DETAILS_val_tbl	Out	Curr_Details_Val_Tbl_Type	No	No

CURR_LISTS_REC_TYPE

The following table shows the parameters for this structure.

Table 5–94 CURR_LISTS_REC_TYPE

Parameter	Usage	Type	Req	Drv
attribute1	Null	Varchar2	No	No
attribute2	Null	Varchar2	No	No
attribute3	Null	Varchar2	No	No
attribute4	Null	Varchar2	No	No
attribute5	Null	Varchar2	No	No
attribute6	Null	Varchar2	No	No
attribute7	Null	Varchar2	No	No
attribute8	Null	Varchar2	No	No
attribute9	Null	Varchar2	No	No
attribute10	Null	Varchar2	No	No
attribute11	Null	Varchar2	No	No
attribute12	Null	Varchar2	No	No
attribute13	Null	Varchar2	No	No
attribute14	Null	Varchar2	No	No
attribute15	Null	Varchar2	No	No
base_currency_code	Null	Varchar2	Yes	No
Context	Null	Varchar2	No	No
created_by	Null	Number	No	No
creation_date	Null	Date	No	No
currency_header_id	Null	Number	No	No
Description	Null	Varchar2	No	No
last_updated_by	Null	Number	No	No
last_update_date	Null	Date	No	No
last_update_login	Null	Number	No	No
Name	Null	Varchar2	No	No

Table 5–94 CURREN_LISTS_REC_TYPE

Parameter	Usage	Type	Req	Drv
base_rounding_factor	Null	Number	No	No
base_markup_formula_id	Null	Number	No	No
base_markup_operator	Null	Varchar2	No	No
base_markup_value	Null	Number	No	No
program_application_id	Null	Number	No	No
program_id	Null	Number	No	No
program_update_date	Null	Date	No	No
prorate_flag	Null	Varchar2	No	No
request_id	Null	Number	No	No
return_status	Null	Varchar2	No	No
db_flag	Null	Varchar2	No	No
Operation	Null	Varchar2	Yes	No

CURREN_LISTS_TBL_TYPE

The following table shows the parameters for this structure.

Table 5–95 CURREN_LISTS_TBL_TYPE

Parameter	Usage	Type	Req	Drv
Curren_Lists_Rec_Type	Null	Record	No	No

CURREN_LISTS_VAL_REC_TYPE

The following table shows the parameters for this structure.

Table 5–96 CURREN_LISTS_VAL_REC_TYPE

Parameter	Usage	Type	Req	Drv
base_currency	Null	Varchar2	No	No
currency_header	Null	Varchar2	No	No
base_markup_formula	Null	Varchar2	No	No

CURR_LISTS_VAL_TBL_TYPE

The following table shows the parameters for this structure.

Table 5-97 CURR_LISTS_VAL_TBL_TYPE

Parameter	Usage	Type	Req	Drv
Curr_Lists_Val_Rec_Type	Null	Record	No	No

CURR_DETAILS_REC_TYPE

The following table shows the parameters for this structure.

Table 5-98 CURR_DETAILS_REC_TYPE

Parameter	Usage	Type	Req	Drv
attribute1	Null	Varchar2	No	No
attribute2	Null	Varchar2	No	No
attribute3	Null	Varchar2	No	No
attribute4	Null	Varchar2	No	No
attribute5	Null	Varchar2	No	No
attribute6	Null	Varchar2	No	No
attribute7	Null	Varchar2	No	No
attribute8	Null	Varchar2	No	No
attribute9	Null	Varchar2	No	No
attribute10	Null	Varchar2	No	No
attribute11	Null	Varchar2	No	No
attribute12	Null	Varchar2	No	No
attribute13	Null	Varchar2	No	No
attribute14	Null	Varchar2	No	No
attribute15	Null	Varchar2	No	No
Context	Null	Varchar2	No	No
conversion_date	Null	Date	No	No
conversion_date_type	Null	Varchar2	No	No

Table 5–98 CURRENDETAILS_REC_TYPE

Parameter	Usage	Type	Req	Drv
conversion_type	Null	Varchar2	Yes	No
created_by	Null	Number	No	No
creation_date	Null	Date	No	No
currency_detail_id	Null	Number	No	No
currency_header_id	Null	Number	No	No
end_date_active	Null	Date	No	No
fixed_value	Null	Number	No	No
last_updated_by	Null	Number	No	No
last_update_date	Null	Date	No	No
last_update_login	Null	Number	No	No
markup_formula_id	Null	Number	No	No
markup_operator	Null	Varchar2	No	No
markup_value	Null	Number	No	No
price_formula_id	Null	Number	No	No
program_application_id	Null	Number	No	No
program_id	Null	Number	No	No
program_update_date	Null	Date	No	No
request_id	Null	Number	No	No
rounding_factor	Null	Number	No	No
selling_rounding_factor	Null	Number	No	No
start_date_active	Null	Date	No	No
to_currency_code	Null	Varchar2	Yes	No
curr_attribute_type	Null	Varchar2	No	No
curr_attribute_context	Null	Varchar2	No	No
curr_attribute	Null	Varchar2	No	No
curr_attribute_value	Null	Varchar2	No	No
Precedence	Null	Number	No	No

Table 5–98 CURRENDETAILS_REC_TYPE

Parameter	Usage	Type	Req	Drv
return_status	Null	Varchar2	No	No
db_flag	Null	Varchar2	No	No
operation	Null	Varchar2	No	No

CURRENDETAILS_TBL_TYPE

The following table shows the parameters for this structure.

Table 5–99 CURRENDETAILS_TBL_TYPE

Parameter	Usage	Type	Req	Drv
Curr_Details_Rec_Type	Null	Record	No	No

CURRENDETAILS_VAL_REC_TYPE

The following table shows the parameters for this structure.

Table 5–100 CURRENDETAILS_VAL_REC_TYPE

Parameter	Usage	Type	Req	Drv
currency_detail	Null	Varchar2	No	No
currency_header	Null	Varchar2	No	No
markup_formula	Null	Varchar2	No	No
price_formula	Null	Varchar2	No	No
to_currency	Null	Varchar2	No	No

CURRENDETAILS_VAL_TBL_TYPE

The following table shows the parameters for this structure.

Table 5–101 CURRENDETAILS_VAL_TBL_TYPE

Parameter	Usage	Type	Req	Drv
Curr_Details_Val_Rec_Type	Null	Record	No	No

Validation of Multi-Currency Conversion API

Standard Validation

Oracle Advanced Pricing validates all required columns in the Multi-Currency Conversion API. For more information, see: Oracle Pricing Technical Reference Manual.

Other Validation

None

Error Handling

If any validation fails, the API will return error status to the calling module. The Multi-Currency Conversion API processes the rows and reports the values in the following table for every record.

Table 5–102 Error Handling

Condition	PROCESS_STATUS	ERROR_MESSAGE
Success	5	null
Failure	4	actual error message

Example of Multi-Currency Conversion API

The following code can also be found in file QPXEXCUR.sql in \$QP_TOP/patch/115/sql directory

```
--Multi-Currency Conversion Creation - USD to FRF and GBP
REM FILETYPE : NOEXEC

REM Added for ARU db drv auto generation
REM dbdrv: none

SET VERIFY OFF
WHenever SQLERROR EXIT FAILURE ROLLBACK;
--set serveroutput on
declare
/* $Header: QPXEXCUR.sql 115.0 2002/05/22 00:38:04 mkarya noship $ */

l_return_status          VARCHAR2(1);
x_msg_count              number;
x_msg_data               Varchar2(2000);
```

```

x_msg_index                number;

l_CURR_LISTS_rec            QP_Currency_PUB.Curr_Lists_Rec_Type;
l_CURR_LISTS_val_rec        QP_Currency_PUB.Curr_Lists_Val_Rec_Type;
l_CURR_DETAILS_tbl          QP_Currency_PUB.Curr_Details_Tbl_Type;
l_CURR_DETAILS_val_tbl      QP_Currency_PUB.Curr_Details_Val_Tbl_Type;

Begin

-- Create a Multi-Currency Conversion List for Base Currency USD

    l_CURR_LISTS_rec.base_currency_code      := 'USD';
    l_CURR_LISTS_rec.name                    := 'New Corporate Multi-Currency
Conversion';
    l_CURR_LISTS_rec.description              := 'Corporate Multi-Currency
Conversion Description';
    l_CURR_LISTS_rec.base_rounding_factor     := -2;
    l_CURR_LISTS_rec.base_markup_operator     := '%';
    l_CURR_LISTS_rec.base_markup_value       := 5;
    l_CURR_LISTS_rec.operation                := QP_GLOBALS.G_OPR_CREATE;
-- Create Multi-Currency Details for Currency FRF and AUD
l_CURR_DETAILS_tbl(1).to_currency_code := 'FRF';
l_CURR_DETAILS_tbl(1).conversion_type := 'FIXED';
l_CURR_DETAILS_tbl(1).fixed_value := 2.2;
l_CURR_DETAILS_tbl(1).curr_attribute_type := 'QUALIFIER';
l_CURR_DETAILS_tbl(1).curr_attribute_context := 'CUSTOMER';
l_CURR_DETAILS_tbl(1).curr_attribute := 'QUALIFIER_ATTRIBUTE2';
l_CURR_DETAILS_tbl(1).curr_attribute_value := '1005';
l_CURR_DETAILS_tbl(1).precedence := 101;
l_CURR_DETAILS_tbl(1).markup_operator := '%';
l_CURR_DETAILS_tbl(1).markup_value := 6;
l_CURR_DETAILS_tbl(1).selling_rounding_factor := -2;
l_CURR_DETAILS_tbl(1).operation := QP_GLOBALS.G_OPR_CREATE;

l_CURR_DETAILS_tbl(2).to_currency_code := 'AUD';
l_CURR_DETAILS_tbl(2).conversion_type := 'Corporate';
l_CURR_DETAILS_tbl(2).conversion_date_type := 'FIXED';
l_CURR_DETAILS_tbl(2).conversion_date := sysdate;
l_CURR_DETAILS_tbl(2).markup_operator := '%';
l_CURR_DETAILS_tbl(2).markup_value := 7;
l_CURR_DETAILS_tbl(2).selling_rounding_factor := -2;
l_CURR_DETAILS_tbl(2).operation := QP_GLOBALS.G_OPR_CREATE;

-- Call the Multi-Currency Conversion Public API to create the header and lines

```

```
QP_Currency_PUB.Process_Currency
( p_api_version_number => 1.0
, p_init_msg_list      => FND_API.G_FALSE
, p_return_values      => FND_API.G_FALSE
, p_commit             => FND_API.G_FALSE
, x_return_status      => l_return_status
, x_msg_count          => x_msg_count
, x_msg_data           => x_msg_data
, p_CURR_LISTS_rec     => l_CURR_LISTS_rec
, p_CURR_LISTS_val_rec => l_CURR_LISTS_val_rec
, p_CURR_DETAILS_tbl   => l_CURR_DETAILS_tbl
, p_CURR_DETAILS_val_tbl => l_CURR_DETAILS_val_tbl
, x_CURR_LISTS_rec     => l_CURR_LISTS_rec
, x_CURR_LISTS_val_rec => l_CURR_LISTS_val_rec
, x_CURR_DETAILS_tbl   => l_CURR_DETAILS_tbl
, x_CURR_DETAILS_val_tbl => l_CURR_DETAILS_val_tbl
);

IF l_return_status <> FND_API.G_RET_STS_SUCCESS THEN

    RAISE FND_API.G_EXC_UNEXPECTED_ERROR;

END IF;

EXCEPTION

WHEN FND_API.G_EXC_ERROR THEN

    l_return_status := FND_API.G_RET_STS_ERROR;

    -- Get message count and data

    --dbms_output.put_line('err msg is : ' || x_msg_data);

WHEN FND_API.G_EXC_UNEXPECTED_ERROR THEN

    l_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;

    for k in 1 .. x_msg_count loop
        x_msg_data := oe_msg_pub.get( p_msg_index => k,
        p_encoded => 'F'

    );
```

```
--Get message count and data
--dbms_output.put_line('err msg ' || k || 'is: ' || x_msg_data);

        end loop;

    WHEN OTHERS THEN

        l_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;

        for k in 1 .. x_msg_count loop
            x_msg_data := oe_msg_pub.get( p_msg_index => k,
                p_encoded => 'F'
            );
        --Get message count and data
        --dbms_output.put_line('err msg ' || k || 'is: ' || x_msg_data);

        end loop;
END;
/
COMMIT;
EXIT;
```

Attaching a multi-currency conversion to a Price List

The multi-currency conversion can be attached to a Price List by passing the currency header id (of the multi-currency conversion to be attached) to the record QP_Price_List_PUB.Price_List_Rec_Type while using the Application Program Interface (QP_Price_List_PUB.Process_Price_List()) for Price List. For more information, see: [Round Price Application Program Interface](#) on page 5-333.

Attaching a multi-currency conversion to 'AGR' agreement

The multi-currency conversion can be attached to a Agreement by passing the currency header id (of the multi-currency conversion to be attached) to the record QP_PRICE_LIST_PUB.Price_List_Rec_Type while using the Application Program Interface (OE_Pricing_Cont_PUB.Process_Agreement()) for Agreement. For more information, see: [Agreements Public Application Program Interface](#) on page 5-3.

Defaulting a multi-currency conversion to a 'STANDARD' agreement from the price list

The multi-currency conversion of price list gets defaulted while creating an agreement of type 'STANDARD'. No extra steps are needed. Just specify the price list id, as used to be, while creating an agreement of type 'STANDARD' using the Application Program Interface (OE_Pricing_Cont_PUB.Process_Agreement ()) for Agreement. For more information, see: [Agreements Public Application Program Interface](#) on page 5-3.

Price List Setup Application Program Interface

This section explains how to use the Price List Setup API and how it functions in Oracle Advanced Pricing. The Price List Setup package consists of entities to set up price lists.

Functional Overview

The Price List Setup package `QP_Price_List_PUB.Process_Price_List` contains the following public record type and table of records entities:

- `Process_Price_List`: `QP_Price_List_PUB.Process_Price_List`. Takes two record types and six table types as input parameters. Use this API to insert, update, and delete price lists and to set up a price list for a given `p_PRICE_LIST_rec` record structure.

You can:

- Set up multiple price list lines by giving multiple price list line definitions in the `p_PRICE_LIST_LINE_tbl` table structure.
- Attach multiple qualifiers at the price list header level by giving multiple qualifiers in the `p_QUALIFIERS_tbl` table structure.
- Attach multiple pricing attributes to price list lines by giving the pricing attributes in the `p_PRICING_ATTR_tbl` table structure.
- `Price_List_Rec_Type`: Corresponds to the columns in the price list header tables `QP_LIST_HEADERS_B` and `QP_LIST_HEADERS_TL`.
- `Price_List_Val_Rec_Type`: Attributes that store the meaning of id or code columns in the price list header table `QP_LIST_HEADERS_B`, for example, Currency.
- `Price_List_Line_Rec_Type`: Corresponds to columns in the price list line table and related modifiers tables `QP_LIST_LINES` and `QP_RLTD_MODIFIERS`.
- `Price_List_Line_Tbl_Type`: Table of `Price_List_Line_Rec_Type`.
- `Price_List_Line_Val_Rec_Type`: Attributes that store the meaning of id or code columns in the price list line table `QP_LIST_LINES`, for example, `Price_By_Formula`.
- `Price_List_Line_Val_Tbl_Type`: Table of `Price_List_Line_Val_Rec_Type`.
- `Qualifiers_Rec_Type`: Corresponds to the columns in the qualifier table `QP_QUALIFIERS`.

- **Qualifiers_Tbl_Type:** Table of Qualifiers_Rec_Type.
- **Qualifiers_Val_Rec_Type:** Made up of attributes that store the meaning of id or code columns in the qualifiers table QP_QUALIFIERS, for example, Qualifier_Rule.
- **Qualifiers_Val_Tbl_Type:** Table of Qualifiers_Val_Rec_Type.
- **Pricing_Attr_Rec_Type:** Corresponds to the columns in the pricing attributes table QP_PRICING_ATTRIBUTES.
- **Pricing_Attr_Tbl_Type:** Table of Pricing_Attr_Rec_Type.
- **Pricing_Attr_Val_Rec_Type:** Attributes that store the meaning of id or code columns in the pricing attributes table QP_PRICING_ATTRIBUTES, for example, Accumulate.
- **Pricing_Attr_Val_Tbl_Type:** Table of Pricing_Attr_Val_Rec_Type.

Setting Up and Parameter Descriptions

The following chart describes all parameters used by the public Price List Setup. All of the inbound and outbound parameters are listed. Additional information on these parameters may follow.

PROCESS_PRICE_LIST

The following table shows the parameters for this structure.

Table 5–103 PROCESS_PRICE_LIST

Parameter	Usage	Type	Req	Drv
p_api_version_number	In	Number	No	No
p_init_msg_list	In	Varchar2	No	No
p_return_values	In	Varchar2	No	No
p_commit	In	Varchar2	No	No
x_return_status	Out	Varchar2	No	No
x_msg_count	Out	Varchar2	No	No
x_msg_data	Out	Varchar2	No	No
p_PRICE_LIST_rec	In	Price_List_Rec_Type	No	No
p_PRICE_LIST_val_rec	In	Price_List_Val_Rec_Type	No	No

Table 5–103 *PROCESS_PRICE_LIST*

Parameter	Usage	Type	Req	Drv
p_PRICE_LIST_tbl	In	Price_List_Line_tbl_type	No	No
p_PRICE_LIST_val_tbl	In	Price_List_Line_Val_tbl_type	No	No
p_QUALIFIERS_tbl	In	QP_Qualifier_Rules_PUB.Qualifiers_tbl_Type	No	No
p_QUALIFIERS_val_tbl	In	QP_Qualifier_Rules_PUB.Qualifiers_val_tbl_Type	No	No
p_PRICING_ATTR_tbl	In	Pricing_Attr_tbl_type	No	No
p_PRICING_ATTR_val_tbl	In	Pricing_Attr_val_tbl_type	No	No
x_PRICE_LIST_rec	Out	Price_List_Rec_Type	No	No
x_PRICE_LIST_val_rec	Out	Price_List_Val_Rec_Type	No	No
x_PRICE_LIST_LINE_tbl	Out	Price_List_Line_tbl_type	No	No
x_PRICE_LIST_LINE_val_tbl	Out	Price_List_Line_Val_tbl_type	No	No
x_QUALIFIERS_tbl	Out	QP_Qualifier_Rules_PUB.Qualifiers_tbl_Type	No	No
x_QUALIFIERS_val_tbl	Out	QP_Qualifier_Rules_PUB.Qualifiers_val_tbl_Type	No	No
x_PRICING_ATTR_tbl	Out	Pricing_Attr_tbl_type	No	No
x_PRICING_ATTR_val_tbl	In	Pricing_Attr_val_tbl_type	No	No

PRICE_LIST_REC_TYPE

The following table shows the parameters for this structure.

Table 5–104 *PRICE_LIST_REC_TYPE*

Parameter	Usage	Type	Req	Drv
attribute1	Null	Varchar2	No	No
attribute2	Null	Varchar2	No	No
attribute3	Null	Varchar2	No	No
attribute4	Null	Varchar2	No	No
attribute5	Null	Varchar2	No	No

Table 5–104 PRICE_LIST_REC_TYPE

Parameter	Usage	Type	Req	Drv
attribute6	Null	Varchar2	No	No
attribute7	Null	Varchar2	No	No
attribute8	Null	Varchar2	No	No
attribute9	Null	Varchar2	No	No
attribute10	Null	Varchar2	No	No
attribute11	Null	Varchar2	No	No
attribute12	Null	Varchar2	No	No
attribute13	Null	Varchar2	No	No
attribute14	Null	Varchar2	No	No
attribute15	Null	Varchar2	No	No
automatic_flag	Null	Varchar2	No	No
comments	Null	Varchar2	No	No
context	Null	Varchar2	No	No
created_by	Null	Number	No	No
creation_date	Null	Date	No	No
currency_code	Null	Varchar2	Yes	No
discount_lines_flag	Null	Varchar2	No	No
end_date_active	Null	Date	No	No
freight_terms_code	Null	Varchar2	No	No
global_flag	Null	Varchar2	No	No
gsa_indicator	Null	Varchar2	No	No
last_updated_by	Null	Number	No	No
last_update_date	Null	Date	No	No
last_update_login	Null	Number	No	No
list_header_id	Null	Number	No	No
list_source_code	Null	Varchar2	No	No
list_type_code	Null	Varchar2	No	No

Table 5–104 PRICE_LIST_REC_TYPE

Parameter	Usage	Type	Req	Drv
orig_org_id	Null	Number	No	No
orig_system_header_ref	Null	Varchar2	No	No
program_application_id	Null	Number	No	No
program_id	Null	Number	No	No
program_update_date	Null	Date	No	No
prorate_flag	Null	Varchar2	No	No
request_id	Null	Number	No	No
rounding_factor	Null	Number	No	No
ship_method_code	Null	Varchar2	No	No
start_date_active	Null	Date	No	No
terms_id	Null	Number	No	No
return_status	Null	Varchar2	No	No
db_flag	Null	Varchar2	No	No
operation	Null	Varchar2	Yes	No
name	Null	Varchar2	Yes	No
description	Null	Varchar2	No	No
version_no	Null	Varchar2	No	No
Active_flag (Default value is Yes.)	Null	Varchar2	No	No
mobile_download	Null	Varchar2	No	No
currency_header_id	Null	Number	No	No

PRICE_LIST_TBL_TYPE

The following table shows the parameters for this structure.

Table 5–105 PRICE_LIST_TBL_TYPE

Parameter	Usage	Type	Req	Drv
Price_List_Rec_Type	Null	Record	No	No

PRICE_LIST_VAL_REC_TYPE

The following table shows the parameters for this structure.

Table 5–106 PRICE_LIST_VAL_REC_TYPE

Parameter	Usage	Type	Req	Drv
automatic	Null	Varchar2	No	No
currency	Null	Varchar2	No	No
discount_lines	Null	Varchar2	No	No
freight_terms	Null	Varchar2	No	No
list_header	Null	Varchar2	No	No
list_type	Null	Varchar2	No	No
prorate	Null	Varchar2	No	No
ship_method	Null	Varchar2	No	No
terms	Null	Varchar2	No	No
currency_header	Null	Varchar2	No	No

PRICE_LIST_VAL_TBL_TYPE

The following table shows the parameters for this structure.

Table 5–107 PRICE_LIST_VAL_TBL_TYPE

Parameter	Usage	Type	Req	Drv
Price_List_Val_Rec_Type	Null	Record	No	No

PRICE_LIST_LINE_REC_TYPE

The following table shows the parameters for this structure.

Table 5–108 PRICE_LIST_LINE_REC_TYPE

Parameter	Usage	Type	Req	Drv
accrual_qty	Null	Number	No	No
accrual_uom_code	Null	Varchar2	No	No
arithmetic_operator	Null	Varchar2	No	No

Table 5–108 PRICE_LIST_LINE_REC_TYPE

Parameter	Usage	Type	Req	Drv
attribute1	Null	Varchar2	No	No
attribute2	Null	Varchar2	No	No
attribute3	Null	Varchar2	No	No
attribute4	Null	Varchar2	No	No
attribute5	Null	Varchar2	No	No
attribute6	Null	Varchar2	No	No
attribute7	Null	Varchar2	No	No
attribute8	Null	Varchar2	No	No
attribute9	Null	Varchar2	No	No
attribute10	Null	Varchar2	No	No
attribute11	Null	Varchar2	No	No
attribute12	Null	Varchar2	No	No
attribute13	Null	Varchar2	No	No
attribute14	Null	Varchar2	No	No
attribute15	Null	Varchar2	No	No
automatic_flag	Null	Varchar2	No	No
base_qty	Null	Number	No	No
base_uom_code	Null	Varchar2	No	No
comments	Null	Varchar2	No	No
context	Null	Varchar2	No	No
created_by	Null	Number	No	No
creation_date	Null	Date	No	No
effective_period_uom	Null	Varchar2	No	No
end_date_active	Null	Date	No	No
estim_accrual_rate	Null	Number	No	No
generate_using_formula_id	Null	Number	No	No

Table 5–108 PRICE_LIST_LINE_REC_TYPE

Parameter	Usage	Type	Req	Drv
inventory_item_id	Null	Number	No	No
last_updated_by	Null	Number	No	No
last_update_date	Null	Date	No	No
last_update_login	Null	Number	No	No
list_header_id	Null	Number	No	No
list_line_id	Null	Number	No	No
list_line_type_code	Null	Varchar2	No	No
list_price	Null	Number	No	No
modifier_level_code	Null	Varchar2	No	No
number_effective_periods	Null	Number	No	No
operand	Null	Number	No	No
organization_id	Null	Number	No	No
override_flag	Null	Varchar2	No	No
percent_price	Null	Number	No	No
price_break_type_code	Null	Varchar2	No	No
price_by_formula_id	Null	Number	No	No
primary_uom_flag	Null	Varchar2	No	No
print_on_invoice_flag	Null	Varchar2	No	No
program_application_id	Null	Number	No	No
program_id	Null	Number	No	No
program_update_date	Null	Date	No	No
rebate_trxn_type_code	Null	Varchar2	No	No
related_item_id	Null	Number	No	No
relationship_type_id	Null	Number	No	No
reprice_flag	Null	Varchar2	No	No
request_id	Null	Number	No	No
revision	Null	Varchar2	No	No

Table 5–108 PRICE_LIST_LINE_REC_TYPE

Parameter	Usage	Type	Req	Drv
revision_date	Null	Date	No	No
revision_reason_code	Null	Varchar2	No	No
start_date_active	Null	Date	No	No
substitution_attribute	Null	Varchar2	No	No
substitution_context	Null	Varchar2	No	No
substitution_value	Null	Varchar2	No	No
return_status	Null	Varchar2	No	No
db_flag	Null	Varchar2	No	No
operation	Null	Varchar2	No	No
from_rltd_modifier_id	Null	Number	No	No
rltd_modifier_group_no	Null	Number	No	No
product_precedence	Null	Number	No	No

PRICE_LIST_LINE_TBL_TYPE

The following table shows the parameters for this structure.

Table 5–109 PRICE_LIST_LINE_TBL_TYPE

Parameter	Usage	Type	Req	Drv
Price_List_Line_Rec_Type	Null	Record	No	No

PRICE_LIST_LINE_VAL_REC_TYPE

The following table shows the parameters for this structure.

Table 5–110 PRICE_LIST_LINE_VAL_REC_TYPE

Parameter	Usage	Type	Req	Drv
accrual_uom	Null	Varchar2	No	No
automatic	Null	Varchar2	No	No
base_uom	Null	Varchar2	No	No

Table 5–110 PRICE_LIST_LINE_VAL_REC_TYPE

Parameter	Usage	Type	Req	Drv
generate_using_formula	Null	Varchar2	No	No
inventory_item	Null	Varchar2	No	No
list_header	Null	Varchar2	No	No
list_line	Null	Varchar2	No	No
list_line_type	Null	Varchar2	No	No
modifier_level	Null	Varchar2	No	No
organization	Null	Varchar2	No	No
override	Null	Varchar2	No	No
price_break_type	Null	Varchar2	No	No
price_by_formula	Null	Varchar2	No	No
primary_uom	Null	Varchar2	No	No
print_on_invoice	Null	Varchar2	No	No
rebate_transaction_type	Null	Varchar2	No	No
related_item	Null	Varchar2	No	No
relationship_type	Null	Varchar2	No	No
reprice	Null	Varchar2	No	No
revision_reason	Null	Varchar2	No	No

PRICE_LIST_VAL_TBL_TYPE

The following table shows the parameters for this structure.

Table 5–111 PRICE_LIST_VAL_TBL_TYPE

Parameter	Usage	Type	Req	Drv
Price_List_Line_Val_Rec_Type	Null	Record	No	No

QUALIFIERS_REC_TYPE

The following table shows the parameters for this structure.

Table 5–112 QUALIFIERS_REC_TYPE

Parameter	Usage	Type	Req	Drv
QP_Qualifier_Rules_PUB.Qualifiers_Rec_Type	Null	Record	No	No

Refer to the Qualifiers public API for the definition.

QUALIFIERS_TBL_TYPE

The following table shows the parameters for this structure.

Table 5–113 QUALIFIERS_TBL_TYPE

Parameter	Usage	Type	Req	Drv
QP_Qualifier_Rules_PUB.Qualifiers_Tbl_Type	Null	Record	No	No

Refer to the Qualifiers public API for the definition.

QUALIFIERS_VAL_REC_TYPE

The following table shows the parameters for this structure.

Table 5–114 QUALIFIERS_VAL_REC_TYPE

Parameter	Usage	Type	Req	Drv
QP_Qualifier_Rules_PUB.Qualifiers_Val_Rec_Type	Null	Record	No	No

Refer to the Qualifiers public API for the definition.

QUALIFIERS_VAL_TBL_TYPE

The following table shows the parameters for this structure.

Table 5–115 QUALIFIERS_VAL_TBL_TYPE

Parameter	Usage	Type	Req	Drv
QP_Qualifier_Rules_PUB.Qualifiers_Val_Tbl_Type	Null	Record	No	No

Refer to the Qualifiers public API for the definition.

Note: For setting up a Secondary Price List, create a qualifier with the following parameters:

QUALIFIER_CONTEXT - 'MODLIST'

QUALIFIER_ATTRIBUTE - 'QUALIFIER_ATTRIBUTE4'

QUALIFIER_ATTR_VALUE - <list_header_id of the primary price list>

LIST_HEADER_ID - <list_header_id of the secondary price list>

COMPARISON_OPERATOR_CODE - '='.

See Example 2 for the details about how to set up a Secondary Price List.

PRICING_ATTR_REC_TYPE

The following table shows the parameters for this structure.

Table 5–116 PRICING_ATTR_REC_TYPE

Parameter	Usage	Type	Req	Drv
accumulate_flag	Null	Varchar2	No	No
attribute1	Null	Varchar2	No	No
attribute2	Null	Varchar2	No	No
attribute3	Null	Varchar2	No	No
attribute4	Null	Varchar2	No	No
attribute5	Null	Varchar2	No	No
attribute6	Null	Varchar2	No	No
attribute7	Null	Varchar2	No	No
attribute8	Null	Varchar2	No	No

Table 5–116 PRICING_ATTR_REC_TYPE

Parameter	Usage	Type	Req	Drv
attribute9	Null	Varchar2	No	No
attribute10	Null	Varchar2	No	No
attribute11	Null	Varchar2	No	No
attribute12	Null	Varchar2	No	No
attribute13	Null	Varchar2	No	No
attribute14	Null	Varchar2	No	No
attribute15	Null	Varchar2	No	No
attribute_grouping_number	Null	Number	No	No
context	Null	Varchar2	No	No
created_by	Null	Number	No	No
creation_date	Null	Date	No	No
excluder_flag	Null	Varchar2	No	No
last_updated_by	Null	Number	No	No
last_update_date	Null	Date	No	No
last_update_login	Null	Number	No	No
list_line_id	Null	Number	No	No
pricing_attribute	Null	Varchar2	No	No
pricing_attribute_context	Null	Varchar2	No	No
pricing_attribute_id	Null	Number	No	No
pricing_attr_value_from	Null	Varchar2	No	No
pricing_attr_value_to	Null	Varchar2	No	No
product_attribute	Null	Varchar2	Yes	No
product_attribute_context	Null	Varchar2	Yes	No
product_attr_value	Null	Varchar2	Yes	No
product_uom_code	Null	Varchar2	Yes	No
program_application_id	Null	Number	No	No
program_id	Null	Number	No	No

Table 5–116 PRICING_ATTR_REC_TYPE

Parameter	Usage	Type	Req	Drv
program_update_date	Null	Date	No	No
request_id	Null	Number	No	No
return_status	Null	Varchar2	No	No
db_flag	Null	Varchar2	No	No
operation	Null	Varchar2	Yes	No
PRICE_LIST_LINE_index	Null	Number	No	No
from_rltd_modifier_id	Null	Number	No	No
comparison_operator_code	Null	Varchar2	Yes	No
product_attribute_datatype	Null	Varchar2	No	Yes
pricing_attribute_datatype	Null	Varchar2	No	Yes

PRICING_ATTR_TBL_TYPE

The following table shows the parameters for this structure.

Table 5–117 PRICING_ATTR_TBL_TYPE

Parameter	Usage	Type	Req	Drv
Pricing_Attr_Rec_Type	Null	Record	No	No

PRICING_ATTR_VAL_REC_TYPE

The following table shows the parameters for this structure.

Table 5–118 PRICING_ATTR_VAL_REC_TYPE

Parameter	Usage	Type	Req	Drv
accumulate	Null	Varchar2	No	No
excluder	Null	Varchar2	No	No
list_line	Null	Varchar2	No	No
pricing_attribute	Null	Varchar2	No	No
product_uom	Null	Varchar2	No	No

PRICING_ATTR_VAL_TBL_TYPE

The following table shows the parameters for this structure.

Table 5–119 PRICING_ATTR_VAL_TBL_TYPE

Parameter	Usage	Type	Req	Drv
Pricing_Attr_Val_Rec_Type	Null	Record	No	No

Validation of Price List Setup API**Standard Validation**

Oracle Advanced Pricing validates all required columns in the Price List Setup API. For more information, see: *Oracle Pricing Technical Reference Manual*.

Other Validation

None

Error Handling

If any validation fails, the API will return error status to the calling module. The Price List Setup API processes the rows and reports the values in the following table for every record.

Table 5–120 Error Handling

Condition	PROCESS_STATUS	ERROR_MESSAGE
Success	5	null
Failure	4	actual error message

Example of Price List Setup API

1. The following code can also be found in file QPPLXMP1.sql in \$QP_TOP/patch/115/sql directory

```
--set serveroutput on
/*****
Sample script which inserts a Price List with 3 price list lines, and the
product information for each of the lines(Product Information is stored in
pricing attributes table in product attribute columns). This sample price list
```

does not have any qualifiers or price breaks or non product-information type of pricing attributes.

This script must be modified by the user such that the gpr_pricing_attr_tbl(J).product_attr_value columns (for J = 1 to 3) are populated with 3 different valid inventory_item_id from the instance where this script is run. Please read the Oracle Advanced Pricing User's Guide (Appendix A & B) to understand the flexfields and seed data.

*****/

declare

```
gpr_return_status varchar2(1) := NULL;
gpr_msg_count number := 0;
gpr_msg_data varchar2(2000);
gpr_price_list_rec QP_PRICE_LIST_PUB.Price_List_Rec_Type;
gpr_price_list_val_rec QP_PRICE_LIST_PUB.Price_List_Val_Rec_Type;
gpr_price_list_line_tbl QP_PRICE_LIST_PUB.Price_List_Line_Tbl_Type;
gpr_price_list_line_val_tbl QP_PRICE_LIST_PUB.Price_List_Line_Val_Tbl_Type;
gpr_qualifiers_tbl QP_Qualifier_Rules_Pub.Qualifiers_Tbl_Type;
gpr_qualifiers_val_tbl QP_Qualifier_Rules_Pub.Qualifiers_Val_Tbl_Type;
gpr_pricing_attr_tbl QP_PRICE_LIST_PUB.Pricing_Attr_Tbl_Type;
gpr_pricing_attr_val_tbl QP_PRICE_LIST_PUB.Pricing_Attr_Val_Tbl_Type;
ppr_price_list_rec QP_PRICE_LIST_PUB.Price_List_Rec_Type;
ppr_price_list_val_rec QP_PRICE_LIST_PUB.Price_List_Val_Rec_Type;
ppr_price_list_line_tbl QP_PRICE_LIST_PUB.Price_List_Line_Tbl_Type;
ppr_price_list_line_val_tbl QP_PRICE_LIST_PUB.Price_List_Line_Val_Tbl_Type;
ppr_qualifiers_tbl QP_Qualifier_Rules_Pub.Qualifiers_Tbl_Type;
ppr_qualifiers_val_tbl QP_Qualifier_Rules_Pub.Qualifiers_Val_Tbl_Type;
ppr_pricing_attr_tbl QP_PRICE_LIST_PUB.Pricing_Attr_Tbl_Type;
ppr_pricing_attr_val_tbl QP_PRICE_LIST_PUB.Pricing_Attr_Val_Tbl_Type;
```

```
K number := 1;
j number := 1;
```

begin

```
--dbms_output.put_line('after get price list ');
```

```
/* set the list_header_id to g_miss_num */
```

```
gpr_price_list_rec.list_header_id := FND_API.G_MISS_NUM;
gpr_price_list_rec.name := 'Sample1-PL 1024';
gpr_price_list_rec.list_type_code := 'PRL';
gpr_price_list_rec.description := 'Sample price list';
```

```

/* you can set the currency of price list to whatever, say FRA */

gpr_price_list_rec.currency_code := 'USD';
gpr_price_list_rec.operation := QP_GLOBALS.G_OPR_CREATE;

FOR K IN 1..3 LOOP

    gpr_price_list_line_tbl(K).list_line_id := FND_API.G_MISS_NUM;
    gpr_price_list_line_tbl(K).list_line_type_code := 'PLL';
    gpr_price_list_line_tbl(K).operation := QP_GLOBALS.G_OPR_CREATE;
    gpr_price_list_line_tbl(K).operand := 10;
    gpr_price_list_line_tbl(K).arithmetic_operator := 'UNIT_PRICE';

END LOOP;

/*
product_attr_value stores inventory item id - product_attribute for Item Number
is Pricing_Attribute1 product_attribute_context is ITEM. Each line can have one
or more pricing attributes. PRICE_LIST_LINE_INDEX is used to link the child
(pricing attributes) to the parent(line).
When you have pricing attributes like color, length, width etc, populate the
fields pricing_attribute_context, pricing_attribute, pricing_attr_value_from,
pricing_attr_value_to and comparison_operator_code ( '=' or 'between') and
repeat the product_attr_value and its attribute and context for each record.
*/
J := 1;

gpr_pricing_attr_tbl(J).pricing_attribute_id := FND_API.G_MISS_NUM;
gpr_pricing_attr_tbl(J).list_line_id := FND_API.G_MISS_NUM;
gpr_pricing_attr_tbl(J).PRODUCT_ATTRIBUTE_CONTEXT := 'ITEM';
gpr_pricing_attr_tbl(J).PRODUCT_ATTRIBUTE := 'PRICING_ATTRIBUTE1';
gpr_pricing_attr_tbl(J).PRODUCT_ATTR_VALUE := '62061';
gpr_pricing_attr_tbl(J).PRODUCT_UOM_CODE := 'Ea';
gpr_pricing_attr_tbl(J).EXCLUDER_FLAG := 'N';
gpr_pricing_attr_tbl(J).ATTRIBUTE_GROUPING_NO := 1;
gpr_pricing_attr_tbl(J).PRICE_LIST_LINE_INDEX := 1;
gpr_pricing_attr_tbl(J).operation := QP_GLOBALS.G_OPR_CREATE;

J := J + 1;

gpr_pricing_attr_tbl(J).pricing_attribute_id := FND_API.G_MISS_NUM;
gpr_pricing_attr_tbl(J).list_line_id := FND_API.G_MISS_NUM;
gpr_pricing_attr_tbl(J).PRODUCT_ATTRIBUTE_CONTEXT := 'ITEM';
gpr_pricing_attr_tbl(J).PRODUCT_ATTRIBUTE := 'PRICING_ATTRIBUTE1';

```

```

gpr_pricing_attr_tbl(J).PRODUCT_ATTR_VALUE := '62081';
gpr_pricing_attr_tbl(J).PRODUCT_UOM_CODE := 'Ea';
gpr_pricing_attr_tbl(J).EXCLUDER_FLAG := 'N';
gpr_pricing_attr_tbl(J).ATTRIBUTE_GROUPING_NO := 1;
gpr_pricing_attr_tbl(J).PRICE_LIST_LINE_INDEX := 2;
gpr_pricing_attr_tbl(J).operation := QP_GLOBALS.G_OPR_CREATE;

J := J + 1;
gpr_pricing_attr_tbl(J).pricing_attribute_id := FND_API.G_MISS_NUM;
gpr_pricing_attr_tbl(J).list_line_id := FND_API.G_MISS_NUM;
gpr_pricing_attr_tbl(J).PRODUCT_ATTRIBUTE_CONTEXT := 'ITEM';
gpr_pricing_attr_tbl(J).PRODUCT_ATTRIBUTE := 'PRICING_ATTRIBUTE1';
gpr_pricing_attr_tbl(J).PRODUCT_ATTR_VALUE := '62083';
gpr_pricing_attr_tbl(J).PRODUCT_UOM_CODE := 'Ea';
gpr_pricing_attr_tbl(J).EXCLUDER_FLAG := 'N';
gpr_pricing_attr_tbl(J).ATTRIBUTE_GROUPING_NO := 1;
gpr_pricing_attr_tbl(J).PRICE_LIST_LINE_INDEX := 3;
gpr_pricing_attr_tbl(J).operation := QP_GLOBALS.G_OPR_CREATE;

--dbms_output.put_line('before process price list ');

```

QP_PRICE_LIST_PUB.Process_Price_List

```

(   p_api_version_number=> 1
,   p_init_msg_list=> FND_API.G_FALSE
,   p_return_values=> FND_API.G_FALSE
,   p_commit=> FND_API.G_FALSE
,   x_return_status=> gpr_return_status
,   x_msg_count=> gpr_msg_count
,   x_msg_data=> gpr_msg_data
,   p_PRICE_LIST_rec=> gpr_price_list_rec
,   p_PRICE_LIST_LINE_tbl=> gpr_price_list_line_tbl
,   p_PRICING_ATTR_tbl=> gpr_pricing_attr_tbl
,   x_PRICE_LIST_rec=> ppr_price_list_rec
,   x_PRICE_LIST_val_rec=> ppr_price_list_val_rec
,   x_PRICE_LIST_LINE_tbl=> ppr_price_list_line_tbl
,   x_PRICE_LIST_LINE_val_tbl=> ppr_price_list_line_val_tbl
,   x_QUALIFIERS_tbl=> ppr_qualifiers_tbl
,   x_QUALIFIERS_val_tbl=> ppr_qualifiers_val_tbl
,   x_PRICING_ATTR_tbl=> ppr_pricing_attr_tbl
,   x_PRICING_ATTR_val_tbl=> ppr_pricing_attr_val_tbl
);

IF gpr_return_status <> FND_API.G_RET_STS_SUCCESS THEN

```



```
        RAISE FND_API.G_EXC_UNEXPECTED_ERROR;

    END IF;

    --dbms_output.put_line('after process price list ');

EXCEPTION

    WHEN FND_API.G_EXC_ERROR THEN

        gpr_return_status := FND_API.G_RET_STS_ERROR;

        --Get message count and data

        --dbms_output.put_line('err msg 1 is : ' || gpr_msg_data);
    Rollback;
    WHEN FND_API.G_EXC_UNEXPECTED_ERROR THEN

        gpr_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;

        --dbms_output.put_line(' msg count 2 is : ' || gpr_msg_count);

        for k in 1 .. gpr_msg_count loop

            gpr_msg_data := oe_msg_pub.get( p_msg_index => k,

p_encoded => 'F'
            );
        /*

            oe_msg_pub.Count_And_Get

        (   p_count=> gpr_msg_count
        ,   p_data=> gpr_msg_data
        );

        */

        --Get message count and data
        --dbms_output.put_line('err msg ' || k || 'is: ' || gpr_msg_data);

        null;

    end loop;
    Rollback;
```

```

        WHEN OTHERS THEN

            gpr_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;

--Get message count and data
--dbms_output.put_line('err msg 3 is : ' || gpr_msg_data);
Rollback;
end;
/
commit;
exit;

```

2. The following code can also be found in file QPPLXMP2.sql in \$QP_TOP/patch/115/sql directory

```

--set serveroutput on
/*****
    Sample script which inserts a Price List with 3 price list lines, and the
    product information for each of the lines(Product Information is stored in
    pricing attributes table in product attribute columns), 1 secondary price list,
    2 price list qualifiers(Only header level qualifiers supported for pricelists)
    and 2 non-product pricing attributes per price list line.
    This sample price list does not have any price breaks. This script must be
    modified by the user such that the qpr_pricing_attr_tbl(J).product_attr_value
    columns are populated with valid inventory_item_id's from the instance where
    this script is run. Also the qpr_qualifiers_tbl(I).list_header_id must be
    populated with a valid list_header_id of a price list.
    For more information, see: Oracle Advanced Pricing User's Guide, Flexfields.
*****/
declare

    gpr_return_status varchar2(1) := NULL;
    gpr_msg_count number := 0;
    gpr_msg_data varchar2(2000);
    gpr_price_list_rec QP_PRICE_LIST_PUB.Price_List_Rec_Type;
    gpr_price_list_val_rec QP_PRICE_LIST_PUB.Price_List_Val_Rec_Type;
    gpr_price_list_line_tbl QP_PRICE_LIST_PUB.Price_List_Line_Tbl_Type;
    gpr_price_list_line_val_tbl QP_PRICE_LIST_PUB.Price_List_Line_Val_Tbl_Type;
    gpr_qualifiers_tbl QP_Qualifier_Rules_Pub.Qualifiers_Tbl_Type;
    gpr_qualifiers_val_tbl QP_Qualifier_Rules_Pub.Qualifiers_Val_Tbl_Type;
    gpr_pricing_attr_tbl QP_PRICE_LIST_PUB.Pricing_Attr_Tbl_Type;
    gpr_pricing_attr_val_tbl QP_PRICE_LIST_PUB.Pricing_Attr_Val_Tbl_Type;
    ppr_price_list_rec QP_PRICE_LIST_PUB.Price_List_Rec_Type;
    ppr_price_list_val_rec QP_PRICE_LIST_PUB.Price_List_Val_Rec_Type;

```

```

ppr_price_list_line_tbl QP_PRICE_LIST_PUB.Price_List_Line_Tbl_Type;
ppr_price_list_line_val_tbl QP_PRICE_LIST_PUB.Price_List_Line_Val_Tbl_Type;
ppr_qualifier_rules_rec QP_Qualifier_Rules_Pub.Qualifier_Rules_Rec_Type;
ppr_qualifier_rules_val_rec QP_Qualifier_Rules_Pub.Qualifier_Rules_Val_Rec_Type;
ppr_qualifiers_tbl QP_Qualifier_Rules_Pub.Qualifiers_Tbl_Type;
ppr_qualifiers_val_tbl QP_Qualifier_Rules_Pub.Qualifiers_Val_Tbl_Type;
ppr_pricing_attr_tbl QP_PRICE_LIST_PUB.Pricing_Attr_Tbl_Type;
ppr_pricing_attr_val_tbl QP_PRICE_LIST_PUB.Pricing_Attr_Val_Tbl_Type;

K number := 1;

J number := 1;

I number := 1;

begin

--dbms_output.put_line('after get price list ');

/* set the list_header_id to g_miss_num */
gpr_price_list_rec.list_header_id := FND_API.G_MISS_NUM;
gpr_price_list_rec.name := 'Sample2-PL 1024-16';
gpr_price_list_rec.list_type_code := 'PRL';
gpr_price_list_rec.description := 'Sample price list';

/* you can set the currency of price list to whatever, say FRA */
gpr_price_list_rec.currency_code := 'USD';
gpr_price_list_rec.operation := QP_GLOBALS.G_OPR_CREATE;

FOR K IN 1..3 LOOP

    gpr_price_list_line_tbl(K).list_line_id := FND_API.G_MISS_NUM;
    gpr_price_list_line_tbl(K).list_line_type_code := 'PLL';
    gpr_price_list_line_tbl(K).operation := QP_GLOBALS.G_OPR_CREATE;
    gpr_price_list_line_tbl(K).operand := 10;
    gpr_price_list_line_tbl(K).arithmetic_operator := 'UNIT_PRICE';

END LOOP;

/*
product_attr_value stores inventory item id - product_attribute for Item
Number is Pricing_Attribute1 product_attribute_context is ITEM. Each line
can have one or more pricing attributes. We use PRICE_LIST_LINE_INDEX to
link the child(pricing attributes) to the parent(line). When you have

```

```

pricing attributes like color, length, width etc, populate the fields
pricing_attribute_context, pricing_attribute, pricing_attr_value_from,
pricing_attr_value_to and comparison_operator_code ( '=' or 'between') and
repeat the product_attr_value and its attribute and context for each record.
*/

J := 1;

/* Pricing Attribute( including Product information) record 1for Price List Line
1 */

gpr_pricing_attr_tbl(J).pricing_attribute_id := FND_API.G_MISS_NUM;
gpr_pricing_attr_tbl(J).list_line_id := FND_API.G_MISS_NUM;
gpr_pricing_attr_tbl(J).PRODUCT_ATTRIBUTE_CONTEXT := 'ITEM';
gpr_pricing_attr_tbl(J).PRODUCT_ATTRIBUTE := 'PRICING_ATTRIBUTE1';
gpr_pricing_attr_tbl(J).PRODUCT_ATTR_VALUE := '62061';
gpr_pricing_attr_tbl(J).PRODUCT_UOM_CODE := 'Ea';
gpr_pricing_attr_tbl(J).EXCLUDER_FLAG := 'N';
gpr_pricing_attr_tbl(J).PRICE_LIST_LINE_INDEX := 1;
gpr_pricing_attr_tbl(J).PRICING_ATTRIBUTE_CONTEXT

:= 'PRICING ATTRIBUTE';

gpr_pricing_attr_tbl(J).PRICING_ATTRIBUTE := 'PRICING_ATTRIBUTE16';
--Corresponds to the Pricing Attribute 'Freight Cost'
gpr_pricing_attr_tbl(J).PRICING_ATTR_VALUE_FROM := '100';
gpr_pricing_attr_tbl(J).PRICING_ATTR_VALUE_TO := '200';
gpr_pricing_attr_tbl(J).COMPARISON_OPERATOR_CODE := 'BETWEEN';
gpr_pricing_attr_tbl(J).operation := QP_GLOBALS.G_OPR_CREATE;
J := J + 1;

/* Pricing Attribute( including Product information) record 2
for Price List Line 1 */

gpr_pricing_attr_tbl(J).pricing_attribute_id := FND_API.G_MISS_NUM;
gpr_pricing_attr_tbl(J).list_line_id := FND_API.G_MISS_NUM;
gpr_pricing_attr_tbl(J).PRODUCT_ATTRIBUTE_CONTEXT := 'ITEM';
gpr_pricing_attr_tbl(J).PRODUCT_ATTRIBUTE := 'PRICING_ATTRIBUTE1';
gpr_pricing_attr_tbl(J).PRODUCT_ATTR_VALUE := '62061';
gpr_pricing_attr_tbl(J).PRODUCT_UOM_CODE := 'Ea';
gpr_pricing_attr_tbl(J).EXCLUDER_FLAG := 'N';
gpr_pricing_attr_tbl(J).PRICE_LIST_LINE_INDEX := 1;
gpr_pricing_attr_tbl(J).PRICING_ATTRIBUTE_CONTEXT

:= 'PRICING ATTRIBUTE';

```

```

gpr_pricing_attr_tbl(J).PRICING_ATTRIBUTE := 'PRICING_ATTRIBUTE11';

- Corresponds to the Pricing Attribute 'Customer Item'

    gpr_pricing_attr_tbl(J).PRICING_ATTR_VALUE_FROM := '3220';
    gpr_pricing_attr_tbl(J).COMPARISON_OPERATOR_CODE := '=';
    gpr_pricing_attr_tbl(J).operation := QP_GLOBALS.G_OPR_CREATE;

    J := J + 1;

/* Pricing Attribute( including Product information) record 1 for Price List
Line 2 */

    gpr_pricing_attr_tbl(J).pricing_attribute_id := FND_API.G_MISS_NUM;
    gpr_pricing_attr_tbl(J).list_line_id := FND_API.G_MISS_NUM;
    gpr_pricing_attr_tbl(J).PRODUCT_ATTRIBUTE_CONTEXT := 'ITEM';
    gpr_pricing_attr_tbl(J).PRODUCT_ATTRIBUTE := 'PRICING_ATTRIBUTE1';
    gpr_pricing_attr_tbl(J).PRODUCT_ATTR_VALUE := '62081';
    gpr_pricing_attr_tbl(J).PRODUCT_UOM_CODE := 'Ea';
    gpr_pricing_attr_tbl(J).EXCLUDER_FLAG := 'N';
    gpr_pricing_attr_tbl(J).PRICE_LIST_LINE_INDEX := 2;
    gpr_pricing_attr_tbl(J).PRICING_ATTRIBUTE_CONTEXT

    := 'PRICING_ATTRIBUTE';

gpr_pricing_attr_tbl(J).PRICING_ATTRIBUTE := 'PRICING_ATTRIBUTE12';

-- Corresponds to the Pricing Attribute 'Insurance Cost'
gpr_pricing_attr_tbl(J).PRICING_ATTR_VALUE_FROM := '12.50';
gpr_pricing_attr_tbl(J).COMPARISON_OPERATOR_CODE := '=';
gpr_pricing_attr_tbl(J).operation := QP_GLOBALS.G_OPR_CREATE;

J := J + 1;

/* Pricing Attribute( including Product information) record 2
for Price List Line 2 */

    gpr_pricing_attr_tbl(J).pricing_attribute_id := FND_API.G_MISS_NUM;
    gpr_pricing_attr_tbl(J).list_line_id := FND_API.G_MISS_NUM;
    gpr_pricing_attr_tbl(J).PRODUCT_ATTRIBUTE_CONTEXT := 'ITEM';
    gpr_pricing_attr_tbl(J).PRODUCT_ATTRIBUTE := 'PRICING_ATTRIBUTE1';
    gpr_pricing_attr_tbl(J).PRODUCT_ATTR_VALUE := '62081';
    gpr_pricing_attr_tbl(J).PRODUCT_UOM_CODE := 'Ea';
    gpr_pricing_attr_tbl(J).EXCLUDER_FLAG := 'N';
    gpr_pricing_attr_tbl(J).PRICE_LIST_LINE_INDEX := 2;

```

```

gpr_pricing_attr_tbl(J).PRICING_ATTRIBUTE_CONTEXT := 'PRICING ATTRIBUTE';
gpr_pricing_attr_tbl(J).PRICING_ATTRIBUTE := 'PRICING_ATTRIBUTE13';
-- Corresponds to the Pricing Attribute 'Handling Cost'
gpr_pricing_attr_tbl(J).PRICING_ATTR_VALUE_FROM := '25.60';
gpr_pricing_attr_tbl(J).COMPARISON_OPERATOR_CODE := '=';
gpr_pricing_attr_tbl(J).operation := QP_GLOBALS.G_OPR_CREATE;

J := J + 1;

/* Pricing Attribute( including Product information) record 1
for Price List Line 3 */

gpr_pricing_attr_tbl(J).pricing_attribute_id := FND_API.G_MISS_NUM;
gpr_pricing_attr_tbl(J).list_line_id := FND_API.G_MISS_NUM;
gpr_pricing_attr_tbl(J).PRODUCT_ATTRIBUTE_CONTEXT := 'ITEM';
gpr_pricing_attr_tbl(J).PRODUCT_ATTRIBUTE := 'PRICING_ATTRIBUTE1';
gpr_pricing_attr_tbl(J).PRODUCT_ATTR_VALUE := '62083';
gpr_pricing_attr_tbl(J).PRODUCT_UOM_CODE := 'Ea';
gpr_pricing_attr_tbl(J).EXCLUDER_FLAG := 'N';
gpr_pricing_attr_tbl(J).PRICE_LIST_LINE_INDEX := 3;
gpr_pricing_attr_tbl(J).PRICING_ATTRIBUTE_CONTEXT := 'PRICING ATTRIBUTE';
gpr_pricing_attr_tbl(J).PRICING_ATTRIBUTE := 'PRICING_ATTRIBUTE14';

-- Corresponds to the Pricing Attribute 'Export Cost'
gpr_pricing_attr_tbl(J).PRICING_ATTR_VALUE_FROM := '3.50';
gpr_pricing_attr_tbl(J).COMPARISON_OPERATOR_CODE := '=';
gpr_pricing_attr_tbl(J).operation := QP_GLOBALS.G_OPR_CREATE;

J := J + 1;

/* Pricing Attribute( including Product information) record 2 for Price List
Line 3 */

gpr_pricing_attr_tbl(J).pricing_attribute_id := FND_API.G_MISS_NUM;
gpr_pricing_attr_tbl(J).list_line_id := FND_API.G_MISS_NUM;
gpr_pricing_attr_tbl(J).PRODUCT_ATTRIBUTE_CONTEXT := 'ITEM';
gpr_pricing_attr_tbl(J).PRODUCT_ATTRIBUTE := 'PRICING_ATTRIBUTE1';
gpr_pricing_attr_tbl(J).PRODUCT_ATTR_VALUE := '62083';
gpr_pricing_attr_tbl(J).PRODUCT_UOM_CODE := 'Ea';
gpr_pricing_attr_tbl(J).EXCLUDER_FLAG := 'N';
gpr_pricing_attr_tbl(J).PRICE_LIST_LINE_INDEX := 3;
gpr_pricing_attr_tbl(J).PRICING_ATTRIBUTE_CONTEXT := 'PRICING ATTRIBUTE';
gpr_pricing_attr_tbl(J).PRICING_ATTRIBUTE := 'PRICING_ATTRIBUTE15';

-- Corresponds to the Pricing Attribute 'Duty Cost'

```

```

gpr_pricing_attr_tbl(J).PRICING_ATTR_VALUE_FROM := '4.50';
gpr_pricing_attr_tbl(J).COMPARISON_OPERATOR_CODE := '=';
gpr_pricing_attr_tbl(J).operation := QP_GLOBALS.G_OPR_CREATE;

I := 1;

/* Price List (Header) Level Qualifier record 1 */
gpr_qualifiers_tbl(I).LIST_HEADER_ID := FND_API.G_MISS_NUM;
gpr_qualifiers_tbl(I).EXCLUDER_FLAG := 'N';
gpr_qualifiers_tbl(I).QUALIFIER_GROUPING_NO := 1;
gpr_qualifiers_tbl(I).QUALIFIER_CONTEXT := 'VOLUME';
gpr_qualifiers_tbl(I).QUALIFIER_ATTRIBUTE := 'QUALIFIER_ATTRIBUTE10';
--Corresponds to Qualifier Attribute 'Order Amount'
gpr_qualifiers_tbl(I).QUALIFIER_ATTR_VALUE := '100';
gpr_qualifiers_tbl(I).QUALIFIER_ATTR_VALUE_TO := '200';
gpr_qualifiers_tbl(I).COMPARISON_OPERATOR_CODE := 'BETWEEN';
gpr_qualifiers_tbl(I).OPERATION := QP_GLOBALS.G_OPR_CREATE;

I := I + 1;

/* Price List (Header) Level Qualifier record 2 */

gpr_qualifiers_tbl(I).LIST_HEADER_ID := FND_API.G_MISS_NUM;
gpr_qualifiers_tbl(I).EXCLUDER_FLAG := 'N';
gpr_qualifiers_tbl(I).QUALIFIER_GROUPING_NO := 2;
gpr_qualifiers_tbl(I).QUALIFIER_CONTEXT := 'ORDER';
gpr_qualifiers_tbl(I).QUALIFIER_ATTRIBUTE := 'QUALIFIER_ATTRIBUTE13';
--Corresponds to Qualifier Attribute 'Order Category'
gpr_qualifiers_tbl(I).QUALIFIER_ATTR_VALUE := 'RETURN';
gpr_qualifiers_tbl(I).COMPARISON_OPERATOR_CODE := '=';
gpr_qualifiers_tbl(I).OPERATION := QP_GLOBALS.G_OPR_CREATE;

--dbms_output.put_line('before process price list ');

```

QP_PRICE_LIST_PUB.Process_Price_List

```

(   p_api_version_number      => 1
,   p_init_msg_list           => FND_API.G_FALSE
,   p_return_values            => FND_API.G_FALSE
,   p_commit=> FND_API.G_FALSE
,   x_return_status=> gpr_return_status
,   x_msg_count=> gpr_msg_count
,   x_msg_data=> gpr_msg_data
,   p_PRICE_LIST_rec=> gpr_price_list_rec

```

```

,   p_PRICE_LIST_LINE_tbl=> gpr_price_list_line_tbl
,   p_QUALIFIERS_tbl=> gpr_qualifiers_tbl
,   p_PRICING_ATTR_tbl=> gpr_pricing_attr_tbl
,   x_PRICE_LIST_rec=> ppr_price_list_rec
,   x_PRICE_LIST_val_rec=> ppr_price_list_val_rec
,   x_PRICE_LIST_LINE_tbl=> ppr_price_list_line_tbl
,   x_PRICE_LIST_LINE_val_tbl=> ppr_price_list_line_val_tbl
,   x_QUALIFIERS_tbl=> ppr_qualifiers_tbl
,   x_QUALIFIERS_val_tbl=> ppr_qualifiers_val_tbl
,   x_PRICING_ATTR_tbl=> ppr_pricing_attr_tbl
,   x_PRICING_ATTR_val_tbl=> ppr_pricing_attr_val_tbl
);

IF gpr_return_status <> FND_API.G_RET_STS_SUCCESS THEN

    RAISE FND_API.G_EXC_UNEXPECTED_ERROR;

END IF;

/** Add the following code in the body only if you want to create secondary
price lists as well. Otherwise the above code is sufficient. */

I := 1;

/* Secondary Price List record 1 */

gpr_qualifiers_tbl(I).LIST_HEADER_ID := '87013';
--Corresponds to list_header_id for the Secondary Price List 'Testing 1019'
gpr_qualifiers_tbl(I).QUALIFIER_CONTEXT := 'MODLIST';
gpr_qualifiers_tbl(I).QUALIFIER_ATTRIBUTE := 'QUALIFIER_ATTRIBUTE4';
--Corresponds to Qualifier Attribute 'Price List'
gpr_qualifiers_tbl(I).QUALIFIER_ATTR_VALUE := ppr_PRICE_LIST_rec.list_
header_id;
gpr_qualifiers_tbl(I).COMPARISON_OPERATOR_CODE := '=';
gpr_qualifiers_tbl(I).OPERATION := QP_GLOBALS.G_OPR_CREATE;

```

QP_QUALIFIER_RULES_PUB.Process_Qualifier_Rules

```

(   p_api_version_number=> 1
,   p_init_msg_list=> FND_API.G_FALSE
,   p_return_values=> FND_API.G_FALSE
,   p_commit=> FND_API.G_FALSE
,   x_return_status=> gpr_return_status
,   x_msg_count=> gpr_msg_count
,   x_msg_data=> gpr_msg_data

```



```

,   p_QUALIFIERS_tbl=> gpr_qualifiers_tbl
,   x_QUALIFIER_RULES_rec=> ppr_qualifier_rules_rec
,   x_QUALIFIER_RULES_val_rec=> ppr_qualifier_rules_val_rec
,   x_QUALIFIERS_tbl=> ppr_qualifiers_tbl
,   x_QUALIFIERS_val_tbl=> ppr_qualifiers_val_tbl
);

IF gpr_return_status <> FND_API.G_RET_STS_SUCCESS THEN

    RAISE FND_API.G_EXC_UNEXPECTED_ERROR;

END IF;

--dbms_output.put_line ('after process price list ');

EXCEPTION

    WHEN FND_API.G_EXC_ERROR THEN

gpr_return_status := FND_API.G_RET_STS_ERROR;

--Get message count and data

--dbms_output.put_line('err msg 1 is : ' || gpr_msg_data);
Rollback;
    WHEN FND_API.G_EXC_UNEXPECTED_ERROR THEN

gpr_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;
--dbms_output.put_line(' msg count 2 is : ' || gpr_msg_count);

    for k in 1 .. gpr_msg_count loop
        gpr_msg_data := oe_msg_pub.get( p_msg_index => k,
p_encoded => 'F'
        );
        /*
            oe_msg_pub.Count_And_Get
        (   p_count=> gpr_msg_count
        ,   p_data => gpr_msg_data
        );
        */

--Get message count and data
--dbms_output.put_line('err msg ' || k || 'is: ' || gpr_msg_data);
null;

```

```

end loop;
    Rollback;
    WHEN OTHERS THEN

        gpr_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;

--Get message count and data
--dbms_output.put_line('err msg 3 is : ' || gpr_msg_data);
Rollback;
end;
/
commit;
exit;

```

3. The following code can also be found in file QPPLXMP3.sql in \$QP_TOP/patch/115/sql directory

```

--set serveroutput on
/*****
Sample script which inserts a Price List with 1 price list line and the
product information for this line (Product Information is stored in pricing
attributes table in product attribute columns), 1 non-product pricing attribute
for the price list line and 1 price list qualifier(Only header level qualifiers
supported for pricelists).
The qualifier and pricing attribute record passes the display value of the
qualifier attribute value and pricing attribute value columns respectively
instead of the hidden id. Simultaneously, the qualifier_attribute and pricing_
attribute segment name must be passed
instead of the pricing attribute itself. This is done using the corresponding
val_tbl structure for qualifiers and pricing attributes.
This script must be run only after applying patch 1473223 to the instance. This
script must also be modified by the user such that the gpr_pricing_attr_val_
tbl(J).PRICING_ATTR_VALUE_FROM_DESC column is populated with a valid 'Customer
Item Number' and the gpr_qualifiers_val_tbl(I). QUALIFIER_ATTR_VALUE_DESC column
is populated with a valid 'Order Category' Description from the instance where
this script is to be run.
For more information, see: Oracle Advanced Pricing User's Guide, Flexfields.
*****/

K number := 1;
J number := 1;
I number := 1;

begin

```

```

--dbms_output.put_line('after get price list ');

/* set the list_header_id to g_miss_num */

    gpr_price_list_rec.list_header_id := FND_API.G_MISS_NUM;
    gpr_price_list_rec.name := 'Sample3-PL 1024-21';
    gpr_price_list_rec.list_type_code := 'PRL';
    gpr_price_list_rec.description := 'Sample price list';

/* you can set the currency of price list to whatever, say FRA */

gpr_price_list_rec.currency_code := 'USD';
gpr_price_list_rec.operation := QP_GLOBALS.G_OPR_CREATE;

--Create a Price List Line
K := 1;

gpr_price_list_line_tbl(K).list_line_id := FND_API.G_MISS_NUM;
gpr_price_list_line_tbl(K).list_line_type_code := 'PLL';
gpr_price_list_line_tbl(K).operation := QP_GLOBALS.G_OPR_CREATE;
gpr_price_list_line_tbl(K).operand := 10;
gpr_price_list_line_tbl(K).arithmetic_operator := 'UNIT_PRICE';

/*
    product_attr_value stores inventory item id -
    product_attribute for Item Number is Pricing_Attribute1
    product_attribute_context is ITEM .
    Each line can have one or more pricing attributes. we use PRICE_LIST_LINE_
    INDEX to link the child (pricing attributes) to the parent(line).
    When you have pricing attributes like color, length, width etc, populate the
    fields pricing_attribute_context, pricing_attribute, pricing_attr_value_
    from, pricing_attr_value_to and comparison_operator_code ( '=' or 'between')
    and repeat the product_attr_value and its attribute and context for each
    record.
*/

J := 1;

/* Pricing Attribute( including Product information) record for Price List Line
1 */

    gpr_pricing_attr_tbl(J).pricing_attribute_id := FND_API.G_MISS_NUM;
    gpr_pricing_attr_tbl(J).list_line_id := FND_API.G_MISS_NUM;
    gpr_pricing_attr_tbl(J).PRODUCT_ATTRIBUTE_CONTEXT := 'ITEM';
    gpr_pricing_attr_tbl(J).PRODUCT_ATTRIBUTE := 'PRICING_ATTRIBUTE1';

```

```

gpr_pricing_attr_tbl(J).PRODUCT_ATTR_VALUE := '62061';
gpr_pricing_attr_tbl(J).PRODUCT_UOM_CODE := 'Ea';
gpr_pricing_attr_tbl(J).EXCLUDER_FLAG := 'N';
gpr_pricing_attr_tbl(J).PRICE_LIST_LINE_INDEX := 1;
gpr_pricing_attr_tbl(J).PRICING_ATTRIBUTE_CONTEXT
:= 'PRICING ATTRIBUTE';

-- Instead of
-- 'gpr_pricing_attr_tbl(J).PRICING_ATTRIBUTE := 'PRICING_ATTRIBUTE11';'
-- the segment name of this pricing attribute is passed using the
-- val_tbl structure as shown below.

gpr_pricing_attr_val_tbl(J).PRICING_ATTRIBUTE_DESC

:= 'CUSTOMER_ITEM_ID';
-- Corresponds to the Pricing Attribute 'Customer Item'
-- Instead of
-- 'gpr_pricing_attr_tbl(J).PRICING_ATTR_VALUE_FROM := '3220';'
-- the display value (from the flex value-set's meaning column)
-- of this pricing attribute value is passed using the val_tbl
-- structure as shown below.

gpr_pricing_attr_val_tbl(J).PRICING_ATTR_VALUE_FROM_DESC
:= 'C01KS';
-- Corresponds to the Customer Item Code for the Item Id 3220.
gpr_pricing_attr_tbl(J).COMPARISON_OPERATOR_CODE := '=';
gpr_pricing_attr_tbl(J).operation := QP_GLOBALS.G_OPR_CREATE;

I := 1;

/* Price List (Header) Level Qualifier record */
gpr_qualifiers_tbl(I).LIST_HEADER_ID := FND_API.G_MISS_NUM;
gpr_qualifiers_tbl(I).EXCLUDER_FLAG := 'N';
gpr_qualifiers_tbl(I).QUALIFIER_GROUPING_NO := 1;
gpr_qualifiers_tbl(I).QUALIFIER_CONTEXT := 'ORDER';

--Instead of
--'gpr_qualifiers_tbl(I).QUALIFIER_ATTRIBUTE := 'QUALIFIER_ATTRIBUTE13';'
--the segment name of this qualifier attribute is passed using the
--val_tbl structure as shown below.

gpr_qualifiers_val_tbl(I).QUALIFIER_ATTRIBUTE_DESC

:= 'ORDER_CATEGORY';

```

```

--Corresponds to Qualifier Attribute 'Order Category'

-- Instead of
-- 'gpr_qualifiers_tbl(I).QUALIFIER_ATTR_VALUE := 'MIXED';'
-- the display value (from the flex value-set's meaning column)
-- of this qualifier attribute value is passed using the val_tbl
-- structure as shown below.

      gpr_qualifiers_val_tbl(I).QUALIFIER_ATTR_VALUE_DESC := 'Mixed';
--Corresponds to the Meaning column for the lookup code 'MIXED'.
      gpr_qualifiers_tbl(I).COMPARISON_OPERATOR_CODE := '=';
      gpr_qualifiers_tbl(I).OPERATION := QP_GLOBALS.G_OPR_CREATE;

--dbms_output.put_line('before process price list ');

```

QP_PRICE_LIST_PUB.Process_Price_List

```

( p_api_version_number=> 1
, p_init_msg_list=> FND_API.G_FALSE
, p_return_values=> FND_API.G_FALSE
, p_commit=> FND_API.G_FALSE
, x_return_status=> gpr_return_status
, x_msg_count=> gpr_msg_count
, x_msg_data=> gpr_msg_data
, p_PRICE_LIST_rec=> gpr_price_list_rec
, p_PRICE_LIST_LINE_tbl=> gpr_price_list_line_tbl
, p_QUALIFIERS_tbl=> gpr_qualifiers_tbl
, p_QUALIFIERS_val_tbl=> gpr_qualifiers_val_tbl
, p_PRICING_ATTR_tbl=> gpr_pricing_attr_tbl
, p_PRICING_ATTR_val_tbl=> gpr_pricing_attr_val_tbl
, x_PRICE_LIST_rec=> ppr_price_list_rec
, x_PRICE_LIST_val_rec=> ppr_price_list_val_rec
, x_PRICE_LIST_LINE_tbl=> ppr_price_list_line_tbl
, x_PRICE_LIST_LINE_val_tbl=> ppr_price_list_line_val_tbl
, x_QUALIFIERS_tbl=> ppr_qualifiers_tbl
, x_QUALIFIERS_val_tbl=> ppr_qualifiers_val_tbl
, x_PRICING_ATTR_tbl=> ppr_pricing_attr_tbl
, x_PRICING_ATTR_val_tbl=> ppr_pricing_attr_val_tbl
);

IF gpr_return_status <> FND_API.G_RET_STS_SUCCESS THEN

      RAISE FND_API.G_EXC_UNEXPECTED_ERROR;

END IF;

```

```

--dbms_output.put_line('after process price list ');

EXCEPTION

    WHEN FND_API.G_EXC_ERROR THEN

gpr_return_status := FND_API.G_RET_STS_ERROR;

    --Get message count and data

    --dbms_output.put_line('err msg 1 is : ' || gpr_msg_data);

Rollback;

WHEN FND_API.G_EXC_UNEXPECTED_ERROR THEN

    gpr_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;
    --dbms_output.put_line(' msg count 2 is : ' || gpr_msg_count);

for k in 1 .. gpr_msg_count loop
    gpr_msg_data := oe_msg_pub.get( p_msg_index => k,
    p_encoded => 'F'

);
/*
    oe_msg_pub.Count_And_Get

    (   p_count=> gpr_msg_count
    ,   p_data=> gpr_msg_data
);

*/
--Get message count and data
--dbms_output.put_line('err msg ' || k || 'is: ' || gpr_msg_data);
null;

end loop;
Rollback;
WHEN OTHERS THEN

    gpr_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;

    --Get message count and data
    --dbms_output.put_line('err msg 3 is : ' || gpr_msg_data);

```

```

Rollback;
end;
/
commit;
exit;

```

4. The following code can also be found in file QPPLXMP4.sql in
\$qp/patch/115/sql directory:

```

/*$Header: QPPLXMP4.sql 115.3 2000/11/10 00:54:12 rchellam noship $*/
--set serveroutput on

/*****
Sample script which inserts a Price List with 1 price list line of type PBH
price break header) and the product information for this line(Product
Information is stored in pricing attributes table in product attribute columns),
a regular pricing attribute(non product) and a Price Break Child Line (To create
a price break child line it is necessary to create a combination of a list line
and a pricing attribute where the pricing attribute can only have the Volume
pricing context and Item Quantity Pricing Attribute).
For more information, see: Oracle Advanced Pricing User's Guide.
*****/
declare

    gpr_return_status varchar2(1) := NULL;
    gpr_msg_count number := 0;
    gpr_msg_data varchar2(2000);

    gpr_price_list_rec QP_PRICE_LIST_PUB.Price_List_Rec_Type;
    gpr_price_list_line_tbl QP_PRICE_LIST_PUB.Price_List_Line_Tbl_Type;
    gpr_pricing_attr_tbl QP_PRICE_LIST_PUB.Pricing_Attr_Tbl_Type;

    ppr_price_list_rec QP_PRICE_LIST_PUB.Price_List_Rec_Type;
    ppr_price_list_val_rec QP_PRICE_LIST_PUB.Price_List_Val_Rec_Type;
    ppr_price_list_line_tbl QP_PRICE_LIST_PUB.Price_List_Line_Tbl_Type;
    ppr_price_list_line_val_tbl QP_PRICE_LIST_PUB.Price_List_Line_Val_Tbl_Type;
    ppr_pricing_attr_tbl QP_PRICE_LIST_PUB.Pricing_Attr_Tbl_Type;
    ppr_pricing_attr_val_tbl QP_PRICE_LIST_PUB.Pricing_Attr_Val_Tbl_Type;
    ppr_qualifiers_tbl QP_QUALIFIER_RULES_PUB.Qualifiers_Tbl_Type;
    ppr_qualifiers_val_tbl QP_QUALIFIER_RULES_PUB.Qualifiers_Val_Tbl_Type;

    K number := 1;
    J number := 1;
    I number := 1;

```

```

begin

    /* set the list_header_id to g_miss_num */

    gpr_price_list_rec.list_header_id := FND_API.G_MISS_NUM;
    gpr_price_list_rec.name := 'Sample4-PL 1109-14';
    gpr_price_list_rec.list_type_code := 'PRL';
    gpr_price_list_rec.description := 'Sample price list';

    /* you can set the currency of price list to whatever, say FRA */

    gpr_price_list_rec.currency_code := 'USD';
    gpr_price_list_rec.operation := QP_GLOBALS.G_OPR_CREATE;

    --Create a Price List Line of type 'PBH'

    K := 1;

    gpr_price_list_line_tbl(K).list_line_id := FND_API.G_MISS_NUM;
    gpr_price_list_line_tbl(K).list_line_type_code := 'PBH';
    gpr_price_list_line_tbl(K).price_break_type_code := 'POINT';
    gpr_price_list_line_tbl(K).operation := QP_GLOBALS.G_OPR_CREATE;
    gpr_price_list_line_tbl(K).operand := 10;
    gpr_price_list_line_tbl(K).arithmetic_operator := 'UNIT_PRICE';

    /*
    product_attr_value stores inventory item id -
    product_attribute for Item Number is Pricing_Attribute1
    product_attribute_context is ITEM .
    Each line can have one or more pricing attributes. we use
    PRICE_LIST_LINE_INDEX to link the child(pricing attributes )
    to the parent(line).
    When you have pricing attributes like color, length, width etc,
    populate the fields pricing_attribute_context, pricing_attribute, pricing_
    attr_value_from, pricing_attr_value_to and comparison_operator_code ( '='
    or 'between') and repeat the product_attr_value and its attribute and
    context for each record.
    */

    J := 1;

    /* Pricing Attribute( with only Product information) record for Price List Line
    of type 'PBH' */

    gpr_pricing_attr_tbl(J).pricing_attribute_id := FND_API.G_MISS_NUM;
    gpr_pricing_attr_tbl(J).list_line_id := FND_API.G_MISS_NUM;

```



```

gpr_pricing_attr_tbl(J).PRODUCT_ATTRIBUTE_CONTEXT := 'ITEM';
gpr_pricing_attr_tbl(J).PRODUCT_ATTRIBUTE := 'PRICING_ATTRIBUTE1';
gpr_pricing_attr_tbl(J).PRODUCT_ATTR_VALUE := '62081';
gpr_pricing_attr_tbl(J).PRODUCT_UOM_CODE := 'Ea';
gpr_pricing_attr_tbl(J).EXCLUDER_FLAG := 'N';
gpr_pricing_attr_tbl(J).PRICE_LIST_LINE_INDEX := 1;
gpr_pricing_attr_tbl(J).operation := QP_GLOBALS.G_OPR_CREATE;

J := J + 1;

/* Pricing Attribute( non Product) record for Price List Line of type 'PBH' */
gpr_pricing_attr_tbl(J).pricing_attribute_id := FND_API.G_MISS_NUM;
gpr_pricing_attr_tbl(J).list_line_id := FND_API.G_MISS_NUM;
gpr_pricing_attr_tbl(J).PRODUCT_ATTRIBUTE_CONTEXT := 'ITEM';
gpr_pricing_attr_tbl(J).PRODUCT_ATTRIBUTE := 'PRICING_ATTRIBUTE1';
gpr_pricing_attr_tbl(J).PRODUCT_ATTR_VALUE := '62081';
gpr_pricing_attr_tbl(J).PRODUCT_UOM_CODE := 'Ea';
gpr_pricing_attr_tbl(J).PRICING_ATTRIBUTE_CONTEXT := 'PRICING_ATTRIBUTE';
gpr_pricing_attr_tbl(J).PRICING_ATTRIBUTE := 'PRICING_ATTRIBUTE12';

-- Corresponds to Pricing Attribute 'Insurance Cost'

gpr_pricing_attr_tbl(J).PRICING_ATTR_VALUE_FROM := '12.50';
gpr_pricing_attr_tbl(J).COMPARISON_OPERATOR_CODE := '=';
gpr_pricing_attr_tbl(J).EXCLUDER_FLAG := 'N';
gpr_pricing_attr_tbl(J).PRICE_LIST_LINE_INDEX := 1;
gpr_pricing_attr_tbl(J).operation := QP_GLOBALS.G_OPR_CREATE;

--Create a Price List Line of type 'PLL', a child price break line

K := K + 1; /* K=2 */
gpr_price_list_line_tbl(K).list_line_id := FND_API.G_MISS_NUM;
gpr_price_list_line_tbl(K).list_line_type_code := 'PLL';
gpr_price_list_line_tbl(K).operation := QP_GLOBALS.G_OPR_CREATE;
gpr_price_list_line_tbl(K).operand := 10;
gpr_price_list_line_tbl(K).arithmetic_operator := 'UNIT_PRICE';
gpr_price_list_line_tbl(K).rltd_modifier_group_no := 1;
gpr_price_list_line_tbl(K).PRICE_BREAK_HEADER_INDEX := 1;

-- This must point to the PBH line. In this example this is a child of the PBH
above where K=1.
--Create a Pricing Attribute with pricing context as 'Volume' and Pricing
Attribute as 'Item Quantity'
--needed to complete the creation of a child price break line.

```

```
J := J + 1;

/* Pricing Attribute( with specific pricing attribute context and attribute)
record for Price List Line of type 'PLL' which is a child Price Break Line */
gpr_pricing_attr_tbl(J).pricing_attribute_id := FND_API.G_MISS_NUM;
gpr_pricing_attr_tbl(J).list_line_id := FND_API.G_MISS_NUM;
gpr_pricing_attr_tbl(J).PRODUCT_ATTRIBUTE_CONTEXT := 'ITEM';
gpr_pricing_attr_tbl(J).PRODUCT_ATTRIBUTE := 'PRICING_ATTRIBUTE1';
gpr_pricing_attr_tbl(J).PRODUCT_ATTR_VALUE := '62081';
gpr_pricing_attr_tbl(J).PRODUCT_UOM_CODE := 'Ea';
gpr_pricing_attr_tbl(J).PRICING_ATTRIBUTE_CONTEXT := 'VOLUME';
gpr_pricing_attr_tbl(J).PRICING_ATTRIBUTE := 'PRICING_ATTRIBUTE10'; --'Item
Quantity'
gpr_pricing_attr_tbl(J).PRICING_ATTR_VALUE_FROM := '10';
gpr_pricing_attr_tbl(J).PRICING_ATTR_VALUE_TO := '20';
gpr_pricing_attr_tbl(J).COMPARISON_OPERATOR_CODE := 'BETWEEN';
gpr_pricing_attr_tbl(J).EXCLUDER_FLAG := 'N';
gpr_pricing_attr_tbl(J).PRICE_LIST_LINE_INDEX := 2;

--Because this is a pricing attribute of the line K = 2

gpr_pricing_attr_tbl(J).operation := QP_GLOBALS.G_OPR_CREATE;

--dbms_output.put_line('before process price list ');
```

QP_PRICE_LIST_PUB.Process_Price_List

```
( p_api_version_number=> 1
, p_init_msg_list=> FND_API.G_FALSE
, p_return_values=> FND_API.G_FALSE
, p_commit=> FND_API.G_FALSE
, x_return_status=> gpr_return_status
, x_msg_count=> gpr_msg_count
, x_msg_data=> gpr_msg_data
, p_PRICE_LIST_rec=> gpr_price_list_rec
, p_PRICE_LIST_LINE_tbl=> gpr_price_list_line_tbl
, p_PRICING_ATTR_tbl=> gpr_pricing_attr_tbl
, x_PRICE_LIST_rec=> ppr_price_list_rec
, x_PRICE_LIST_val_rec=> ppr_price_list_val_rec
, x_PRICE_LIST_LINE_tbl=> ppr_price_list_line_tbl
, x_PRICE_LIST_LINE_val_tbl=> ppr_price_list_line_val_tbl
, x_QUALIFIERS_tbl=> ppr_qualifiers_tbl
, x_QUALIFIERS_val_tbl=> ppr_qualifiers_val_tbl
, x_PRICING_ATTR_tbl=> ppr_pricing_attr_tbl
```

```

,   x_PRICING_ATTR_val_tbl=> ppr_pricing_attr_val_tbl
);

    IF gpr_return_status <> FND_API.G_RET_STS_SUCCESS THEN

        RAISE FND_API.G_EXC_UNEXPECTED_ERROR;

    END IF;

--dbms_output.put_line('after process price list ');

EXCEPTION

    WHEN FND_API.G_EXC_ERROR THEN
        gpr_return_status := FND_API.G_RET_STS_ERROR;

--Get message count and data
--dbms_output.put_line('err msg 1 is : ' || gpr_msg_data);
Rollback;

    WHEN FND_API.G_EXC_UNEXPECTED_ERROR THEN
        gpr_return_status := FND_API.G_RET_STS_UNEXP_ERROR;

--dbms_output.put_line(' msg count 2 is : ' || gpr_msg_count);

    for k in 1 .. gpr_msg_count loop
        gpr_msg_data := oe_msg_pub.get( p_msg_index => k,

            p_encoded=> 'F'

        );
        /*

        oe_msg_pub.Count_And_Get

        (   p_count=> gpr_msg_count
        ,   p_data=> gpr_msg_data
        );
        */

--Get message count and data
--dbms_output.put_line('err msg ' || k || 'is: ' || gpr_msg_data);

    null;

```

```

        end loop;
Rollback;

        WHEN OTHERS THEN
            gpr_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;

--Get message count and data
--dbms_output.put_line('err msg 3 is : ' || gpr_msg_data);
Rollback;
    end;
    /
    commit;
    exit;

```

5. The following code can also be found in file QPPLXMP5.sql in \$QP_TOP/patch/115/sql directory:

```

--set serveroutput on

/*****
    Sample script to which updates the list price on a Price List Line ,

    This script must be modified by the user such that the gpr_list_line_
    tbl(K).list_line_id column is populated with a valid list_line_id from the
    instance where this script is run.

    Please read the Oracle Advanced Pricing User's Guide (Appendix A & B) to
    understand the flexfields and seed data.
    *****/
/

declare

    gpr_return_status varchar2(1) := NULL;
    gpr_msg_count number := 0;
    gpr_msg_data varchar2(2000);
    gpr_price_list_line_tbl QP_PRICE_LIST_PUB.Price_List_Line_Tbl_Type;
    gpr_price_list_line_val_tbl QP_PRICE_LIST_PUB.Price_List_Line_Val_Tbl_Type;

    ppr_price_list_rec QP_PRICE_LIST_PUB.Price_List_Rec_Type;
    ppr_price_list_val_rec QP_PRICE_LIST_PUB.Price_List_Val_Rec_Type;
    ppr_price_list_line_tbl QP_PRICE_LIST_PUB.Price_List_Line_Tbl_Type;
    ppr_price_list_line_val_tbl QP_PRICE_LIST_PUB.Price_List_Line_Val_Tbl_Type;
    ppr_qualifiers_tbl QP_Qualifier_Rules_Pub.Qualifiers_Tbl_Type;
    ppr_qualifiers_val_tbl QP_Qualifier_Rules_Pub.Qualifiers_Val_Tbl_Type;

```

```

ppr_pricing_attr_tbl QP_PRICE_LIST_PUB.Pricing_Attr_Tbl_Type;
ppr_pricing_attr_val_tbl QP_PRICE_LIST_PUB.Pricing_Attr_Val_Tbl_Type;

K number := 1;

begin

/* Populate the list_line_id (PK) of the price list line whose list_price is to
be updated, the operation and the columns to be updated with the new values. All
other values are not required to be populated.*/

K := 1;

    gpr_price_list_line_tbl(K).list_line_id := 293195;

-- Corresponds to a the item 'dw01' on Price List 'Testing 1023'

    gpr_price_list_line_tbl(K).operation := QP_GLOBALS.G_OPR_UPDATE;
    gpr_price_list_line_tbl(K).operand := 25;

--The operand column corresponds to the listprice on a pricelist line.
--dbms_output.put_line('before process price list ');

```

QP_PRICE_LIST_PUB.Process_Price_List

```

(   p_api_version_number=> 1
,   p_init_msg_list=> FND_API.G_FALSE
,   p_return_values=> FND_API.G_FALSE
,   p_commit=> FND_API.G_FALSE
,   x_return_status=> gpr_return_status
,   x_msg_count=> gpr_msg_count
,   x_msg_data=> gpr_msg_data
,   p_PRICE_LIST_LINE_tbl=> gpr_price_list_line_tbl
,   x_PRICE_LIST_rec=> ppr_price_list_rec
,   x_PRICE_LIST_val_rec=> ppr_price_list_val_rec
,   x_PRICE_LIST_LINE_tbl=> ppr_price_list_line_tbl
,   x_PRICE_LIST_LINE_val_tbl=> ppr_price_list_line_val_tbl
,   x_QUALIFIERS_tbl=> ppr_qualifiers_tbl
,   x_QUALIFIERS_val_tbl=> ppr_qualifiers_val_tbl
,   x_PRICING_ATTR_tbl=> ppr_pricing_attr_tbl
,   x_PRICING_ATTR_val_tbl=> ppr_pricing_attr_val_tbl
);

IF gpr_return_status <> FND_API.G_RET_STS_SUCCESS THEN

```

```

        RAISE FND_API.G_EXC_UNEXPECTED_ERROR;

    END IF;

    --dbms_output.put_line('after process price list ');

    EXCEPTION

        WHEN FND_API.G_EXC_ERROR THEN

            gpr_return_status := FND_API.G_RET_STS_ERROR;

            --Get message count and data

            --dbms_output.put_line('err msg 1 is : ' || gpr_msg_data);
            Rollback;

        WHEN FND_API.G_EXC_UNEXPECTED_ERROR THEN

            gpr_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;

            --dbms_output.put_line(' msg count 2 is : ' || gpr_msg_count);

            for k in 1 .. gpr_msg_count loop

                gpr_msg_data := oe_msg_pub.get( p_msg_index => k,

                p_encoded => 'F'

                );

                /*

                oe_msg_pub.Count_And_Get
                (   p_count=> gpr_msg_count
                ,   p_data=> gpr_msg_data
                );

                */

            -- Get message count and data

            --dbms_output.put_line('err msg ' || k || 'is: ' || gpr_msg_data);

```

```
null;

end loop;
    Rollback;
    WHEN OTHERS THEN

        gpr_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;

--Get message count and data
        --dbms_output.put_line('err msg 3 is : ' || gpr_msg_data);
Rollback;
end;
/
commit;
exit;
```

See

Oracle Pricing Technical Reference Manual

Price List Setup Group Application Program Interface

This section explains how to use the Price List Setup Group API and how it functions in Oracle Advanced Pricing. The Price List Setup package consists of entities to set up price lists.

Functional Overview

The Price List Setup package `QP_Price_List_GRP.Process_Price_List` contains the following public record type and table of records entities:

- `Process_Price_List`: `QP_Price_List_GRP.Process_Price_List`: Takes two record types and six table types as input parameters. Use this API to insert, update, and delete price lists and to set up a price list for a given `p_PRICE_LIST_rec` record structure.

You can:

- Set up multiple price list lines by giving multiple price list line definitions in the `p_PRICE_LIST_LINE_tbl` table structure.
- Attach multiple qualifiers at the price list header level by giving multiple qualifiers in the `p_QUALIFIERS_tbl` table structure.
- Attach multiple pricing attributes to price list lines by giving the pricing attributes in the `p_PRICING_ATTR_tbl` table structure.
- `Price_List_Rec_Type`: Corresponds to the columns in the price list header tables `QP_LIST_HEADERS_B` and `QP_LIST_HEADERS_TL`.
- `Price_List_Val_Rec_Type`: Attributes that store the meaning of id or code columns in the price list header table `QP_LIST_HEADERS_B`, for example, Currency.
- `Price_List_Line_Rec_Type`: Corresponds to columns in the price list line table and related modifiers tables `QP_LIST_LINES` and `QP_RLTD_MODIFIERS`.
- `Price_List_Line_Tbl_Type`: Table of `Price_List_Line_Rec_Type`.
- `Price_List_Line_Val_Rec_Type`: Attributes that store the meaning of id or code columns in the price list line table `QP_LIST_LINES`, for example, Price_By_Formula.
- `Price_List_Line_Val_Tbl_Type`: Table of `Price_List_Line_Val_Rec_Type`.
- `Qualifiers_Rec_Type`: Corresponds to the columns in the qualifier table `QP_QUALIFIERS`.

- Qualifiers_Tbl_Type: Table of Qualifiers_Rec_Type.
- Qualifiers_Val_Rec_Type: Made up of attributes that store the meaning of id or code columns in the qualifiers table QP_QUALIFIERS, for example, Qualifier_Rule.
- Qualifiers_Val_Tbl_Type: Table of Qualifiers_Val_Rec_Type.
- Pricing_Attr_Rec_Type: Corresponds to the columns in the pricing attributes table QP_PRICING_ATTRIBUTES.
- Pricing_Attr_Tbl_Type: Table of Pricing_Attr_Rec_Type.
- Pricing_Attr_Val_Rec_Type: Attributes that store the meaning of id or code columns in the pricing attributes table QP_PRICING_ATTRIBUTES, for example, Accumulate.
- Pricing_Attr_Val_Tbl_Type: Table of Pricing_Attr_Val_Rec_Type.

Setting Up and Parameter Descriptions

The following chart describes all parameters used by the Price List Group API. All of the inbound and outbound parameters are listed. Additional information on these parameters may follow.

PROCESS_PRICE_LIST

The following table shows the parameters for this structure.

Table 5–121 PROCESS_PRICE_LIST

Parameter	Usage	Type	Req	Drv
p_api_version_number	In	Number	No	No
p_init_msg_list	In	Varchar2	No	No
p_return_values	In	Varchar2	No	No
p_commit	In	Varchar2	No	No
x_return_status	Out	Varchar2	No	No
x_msg_count	Out	Number	No	No
x_msg_data	Out	Varchar2	No	No
p_PRICE_LIST_rec	In	Price_List_Rec_Type	No	No
p_PRICE_LIST_val_rec	In	Price_List_Val_Rec_Type	No	No

Table 5–121 PROCESS_PRICE_LIST

Parameter	Usage	Type	Req	Drv
p_PRICE_LIST_tbl	In	Price_List_Line_tbl_type	No	No
p_PRICE_LIST_val_tbl	In	Price_List_Line_Val_tbl_type	No	No
p_QUALIFIERS_tbl	In	QP_Qualifier_Rules_PUB.Qualifiers_tbl_Type	No	No
p_QUALIFIERS_val_tbl	In	QP_Qualifier_Rules_PUB.Qualifiers_val_tbl_Type	No	No
p_PRICING_ATTR_tbl	In	Pricing_Attr_tbl_type	No	No
p_PRICING_ATTR_val_tbl	In	Pricing_Attr_val_tbl_type	No	No
x_PRICE_LIST_rec	Out	Price_List_Rec_Type	No	No
x_PRICE_LIST_val_rec	Out	Price_List_Val_Rec_Type	No	No
x_PRICE_LIST_LINE_tbl	Out	Price_List_Line_tbl_type	No	No
x_PRICE_LIST_LINE_val_tbl	Out	Price_List_Line_Val_tbl_type	No	No
x_QUALIFIERS_tbl	Out	QP_Qualifier_Rules_PUB.Qualifiers_tbl_Type	No	No
x_QUALIFIERS_val_tbl	Out	QP_Qualifier_Rules_PUB.Qualifiers_val_tbl_Type	No	No
x_PRICING_ATTR_tbl	Out	Pricing_Attr_tbl_type	No	No
x_PRICING_ATTR_val_tbl	Out	Pricing_Attr_val_tbl_type	No	No

A key of the short names and definitions used in the API tables are provided in the following table:

Table 5–122 Short Name Key

Short name	Definition
Drv	Derived
Req	Required Yes : This is a required parameter. No : This is an optional parameter.
N/A (no entry)	No value/not applicable

PRICE_LIST_REC_TYPE

The following table shows the parameters for this structure.

Table 5–123 PRICE_LIST_REC_TYPE

Parameter	Usage	Type	Req	Drv
Attribute1	Null	Varchar2	No	No
Attribute2	Null	Varchar2	No	No
Attribute3	Null	Varchar2	No	No
Attribute4	Null	Varchar2	No	No
Attribute5	Null	Varchar2	No	No
Attribute6	Null	Varchar2	No	No
Attribute7	Null	Varchar2	No	No
Attribute8	Null	Varchar2	No	No
Attribute9	Null	Varchar2	No	No
Attribute10	Null	Varchar2	No	No
Attribute11	Null	Varchar2	No	No
Attribute12	Null	Varchar2	No	No
Attribute13	Null	Varchar2	No	No
Attribute14	Null	Varchar2	No	No
Attribute15	Null	Varchar2	No	No
automatic_flag	Null	Varchar2	No	No
comments	Null	Varchar2	No	No
context	Null	Varchar2	No	No
created_by	Null	Number	No	No
creation_date	Null	Date	No	No
currency_code	Null	Varchar2	Yes	No
discount_lines_flag	Null	Varchar2	No	No
end_date_active	Null	Date	No	No
freight_terms_code	Null	Varchar2	No	No
gsa_indicator	Null	Varchar2	No	No
last_updated_by	Null	Number	No	No
last_update_date	Null	Date	No	No

Table 5–123 PRICE_LIST_REC_TYPE

Parameter	Usage	Type	Req	Drv
last_update_login	Null	Number	No	No
list_header_id	Null	Number	No	No
list_type_code	Null	Varchar2	No	No
program_application_id	Null	Number	No	No
program_id	Null	Number	No	No
program_update_date	Null	Date	No	No
prorate_flag	Null	Varchar2	No	No
request_id	Null	Number	No	No
rounding_factor	Null	Number	No	No
ship_method_code	Null	Varchar2	No	No
start_date_active	Null	Date	No	No
terms_id	Null	Number	No	No
return_status	Null	Varchar2	No	No
db_flag	Null	Varchar2	No	No
operation	Null	Varchar2	Yes	No
name	Null	Varchar2	Yes	No
description	Null	Varchar2	No	No
version_no	Null	Varchar2	No	No
Active_flag	Null	Varchar2	No	No
mobile_download	Null	Varchar2	No	No
currency_header_id	Null	Number	No	No
list_source_code	Null	Varchar2	No	No
orig_system_header_ref	Null	Varchar2	No	No
global_flag	Null	Varchar2	No	No
orig_org_id	Null	Number	No	No

PRICE_LIST_TBL_TYPE

The following table shows the parameters for this structure.

Table 5–124 PRICE_LIST_TBL_TYPE

Parameter	Usage	Type	Req	Drv
Price_List_Rec_Type	Null	Record	No	No

PRICE_LIST_VAL_REC_TYPE

The following table shows the parameters for this structure.

Table 5–125 PRICE_LIST_VAL_REC_TYPE

Parameter	Usage	Type	Req	Drv
automatic	Null	Varchar2	No	No
Currency	Null	Varchar2	No	No
discount_lines	Null	Varchar2	No	No
freight_terms	Null	Varchar2	No	No
list_header	Null	Varchar2	No	No
list_type	Null	Varchar2	No	No
Prorate	Null	Varchar2	No	No
ship_method	Null	Varchar2	No	No
Terms	Null	Varchar2	No	No
currency_header	Null	Varchar2	No	No

PRICE_LIST_VAL_TBL_TYPE

The following table shows the parameters for this structure.

Table 5–126 PRICE_LIST_VAL_TBL_TYPE

Parameter	Usage	Type	Req	Drv
Price_List_Val_Rec_Type	Null	Record	No	No

PRICE_LIST_LINE_REC_TYPE

The following table shows the parameters for this structure.

Table 5–127 PRICE_LIST_LINE_REC_TYPE

Parameter	Usage	Type	Req	Drv
accrual_qty	Null	Number	No	No
accrual_uom_code	Null	Varchar2	No	No
arithmetic_operator	Null	Varchar2	No	No
attribute1	Null	Varchar2	No	No
attribute2	Null	Varchar2	No	No
attribute3	Null	Varchar2	No	No
attribute4	Null	Varchar2	No	No
attribute5	Null	Varchar2	No	No
Attribute6	Null	Varchar2	No	No
Attribute7	Null	Varchar2	No	No
Attribute8	Null	Varchar2	No	No
Attribute9	Null	Varchar2	No	No
Attribute10	Null	Varchar2	No	No
Attribute11	Null	Varchar2	No	No
Attribute12	Null	Varchar2	No	No
Attribute13	Null	Varchar2	No	No
Attribute14	Null	Varchar2	No	No
Attribute15	Null	Varchar2	No	No
automatic_flag	Null	Varchar2	No	No
base_qty	Null	Number	No	No
base_uom_code	Null	Varchar2	No	No
comments	Null	Varchar2	No	No
context	Null	Varchar2	No	No
created_by	Null	Number	No	No
creation_date	Null	Date	No	No
effective_period_uom	Null	Varchar2	No	No
end_date_active	Null	Date	No	No

Table 5–127 PRICE_LIST_LINE_REC_TYPE

Parameter	Usage	Type	Req	Drv
estim_accrual_rate	Null	Number	No	No
generate_using_formula_id	Null	Number	No	No
inventory_item_id	Null	Number	No	No
last_updated_by	Null	Number	No	No
last_update_date	Null	Date	No	No
last_update_login	Null	Number	No	No
list_header_id	Null	Number	No	No
list_line_id	Null	Number	No	No
list_line_type_code	Null	Varchar2	No	No
list_price	Null	Number	No	No
modifier_level_code	Null	Varchar2	No	No
number_effective_periods	Null	Number	No	No
operand	Null	Number	No	No
organization_id	Null	Number	No	No
override_flag	Null	Varchar2	No	No
percent_price	Null	Number	No	No
price_break_type_code	Null	Varchar2	No	No
price_by_formula_id	Null	Number	No	No
primary_uom_flag	Null	Varchar2	No	No
print_on_invoice_flag	Null	Varchar2	No	No
program_application_id	Null	Number	No	No
program_id	Null	Number	No	No
program_update_date	Null	Date	No	No
rebate_trxn_type_code	Null	Varchar2	No	No
related_item_id	Null	Number	No	No
relationship_type_id	Null	Number	No	No
reprice_flag	Null	Varchar2	No	No

Table 5–127 PRICE_LIST_LINE_REC_TYPE

Parameter	Usage	Type	Req	Drv
request_id	Null	Number	No	No
revision	Null	Varchar2	No	No
revision_date	Null	Date	No	No
revision_reason_code	Null	Varchar2	No	No
start_date_active	Null	Date	No	No
substitution_attribute	Null	Varchar2	No	No
substitution_context	Null	Varchar2	No	No
substitution_value	Null	Varchar2	No	No
return_status	Null	Varchar2	No	No
db_flag	Null	Varchar2	No	No
operation	Null	Varchar2	No	No
from_rltd_modifier_id	Null	Number	No	No
rltd_modifier_group_no	Null	Number	No	No
product_precedence	Null	Number	No	No

PRICE_LIST_LINE_TBL_TYPE

The following table shows the parameters for this structure.

Table 5–128 PRICE_LIST_LINE_TBL_TYPE

Parameter	Usage	Type	Req	Drv
Price_List_Line_Rec_Type	Null	Record	No	No

PRICE_LIST_LINE_VAL_REC_TYPE

The following table shows the parameters for this structure.

Table 5–129 PRICE_LIST_LINE_VAL_REC_TYPE

Parameter	Usage	Type	Req	Drv
accrual_uom	Null	Varchar2	No	No
automatic	Null	Varchar2	No	No
base_uom	Null	Varchar2	No	No
generate_using_formula	Null	Varchar2	No	No
inventory_item	Null	Varchar2	No	No
list_header	Null	Varchar2	No	No
list_line	Null	Varchar2	No	No
list_line_type	Null	Varchar2	No	No
modifier_level	Null	Varchar2	No	No
organization	Null	Varchar2	No	No
override	Null	Varchar2	No	No
price_break_type	Null	Varchar2	No	No
price_by_formula	Null	Varchar2	No	No
primary_uom	Null	Varchar2	No	No
print_on_invoice	Null	Varchar2	No	No
rebate_transaction_type	Null	Varchar2	No	No
related_item	Null	Varchar2	No	No
relationship_type	Null	Varchar2	No	No
reprice	Null	Varchar2	No	No
revision_reason	Null	Varchar2	No	No

PRICE_LIST_LINE_VAL_TBL_TYPE

The following table shows the parameters for this structure.

Table 5–130 PRICE_LIST_LINE_VAL_TBL_TYPE

Parameter	Usage	Type	Req	Drv
Price_List_Line_Val_Rec_Type	Null	Record	No	No

QUALIFIERS_REC_TYPE

The following table shows the parameters for this structure.

Table 5–131 QUALIFIERS_REC_TYPE

Parameter	Usage	Type	Req	Drv
QP_Qualifier_Rules_PUB.Qualifiers_Rec_Type	Null	Record	No	No

Refer to the Qualifiers public API for the definition.

QUALIFIERS_TBL_TYPE

The following table shows the parameters for this structure.

Table 5–132 QUALIFIERS_TBL_TYPE

Parameter	Usage	Type	Req	Drv
QP_Qualifier_Rules_PUB.Qualifiers_Tbl_Type	Null	Record	No	No

Refer to the Qualifiers public API for the definition.

QUALIFIERS_VAL_REC_TYPE

The following table shows the parameters for this structure.

Table 5–133 QUALIFIERS_VAL_REC_TYPE

Parameter	Usage	Type	Req	Drv
QP_Qualifier_Rules_PUB.Qualifiers_Val_Rec_Type	Null	Record	No	No

Refer to the Qualifiers public API for the definition.

QUALIFIERS_VAL_TBL_TYPE

The following table shows the parameters for this structure.

Table 5–134 QUALIFIERS_VAL_TBL_TYPE

Parameter	Usage	Type	Req	Drv
QP_Qualifier_Rules_PUB.Qualifiers_Val_Tbl_Type	Null	Record	No	No

Refer to the Qualifiers public API for the definition.

Note: For setting up Secondary Price List, Create a qualifier with the parameters:

QUALIFIER_CONTEXT - 'MODLIST'

QUALIFIER_ATTRIBUTE - 'QUALIFIER_ATTRIBUTE4'

QUALIFIER_ATTR_VALUE - <list_header_id of the primary price list>

LIST_HEADER_ID - <list_header_id of the secondary price list>

COMPARISON_OPERATOR_CODE - '='.

See the Example 2 for the details about how to setup a Secondary Price List.

PRICING_ATTR_REC_TYPE

The following table shows the parameters for this structure.

Table 5–135 PRICING_ATTR_REC_TYPE

Parameter	Usage	Type	Req	Drv
accumulate_flag	Null	Varchar2	No	No
attribute1	Null	Varchar2	No	No
attribute2	Null	Varchar2	No	No
attribute3	Null	Varchar2	No	No
attribute4	Null	Varchar2	No	No
attribute5	Null	Varchar2	No	No
attribute6	Null	Varchar2	No	No
attribute7	Null	Varchar2	No	No
attribute8	Null	Varchar2	No	No
attribute9	Null	Varchar2	No	No

Table 5–135 PRICING_ATTR_REC_TYPE

Parameter	Usage	Type	Req	Drv
attribute10	Null	Varchar2	No	No
attribute11	Null	Varchar2	No	No
attribute12	Null	Varchar2	No	No
attribute13	Null	Varchar2	No	No
attribute14	Null	Varchar2	No	No
attribute15	Null	Varchar2	No	No
attribute_grouping_number	Null	Number	No	No
Context	Null	Varchar2	No	No
created_by	Null	Number	No	No
creation_date	Null	Date	No	No
excluder_flag	Null	Varchar2	No	No
last_updated_by	Null	Number	No	No
last_update_date	Null	Date	No	No
last_update_login	Null	Number	No	No
list_line_id	Null	Number	No	No
pricing_attribute	Null	Varchar2	No	No
pricing_attribute_context	Null	Varchar2	No	No
pricing_attribute_id	Null	Number	No	No
pricing_attr_value_from	Null	Varchar2	No	No
pricing_attr_value_to	Null	Varchar2	No	No
product_attribute	Null	Varchar2	Yes	No
product_attribute_context	Null	Varchar2	Yes	No
product_attr_value	Null	Varchar2	Yes	No
product_uom_code	Null	Varchar2	Yes	No
program_application_id	Null	Number	No	No
program_id	Null	Number	No	No
program_update_date	Null	Date	No	No

Table 5–135 PRICING_ATTR_REC_TYPE

Parameter	Usage	Type	Req	Drv
request_id	Null	Number	No	No
return_status	Null	Varchar2	No	No
db_flag	Null	Varchar2	No	No
Operation	Null	Varchar2	Yes	No
PRICE_LIST_LINE_index	Null	Number	No	No
from_rltd_modifier_id	Null	Number	No	No
comparison_operator_code	Null	Varchar2	Yes	No
product_attribute_datatype	Null	Varchar2	No	Yes
pricing_attribute_datatype	Null	Varchar2	No	Yes

PRICING_ATTR_TBL_TYPE

The following table shows the parameters for this structure.

Table 5–136 PRICING_ATTR_TBL_TYPE

Parameter	Usage	Type	Req	Drv
Pricing_Attr_Rec_Type	Null	Record	No	No

PRICING_ATTR_VAL_REC_TYPE

The following table shows the parameters for this structure.

Table 5–137 PRICING_ATTR_VAL_REC_TYPE

Parameter	Usage	Type	Req	Drv
Accumulate	Null	Varchar2	No	No
Excluder	Null	Varchar2	No	No
list_line	Null	Varchar2	No	No
pricing_attribute	Null	Varchar2	No	No
product_uom	Null	Varchar2	No	No

PRICING_ATTR_VAL_TBL_TYPE

The following table shows the parameters for this structure.

Table 5–138 PRICING_ATTR_VAL_TBL_TYPE

Parameter	Usage	Type	Req	Drv
Pricing_Attr_Val_Rec_Type	Null	Record	No	No

Validation of Price List Group API

Standard Validation

Oracle Advanced Pricing validates all required columns in the Price List Group API. For more information, see: *Oracle Pricing Technical Reference Manual*.

Other Validation

None

Error Handling

If any validation fails, the API will return error status to the calling module. The Price List Group API processes the rows and reports the values in the following table for every record.

Table 5–139 Error Handling

Condition	PROCESS_STATUS	ERROR_MESSAGE
Success	5	null
Failure	4	actual error message

Price Request Application Program Interface

This section explains how to use the Price Request API and how it functions in Oracle Pricing. The Price Request group API has procedures to be called to pass the request information to the pricing engine. This API also has the different constants that the pricing engine uses for error status, codes and so on.

Price Request API Features

The Price Request Application Program Interface (API) is a public API that allows you to get a base price and to apply price adjustments, other benefits, and charges to a transaction.

Oracle Applications products request it for pricing calculations and you can request it from custom applications and legacy systems.

A pricing request consists of numerous price request lines which mirror the transaction lines of the calling application and may include a transaction header request line. Since it is PL/SQL based, the Pricing Request processes one pricing request per call.

To properly use the Price Request Application Program Interface, pass all lines that need prices and that the pricing engine needs to consider as part of pricing request. For example, you may have frozen the price of one order line but, if you include it in the pricing request, the pricing engine may be able to use the quantity on that line to qualify the order to receive another discount based on quantities across multiple lines.

The Price Request Application Program Interface consists of two engines:

- Search engine: Uses qualifiers and pricing attributes passed from the calling application to select the price list lines and the modifier list lines that may apply to the pricing request. As part of this process, the search engine uses rules of eligibility, incompatibility, exclusivity, and precedence.

For each pricing phase the search engine executes the following functions:

- Selects eligible price list lines and modifier list lines using predefined pricing rules.
- Resolve incompatibilities among eligible benefits.
- Applies the eligible benefits to the pricing request.

- Calculation engine: For each pricing request line and its associated pricing request line details, calculates the base price, adjusted price, and extended price.

You can call one or both of the engines by setting the calculate flag on the control record.

Functional Overview

The Price Request public API QP_PREQ_PUB contains the following entities: PRICE_REQUEST

- CONTROL_RECORD_TYPE: Parameters which control the behavior of the pricing engine.
- LINE_REC_TYPE: A record which contains the elements in the calling application that requires a base and adjusted price. It may equate to a transaction line or transaction header record in the calling application.
- LINE_DETAIL_REC_TYPE: A record that contains the details of the derivation of the base and adjusted prices. Each pricing request line detail provides details for a price list line or modifier list line. The pricing engine may apply many pricing request line detail records to a pricing request.
- QUAL_REC_TYPE: A record that contains qualifier information. Qualifier information helps the pricing engine to determine the price list lines and modifier list lines for which a pricing request is eligible. The pricing engine returns all qualifiers that determined eligibility on the request line detail record.
- LINE_ATTR_REC_TYPE: A record that contains pricing attributes. Pricing attribute information helps the pricing engine to determine the price list lines and modifier list lines for which a pricing request is eligible. The calling application must load all pricing attributes into this record type that the pricing engine should use to determine if a pricing request line qualifies for a price or modifier.
- RELATED_LINES_REC_TYPE: A record that contains relationships between request lines and between request lines and request line details. Types of relationships are as follows:
 - PBH_LINE: Relates a price break header to modifier price break lines.
 - SERVICE_LINE: Relates an order line for a service item and its parent, the serviceable item. The pricing engine needs to know this relationship when it must price service items based on a percent of the serviceable item price.

- **GENERATED_LINE**: Indicates the lines—both request and detail that a pricing request line or pricing request detail line created.

Setting Up and Parameter Descriptions

The following chart describes all parameters used by the public Price Request. All of the inbound and outbound parameters are listed. Additional information on these parameters follows.

Insert_Lines2

This API takes the request line information and does a bulk insert into the pricing temporary table which holds the request lines, qp_preq_lines_tmp. Each of the input parameter is a pl/sql table to enable the bulk insert. In case there is an error, this API returns x_return_status as FND_API.G_RET_STS_ERROR.

QP_PREQ_GRP.INSERT_LINES2

Table 5–140 QP_PREQ_GRP.INSERT_LINES2

Parameter	Usage	Type
p_LINE_INDEX	In	QP_PREQ_GRP.PLS_INTEGER_TYPE,
p_LINE_TYPE_CODE	In	QP_PREQ_GRP.VARCHAR_TYPE,
p_PRICING_EFFECTIVE_DATE	In	QP_PREQ_GRP.DATE_TYPE
p_ACTIVE_DATE_FIRST	In	QP_PREQ_GRP.DATE_TYPE
p_ACTIVE_DATE_FIRST_TYPE	In	QP_PREQ_GRP.VARCHAR_TYPE,
p_ACTIVE_DATE_SECOND	In	QP_PREQ_GRP.DATE_TYPE
p_ACTIVE_DATE_SECOND_TYPE	In	QP_PREQ_GRP.VARCHAR_TYPE ,
p_LINE_QUANTITY	In	QP_PREQ_GRP.NUMBER_TYPE ,
p_LINE_UOM_CODE	In	QP_PREQ_GRP.VARCHAR_TYPE,
p_REQUEST_TYPE_CODE	In	QP_PREQ_GRP.VARCHAR_TYPE,
p_PRICED_QUANTITY	In	QP_PREQ_GRP.NUMBER_TYPE,
p_PRICED_UOM_CODE	In	QP_PREQ_GRP.VARCHAR_TYPE,
p_CURRENCY_CODE	In	QP_PREQ_GRP.VARCHAR_TYPE,
p_UNIT_PRICE	In	QP_PREQ_GRP.NUMBER_TYPE,

Table 5–140 QP_PREQ_GRP.INSERT_LINES2

Parameter	Usage	Type
p_PERCENT_PRICE	In	QP_PREQ_GRP.NUMBER_TYPE,
p_UOM_QUANTITY	In	QP_PREQ_GRP.NUMBER_TYPE,
p_ADJUSTED_UNIT_PRICE	In	QP_PREQ_GRP.NUMBER_TYPE,
p_UPD_ADJUSTED_UNIT_PRICE	In	QP_PREQ_GRP.NUMBER_TYPE,
p_PROCESSED_FLAG	In	QP_PREQ_GRP.VARCHAR_TYPE,
p_PRICE_FLAG	In	QP_PREQ_GRP.VARCHAR_TYPE,
p_LINE_ID	In	QP_PREQ_GRP.NUMBER_TYPE,
p_PROCESSING_ORDER	In	QP_PREQ_GRP.PLS_INTEGER_TYPE,
p_PRICING_STATUS_CODE	In	QP_PREQ_GRP.VARCHAR_TYPE,
p_PRICING_STATUS_TEXT	In	QP_PREQ_GRP.VARCHAR_TYPE,
p_ROUNDING_FLAG	In	QP_PREQ_GRP.FLAG_TYPE,
p_ROUNDING_FACTOR	In	QP_PREQ_GRP.PLS_INTEGER_TYPE,
p_QUALIFIERS_EXIST_FLAG	In	QP_PREQ_GRP.VARCHAR_TYPE,
p_PRICING_ATTRS_EXIST_FLAG	In	QP_PREQ_GRP.VARCHAR_TYPE,
p_PRICE_LIST_ID	In	QP_PREQ_GRP.NUMBER_TYPE,
p_VALIDATED_FLAG	In	QP_PREQ_GRP.VARCHAR_TYPE,
p_PRICE_REQUEST_CODE	In	QP_PREQ_GRP.VARCHAR_TYPE,
p_USAGE_PRICING_TYPE	In	QP_PREQ_GRP.VARCHAR_TYPE,
p_LINE_CATEGORY	In	QP_PREQ_GRP.VARCHAR_TYPE := QP_PREQ_GRP.G_LINE_CATEGORY_DEF_TBL
x_status_code	OUT	VARCHAR2
x_status_text	OUT	VARCHAR2

For information on each of these parameters, refer the LINE_REC_TYPE defined in this manual under the price_request API.

Insert_Line_Attrs2

This API takes the attribute information and does a bulk insert into the pricing temporary table which holds the request line attributes, qp_preq_line_attrs_tmp.

Each of the input parameter is a pl/sql table to enable the bulk insert. In case there is an error, this API returns x_return_status as FND_API.G_RET_STS_ERROR.

QP_PREQ_GRP.INSERT_LINE_ATTRS2

Table 5–141 QP_PREQ_GRP.INSERT_LINE_ATTRS2

Parameter	Type
p_LINE_INDEX_tbl	QP_PREQ_GRP.pls_integer_type
p_LINE_DETAIL_INDEX_tbl	QP_PREQ_GRP.pls_integer_type
p_ATTRIBUTE_LEVEL_tbl	QP_PREQ_GRP.varchar_type
p_ATTRIBUTE_TYPE_tbl	QP_PREQ_GRP.varchar_type
p_LIST_HEADER_ID_tbl	QP_PREQ_GRP.number_type
p_LIST_LINE_ID_tbl	QP_PREQ_GRP.number_type
p_CONTEXT_tbl	QP_PREQ_GRP.varchar_type
p_ATTRIBUTE_tbl	QP_PREQ_GRP.varchar_type
p_VALUE_FROM_tbl	QP_PREQ_GRP.varchar_type
p_SETUP_VALUE_FROM_tbl	QP_PREQ_GRP.varchar_type
p_VALUE_TO_tbl	QP_PREQ_GRP.varchar_type
p_SETUP_VALUE_TO_tbl	QP_PREQ_GRP.varchar_type
p_GROUPING_NUMBER_tbl	QP_PREQ_GRP.pls_integer_type
p_NO_QUALIFIERS_IN_GRP_tbl	QP_PREQ_GRP.pls_integer_type
p_COMPARISON_OPERATOR_TYPE_tbl	QP_PREQ_GRP.varchar_type
p_VALIDATED_FLAG_tbl	QP_PREQ_GRP.varchar_type
p_APPLIED_FLAG_tbl	QP_PREQ_GRP.varchar_type
p_PRICING_STATUS_CODE_tbl	QP_PREQ_GRP.varchar_type
p_PRICING_STATUS_TEXT_tbl	QP_PREQ_GRP.varchar_type
p_QUALIFIER_PRECEDENCE_tbl	QP_PREQ_GRP.pls_integer_type
p_DATATYPE_tbl	QP_PREQ_GRP.varchar_type
p_PRICING_ATTR_FLAG_tbl	QP_PREQ_GRP.varchar_type
p_QUALIFIER_TYPE_tbl	QP_PREQ_GRP.varchar_type

Table 5–141 QP_PREQ_GRP.INSERT_LINE_ATTRS2

Parameter	Type
p_PRODUCT_UOM_CODE_TBL	QP_PREQ_GRP.varchar_type
p_EXCLUDER_FLAG_TBL	QP_PREQ_GRP.varchar_type
p_PRICING_PHASE_ID_TBL	QP_PREQ_GRP.pls_integer_type
p_INCOMPATABILITY_GRP_CODE_TBL	QP_PREQ_GRP.varchar_type
p_LINE_DETAIL_TYPE_CODE_TBL	QP_PREQ_GRP.varchar_type
p_MODIFIER_LEVEL_CODE_TBL	QP_PREQ_GRP.varchar_type
p_PRIMARY_UOM_FLAG_TBL	QP_PREQ_GRP.varchar_type
x_status_code	OUT VARCHAR2
x_status_text	OUT VARCHAR2);

For more information on each of these input parameters, refer to the line_attr_rec_type in the price_request API.

PRICE_REQUEST

The Derived value for each parameter is Null.

Table 5–142 PRICE_REQUEST

Parameter	Usage	Type	Req
p_line_tbl	In	LINE_TBL_TYPE	Yes
p_qual_tbl	In	QUAL_TBL_TYPE	No
p_line_attr_tbl	In	LINE_ATTR_TBL_TYPE	No
p_line_detail_tbl	In	LINE_DETAIL_TBL_TYPE	No
p_line_detail_qual_tbl	In	LINE_DETAIL_QUAL_TBL_TYPE	No
p_line_detail_attr_tbl	In	LINE_DETAIL_ATTR_TBL_TYPE	No
p_related_lines_tbl	In	RELATED_LINES_TBL_TYPE	No (req. for Service Line Pricing)
p_control_rec	In	CONTROL_RECORD_TYPE	Yes
x_line_tbl	Out	LINE_TBL_TYPE	Yes
x_qual_tbl	Out	QUAL_TBL_TYPE	No

Table 5–142 PRICE_REQUEST

Parameter	Usage	Type	Req
x_line_attr_tbl	Out	LINE_ATTR_TBL_TYPE	No
x_line_detail_tbl	Out	LINE_DETAIL_TBL_TYPE	Yes
x_line_detail_qual_tbl	Out	LINE_DETAIL_QUAL_TBL_TYPE	No
x_line_detail_attr_tbl	Out	LINE_DETAIL_ATTR_TBL_TYPE	No
x_related_lines_tbl	Out	RELATED_LINES_TBL_TYPE	No
x_return_status	Out	Varchar2	Yes
x_return_status_text	Out	Varchar2	Yes

PRICE_REQUEST(Overloaded)

This API has been overloaded for performance features. The request line and attributes are inserted into the pricing temporary tables prior to calling this API. Please refer to the performance features section under this API for more details regarding this call.

Table 5–143 PRICE_REQUEST(Overloaded)

Parameter	Usage	Type	Req
p_control_rec	In	CONTROL_REC_TYPE	Yes
x_return_status	Out	Varchar2	Yes
x_return_status_text	Out	Varchar2	Yes

CONTROL_REC_TYPE

The Derived value for each parameter is Null.

Table 5–144 CONTROL_REC_TYPE

Parameter	Type	Req	Usage
PRICING_EVENT	Varchar2	Yes	A point in the transaction life cycle of your transaction at which you wish to price it. The pricing event determines which phases the search engine processes according to the mapping in QP_EVENT_PHASES.

Table 5–144 CONTROL_REC_TYPE

Parameter	Type	Req	Usage
CALCULATE_FLAG	Varchar2	Yes	<p>Use to call the engines. Allowable values are:</p> <p>QP_PREQ_GRP.G_SEARCH_ONLY: Search engine: If you do not want the engine to calculate the selling price.</p> <p>QP_PREQ_GRP.G_CALCULATE_ONLY: Calculate engine: If you are passing the adjustment records to the engine and you want the engine to recalculate the selling price, without retrieving new adjustments.</p> <p>QP_PREQ_GRP.G_SEARCH_N_CALCULATE: Both the calculate and search engines: Regular engine call. Retrieves new adjustments and calculates the selling price</p>
SIMULATION_FLAG	Varchar2	Yes	<p>A value of Yes indicates that the call to is a pricing simulation and that the pricing engine should not make permanent record changes and neither issue or redeem coupons.</p>
ROUNDING_FLAG	Varchar2	Yes	<p>Indicates whether the calculation engine should round the list price and selling price based on the price list rounding factor or the pricing request line record rounding factor. When rounding, the calculation engine rounds all intermediate subtotals for each pricing group sequence. By default, the rounding is enabled in the pricing engine unless the rounding flag is explicitly set to No.</p> <p>Allowable values are:</p> <p>Y: engine applies the rounding factor defined in the price list</p> <p>N: unrounded figures would be returned</p> <p>Q: Refer to the value of the profile QP: Selling Price Rounding Options</p>

Table 5–144 CONTROL_REC_TYPE

Parameter	Type	Req	Usage
GSA_CHECK_FLAG	Varchar2	No	<p>Indicates whether the GSA Violation needs to be checked, provided the customer is not a GSA customer and the selling price falls below the price on the GSA price list.</p> <p>Allowable values are:</p> <p>Y: engine applies the rounding factor defined in the price list</p> <p>N: unrounded figures are returned</p>
TEMP_TABLE_INSERT_FLAG	Varchar2	No	<p>Indicates if the calling application wants the engine to insert the order lines into the QP temporary table(s). The values for this are:</p> <p>Y: The pricing engine will insert the lines into the QP temporary tables</p> <p>N – This means to the engine that the order lines are already there in the temporary tables.</p>
MANUAL_DISCOUNT_FLAG	Varchar2	No	<p>This flag is introduced to support new Release 11i functionality. This value is set by the calling application and the value is based on the profile QP: Return Manual Discounts.</p> <p>Indicates how pricing engine should perform incompatibility processing for manual discounts. he possible values for this profile are:</p> <p>Yes: All the manual discounts will be returned. All the automatic discounts that get deleted as part of incompatibility processing will be returned as manual discounts.</p> <p>No: All automatic and manual discounts will go through incompatibility processing and one of them in each incompatibility group will be returned. In this process an automatic discount might get deleted and a manual discount might get selected.</p>

Table 5–144 CONTROL_REC_TYPE

Parameter	Type	Req	Usage
SOURCE_ORDER_AMOUNT_FLAG	Varchar2	No	<p>This is for internal use of the pricing engine. Indicates whether the pricing engine will source the order amount attribute.</p> <p>Allowable values are:</p> <p>Y: Indicates to the pricing engine to source the order amount. It means the calling application will provide the order amount</p> <p>N: The pricing engine should calculate the order amount</p>
CHECK_CUST_VIEW_FLAG	Varchar2	No	<p>This is for internal use only. Indicates whether the pricing engine should take a look at caller's pricing tables.</p> <p>Allowable values are:</p> <p>Y: Indicates that to access the calling applications data, the source system uses the view provided by calling application.</p> <p>N: Indicates that the calling application directly provides the data to the source system</p>
REQUEST_TYPE_CODE	Varchar2	Yes	Identifies the transaction system making the pricing request.
VIEW_CODE	Varchar2	No	<p>This is for internal use only. Indicates the view code for the transaction system of the caller.</p> <p>It indicates database view id, which the source system uses to access the calling application's data.</p>
USE_MULTI_CURRENCY	Varchar2	No	<p>Indicates if the calling application wants to use multi-currency.</p> <p>Valid values: 'Y' or 'N'</p> <p>Default value: 'N'</p>
USER_CONVERSION_RATE	Number	No	User defined Conversion rate, used for multi-currency
USER_CONVERSION_TYPE	Varchar2	No	User defined conversion type, used for multi-currency

Table 5–144 CONTROL_REC_TYPE

Parameter	Type	Req	Usage
FUNCTION_CURRENCY	Varchar2	No	Functional currency, used for multi-currency
FULL_PRICING_CALL	Varchar2	No	Indicates if the calling application passed only changed lines to the pricing engine or all the lines during reprice. Set to 'Y' if passed all the lines, 'N' if passed only changed lines. Refer to the changed lines feature
GET_FREIGHT_FLAG	Varchar2	No	When the control record get_freight_flag is set to Y, the pricing engine will only process the phases that have freight charges existing. The Pricing engine will return the modifiers (including freight charge modifiers and non freight charge modifiers) that are qualified in these phases.

The pricing event determines the pricing phases to be run during the call to pricing engine. If you create your own pricing phase, associate the pricing events to phases using screen setup _ event phases. Each modifier is associated to one pricing phase. Pricing engine looks only at those modifiers associated with the phases defined for the pricing event passed.

The following are some of the seeded pricing events:

Table 5–145 Seeded Pricing Events

Pricing Event	Pricing Phase
PRICE	List Line Base Price
LINE	List Line Base Price
LINE	List Line Adjustment
LINE	Line Charges
LINE	Line Charges - Manual
BOOK	Modifiers for Book event
ORDER	All Lines Adjustments

Table 5–145 Seeded Pricing Events

Pricing Event	Pricing Phase
ORDER	Header Level Adjustments
ORDER	Header Level Charges
SHIP	Line Charges
BATCH	Pricing phases in LINE+ORDER
ICBATCH	Pricing phases in LINE+ORDER. This event is specific to Inventory Intercompany Invoice Transfer Pricing

LINE_REC_TYPE**Table 5–146 LINE_REC_TYPE**

Parameter	Datatype	Req	Drv	Type	Usage
REQUEST_TYPE_CODE	Varchar2	No	No	In	Identifies the transaction system making the pricing request.
PRICING_EVENT	Varchar2	No	No	N/A	Not used
HEADER_ID	Number	No	No	N/A	Not used
LINE_INDEX	Number	Yes	No	In/Out	PL/SQL unique identifier for request line.
LINE_ID	Number	No	No	In	Unique identifier of the request line in the calling application.
LINE_TYPE_CODE	Varchar2	Yes	No	In	Type of line within the request. Allowable values are: <ul style="list-style-type: none">■ ORDER■ LINE
PRICING_EFFECTIVE_DATE	Date	Yes	No	In/Out	Date for which the pricing engine calculates the prices.

Table 5–146 *LINE_REC_TYPE*

Parameter	Datatype	Req	Drv	Type	Usage
ACTIVE_DATE_FIRST	Date	No	No	In/Out	In addition to the pricing effective date, you can specify two additional dates for the pricing engine to use to qualify pricing entities. The pricing engine compares this date against the first date range on the modifier list—QP_LIST_HEADERS_B.START_DATE_ACTIVE_FIRST and QP_LIST_HEADERS_B.END_DATE_ACTIVE_FIRST.
ACTIVE_DATE_FIRST_TYPE	Varchar2	No	No	In/Out	The date type of ACTIVE_DATE_FIRST based on lookup type EFFECTIVE_DATE_TYPES.
ACTIVE_DATE_SECOND	Date	No	No	In/Out	In addition to the pricing effective date, you can specify two additional dates for the pricing engine to use to qualify pricing entities. The pricing engine compares this date against the first date range on the modifier list—QP_LIST_HEADERS_B.START_DATE_ACTIVE_SECOND and QP_LIST_HEADERS_B.END_DATE_ACTIVE_SECOND.
ACTIVE_DATE_SECOND_TYPE	Varchar2	No	No	In/Out	The date type of ACTIVE_DATE_SECOND based on lookup type EFFECTIVE_DATE_TYPES.
LINE_QUANTITY	Number	Yes	No	In/Out	Pricing request line quantity.
LINE_UOM_CODE	Varchar2	Yes	No	In/Out	Pricing request line unit of measure.
UOM_QUANTITY	Number	No	No	In/Out	Unit of measure quantity, internal value used by engine to perform service pricing.
PRICED_QUANTITY	Number	No	Yes	Out	Quantity of pricing request line that pricing engine has priced.
PRICED_UOM_CODE	Varchar2	No	Yes	Out	Unit of measure in which the pricing engine priced.
CURRENCY_CODE	Varchar2	Yes	No	In/Out	Currency in which the pricing engine priced.
UNIT_PRICE	Number	No	Yes	Out	Unit price of the item that is expressed in <i>Priced UOM Code</i>
PERCENT_PRICE	Number	No	Yes	Out	Price calculated as a percentage of the price of another item.

Table 5–146 *LINE_REC_TYPE*

Parameter	Datatype	Req	Drv	Type	Usage
UPDATED_ADJUSTED_UNIT_PRICE	Number	No	No	In	This is used for overriding the unit selling price of an item to apply a suitable manual adjustment. In this case, the pricing engine will apply a suitable manual adjustment after calculating the unit selling price to match the updated_adjusted_unit_price that the caller has passed in.
ADJUSTED_UNIT_PRICE	Number	No	Yes	Out	Price per unit after the pricing engine applies discounts and surcharges. It indicates the unit price for the service item, which has the percent price.
PARENT_PRICE	Number	No	Yes	Out	When the pricing engine determines the price of an item from the price of another item, the price of the related item. This is used only for service items and it is populated from the serviceable item.
PARENT_QUANTITY	Number	No	Yes	Out	When the pricing engine determines the price of an item from the price of another item, the quantity of the related item.
ROUNDING_FACTOR	Number	No	No	In/Out	The pricing engine will use this factor to which the selling price is rounded to when the pricing engine returns the price. If the caller does not pass this field, the pricing engine will populate the rounding factor from the qualified price list into this field and use it for rounding. If multi-currency is installed and used, the pricing engine will populate the selling rounding factor of the matching currency from the attached multi-currency list (currency conversion criteria) of the qualified price list into this field and use it for rounding
PARENT_UOM_CODE	Varchar2	No	Yes	Out	When the pricing engine determines the price of an item from the price of another item, the unit of measure of the related item.
PRICING_PHASE_ID	Number	No	No		Not used

Table 5–146 LINE_REC_TYPE

Parameter	Datatype	Req	Drv	Type	Usage
PRICE_FLAG	Varchar2	Yes	No	In	<p>Indicates the degree to which the price is frozen. Allowable values, based on lookup type CALCULATE_PRICE_FLAG are:</p> <p>Y (Calculate Price): Apply all prices and modifiers to the request line.</p> <p>N (Freeze Price): Do not apply any prices or modifiers to the request line. Consider the volume of the request line when processing LINEGROUP modifiers for other lines.</p> <p>P (Partial Price): Apply prices and modifiers in phases whose freeze override flag is Y.</p>
PROCESSED_CODE	Varchar2	No	No	Out	Internal code which indicates the stage of engine processing when an error occurred.
STATUS_CODE	Varchar2	No	No	In/Out	<p>Returned status. Allowable values are:</p> <p>N: New record created(All 'N' records are returned back from the pricing engine. These are success records)</p> <p>X: Unchanged(Default status when the line is passed to the pricing engine for processing)</p> <p>U: Updated</p> <p>IPL: Invalid price list(When passed in price list is not found , then an error is given)</p> <p>GSA: GSA violation</p> <p>FER: Error processing formula</p> <p>OER: Other error</p> <p>CALC: Error in calculation engine</p> <p>UOM: Failed to price using unit of measure</p> <p>INVALID_UOM: Invalid unit of measure</p> <p>DUPLICATE_PRICE_LIST: Duplicate price list</p> <p>INVALID_UOM_CONV: Unit of measure conversion not found</p> <p>INVALID_INCOMP: Could not resolve incompatibility</p> <p>INVALID_BEST_PRICE: Could not resolve best price</p>

Table 5–146 LINE_REC_TYPE

Parameter	Datatype	Req	Drv	Type	Usage
STATUS_TEXT	Varchar2	No	No	Out	Returned message.
PRICE_REQUEST_CODE	Varchar2	No	Yes	In/Out	This is unique code to identify each order line consuming a modifier. It is request_type_code concatenated to the header_id and the line_id(applicable only for order lines as line_id is null for header lines) of the order line separated .
HOLD_CODE	Varchar2	No	No	Out	This indicates that a limit having the hold_flag checked has been exceeded or adjusted and is a recommendation to the caller to put the order/line on hold.
HOLD_TEXT	Varchar2	No	No	Out	This is a detailed message corresponding to the hold_code.
USAGE_PRICING_TYPE	Varchar2	No	No	In/Out	Indicates the usage pricing type. REGULAR – Regular pricing AUTHORING – Authoring call BILLING – Billing call
LINE_CATEGORY	Varchar2	Yes	No	In/Out	Indicates the line category. The applicable values: RETURN – Returned line CANCEL – Cancelled line NULL – Regular line
CONTRACT_START_DATE	Date	No	No	In	When UOM conversion is time related and Profile QP: Time UOM Conversion is set to Oracle Contracts, order quantity by order UOM and pricing quantity by pricing UOM are calculated based on contract_start_date, contract_end_date.
CONTRACT_END_DATE	Date	No	No	In	When UOM conversion is time related and Profile QP: Time UOM Conversion is set to Oracle Contracts, order quantity by order UOM and pricing quantity by pricing UOM are calculated based on contract_start_date, contract_end_date.
LINE_UNIT_PRICE	Number	No	Yes	Out	Unit price by order UOM

Types Of Request Lines

1. Regular Order Line where line_type_code = 'LINE'.
2. Summary Line where line_type_code = 'ORDER' which is the summary line for the whole Sales Order. This line has all the order level attributes attached to it. This line is required to be passed to the pricing engine to get any Order Level Discounts based on the attributes or qualifiers that are attached to this Line.
3. Lines that are generated or derived by the pricing engine , in case of discounts like Promotional Good , where in a new order line is created.This line would have the processed code as 'ENGINE'. The relationship between the new line and the original request line is stored in the Relationship record structure RELATED_LINES_REC_TYPE.So, if the engine returns a free good (PRG Modifier) , then there will be a new request line generated by the pricing engine.

LINE_DETAIL_REC_TYPE

Table 5–147 LINE_DETAIL_REC_TYPE

Parameter	Type	Req	Drv	Type	Usage
LINE_DETAIL_INDEX	Number	Yes	Yes	In/Out	PL/SQL unique identifier. Unique identifier of request line detail in calling application.
LINE_DETAIL_ID	Number	No	No	N/A	Not Used
LINE_DETAIL_TYPE_CODE	Varchar2	Yes	No	Out	Type of detail line.
LINE_INDEX	Number	Yes	No	In/Out	Identifier for parent request line.
LIST_HEADER_ID	Number	Yes	No	In/Out	Identifier of the list header used to create or update the pricing line.
LIST_LINE_ID	Number	Yes	No	In/Out	Identifier of the list line used to create or update the pricing line.
LIST_LINE_TYPE_CODE	Varchar2	Yes	No	In/Out	Line type of the list line used to update the pricing line. Possible values can be found from the lookup type LIST_LINE_TYPE_CODE from qp_lookups table.
SUBSTITUTION_TYPE_CODE	Varchar2	No	No		Not used.
SUBSTITUTION_FROM	Varchar2	No	No		Not used.
SUBSTITUTION_TO	Varchar2	No	No	Out	Value for terms substitution attribute.

Table 5–147 LINE_DETAIL_REC_TYPE

Parameter	Type	Req	Drv	Type	Usage
AUTOMATIC_FLAG	Varchar2	Yes	No	Out	Indicates if the pricing engine should automatically apply the request line detail to the request line. The engine derives the value from the list line.
OPERAND_CALCULATION_CODE	Varchar2	Yes	No	In/Out	Type of operand. Allowable values are: Adjustment percent(for discounts) Adjustment amount (for discounts) Adjustment NewPrice(for discounts) UNIT_PRICE(for price lists) PERCENT_PRICE(for price lists)
OPERAND_VALUE	Number	Yes	No	In/Out	Value of pricing request detail line, for example, 10 currency unit list price with 3% discount
PRICING_GROUP_SEQUENCE	Number	Yes	No	In/Out	Indicates the pricing bucket in which the pricing engine applied this list line.
PRICE_BREAK_TYPE_CODE	Varchar2	Yes	No	In/Out	Type of price break based on lookup type PRICE_BREAK_TYPE_CODE. Possible Values: POINT , RANGE , RECURRING
CREATED_FROM_LIST_TYPE_CODE	Varchar2	Yes	No	In/Out	List type used to create or update the pricing line.Possible values can be found from the lookup_type LIST_TYPE_CODE from qp_lookups table.
PRICING_PHASE_ID	Number	Yes	No	In/Out	The pricing phase which created the request line detail.
LIST_PRICE	Number	No	No	Out	Not Used
LINE_QUANTITY	Number	No	Yes	Out	Quantity on the price break line. Populated if the pricing engine derived the value of the request line or request line detail from a price break. A not null value indicates that this particular break line was used in the calculation.

Table 5–147 LINE_DETAIL_REC_TYPE

Parameter	Type	Req	Drv	Type	Usage
ADJUSTMENT_AMOUNT	Number	No	No	Out	It holds the value of the bucketed adjusted amount for line types like PLL , DIS ,SUR etc. For price break(PBH) child lines , the field is populated if the pricing engine derived the value of the request line or request line detail from a price break.
APPLIED_FLAG	Varchar2	No	No	In/Out	The lists or list lines that this pricing event or a prior pricing event applied. Allowable values are Yes: Applicable when the attribute context is a list or list line No: Applicable when the attribute context is a list or list line Null
MODIFIER_LEVEL_CODE	Varchar2	Yes	No	In/Out	The level at which the list line qualified for the transaction. Based on lookup type MODIFIER_LEVEL_CODE.
STATUS_CODE	Varchar2	No	No	Out	Returned status. Possible Values: N: New record created(All 'N' records are returned back from the pricing engine.These are success records) UPDATED: Indicates Success status and that the record is Updated record and is not new. X: Indicates Success status and that the record is Unchanged BACK_CALCULATION_ERROR : Indicates that there were no qualified manual overrideable adjustments to adjust the price. D : Deleted. This indicates Failure. D_PBH : deleted in PBH processing
STATUS_TEXT	Varchar2	No	No	Out	Returned message.

Table 5–147 *LINE_DETAIL_REC_TYPE*

Parameter	Type	Req	Drv	Type	Usage
SUBSTITUTION_ ATTRIBUTE	Varchar2	No	No	Out	Modifier details. The attribute in the TERMS context that the pricing engine substituted, for example, Payment Terms. Used for Term Substitution-type modifiers.
ACCRUAL_FLAG	Varchar2	No	No	In/Out	Indicates that the modifier is an accrual
LIST_LINE_NO	Varchar2	No	No	In/Out	Modifier number. This field is applicable in case of Coupon Issue kind of modifier line.
ESTIM_GL_VALUE	Number	No	No	Out	The discount or surcharge value of the modifier. Used to estimate the discount cost for non-monetary modifiers.
ACCRUAL_ CONVERSION_ RATE	Number	No	No	Out	The rate to use when converting a non-monetary accrual to a monetary value.
OVERRIDE_FLAG	Varchar2	No	No	In/Out	Indicates if a user in the calling application can override the modifier value.
PRINT_ON_INVOICE_ FLAG	Varchar2	No	No	Out	Not used.
INVENTORY_ITEM_ID	Number	No	No	Out	Inventory item identifier in an item relationship. Used for list line type Item Upgrade.
ORGANIZATION_ID	Number	No	No	Out	Organization identifier in an item relationship. Used for list line type Item Upgrade.
RELATED_ITEM_ID	Number	No	No	Out	Related inventory item identifier in an item relationship. Used for list line type Item Upgrade.
RELATIONSHIP_TYPE_ID	Number	No	No	Out	Relationship type identifier in an item relationship. Used for list line type Item Upgrade.
ESTIM_ACCRUAL_RATE	Number	No	No	Out	Indicates the percentage at which to accrue or, for a coupon, the expected rate of redemption of the coupon. Liability is defined as: ACCRUAL OR COUPON VALUE * ESTIM_ ACCRUAL_RATE. Default Value: 100

Table 5–147 LINE_DETAIL_REC_TYPE

Parameter	Type	Req	Drv	Type	Usage
EXPIRATION_DATE	Date	No	No	Out	The expiration date of the accrual or coupon.
BENEFIT_PRICE_LIST_LINE_ID	Number	No	No	Out	The price list_line_id which has the list price before promotional discount. Used for Promotional Goods-type modifiers when the pricing engine creates a new transaction line.
RECURRING_FLAG	Varchar2	No	No		Not Used
BENEFIT_LIMIT	Number	No	No	Out	Not Used.
CHARGE_TYPE_CODE	Varchar2	No	No	In/Out	Indicates the type of charge based on lookup type FREIGHT_CHARGES_TYPE. Used for Freight/Special Charge-type modifiers.
CHARGE_SUBTYPE_CODE	Varchar2	No	No	In/Out	Indicates the type of charge based on lookup type CHARGE_TYPE_CODE.
INCLUDE_ON_RETURNS_FLAG	Varchar2	No	No	Out	Indicates whether the pricing engine should include the charge on a return transaction. Used for Freight/Special Charge-type modifiers.
BENEFIT_QTY	Number	No	No	Out	The accrual quantity for non-monetary accruals or, for promotional goods, item quantity.
BENEFIT_UOM_CODE	Varchar2	No	No	Out	The accrual unit of measure for non-monetary accruals, or, for promotional goods, item unit of measure.
PRORATION_TYPE_CODE	Varchar2	No	No	Out	Not Used.
SOURCE_SYSTEM_CODE	Varchar2	No	No	Out	Not Used.
REBATE_TRANSACTION_TYPE_CODE	Varchar2	No	No	Out	Not Used.
SECONDARY_PRICE_LIST_IND	Varchar2	No	No	Out	Indicates that the pricing used a secondary price list instead of the price list that the calling application requested.

Table 5–147 LINE_DETAIL_REC_TYPE

Parameter	Type	Req	Drv	Type	Usage
GROUP_VALUE	Varchar2	No	No	Out	This is populated for modifiers with modifier level code 'Group of Lines' and arithmetic operator as 'LUMPSUM'. It indicates the quantity/amount applied for that order line to which this belongs to. It is the quantity if the pricing attribute has 'Item Quantity' and amount if the pricing attribute is 'Item Amount'.
COMMENTS	Varchar2	No	No	Out	The comments on a modifier
UPDATED_FLAG	Varchar2	No	No	In/Out	Indicates that this modifier has been overridden.
PROCESS_CODE	Varchar2	No	No	Out	Indicates the pricing engine processing code for internal use.
LIMIT_CODE	Varchar2	No	No	Out	Indicates that the limit on this modifier has either exceeded or adjusted or consumed. Applicable values: <ul style="list-style-type: none"> ■ EXCEEDED ■ ADJUSTED ■ CONSUMED
LIMIT_TEXT	Varchar2	No	No	Out	The detailed message corresponding to the limit_code
FORMULA_ID	Varchar2	No	No	Out	The formula_id setup on the modifier.
CALCULATION_CODE	Varchar2	No	No	Out	Indicates the calculation code on the adjustment. BACK_CALCULATE : Indicates that this adjustment is a back calculated adjustment and it got applied to adjust the overridden selling price. This is related to the field updated_adjusted_unit_price on line_rec_type.
ROUNDING_FACTOR	Number	No	No	Out	Conversion rounding factor, populated when multi-currency set-up is being used

Table 5–147 *LINE_DETAIL_REC_TYPE*

Parameter	Type	Req	Drv	Type	Usage
CURRENCY_DETAIL_ID	Number	No	No	Out	Unique identifier for the multi-currency detail record, populated when multi-currency set-up is being used
CURRENCY_HEADER_ID	Number	No	No	Out	Unique identifier for the multi-currency header record, populated when multi-currency set-up is being used
SELLING_ROUNDING_FACTOR	Number	No	No	Out	Rounding Factor, populated when multi-currency set-up is being used
ORDER_CURRENCY	Varchar2	No	No	Out	Order Currency, populated when multi-currency set-up is being used
PRICING_EFFECTIVE_DATE	Date	No	No	Out	Pricing effective date, populated when multi-currency set-up is being used
BASE_CURRENCY_CODE	Varchar2	No	No	Out	Base currency code, populated when multi-currency set-up is being used

Inserting Lines into Temporary Tables The pricing engine takes in the input data, processes and inserts to the temporary table. By inserting the request lines, passed in attributes, qualifiers and modifiers directly to the temporary tables, the performance can be improved. The caller can indicate to the pricing engine that the data is inserted to the temporary tables by passing a value of 'N' to the TEMP_TABLE_INSERT_FLAG on the control record. There are procedures to do bulk inserts to the pricing temporary tables.

1. To insert the request lines to the temporary table, the procedure INSERT_LINES2 can be used. This procedure is in the pricing engine API QP_PREQ_GRP. This procedure does a bulk insert to the temporary tables. So the user needs to load the request line details PL/SQL table of records to individual PL/SQL tables one for each column. The temporary table for request lines QP_PREQ_LINES_TMP has columns that map to the columns of the P_LINE_TBL PL/SQL table.
2. To insert the passed in attributes, the procedure INSERT_LINE_ATTRS2 in the API QP_PREQ_GRP. This also does a bulk insert. Each column of the P_LINE_ATTR_TBL maps to a column in the temporary table for line attributes QP_PREQ_LINE_ATTRS_TMP .

3. The manual modifiers that need to be applied can be inserted into the temporary table for modifiers QP_PREQ_LDETS_TMP.
4. The build sourcing API has been modified to insert into the temporary tables directly. The procedure BUILD_CONTEXTS in the QP_ATTR_MAPPING_PUB API populates the temporary tables directly. This can be called to source the necessary attributes.
5. The procedure to make the pricing engine call, PRICE_REQUEST in the pricing engine API QP_PREQ_PUB has been overloaded to pass only the control record.
6. The output of the pricing engine exists in the temporary tables and the caller can process the data from the pricing temporary tables.

This is the process to insert records into the temporary tables and this will enhance the performance of the pricing engine. There is a example script to demonstrate the direct insertion. Please refer to the example scripts mentioned in this document.

Types of Request Line Details

1) Price List Line (PLL)

OPERAND_VALUE and OPERAND_CALCULATION_CODE columns store the price information and the type of price(UNIT_PRICE or PERCENT_PRICE).

2) Discount Line (DIS)

OPERAND_VALUE and OPERAND_CALCULATION_CODE columns store the price information and the type of price(AMT,%,or NEW_PRICE).

3) Surcharge Line (SUR)

OPERAND_VALUE and OPERAND_CALCULATION_CODE columns store the price information and the type of price(AMT,%,or NEW_PRICE).

4) Other Item Discount (OID)

OID processing: For OID processing, two request lines need to be passed to the pricing engine; for example, Buy A get 10% of B.

In the above example A and B need to be ordered on 2 order lines. So 2 request lines are passed to the pricing engine. So , when the engine does the OID processing , it creates a discount line for 10% on the 2nd request line . So, the 10% discount gets applied to the 2nd request line. Also a relationship record is created in the

RELATED_LINES_REC_TYPE record structure. Line Detail Index 8 is the actual Discount Line which is the OID line and Line Detail Index 9 is the actual benefit line which is 10% off line. So, the relationship is as follows:

Table 5–148 Relationship

Line Index	Line Detail Index	Related Line Index	Related Line Detail Index	Relation Ship Type
1	8(OID Line)	2	9(10% line)	GENERATED_LINE

5) Promotional Goods Discount (PRG)

PRG Processing: For example, Buy A get B for free, only Buy Item A request line needs to be passed to the pricing engine.

In the above example only request line/order line with Item A is sent to the pricing engine. Item B need not be ordered . This is the basic difference between OID and PRG. The pricing engine selects the PRG Modifier because of purchase of Item A and creates a record in the LINE_DETAIL_REC_TYPE record structure (Line Index 1-Line Detail Index 1). Then it tries to give the benefit which is a free B Item. In the process it does the following things:

- a. A new request line is created in the LINE_REC_TYPE record structure.(Line Index 2).
- b. A new relationship between the Item A line and Item B line is created ,in the RELATED_LINES_REC_TYPE record structure. It is a Line-Line relationship(Line Index 1 - Line Index 2).
- c. A new Price List Line is created for the new request line in the LINE_DETAIL_REC_TYPE record structure.(Line Index 2 - Line Detail Index 3).
- d. A new adjustment line for 100% discount is created for the new request line in the LINE_DETAIL_REC_TYPE record structure.(Line Index 2 - Line Detail Index 4).
- e. A new relationship line between the original PRG Line Detail Line and the new 100% off line detail is created in the RELATED_LINES_REC_TYPE record structure .(Line Detail Index 1 - Line Detail Index 4).
- f. A new record is created in the LINE_ATTR_REC_TYPE record structure for the new item B. (Line Index 2 - PRICING_CONTEXT = 'ITEM' , PRICING_ATTRIBUTE = 'PRICING_ATTRIBUTE1', PRICING_ATTR_VALUE = 'Item B').

6) Terms Substitution (TSN)

- SUBSTITUTION ATTRIBUTE
- SUBSTITUTION TO

7) Freight Charge

Freight Charges Processing: All the freight charge discounts do not affect the selling price. The pricing engine calculates the adjustment amount for the freight charge discounts but, this does not effect the selling price.

8) Manual Modifiers Processing

All the automatic modifiers of type Discounts and Surcharges that the user has qualified for , that are deleted as part of incompatibility resolution(due to incompatibility setup rules) , are returned as manual discounts to the caller. In addition to these discounts , all the qualified manual modifiers of type Discounts and Surcharge discounts are also returned to the caller , unapplied.

Manual adjustments can be applied in 2 ways:

- a. The caller can pass the manual adjustment to the pricing engine with Applied_Flag = 'Y' and Updated_Flag = 'Y'. The engine will apply this manual adjustment. The caller can override the manual adjustment by passing the new operand on the line_detail_rec_type.adjustment_amount.

Example: The caller makes a pricing engine call with 3 order lines. He wants to apply a manual adjustment of 10% to be applied to the second order line. The caller should pass the manual adjustment in the line_detail_tbl with columns updated_flag = 'Y' and applied_flag = 'Y' and with the line_index of the second order line. The pricing engine API, will calculate the adjustment amount and will apply this manual adjustment to the second order line. The applied_flag and updated_flag will be returned as 'Y' to indicate that it has been applied.

- b. The caller can override the selling price by passing the new selling price in the line_detail_rec_type.updated_adjusted_unit_price. The engine will then pick up a suitable manual overrideable modifier that is qualified and back calculate the adjustment amount and operand to match the new selling price. In this case the pricing engine will pass back this manual modifier with calculation_code as 'BACK_CALCULATE', updated_flag = 'Y' and applied_flag = 'Y'.

Example: If the caller passes 3 order lines to the pricing engine and the unit selling price on the second order line is \$80 and the unit list price is \$100

and he wants to override the selling price to \$90. In this case, the user has to pass 80 in the column UPDATE_ADJUSTED_UNIT_PRICE on the request line in the record corresponding to the second order line in the LINE_TBL. Then the pricing engine picks up all the qualifying manual overrideable adjustments, decides whether it needs to apply a discount or a surcharge. It prefers a manual overrideable adjustment that has been applied already. If none has been applied already, it randomly picks up a manual adjustment, back calculates the adjustment amount, \$10 surcharge in this case, and returns it with calculation_code as 'BACK_CALCULATE'. If there are no qualified manual overrideable adjustments, it returns an error status on the second order line indicating that there are no manual adjustments. In case, there is an error during the back calculation, the engine returns an error status on the second order line. The pricing_status_code on the second order line has the error code 'BACK_CALCULATION_ERROR' in case of an error.

9) Coupon Issue (CIE)

Coupon processing: For example, Buy Item A , get a Coupon for 10% of Item B.

The pricing engine gets one request line with Item A . Based on the search , the engine loads the Coupon Issue Modifier Line (CIE) in the LINE_DETAIL_REC_TYPE record structure . As part of the coupon issue processing , the pricing engine generates a coupon no dynamically which can be found from the LIST_LINE_NO column of the LINE_DETAIL_REC_TYPE record structure on the CIE modifier line . Also, the pricing engine creates a qualifier with this generated list_line_no for the 10% discount on Item B. That way , this discount will not be given , unless this coupon number is presented, next time as a qualifier to the pricing engine.

In the above example it say generated C10B as the coupon number. The next time , the customer gets this Coupon and this coupon number is punched in before making a pricing engine call and then the 10% discount on Item B is given , if the Customer Buys Item B.

10) Price Break Header (PBH)

Price Break Processing: The price break processing is same for both Price List Breaks and Discount/Surcharge Breaks. Price Breaks can be either POINT or RANGE breaks. This is found in the PRICE_BREAK_TYPE_CODE column in the record structure. Following is an example of how a Price Break is setup:

Price Break setup has a Price Break Parent Record which has a line type called PBH. This PBH record can have more than 1 child record, which actually define the

breaks. The Parent and the child records are all there in the LINE_DETAIL_REC_ TYPE record structure.

Example:

Table 5–149 Example

List Line Type Code	Parent/Child	Price Break From	Price Break To	Value	Break Type
PBH	Parent				POINT/RANGE
PLL	Child	1	100	\$100	POINT/RANGE
PLL	Child	101	200	\$90	POINT/RANGE
PLL	Child	201	300	\$80	POINT/RANGE
PLL	Child	301		\$70	POINT/RANGE

The Adjustment Amount and Line Quantity Fields exactly tell how and which child line has been used in deriving the break . Ex: If the ordered Quantity is 350. Then the record structure will have the information as follows:

Table 5–150 Record Structure Information

List Line Type Code	Adjustment Amount	Line Quantity	Break Type
PBH	\$90	350	POINT
PLL	\$100	0	POINT
PLL	\$90	0	POINT
PLL	\$80	0	POINT
PLL	\$70	350	POINT
PBH	\$87.1428571	350	RANGE
PLL	\$100	100	RANGE
PLL	\$90	100	RANGE
PLL	\$80	100	RANGE
PLL	\$70	50	RANGE

The relationship between the Parent and Child Records are stored in the RELATED_LINES_REC_TYPE record structure. The relationship type would be PBH_LINE

Table 5–151 RELATED_LINES_REC_TYPE

Line Index	Line Detail Index	Related Line Index	Related Line Detail Index	Relation Ship Type
1	8(PBH Line Detail)	1	9(PLL Line Detail 1)	PBH_LINE
1	8	1	10(PLL Line Detail 2)	PBH_LINE
1	8	1	11(PLL Line Detail 3)	PBH_LINE
1	8	1	12(PLL Line Detail 4)	PBH_LINE

11) Item Upgrade (IUE)

Important Columns:

INVENTORY_ITEM_ID

RELATED_ITEM_ID

QUAL_REC_TYPE

The Required and Derived values for the following parameters are Null.

Table 5–152 QUAL_REC_TYPE

Parameter	Data Type	Type	Usage
LINE_INDEX	Number	In/Out	Unique identifier for request line or request line detail.
QUALIFIER_CONTEXT	Varchar2	In/Out	Context for qualifier, for example, Customer Hierarchy.
QUALIFIER_ATTRIBUTE	Varchar2	In/Out	Qualifier attribute, for example, QUALIFIER_ATTRIBUTE1: Customer Class.
QUALIFIER_ATTR_VALUE_FROM	Varchar2	In/Out	Value for qualifier attribute
QUALIFIER_ATTR_VALUE_TO	Varchar2	In/Out	Return value for qualifier attribute. Populated when the pricing engine returns details of a volume break.

Table 5–152 QUAL_REC_TYPE

Parameter	DataType	Type	Usage
COMPARISION_OPERATOR_CODE	Varchar2	Out	The pricing engine creates qualifier attributes to indicate to the calling application which qualifier attribute caused it to give a benefit, for example, Order Amount > 1000 currency units (> is the operator code).
VALIDATED_FLAG	Varchar2	In/Out	Indicates that a price list or modifier list(asked for promotion) is valid for the pricing request. Applicable to price list and modifier list qualifiers; the pricing engine assumes that other qualifiers are valid.
STATUS_CODE	Varchar2	In/Out	Return status.
STATUS_TEXT	Varchar2	Out	Return message

LINE_ATTR_REC_TYPE

The Required and Derived values for the following parameters are Null.

Table 5–153 LINE_ATTR_REC_TYPE

Parameter	Datatype	Type	Usage
LINE_INDEX	Number	In/Out	Unique identifier for request line or request line detail.
PRICING_CONTEXT	Varchar2	In/Out	Context for a product or pricing attribute, for example, Product Hierarchy.
PRICING_ATTRIBUTE	Varchar2	In/Out	Product or pricing attribute, for example, PRICING_ATTRIBUTE11: Customer Item ID.
PRICING_ATTR_VALUE_FROM	Varchar2	In/Out	Value for product or pricing attribute.
PRICING_ATTR_VALUE_TO	Varchar2	In/Out	Return value for Pricing attribute. Populated when the pricing engine returns details of a volume break
VALIDATED_FLAG	Varchar2	In/Out	Not used.
STATUS_CODE	Varchar2	In/Out	Return status
STATUS_TEXT	Varchar2	In/Out	Return Message

RELATED_LINES_REC_TYPE

The Required and Derived values for the following parameters are Null.

Table 5–154 RELATED_LINES_REC_TYPE

Parameter	Datatype	Type	Usage
LINE_INDEX	Number	In/Out	PL/SQL unique identifier for request line.
LINE_DETAIL_INDEX	Number	In/Out	PL/SQL unique identifier for request detail line
RELATIONSHIP_TYPE_CODE	Varchar2	In/Out	Type of relationship between pricing lines.
RLTD_LINE_INDEX	Number	In/Out	PL/SQL identifier for related request line.
RLTD_LINE_DETAIL_INDEX	Number	In/Out	PL/SQL unique identifier for related request detail line.

Validation of Price Request API

Standard Validation

Oracle Pricing validates all required columns in the Price Request API. For specific information on the data implied by these columns, see: *Oracle Pricing Technical Reference Manual* for details.

Other Validation

None

Error Handling

If any validation fails, the API will return error status to the calling module. The Price Request API processes the rows and reports the following values for every record.

Table 5–155 Error Handling

Condition	PROCESS_STATUS	ERROR_MESSAGE
Success	5	Null
Failure	4	actual error message

See

Oracle Pricing Technical Reference Manual

Example of Price Request Application Program Interface

The follow are script examples for the Price Request Application Program Interface:

Script Names

- qp_engine_testing.sql(All regular Price Lists, Discounts , Surcharges etc.)
- qp_test_service.sql(Service Item Pricing Setup)
- qp_test_oid.sql(Other Item Discount Setup)
- qp_direct_insert.sql(Direct insertion into temporary tables to improve performance)
- qp_manual_adjustments(Apply manual adjustments)
- qp_override_selling_price.sql(Override the unit selling price)

Location

\$QP_TOP/patch/115/sql

Purpose

This script helps the user in setting up data and making a call to the pricing engine which would return the price and discounts(if any).

Setup

Involves passing information to the pricing engine

1. Control Record Information

Control Record has parameters which control the behaviour of the pricing engine. Please refer to the example script for more details

2. Request/Order Line Information

Request Line has information which contains the elements in the calling application that requires a base and adjusted price . It may equate to a transaction line or transaction header record in the calling application. Refer to qp_engine_testing.sql for information on setting up the request line data.

LINE_REC_TYPE is the record structure that needs to be filled in when passing information related to a request line.

3. Qualifiers and Pricing Attributes Information

Qualifiers and Pricing Attributes information that helps the pricing engine to determine the price list lines and modifier list lines for which a price request is eligible.

Pricing Attributes and Qualifier Attributes Setup

LINE_ATTR_REC_TYPE is the record structure that needs to be filled in when passing information related to a pricing attribute. The following are the important columns:

- PRICING_CONTEXT
- PRICING_ATTRIBUTE
- PRICING_ATTR_VALUE_FROM

QUAL_REC_TYPE is the record structure that needs to be filled in when passing information related to a qualifiers. The following are the important columns:

- QUALIFIER_CONTEXT

- QUALIFIER_ATTRIBUTE
- QUALIFIER_ATTR_VALUE_FROM

Any attribute for example Item, Price List Id, Customer Name, Order Type etc. will have a mapping to give the Context, Attribute, Value_Id (will be the actual value). For more information on seeded attributes, see: *Oracle Advanced Pricing User's Guide*. Please refer to the following examples:

Table 5–156 Pricing Attributes and Qualifier Attributes

Attribute	Attribute Name	Context	Attribute	Value_Id (id)	Value (Name)
Pricing Attribute	Inventory Item Id	ITEM	PRICING_ATTRIBUTE1	149	AS54888
Pricing Attribute	Plastics	ITEM	PRICING_ATTRIBUTE2	29	PLASTIC
Qualifier	Order Type	ORDER	QUALIFIER_ATTRIBUTE1	1000	Standard
Qualifier	Customer	CUSTOMER	QUALIFIER_ATTRIBUTE1	100	Business World
Qualifier	Price List	MODLIST	QUALIFIER_ATTRIBUTE4	1000	Corporate
Qualifier	Agreement Type	ORDER	QUALIFIER_ATTRIBUTE9	RESELLER	RESELLER
Qualifier	Order Date	ORDER	QUALIFIER_ATTRIBUTE5	2000/12/10 00:00:00	

Note: Dates and Numbers are to be passed in canonical format
Canonical Date Format: YYYY/MM/DD HH24:MM:SS Ex: select
fnd_date.date_to_canonical ('01-OCT-2000') from dual; Canonical
Number Format, '.' is the decimal separator. For example, select
fnd_number.number_to_canonical(2000.45) from dual.

See Order Date in the preceding matrix.

Interpreting the results from the pricing engine

1. Request Line Information

The output request line record would have the unit_price and adjusted unit price information. Please refer to the example to see more details.

2. Price List/Discount Related Information

The output request line detail record would have the price list line and modifier list line detail information like the unit_price value, discount percentage etc. Please refer to API Doc for more explanation on each of the columns of the output record structure.

3. Qualifiers and Pricing Attributes

The output qualifier and pricing attribute records would have the qualifiers and pricing attributes which qualified the price list line or modifier list line . Please refer to API Doc for more explanation on each of the columns of the output record structure.

4. Related Lines Information

This record structure has information related to Service Lines Relationship , Price Break Lines Relationship, OID Modifier Line relationship , PRG Modifier Line Relationship and Order To Line Relationship (Type - ORDER_TO_LINE) which tells an Order Level Discount was applied to which Lines.

Pricing Object Security - Check Function API

This section explains how to use the Pricing Object Security CHECK_FUNCTION API and how it functions in Oracle Advanced Pricing.

The CHECK_FUNCTION API is used to check if a specific user, logging in with a specific responsibility and within a specific operating unit, can have the functional access to a specific pricing object or not. The package QP_SECURITY contains the function CHECK_FUNCTION. In our pricing data security system, we define two levels of functional access: QP_SECU_VIEW access and QP_SECU_UPDATE access. The data model is additive, so, this API will check all existing security rules and decide to authorize the user with the functional access or deny him.

Functional Overview

This API will take the parameters and check the pricing object security rules which have been setup in the pricing data security system.

The parameter p_function_name is required and can take 2 different values – 'QP_SECU_VIEW' or 'QP_SECU_UPDATE'.

The parameter p_instance_type is required and can take 3 different values – 'PRL' for type of standard pricelist, 'MOD' for type of modifier list, and 'AGR' for type of agreement pricelist.

The parameter p_instance_pk1_value is required and is the list_header_id from qp_list_headers_b for a specific pricing object.

The parameter p_instance_pk2_value and p_instance_pk3_value are not in use now. This is for future consideration, in case that the pricing objects to be secured have composite keys.

The parameter p_user_name, p_resp_id and p_org_id are optional. If they are not passed in, CHECK_FUNCTION will take the values from the current user and the corresponding settings for responsibility and operating unit.

This API will return one of the three values as follows:

'T', means the user has the asking-for functional access to the pricing object.

'F', means the user has been denied with the access.

'E-<error message>', means error happened within the API.

Setting Up and Parameter Descriptions

The following chart describes all parameters used by the public Pricing Object Security Check API. All of the inbound and outbound parameters are listed. Additional information on these parameters may follow.

CHECK_FUNCTION

The following table shows the parameters for this structure.

Table 5–157 CHECK_FUNCTION

Parameter	Usage	Type	Req	Drv
p_function_name	In	Varchar2	Yes	No
p_instance_type	In	Varchar2	Yes	No
p_instance_pk1	In	Number	Yes	No
p_instance_pk2	In	Number	No	No
p_instance_pk3	In	Number	No	No
p_user_name	In	Varchar2	No	Yes
p_resp_id	In	Number	No	Yes
p_org_id	In	Number	No	Yes
Function return value	Out	Varchar2	Yes	No

A key of the short names and definitions used in the API tables are provided in the following table:

Table 5–158 Short Names Key

Short name	Definition
Drv	Derived
Req	Required Yes : This is a required parameter. No : This is an optional parameter.
N/A (no entry)	No value/not applicable

Validation of Pricing Object Security API

Standard Validation

Oracle Advanced Pricing validates all required columns in the CHECK_FUNCTION API. For more information, see: *Oracle Pricing Technical Reference Manual*.

Other Validation

None

Error Handling

If any validation fails, the API will return error to the calling module.

Table 5–159 Error Handling

Condition	Function Return Value
Success	T or F
Failure	E-error message

QP_ATTRIBUTES_PUB Application Program Interface

Functional Overview

The Public package QP_ATTRIBUTES_PUB is a Business Object API, based on the following tables. QP_PRC_CONTEXTS_B/TL, QP_SEGMENTS_B/TL tables. This API does all the transaction processing for all the data that is inserted, updated or deleted from Contexts and Attributes. The Business API Model is as shown below :

The Object Name is Attributes and the object code is ATR.

Procedure PROCESS_ATTRIBUTES

The following table shows the parameters for this structure. This procedure is used to add, update or delete contexts and attributes.

Table 5–160 PROCESS_ATTRIBUTES

Parameter	Usage	Type	Req	Drv
p_api_version_number	In	Number	No	No
p_init_msg_list	In	Varchar2	No	No
p_return_values	In	Varchar2	No	No
p_commit	In	Varchar2	No	No
x_return_status	Out	Number	No	No
x_msg_count	Out	Varchar2	No	No
x_msg_data	Out	Varchar2	No	No
p_CON_rec	In	Con_Rec_Type	No	No
p_CON_val_rec	In	Con_Val_Rec_Type	No	No
p_SEG_tbl	In	Con_Tbl_Type	No	No
p_SEG_val_tbl	In	Con_Val_Tbl_Type	No	No
x_CON_rec	Out	Con_Rec_Type	No	No
x_CON_val_rec	Out	Con_Val_Rec_Type	No	No
x_SEG_tbl	Out	Con_Tbl_Type	No	No
x_SEG_val_tbl	Out	Con_Val_Tbl_Type	No	No

p_api_version_number

This the version number of the API.

P_CON_REC

The following table shows the parameters for this structure.

Table 5–161 P_CON_REC

Parameter	Usage	Type	Req	Drv
Prc_context_code	Null	Varchar2	Yes	No
Prc_context_id	Null	Number	Yes	No
Prc_context_type	Null	Varchar2	Yes	No
Program_application_id	Null	Number	No	No
Program_id	Null	Number	No	No
Program_update_date	Null	Date	No	No
Seeded_description	Null	Varchar2	No	No
Seeded_flag	Null	Varchar2	Yes	No
Seeded_prc_context_name	Null	Varchar2	No	No
User_description	Null	Varchar2	No	No
User_prc_context_name	Null	Varchar2	No	No
Return_status	Null	Varchar2	No	No
Db_flag	Null	Varchar2	No	No
operation	Null	Varchar2	No	No

P_CON_VAL_REC

The following table shows the parameters for this structure.

Table 5–162 P_CON_VAL_REC

Parameter	Usage	Type	Req	Drv
Enabled	Null	Varchar2	Yes	No
Prc_context	Null	Varchar2	Yes	No
Seeded	Null	Varchar2	Yes	No

P_SEG_TBL

The following table shows the parameters for this structure.

Table 5–163 P_SEG_TBL

Parameter	Usage	Type	Req	Drv
P_Seg_Tbl	Null	TABLE OF Seg_Rec_Type		

P_SEG_VAL_TBL

The following table shows the parameters for this structure.

Table 5–164 P_SEG_VAL_TBL

Parameter	Usage	Type	Req	Drv
P_Seg_Val_Tbl	Null	TABLE OF Seg_Val_Rec_Type		

X_CON_REC

The following table shows the parameters for this structure.

Table 5–165 X_CON_REC

Parameter	Usage	Type	Req	Drv
Prc_context_code	Null	Varchar2	No	No
Prc_context_id	Null	Number	No	No
Prc_context_type	Null	Varchar2	No	No
Program_application_id	Null	Number	No	No
Program_id	Null	Number	No	No
Program_update_date	Null	Date	No	No
Seeded_description	Null	Varchar2	No	No
Seeded_flag	Null	Varchar2	No	No
Seeded_prc_context_name	Null	Varchar2	No	No
User_description	Null	Varchar2	No	No
User_prc_context_name	Null	Varchar2	No	No
Return_status	Null	Varchar2	No	No

Table 5–165 X_CON_REC

Parameter	Usage	Type	Req	Drv
Db_flag	Null	Varchar2	No	No
operation	Null	Varchar2	No	No

X_CON_VAL_REC

The following table shows the parameters for this structure.

Table 5–166 X_CON_VAL_REC

Parameter	Usage	Type	Req	Drv
Enabled	Null	Varchar2	No	No
Prc_context	Null	Varchar2	No	No
Seeded	Null	Varchar2	No	No

X_SEG_TBL

The following table shows the parameters for this structure.

Table 5–167 X_SEG_TBL

Parameter	Usage	Type	Req	Drv
-	Null	TABLE OF Seg_Rec_Type	No	No

X_SEG_VAL_TBL

The following table shows the parameters for this structure.

Table 5–168 X_SEG_VAL_TBL

Parameter	Usage	Type	Req	Drv
-	Null	TABLE OF Seg_Val_Rec_Type	No	No

Procedure LOCK_ATTRIBUTES

The following table shows the parameters for this structure. Users can use this procedure to lock a context row and its attributes from getting updated by another user concurrently.

Table 5–169 LOCK_ATTRIBUTES

Parameter	Usage	Type	Req	Drv
p_api_version_number	In	Number	Yes	No
p_init_msg_list	In	Varchar2	No	No
p_return_values	In	Varchar2	No	No
x_return_status	Out	Number	No	No
x_msg_count	Out	Varchar2	No	No
x_msg_data	Out	Varchar2	No	No
p_CON_rec	In	Con_Rec_Type	No	No
p_CON_val_rec	In	Con_Val_Rec_Type	No	No
p_SEG_tbl	In	Con_Tbl_Type	No	No
p_SEG_val_tbl	In	Con_Val_Tbl_Type	No	No
x_CON_rec	Out	Con_Rec_Type	No	No
x_CON_val_rec	Out	Con_Val_Rec_Type	No	No
x_SEG_tbl	Out	Con_Tbl_Type	No	No
x_SEG_val_tbl	Out	Con_Val_Tbl_Type	No	No

p_api_version_number

This is the version number of the API.

P_CON_REC

The following table shows the parameters for this structure.

Table 5–170 P_CON_REC

Parameter	Usage	Type	Req	Drv
Prc_context_code	Null	Varchar2	Yes	No
Prc_context_id	Null	Number	Yes	No
Prc_context_type	Null	Varchar2	Yes	No
Program_application_id	Null	Number	No	No
Program_id	Null	Number	No	No

Table 5–170 P_CON_REC

Parameter	Usage	Type	Req	Drv
Program_update_date	Null	Date	No	No
Seeded_description	Null	Varchar2	No	No
Seeded_flag	Null	Varchar2	Yes	No
Seeded_prc_context_name	Null	Varchar2	No	No
User_description	Null	Varchar2	No	No
User_prc_context_name	Null	Varchar2	No	No
Return_status	Null	Varchar2	No	No
Db_flag	Null	Varchar2	No	No
operation	Null	Varchar2	No	No

P_CON_VAL_REC

The following table shows the parameters for this structure.

Table 5–171 P_CON_VAL_REC

Parameter	Usage	Type	Req	Drv
Enabled	Null	Varchar2	Yes	No
Prc_context	Null	Varchar2	Yes	No
Seeded	Null	Varchar2	Yes	No

The following table shows the parameters for these table structures:

Table 5–172 Parameters

Table name	Usage	Type
P_SEG_TBL	Null	TABLE OF Seg_Rec_Type
P_SEG_VAL_TBL	Null	TABLE OF Seg_Val_Rec_Type

X_CON_REC

The following table shows the parameters for this structure.

Table 5–173 X_CON_REC

Parameter	Usage	Type	Req	Drv
Prc_context_code	Null	Varchar2	No	No
Prc_context_id	Null	Number	No	No
Prc_context_type	Null	Varchar2	No	No
Program_application_id	Null	Number	No	No
Program_id	Null	Number	No	No
Program_update_date	Null	Date	No	No
Seeded_description	Null	Varchar2	No	No
Seeded_flag	Null	Varchar2	No	No
Seeded_prc_context_name	Null	Varchar2	No	No
User_description	Null	Varchar2	No	No
User_prc_context_name	Null	Varchar2	No	No
Return_status	Null	Varchar2	No	No
Db_flag	Null	Varchar2	No	No
operation	Null	Varchar2	No	No

X_CON_VAL_REC

The following table shows the parameters for this structure.

Table 5–174 X_CON_VAL_REC

Parameter	Usage	Type	Req	Drv
Enabled	Null	Varchar2	No	No
Prc_context	Null	Varchar2	No	No
Seeded	Null	Varchar2	No	No

X_SEG_TBL

The following table shows the parameters for this structure.

Table 5–175 X_SEG_TBL

Parameter	Usage	Type	Req	Drv
X_Seg_Tbl	Null	TABLE OF Seg_Rec_Type		

X_SEG_VAL_TBL

The following table shows the parameters for this structure.

Table 5–176 X_SEG_VAL_TBL

Parameter	Usage	Type	Req	Drv
X_Seg_Val_Tbl	Null	TABLE OF Seg_Val_Rec_Type		

Procedure GET_ATTRIBUTES

The following table shows the parameters for this structure. This procedure fetches the context and all its attributes, when prc_context_id is provided.

Table 5–177 GET_ATTRIBUTES

Parameter	Usage	Type	Req	Drv
p_api_version_number	In	Number	Yes	No
p_init_msg_list	In	Varchar2	No	No
p_return_values	In	Varchar2	No	No
x_return_status	Out	Number	No	No
x_msg_count	Out	Varchar2	No	No
x_msg_data	Out	Varchar2	No	No
p_prc_context_id	In	Number	No	No
p_prc_context	In	Varchar2	No	No
x_CON_rec	Out	Con_Rec_Type	No	No
x_CON_val_rec	Out	Con_Val_Rec_Type	No	No
x_SEG_tbl	Out	Con_Tbl_Type	No	No
x_SEG_val_tbl	Out	Con_Val_Tbl_Type	No	No

p_api_version_number

This the version number of the API.

p_prc_context_id

This the primary key for a given context. The API will fetch all the context and all the attributes for this value.

p_prc_context

This the unique code for a given context. The API will fetch all the context and all the attributes for this value.

X_CON_REC

The following table shows the parameters for this structure.

Table 5–178 X_CON_REC

Parameter	Usage	Type	Req	Drv
Prc_context_code	Null	Varchar2	No	No
Prc_context_id	Null	Number	No	No
Prc_context_type	Null	Varchar2	No	No
Program_application_id	Null	Number	No	No
Program_id	Null	Number	No	No
Program_update_date	Null	Date	No	No
Seeded_description	Null	Varchar2	No	No
Seeded_flag	Null	Varchar2	No	No
Seeded_prc_context_name	Null	Varchar2	No	No
User_description	Null	Varchar2	No	No
User_prc_context_name	Null	Varchar2	No	No
Return_status	Null	Varchar2	No	No
Db_flag	Null	Varchar2	No	No
operation	Null	Varchar2	No	No

X_CON_VAL_REC The following table shows the parameters for this structure.

Table 5–179 X_CON_VAL_REC

Parameter	Usage	Type	Req	Drv
Enabled	Null	Varchar2	No	No
Prc_context	Null	Varchar2	No	No
Seeded	Null	Varchar2	No	No

X_SEG_TBL The following table shows the parameters for this structure.

Table 5–180 X_SEG_TBL

Parameter	Usage	Type	Req	Drv
X_Seg_Tbl	Null	TABLE OF Seg_Rec_Type		

X_SEG_VAL_TBL The following table shows the parameters for this structure.

Table 5–181 X_SEG_VAL_TBL

Parameter	Usage	Type	Req	Drv
X_Seg_Val_Tbl	Null	TABLE OF Seg_Val_Rec_Type		

QP_ATTR_MAPPING_PUB Application Program Interface

Functional Overview

The build_contexts API calls QP_BUILD_SOURCING_PVT, which contains sourcing calls to build attribute mapping rules for Attributes. This API is called from the OM pricing integration. OM passes request_type_code and pricing_type and this API builds the attribute mapping rules of the attributes used in the setup that can be sourced at run time. Attribute mapping rules thus built are returned in the PLSQL table structures x_price_contexts_result_tbl & x_qual_contexts_result_tbl.

The following table shows the parameters for this structure.

Setting Up and Parameter Descriptions

The following table describes all parameters used by the public QP_ATTR_MAPPING_PUB. All of the inbound and outbound parameters are listed. Additional information on these parameters follows.

Table 5–182 QP_ATTR_MAPPING_PUB

Parameter	Usage	Type	Req	Drv
p_request_type_code	In	Varchar2	Yes	No
p_pricing_type	In	Varchar2	Yes	No
x_price_contexts_result_tbl	Out	CONTEXTS_RESULT_TBL_TYPE	No	No
x_qual_contexts_result_tbl	Out	CONTEXTS_RESULT_TBL_TYPE	No	No

p_request_type_code

An allowable value from the QP_PRICING_REQ_SOURCES table. This is different from source system code; groups of source system codes form a request type.

For example, CRM request type code can have the I-marketing and I-store source system codes associated with it (using QP_PRICING_REQ_SOURCES). When you call Build_Contexts with the CRM request type, the process sources all contexts for both I-marketing and I-store source systems.

p_pricing_type

Indicates the sourcing rules to use when building the attributes. Allowable values are:

- L (Line): The process builds the pricing contexts based on the rules associated with the line pricing attributes entity and builds the qualifier contexts based on the rules associated with the line qualifier attribute
- H (Header): The process builds the pricing contexts based on the rules associated with the header pricing attributes entity and builds the qualifier contexts based on the rules associated with the header qualifier attributes.

CONTEXTS_RESULT_TBL_TYPE

The following table shows the parameters for this structure.

Table 5–183 CONTEXTS_RESULT_TBL_TYPE

Parameter	Usage	Type	Req	Drv
Contexts_Result_Tbl_Type	Out	Contexts_Result_Rec_Type	No	NO

CONTEXTS_RESULT_REC_TYPE

The following table shows the parameters for this structure.

Table 5–184 CONTEXTS_RESULT_REC_TYPE

Parameter	Usage	Type	Req	Drv
context_name	Null	Varchar2	No	No
attribute_name	Null	Varchar2	No	No
Attribute_value	Null	Varchar2	No	No

Procedure BUILD_CONTEXTS (overloaded)

This API is called from the Order Management (OM) pricing integration in the performant code path. OM passes request_type_code and pricing_type and this API builds the attribute mapping rules of the attributes used in the setup that can be sourced at run time. Attribute mapping rules thus built are inserted into the pricing temporary table QP_PREQ_LINE_ATTRS_TMP for the passed in line_index.

The following table shows the parameters for this structure.

Table 5–185 BUILD_CONTEXTS (overloaded)

Parameter	Usage	Type	Req	Drv
p_request_type_code	In	Varchar2	Yes	No

Table 5–185 BUILD_CONTEXTS (overloaded)

Parameter	Usage	Type	Req	Drv
p_line_index	In	Number	Yes	No
p_pricing_type_code	In	Varchar2	Yes	No

p_request_type_code

An allowable value from the QP_PRICING_REQ_SOURCES table. This is different from source system code; groups of source system codes form a request type.

For example, CRM request type code can have the I-marketing and I-store source system codes associated with it (using QP_PRICING_REQ_SOURCES). When you call Build_Contexts with the CRM request type, the process sources all contexts for both I-marketing and I-store source systems.

P_line_index

Unique identifier for line in the table qp_preq_lines_tmp.

p_pricing_type

Indicates the sourcing rules to use when building the attributes. Allowable values are:

- L (Line): The process builds the pricing contexts based on the rules associated with the line pricing attributes entity and builds the qualifier contexts based on the rules associated with the line qualifier attribute
- H (Header): The process builds the pricing contexts based on the rules associated with the header pricing attributes entity and builds the qualifier contexts based on the rules associated with the header qualifier attributes.

Procedure Get_User_Item_Pricing_Contexts

This API returns the pricing contexts whose sourcing method is USER ENTERED. This API is called from the OM pricing integration. OM passes request_type_code and this API returns the pricing contexts whose sourcing method is USER ENTERED in the PLSQL table structure p_user_attrbs_tbl.

The following table shows the parameters for this structure.

Table 5–186 Get_User_Item_Pricing_Contexts

Parameter	Usage	Type	Req	Drv
p_request_type_code	In	Varchar2	Yes	No
p_user_attrbs_tbl	Out	USER_ATTRIBUTE_TBL_TYPE	No	No

p_request_type_code

An allowable value from the QP_PRICING_REQ_SOURCES table. This is different from source system code; groups of source system codes form a request type.

For example, CRM request type code can have the I-marketing and I-store source system codes associated with it (using QP_PRICING_REQ_SOURCES). When you call Build_Contexts with the CRM request type, the process sources all contexts for both I-marketing and I-store source systems.

USER_ATTRIBUTE_TBL_TYPE

The following table shows the parameters for this structure.

Table 5–187 USER_ATTRIBUTE_TBL_TYPE

Parameter	Usage	Type	Req	Drv
User_Attribute_Tbl_Type	Out	User_Attribute_Rec_Type	No	No

USER_ATTRIBUTE_REC_TYPE

The following table shows the parameters for this structure.

Table 5–188 USER_ATTRIBUTE_REC_TYPE

Parameter	Usage	Type	Req	Drv
context_name	Null	Varchar2	No	No
attribute_name	Null	Varchar2	No	No

Function Is_attribute_used

This API is used to find out whether pricing attribute is used in the pricing setup. This API is called by the pricing engine. The pricing engine passes context and pricing attribute to this function and this API returns the attribute used flag (Y/N).

The following table shows the parameters for this structure.

Table 5–189 *Is_attribute_used*

Parameter	Usage	Type	Req	Drv
p_attribute_context	In	Varchar2	No	No
p_attribute_code	In	Varchar2	No	No

p_attribute_context

A context name set up in the context definition table, for example, item, customer, or volume.

p_attribute_code

Name of the attribute for the sourcing rule, for example, PRICING_ATTRIBUTE1, PRICING_ATTRIBUTE2, or QUALIFIER_ATTRIBUTE1.

Validation of Attribute Mapping API

Standard Validation

Oracle Advanced Pricing validates all required columns in the Attribute Mapping API. For more information, see: *Oracle Pricing Technical Reference Manual*.

Other Validation

None

Error Handling

If any validation fails, the API will return error status to the calling module. The Attribute Mapping API processes the rows and reports the values in the following table for every record.

Table 5–190 *Error Handling*

Condition	PROCESS_STATUS	ERROR_MESSAGE
Success	5	Null
Failure	4	actual error message

See:

Oracle Pricing Technical Reference Manual

Qualifiers Application Program Interface

This section explains how to use the Qualifiers API and how it functions in Oracle Advanced Pricing. The Qualifiers package consists of entities to set up qualifiers.

Functional Overview

The Qualifiers package QP_QUALIFIERS_RULES_PUB contains the following APIs and record type definitions:

- Qualifier_Rules_Rec_Type
- Qualifier_Rules_Tbl_Type
- Qualifier_Rules_Val_Rec_Type
- Qualifier_Rules_Val_Tbl_Type
- Qualifiers_Rec_Type
- Qualifiers_Tbl_Type
- Qualifiers_Val_Rec_Type
- Qualifiers_Val_Tbl_Type
- Process_Qualifier_Rules: Creates, updates, and deletes pricing qualifier rules and pricing qualifiers belonging to those rules.
- Lock_Qualifier_Rules: Locks table rows.
- Get_Qualifier_Rules: Retrieves the qualifier rule and qualifiers for a given qualifier rule ID or qualifier rule name.
- Copy_Qualifier_Rules: Creates a new qualifier rule with the name in p_to_qualifier_rule. Also, copies the qualifiers belonging to the qualifier rule specified by the qualifier rule in p_qualifier_rule or the qualifier rule ID in p_qualifier_rule_id into a new qualifier rule. Returns qualifier rule ID for the new qualifier rule.

Setting Up and Parameter Descriptions

The following chart describes all parameters used by the public Qualifiers. All of the inbound and outbound parameters are listed. Additional information on these parameters may follow.

PROCESS_QUALIFIER_RULES

The following table shows the parameters for this structure.

Table 5–191 PROCESS_QUALIFIER_RULES

Parameter	Usage	Type	Req	Drv
p_api_version_number	In	Number	No	No
p_init_msg_list	In	Varchar2	No	No
p_return_values	In	Varchar2	No	No
p_commit	In	Varchar2	No	No
x_return_status	Out	Varchar2	No	No
x_msg_count	Out	Number	No	No
x_msg_data	Out	Varchar2	No	No
p_QUALIFIER_RULES_rec	In	Qualifier_Rules_Rec_Type	No	No
p_QUALIFIER_RULES_val_rec	In	Qualifier_Rules_Val_Rec_Type	No	No
p_QUALIFIERS_tbl	In	Qualifiers_Tbl_Type	No	No
p_QUALIFIERS_val_tbl	In	Qualifiers_Val_Tbl_Type	No	No
x_QUALIFIER_RULES_rec	Out	Qualifier_Rules_Rec_Type	No	No
x_QUALIFIER_RULES_val_rec	Out	Qualifier_Rules_Val_Rec_Type	No	No
x_QUALIFIERS_tbl	Out	Qualifiers_Tbl_Type	No	No
x_QUALIFIERS_val_tbl	Out	Qualifiers_Val_Tbl_Type	No	No

p_init_msg_list

Default Value: FND_API.G_FALSE

p_return_values

Default Value: FND_API.G_FALSE

p_commit

Default Value: FND_API.G_FALSE

p_QUALIFIER_RULES_rec

In this procedure, P_QUALIFIER_RULES_rec.operation states the operation that the process should perform. Allowable values of P_QUALIFIER_RULES_rec.operation are:

- QP_GLOBALS.G_OPR_CREATE
- QP_GLOBALS.G_OPR_DELETE
- QP_GLOBALS.G_OPR_UPDATE
- QP_GLOBALS.G_OPR_LOCK
- QP_GLOBALS.G_OPR_NONE

Default Value: G_MISS_QUALIFIER_RULES_REC

p_QUALIFIER_RULES_val_rec

Default Value: G_MISS_QUALIFIER_RULES_VAL_REC

p_QUALIFIERS_tbl

Default Value: G_MISS_QUALIFIERS_TBL

p_QUALIFIERS_val_tbl

Default Value: G_MISS_QUALIFIERS_VAL_TBL

QUALIFIER_RULES_REC_TYPE

The following table shows the parameters for this structure.

Table 5–192 QUALIFIER_RULES_REC_TYPE

Parameter	Usage	Type	Req	Drv
attribute1	Null	Varchar2	No	No
attribute2	Null	Varchar2	No	No
attribute3	Null	Varchar2	No	No
attribute4	Null	Varchar2	No	No
attribute5	Null	Varchar2	No	No
attribute6	Null	Varchar2	No	No
attribute7	Null	Varchar2	No	No
attribute8	Null	Varchar2	No	No

Table 5–192 QUALIFIER_RULES_REC_TYPE

Parameter	Usage	Type	Req	Drv
attribute9	Null	Varchar2	No	No
attribute10	Null	Varchar2	No	No
attribute11	Null	Varchar2	No	No
attribute12	Null	Varchar2	No	No
attribute13	Null	Varchar2	No	No
attribute14	Null	Varchar2	No	No
attribute15	Null	Varchar2	No	No
context	Null	Varchar2	No	No
created_by	Null	Number	No	No
creation_date	Null	Date	No	No
description	Null	Varchar2	No	No
last_updated_by	Null	Number	No	No
last_update_date	Null	Date	No	No
last_update_login	Null	Number	No	No
name	Null	Varchar2	Yes	No
program_application_id	Null	Number	No	No
program_id	Null	Number	No	No
program_update_date	Null	Date	No	No
qualifier_rule_id	Null	Number	No	No
request_id	Null	Number	No	No
return_status	Null	Varchar2	No	No
db_flag	Null	Varchar2	No	No
operation	Null	Varchar2	Yes	No

attribute1-15

Default Value: FND_API.G_MISS_CHAR

context

Default Value: FND_API.G_MISS_CHAR

created_by

Default Value: FND_API.G_MISS_NUM

creation_date

Default Value: FND_API.G_MISS_DATE

description

Default Value: FND_API.G_MISS_CHAR

last_updated_by

Default Value: FND_API.G_MISS_NUM

last_update_date

Default Value: FND_API.G_MISS_DATE

last_update_login

Default Value: FND_API.G_MISS_NUM

name

Default Value: FND_API.G_MISS_CHAR

program_application_id

Default Value: FND_API.G_MISS_NUM

program_id

Default Value: FND_API.G_MISS_NUM

program_update_date

Default Value: FND_API.G_MISS_DATE

qualifier_rule_id

Default Value: Comes from the sequence QP_QUALIFIER_RULES_S

request_id

Default Value: FND_API.G_MISS_NUM

return_status

Default Value: FND_API.G_MISS_CHAR

db_flag

Default Value: FND_API.G_MISS_CHAR

operation

Default Value: FND_API.G_MISS_CHAR

QUALIFIER_RULES_TBL_TYPE

The following table shows the parameters for this structure.

Table 5–193 QUALIFIER_RULES_TBL_TYPE

Parameter	Usage	Type	Req	Drv
Qualifier_Rules_Rec_Type	Null	Record	No	No

QUALIFIER_RULES_VAL_REC_TYPE

The following table shows the parameters for this structure.

Table 5–194 QUALIFIER_RULES_VAL_REC_TYPE

Parameter	Usage	Type	Req	Drv
qualifier_rule	Null	Varchar2	No	No

qualifier_rule

Default Value: FND_API.G_MISS_CHAR

QUALIFIER_RULES_VAL_TBL_TYPE

The following table shows the parameters for this structure.

Table 5–195 QUALIFIER_RULES_VAL_TBL_TYPE Parameters

Parameter	Usage	Type	Req	Drv
Optional	Qualifier_Rules_Val_Rec_Type	Null	Record	No
No	No			

QUALIFIERS_REC_TYPE

The following table shows the parameters for this structure.

Table 5–196 QUALIFIERS_REC_TYPE Parameters

Parameter	Usage	Type	Req	Drv
attribute1	Null	Varchar2	No	No
attribute2	Null	Varchar2	No	No
attribute3	Null	Varchar2	No	No
attribute4	Null	Varchar2	No	No
attribute5	Null	Varchar2	No	No
attribute6	Null	Varchar2	No	No
attribute7	Null	Varchar2	No	No
attribute8	Null	Varchar2	No	No
attribute9	Null	Varchar2	No	No
attribute10	Null	Varchar2	No	No
attribute11	Null	Varchar2	No	No
attribute12	Null	Varchar2	No	No
attribute13	Null	Varchar2	No	No
attribute14	Null	Varchar2	No	No
attribute15	Null	Varchar2	No	No
comparison_operator_code	Null	Varchar2	Yes	No
context	Null	Varchar2	No	No

Table 5–196 QUALIFIERS_REC_TYPE Parameters

Parameter	Usage	Type	Req	Drv
created_by	Null	Number	No	No
created_from_rule_id	Null	Number	No	No
creation_date	Null	Date	No	No
end_date_active	Null	Date	No	No
excluder_flag	Null	Varchar2	No	No
last_updated_by	Null	Number	No	No
last_update_date	Null	Date	No	No
last_update_login	Null	Number	No	No
list_header_id	Null	Number	No	No
list_line_id	Null	Number	No	No
program_application_id	Null	Number	No	No
program_id	Null	Number	No	No
program_update_date	Null	Date	No	No
qualifier_attribute	Null	Varchar2	Yes	No
qualifier_attr_value	Null	Varchar2	Yes	No
qualifier_attr_value_to	Null	Varchar2	No	No
qualifier_context	Null	Varchar2	Yes	No
qualifier_datatype	Null	Varchar2	No	Yes
qualifier_grouping_no	Null	Number	No	No
qualifier_id	Null	Number	No	No
qualifier_precedence	Null	Number	No	No
qualifier_rule_id	Null	Number	No	No
request_id	Null	Number	No	No
start_date_active	Null	Date	No	No
qual_attr_value_from_ number	Null	Number	No	Yes
qual_attr_value_to_ number	Null	Number	No	Yes

Table 5–196 QUALIFIERS_REC_TYPE Parameters

Parameter	Usage	Type	Req	Drv
active_flag	Null	Varchar2	No	No
search_ind	Null	Number	No	Yes
qualifier_group_cnt	Null	Number	No	Yes
header_qual_exists_flag	Null	Varchar2	No	Yes
distinct_row_count	Null	Number	No	Yes
return_status	Null	Varchar2	No	No
db_flag	Null	Varchar2	No	No
operation	Null	Varchar2	Yes	No

attribute1–15

Default Value: FND_API.G_MISS_CHAR

comparison_operator_code

Default Value: FND_API.G_MISS_CHAR

context

Default Value: FND_API.G_MISS_CHAR

created_by

Default Value: FND_API.G_MISS_NUM

created_from_rule_id

Default Value: FND_API.G_MISS_NUM

creation_date

Default Value: FND_API.G_MISS_DATE

end_date_active

Default Value: FND_API.G_MISS_DATE

excluder_flag

Default Value: 'N'

last_updated_by

Default Value: FND_API.G_MISS_NUM

last_update_date

Default Value: FND_API.G_MISS_DATE

last_update_login

Default Value: FND_API.G_MISS_NUM

list_header_id

Default Value: FND_API.G_MISS_NUM

list_line_id

Default Value: FND_API.G_MISS_NUM

program_application_id

Default Value: FND_API.G_MISS_NUM

program_id

Default Value: FND_API.G_MISS_NUM

program_update_date

Default Value: FND_API.G_MISS_DATE

qualifier_attribute

Default Value: FND_API.G_MISS_CHAR

qualifier_attr_value

Default Value: FND_API.G_MISS_CHAR

qualifier_attr_value_to

Default Value: FND_API.G_MISS_CHAR

qualifier_context

Default Value: FND_API.G_MISS_CHAR

qualifier_datatype

Default Value: FND_API.G_MISS_CHAR

qualifier_grouping_no

Default Value: FND_API.G_MISS_NUM

qualifier_id

Default Value: From sequence QP_QUALIFIERS_S

qualifier_precedence

Default Value: FND_API.G_MISS_NUM

qualifier_rule_id

Default Value: FND_API.G_MISS_NUM

request_id

Default Value: FND_API.G_MISS_NUM

start_date_active

Default Value: FND_API.G_MISS_DATE

qual_attr_value_from_number

Default Value: FND_API.G_MISS_NUM

qual_attr_value_to_number

Default Value: FND_API.G_MISS_NUM

active_flag

Default Value: FND_API.G_MISS_CHAR

search_ind

Default Value: FND_API.G_MISS_NUM

qualifier_group_cnt

Default Value: FND_API.G_MISS_NUM

header_qual_exists_flag

Default Value: FND_API.G_MISS_CHAR

distinct_row_count

Default Value: FND_API.G_MISS_NUM

return_status

Default Value: FND_API.G_MISS_CHAR

db_flag

Default Value: FND_API.G_MISS_CHAR

operation

Default Value: FND_API.G_MISS_CHAR

QUALIFIERS_TBL_TYPE

The following table shows the parameters for this structure.

Table 5–197 QUALIFIERS_TBL_TYPE Parameters

Parameter	Usage	Type	Req	Drv
Qualifiers_Rec_Type	Null	Record	No	No

QUALIFIER_VAL_REC_TYPE

The following table shows the parameters for this structure.

Table 5–198 QUALIFIER_VAL_REC_TYPE

Parameter	Usage	Type	Req	Drv
created_from_rule	Null	Varchar2	No	No
list_header	Null	Varchar2	No	No
list_line	Null	Varchar2	No	No
qualifier_rule	Null	Varchar2	No	No
qualifier_attribute_desc	Null	Varchar2	No	No
qualifier_attr_value_desc	Null	Varchar2	No	No
qualifier_attr_value_to_desc	Null	Varchar2	No	No

created_from_rule

Default Value: FND_API.G_MISS_CHAR

list_header

Default Value: FND_API.G_MISS_CHAR

list_line

Default Value: FND_API.G_MISS_CHAR

qualifier_rule

Default Value: FND_API.G_MISS_CHAR

qualifier_attribute_desc

Default Value: FND_API.G_MISS_CHAR

qualifier_attr_value_desc

Default Value: FND_API.G_MISS_CHAR

qualifier_rule_value_to_desc

Default Value: FND_API.G_MISS_CHAR

QUALIFIERS_VAL_TBL_TYPE

The following table shows the parameters for this structure.

Table 5–199 QUALIFIERS_VAL_TBL_TYPE Parameters

Parameter	Usage	Type	Req	Drv
Qualifiers_Val_Rec_Type	Null	Record	No	No

LOCK_QUALIFIER_RULES

The following table shows the parameters for this structure.

Table 5–200 LOCK_QUALIFIER_RULES Parameters

Parameter	Usage	Type	Req	Drv
p_api_version_number	In	Number	No	No
p_init_msg_list	In	Varchar2	No	No

Table 5–200 LOCK_QUALIFIER_RULES Parameters

Parameter	Usage	Type	Req	Drv
p_return_values	In	Varchar2	No	No
x_return_status	Out	Varchar2	No	No
x_msg_count	Out	Number	No	No
x_msg_data	Out	Varchar2	No	No
p_QUALIFIER_RULES_rec	In	Qualifier_Rules_Rec_Type	No	No
p_QUALIFIER_RULES_val_rec	In	Qualifier_Rules_Val_Rec_Type	No	No
p_QUALIFIERS_tbl	In	Qualifiers_Tbl_Type	No	No
p_QUALIFIERS_val_tbl	In	Qualifiers_Val_Tbl_Type	No	No
x_QUALIFIER_RULES_rec	Out	Qualifier_Rules_Rec_Type	No	No
x_QUALIFIER_RULES_val_rec	Out	Qualifier_Rules_Val_Rec_Type	No	No
x_QUALIFIERS_tbl	Out	Qualifiers_Tbl_Type	No	No
x_QUALIFIERS_val_tbl	Out	Qualifiers_Val_Tbl_Type	No	No

p_init_msg_list

Default Value: FND_API.G_FALSE

p_return_values

Default Value: FND_API.G_FALSE

p_QUALIFIER_RULES_rec

Default Value: G_MISS_QUALIFIER_RULES_REC

p_QUALIFIER_RULES_val_rec

Default Value: G_MISS_QUALIFIER_RULES_VAL_REC

p_QUALIFIERS_tbl

Default Value: G_MISS_QUALIFIERS_TBL

p_QUALIFIERS_val_tbl

Default Value: G_MISS_QUALIFIERS_VAL_TBL

GET_QUALIFIER_RULES

The following table shows the parameters for this structure.

Table 5–201 GET_QUALIFIER_RULES Parameters

Parameter	Usage	Type	Req	Drv
p_api_version_number	In	Number	No	No
p_init_msg_list	In	Varchar2	No	No
p_return_values	In	Varchar2	No	No
x_return_status	Out	Varchar2	No	No
x_msg_count	Out	Number	No	No
x_msg_data	Out	Varchar2	No	No
p_qualifier_rule_id	In	Number	No	No
p_qualifier_rule	In	Varchar2	No	No
x_QUALIFIER_RULES_rec	Out	Qualifier_Rules_Rec_Type	No	No
x_QUALIFIER_RULES_val_rec	Out	Qualifier_Rules_Val_Rec_Type	No	No
x_QUALIFIERS_tbl	Out	Qualifiers_Tbl_Type	No	No
x_QUALIFIERS_val_tbl	Out	Qualifiers_Val_Tbl_Type	No	No

p_init_msg_list

Default Value: FND_API.G_FALSE

p_return_values

Default Value: FND_API.G_FALSE

p_qualifier_rule_id

Default Value: FND_API.G_MISS_NUM

p_qualifier_rule

Default Value: FND_API.G_MISS_CHAR

COPY_QUALIFIER_RULES

The following table shows the parameters for this structure.

Table 5–202 COPY_QUALIFIER_RULES Parameters

Parameter	Usage	Type	Req	Drv
p_api_version_number	In	Number	No	No
p_init_msg_list	In	Varchar2	No	No
p_return_values	In	Varchar2	No	No
p_commit	In	Varchar2	No	No
x_return_status	Out	Varchar2	No	No
x_msg_count	Out	Number	No	No
x_msg_data	Out	Varchar2	No	No
p_qualifier_rule_id	In	Number	No	No
p_qualifier_rule	In	Varchar2	No	No
p_to_qualifier_rule	In	Varchar2	No	No
p_to_description	In	Varchar2	No	No
x_qualifier_rule_id	Out	Number	No	No

p_init_msg_list

Default Value: FND_API.G_FALSE

p_return_values

Default Value: FND_API.G_FALSE

p_commit

Default Value: FND_API.G_FALSE

p_qualifier_rule_id

Default Value: FND_API.G_MISS_NUM

p_qualifier_rule

Default Value: FND_API.G_MISS_CHAR

p_to_description
Default Value: FND_API.G_MISS_CHAR

Validation of Qualifiers API

Standard Validation
Oracle Advanced Pricing validates all required columns in the Qualifiers API. For more information, see: *Oracle Pricing Technical Reference Manual*.

Other Validation
None

Error Handling
If any validation fails, the API will return error status to the calling module. The Qualifiers API processes the rows and reports the values in the following table for every record.

Table 5–203 Error Handling

Condition	PROCESS_STATUS	ERROR_MESSAGE
Success	5	null
Failure	4	actual error message

Example of Qualifiers API

The following code can also be found in file QPQFXMP1.sql in \$QP_TOP/patch/115/sql directory

```
--set serveroutput on

/*****
   Sample script which inserts a Qualifier Rule and a qualifier. Please read the
   Oracle Advanced Pricing User's Guide (Appendix A & B) to understand the
   flexfields and seed data.
   *****/

declare

gpr_return_status varchar2(1) := NULL;
gpr_msg_count number := 0;
```

```

gpr_msg_data varchar2(2000);
gpr_qualifier_rules_rec QP_Qualifier_Rules_Pub.Qualifier_Rules_Rec_Type;
gpr_qualifier_rules_val_rec QP_Qualifier_Rules_Pub.Qualifier_Rules_Val_Rec_Type;
gpr_qualifiers_tbl QP_Qualifier_Rules_Pub.Qualifiers_Tbl_Type;
gpr_qualifiers_val_tbl QP_Qualifier_Rules_Pub.Qualifiers_Val_Tbl_Type;
pqr_qualifier_rules_rec QP_Qualifier_Rules_Pub.Qualifier_Rules_Rec_Type;
pqr_qualifier_rules_val_rec QP_Qualifier_Rules_Pub.Qualifier_Rules_Val_Rec_Type;
pqr_qualifiers_tbl QP_Qualifier_Rules_Pub.Qualifiers_Tbl_Type;
pqr_qualifiers_val_tbl QP_Qualifier_Rules_Pub.Qualifiers_Val_Tbl_Type;
I number := 1;

begin

/* Create qualifier rule. Set the qualifier_rule_id to g_miss_num */
gpr_qualifier_rules_rec.qualifier_rule_id := FND_API.G_MISS_NUM;
gpr_qualifier_rules_rec.name := 'Sample1-QF 1024-2';
gpr_qualifier_rules_rec.description := 'Sample Qualifier Rule';
gpr_qualifier_rules_rec.operation := QP_GLOBALS.G_OPR_CREATE;

/* Create a qualifier */
I := 1;

gpr_qualifiers_tbl(I).QUALIFIER_ID := FND_API.G_MISS_NUM;
gpr_qualifiers_tbl(I).EXCLUDER_FLAG := 'N';
gpr_qualifiers_tbl(I).QUALIFIER_GROUPING_NO := 1;
gpr_qualifiers_tbl(I).QUALIFIER_CONTEXT := 'ORDER';

gpr_qualifiers_tbl(I).QUALIFIER_ATTRIBUTE := 'QUALIFIER_ATTRIBUTE13';

--Corresponds to Qualifier Attribute 'Order Category'
gpr_qualifiers_tbl(I).QUALIFIER_ATTR_VALUE := 'MIXED';
gpr_qualifiers_tbl(I).COMPARISON_OPERATOR_CODE := '=';
gpr_qualifiers_tbl(I).OPERATION := QP_GLOBALS.G_OPR_CREATE;
--dbms_output.put_line('before process qualifier rules');

```

QP_QUALIFIER_RULES_PUB.Process_Qualifier_Rules

```

( p_api_version_number=> 1
, p_init_msg_list=> FND_API.G_FALSE
, p_return_values=> FND_API.G_FALSE
, p_commit=> FND_API.G_FALSE
, x_return_status=> gpr_return_status
, x_msg_count=> gpr_msg_count
, x_msg_data=> gpr_msg_data

```

```
,    p_QUALIFIER_RULES_rec=> gpr_qualifier_rules_rec
,    p_QUALIFIER_RULES_val_rec=> gpr_qualifier_rules_val_rec
,    p_QUALIFIERS_tbl=> gpr_qualifiers_tbl
,    p_QUALIFIERS_val_tbl=> gpr_qualifiers_val_tbl
,    x_QUALIFIER_RULES_rec=> ppr_qualifier_rules_rec
,    x_QUALIFIER_RULES_val_rec=> ppr_qualifier_rules_val_rec
,    x_QUALIFIERS_tbl=> ppr_qualifiers_tbl
,    x_QUALIFIERS_val_tbl=> ppr_qualifiers_val_tbl
);

IF gpr_return_status <> FND_API.G_RET_STS_SUCCESS THEN

    RAISE FND_API.G_EXC_UNEXPECTED_ERROR;

END IF;

--dbms_output.put_line('after process qualifier rules');

EXCEPTION

    WHEN FND_API.G_EXC_ERROR THEN

gpr_return_status := FND_API.G_RET_STS_ERROR;

--Get message count and data

--dbms_output.put_line('err msg 1 is : ' || gpr_msg_data);

Rollback;

    WHEN FND_API.G_EXC_UNEXPECTED_ERROR THEN

        gpr_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;

--dbms_output.put_line(' msg count 2 is : ' || gpr_msg_count);

for k in 1 .. gpr_msg_count loop

gpr_msg_data := oe_msg_pub.get( p_msg_index => k,

p_encoded => 'F'
);

/*
        oe_msg_pub.Count_And_Get
(    p_count=> gpr_msg_count
```

```
,    p_data=> gpr_msg_data

);

*/

--Get message count and data
--dbms_output.put_line('err msg ' || k || 'is: ' || gpr_msg_data);

null;
end loop;
    Rollback;

    WHEN OTHERS THEN

        gpr_return_status := FND_API.G_RET_STS_UNEXP_ERROR ;

--Get message count and data
--dbms_output.put_line('err msg 3 is : ' || gpr_msg_data);

Rollback;
end;
/
commit;
exit;
```

See

Oracle Pricing Technical Reference Manual

Reverse Limits Application Program Interface

This section explains how to use the Reverse Limits API and how it functions in Oracle Advanced Pricing.

Reverse Limits API Features

The Reverse Limits API is a procedure that must be called directly by Oracle Order Management (OM) to reverse any promotional limits that were consumed by an order line when the line is either cancelled, amended or returned and no pricing engine call is made. The package QP_UTIL_PUB contains the procedure Reverse_Limits.

Note: The follow is not supported by this API:

- When the API is called with p_action_code = 'AMEND', an amended quantity that is greater than the originally ordered or consumed quantity is not supported.
 - Reversal of order level discounts for an order line cancellation, amendment, or return is not supported.
-
-

Functional Overview

When an item is ordered in OM, a call is made to the pricing engine to price the item and to determine the promotions that are to be given away for the ordered item. If promotional limits have been set up for the promotions, then the promotions that are given away are consumed from this limit.

However, when the order line is either cancelled, amended or returned without making a pricing engine call (since there was no need to calculate the price of return), the promotional limits that were consumed do not get reversed.

To reverse the consumed limit when such a return, amend or cancellation occurs, OM must call this API with the right parameters to increment the available limit balance(s).

Setting Up and Parameter Descriptions

The following chart describes all parameters used by the public Reverse Limits API. All of the inbound and outbound parameters are listed. Additional information on these parameters may follow. The input parameters must be passed to the Reverse Limits API by OM and the output parameters are passed to OM.

REVERSE_LIMITS

Table 5–204 Reverse_Limits Parameters

Parameter	Usage	Type	Req	Description
p_action_code	In	Varchar2	Yes	Valid values are 'RETURN', 'CANCEL' and 'AMEND'
p_cons_price_request_code	In	Varchar2	Yes	<p>Price_Request_Code of the last, successfully priced order_line that consumed the promotional limit:</p> <ul style="list-style-type: none"> ■ If the p_action_code is 'RETURN', then the price_request_code of the order line that is referenced by the returned line must be passed to this parameter. ■ If the p_action_code is 'CANCEL', then to this parameter must be passed the price_request_code of the order line/order that is being cancelled. ■ If the p_action_code is 'AMEND', then to this parameter must be passed the price_request_code of the order line that is being amended
p_orig_ordered_qty	In	Number	Yes ¹	<p>Priced_quantity of the last successfully priced order line that consumed the promotional limit:</p> <ul style="list-style-type: none"> ■ If the p_action_code is 'RETURN', then to this parameter must be passed the priced quantity of the order line that is referenced by the returned line. ■ If the p_action_code is 'AMEND', then to this parameter must be passed the originally priced quantity of the order line that is being amended.
p_amended_qty	In	Number	Yes ²	The new quantity to which the original ordered or priced quantity has been amended to. The Amended Qty can be less than or equal to the originally priced qty but not greater.
p_ret_price_request_code	In	Varchar2	Yes ³	The price request code of the return line.
P_returned_qty	In	Number	Yes ⁴	The returned quantity (the quantity on the return line).

Table 5–204 Reverse_Limits Parameters

Parameter	Usage	Type	Req	Description
x_return_status	Out	Varchar2	Yes	See Error Handling for possible values returned.
x_return_message	Out	Varchar2	Yes	Suitable Error Message if the API returns an error status.

The following table describes the notations listed in the preceding table:

Table 5–205 Notations

Note	Description
1	When p_action_code is 'RETURN' or 'AMEND'
2	When p_action_code is 'AMEND'
3	When p_action_code is 'RETURN'
4	When p_action_code is 'RETURN'

Validation of Reverse Limits API

Standard Validation

The calling application is responsible for passing the correct parameters to the Reverse Limits API. For more information on the data described in these columns, See *Oracle Pricing Technical Reference Manual*.

Other Validation

None

Error Handling

If any exception occurs in the Reverse Limits API, it should return the error status to the calling module. The Order Management application must call the Reverse Limits API while processing Cancellations, Amendments or Returned Order Lines, when the pricing engine is not called.

Table 5–206 Error Handling

Condition	x_return_status	x_return_message
Success	FND_API.G_RET_STS_SUCCE	null
Expected Error	FND_API.G_RET_STS_ERROR	actual error message
Unexpected Error	FND_API.G_RET_STS_UNEXP_ERROR	actual error message

Note: The Package Specification and Body files are QPXRTCN.S.pls and QPXRTCNB.pls and are available under the source control directory \$qp/patch/115/sql.

Additional Notes

- When the API is called with p_action_code = 'AMEND', the amended quantity can less than or equal to the originally ordered or consumed quantity but not greater than the originally priced quantity. If the amended quantity is greater, then the promotional limit may potentially be exceeded, in which case, this API (which is independent of the Limits Engine) cannot perform the logic that the Pricing Limits Engine would have performed under such circumstances. (Also, Oracle Order Management handles limit reversal for under-shipped or over-shipped quantities).
- When an order line is cancelled, amended or returned, this API may be used to only reverse limits on order-line level Discounts but not order-level discounts.
- Order Cancellations are supported by this API. To achieve this, the calling application (OM) must call this API for each order line on the order to be cancelled and once with the Order's price_request_code. Number of times to call the API = no. of order lines on the order to be cancelled + 1.
- The limit amount to be reversed/amended is calculated as a prorated value of the originally consumed amount.

Example of Reverse Limits API

The limit amount to be reversed/amended is calculated as a prorated value of the originally consumed amount as in the following examples:

- a. If an order line item has a unit price of \$100 and the original priced/ordered quantity is 10 then the gross revenue is \$1000. If the order line is eligible for a lumpsum discount of \$20, then the actual revenue is

$\$1000 - \$20 = \$980$ which is a selling price of $\$98$ ($\$980/10$) per unit of the item effectively.

If the priced quantity is amended to 5 from 10, then since selling price must be fixed at $\$98$ (since there's no repricing), the new total should be $5 \times \$98 = \490 which is effectively a lumpsum discount of $\$10$ for an order line of qty 5 with unit price = $\$10$ (for example, $5 \times \$10 - \10).

- b. With the pro-rated approach, the new benefit consumption against the limit would be calculated as follows:

$(\text{amended_qty} / \text{original_ordered_qty}) * \text{original consumed amount}$

$(5/10) * \$20$ (lumpsum benefit) = $\$10$ (new lumpsum) which is the same as the benefit(discount) calculated in a) above.

So pro-rating the original consumption correctly evaluates the new consumption against a limit when the quantity is amended.

Example 2

- a. If an order line item has a unit price of $\$100$ and the original priced/ordered quantity is 10 then the gross revenue is $\$1000$. If the order line is eligible for a 30% discount (for example, $30/100 \times \$1000 = \300), then the actual revenue is $\$1000 - \$300 = \$700$ which is a selling price of $\$70$ ($\$700/10$) per unit of the item effectively.

If the priced quantity is amended to 6 from 10, then the selling price must be fixed at $\$70$ (since there is no repricing), and the new total should be $6 \times \$70 = \420 . This is effectively a discount of $\$180$ ($6 \times \$100 - \420) which is an 18% percent discount consumption ($\$180/\1000)

- b. With the pro-rated approach, the new benefit consumption against the limit would be:

$(\text{amended_qty} / \text{original_ordered_qty}) * \text{original consumed amount}$

$(6/10) * 30$ (discount%) = 18 (new discount%) which is the same as the benefit (discount %) calculated in a) above.

So pro-rating the original consumption correctly evaluates the new consumption against a limit when the quantity is amended.

Round Price Application Program Interface

This section explains how to use the round_price API and how it functions in Oracle Advanced Pricing.

Round Price API Features

The round_price API is used to round the price using the rounding factor. The package QP_UTIL_PUB contains the procedure round_price.

Functional Overview

This API is called by pricing engine as well as directly from OM.

When it is called from pricing engine, the pricing engine passes the rounding factor and the operand (i.e. the price of an item) and so this API just uses the round() function and return the rounded operand.

But when this API is called directly from OM (while re-pricing an item), it passes the price_list_id and the operand. In this case, this API finds the rounding factor associated with price list, round the passed operand and then return the rounded operand.

The parameter p_operand_type can take 3 different values – ‘A’, ‘S’ or ‘R’, with the default value ‘S’. When ‘R’ is passed for p_operand_type then rounding_factor is returned in the out parameter p_rounded_operand, else rounded value (after applying rounding factor) of p_operand is returned.

The rounded operand / rounding factor is returned only if one of the following conditions is met :

- a. p_operand_type = ‘A’ and profile “QP: Selling Price Rounding Options” = ‘ROUND_ADJ’
- b. p_operand_type = ‘S’ and profile “QP: Selling Price Rounding Options” = ‘ROUND_ADJ’ or ‘NO_ROUND_ADJ’
- c. p_operand_type = ‘R’
- d. profile OE_UNIT_PRICE_ROUNDING = ‘Y’

When profile “QP: Multi Currency Installed” is ‘Y’ then rounding factor is retrieved by joining the tables qp_list_headers_b and qp_currency_details for the passed values of price list id, currency code and effective date else rounding factor is retrieved from table qp_list_headers_b for the price list id (without taking into consideration currency and effective date).

Setting Up and Parameter Descriptions

The following chart describes all parameters used by the round_price API. All of the inbound and outbound parameters are listed.

Table 5–207 round_price Parameters

Parameter	Usage	Type	Req	Drv
p_operand	IN	Number	Yes, if the p_operand_type is not 'R'	No
p_rounding_factor	IN	Number	No	No
p_use_multi_currency	IN	Varchar2	No	No
p_price_list_id	IN	Number	No	No
p_currency_code	IN	Varchar2	No	No
p_pricing_effective_date	IN	Date	No	No
x_rounded_operand	OUT	Number	No	No
x_status_code	OUT	Varchar2	No	No
p_operand_type	IN	Varchar2	No	No

Validation of Round Price API

Standard Validation

The caller is responsible for passing the right parameters to Round_Price.

For specific information on the data implied by these columns, See *Oracle Pricing Technical Reference Manual* for details.

Other Validation

None.

Error Handling

If any exception occurs in the Round_Price API, it returns appropriate status code as below.

Table 5–208 Status Code

Condition	x_status_code
Success	FND_API.G_RET_STS_SUCCE
Expected Error	FND_API.G_RET_STS_ERROR
Unexpected Error	FND_API.G_RET_STS_UNEXP_ERROR

NOTE: The Package Specification and Body files are QPXRTCNS.pls and QPXRTCNB.pls and are available under the source control directory \$QP_TOP/patch/115/sql.

Validate_Price_list_Curr_code Application Program Interface

This section explains how to use the Validate_Price_list_Curr_code API and how it functions in Oracle Advanced Pricing.

Validate_Price_list_Curr_code API Features

The Validate_Price_list_Curr_code API is used to validate a given price list with a currency code. The package QP_UTIL_PUB contains the procedure Validate_Price_list_Curr_code.

Functional Overview

While processing an order using a particular price list and currency code, it is imperative to verify whether the combination of price list and currency code is correct for the pricing effective date. If pricing effective date is not passed by calling application, the current date is defaulted. If profile QP: Multi Currency Installed is 'Y' then validation is done by joining the tables qp_list_headers_b and qp_currency_details else only table qp_list_headers_b is used.

Setting Up and Parameter Descriptions

The following chart describes all parameters used by the Validate_Price_list_Curr_code API. All of the inbound and outbound parameters are listed.

Table 5–209 Validate_Price_list_Curr_code Parameters

Parameter	Usage	Type	Req	Drv
l_price_list_id	IN	Number	Yes	No
l_currency_code	IN	Varchar2	Yes	No
l_pricing_effective_date	IN	Date	No	No
l_validate_result	OUT	Varchar2	No	No

Validation of Validate_Price_list_Curr_code API

Standard Validation

The caller is responsible for passing the right parameters to Validate_Price_list_Curr_code.

For more information on the data implied by these columns, See *Oracle Pricing Technical Reference Manual* for details.

Other Validation

None.

Error Handling

If any exception occurs in the Validate_Price_list_Curr_code API, it returns the value 'N' in outbound parameter l_validate_result.

NOTE: The Package Specification and Body files are QPXRTCNS.pls and QPXRTCNB.pls and are available under the source control directory \$QP_TOP/patch/115/sql.

Oracle Order Management EDI Transactions

Topics covered include:

- [Oracle Order Management](#) on page 6-2
- [Inbound Purchase Order \(POI/850/ORDERS\)](#) on page 6-4
- [Comment Text](#) on page 6-22
- [Review Order Management Open Interface Exceptions](#) on page 6-45
- [Order Import Item Cross-Referencing](#) on page 6-46
- [Data in the e-Commerce Gateway Transaction](#) on page 6-49
- [Order Management Transaction Summaries](#) on page 6-60
- [User Guide Update for Order Management Transactions](#) on page 6-92
- [Purchase Order Change Inbound \(POCI\) Program](#) on page 6-95
- [ReadMe for November, 2000 Patch](#) on page 6-102
- [Additional Information for the March 2001 Patch](#) on page 6-104

Oracle Order Management

The following Oracle Release Management transactions are summarized:

Table 6–1 Release Management Transactions

Transaction Name	Direction	Tran. Code	ASC X12	EDIFACT
Purchase Order	Inbound	POI	850	ORDERS
Purchase Order Change	Inbound	POCI	860	ORDCHG
Purchase Order Acknowledgement	Outbound	POAO	855	ORDRSP
Purchase Order Change Acknowledgement	Outbound	POCO	865	ORDRSP
Purchase Order Change				

The topics covered for inbound transactions include the following:

- Trading Partner Link to Oracle e-Commerce Gateway
- Oracle e-Commerce Gateway Required Fields
- Review Oracle e-Commerce Gateway Exceptions
- Resolve Oracle e-Commerce Gateway Exceptions
- Relevant Oracle Order Management Profiles and Setup Steps
- Order Management Open Interface Required Fields
- Review Order Management Open Interface Exceptions
- Resolve Order Management Open Interface Exceptions.

The topics covered for outbound transactions include the following:

- Transaction Handling Options
- Trading Partner Link to Oracle e-Commerce Gateway
- Relevant Oracle Order Management Profiles and Setup steps
- Extract Criteria
- Columns Updated Upon Extraction

The transaction requirements may change when enhancements are made such as additional data added to the transaction. Current transaction details can be found on Oracle Support's web site.

Current detail record layouts are reported via the Transaction Layout Definition Report and the Interface File Data Report.

Refer to Oracle Order Management appendix in the supplemental *Oracle e-Commerce Gateway Implementation Manual* for a diagram of the record hierarchy; record looping structure, and record summaries.

Inbound Purchase Order (POI/850/ORDERS)

This transaction is the inbound purchase order from the customer's procurement application into the Oracle Order Management application. This transaction uses the Order Management Order Import open interface. Refer to the *Oracle Manufacturing APIs and Open Interface Manual* and the *Oracle Order Management Open Interfaces, API & Electronic Messaging Guide* for more detail.

Trading Partner Link to Oracle e-Commerce Gateway

Customers and Customer sites are defined in either Oracle Receivables or Oracle Order Management. Included in the definition is the EDI Location Code that trading partners agree to exchange to represent the full detailed address. Often they do not send the full address, but just the EDI Location Code. This is a critical data field to Oracle e-Commerce Gateway.

The EDI Location Code is the link between a Customer/Customer site in Oracle Applications and the trading partner site definition in Oracle e-Commerce Gateway. This enables Oracle e-Commerce Gateway to access the detailed data about the Customer or Customer site in the base Oracle Applications without maintaining the detail data in Oracle e-Commerce Gateway.

To ensure that the trading partner link between Oracle e-Commerce Gateway and Oracle Applications is set up properly, verify that the Customer/Customer site and the EDI Location Code in Oracle Applications is the correct Customer/Customer site selected for the Trading Partner definition in Oracle e-Commerce Gateway. The selected Customer/Customer site and the EDI Location Code defined in Oracle Applications are displayed in the Define Trading Partners window, Assignment tab. If the data is not correct, you must make the appropriate changes for the transaction to be imported for the correct trading partner. This could involve either altering the Customer/Customer site in the base Oracle Application, or assigning a different Customer/Customer site to that EDI Location Code in Oracle e-Commerce Gateway.

Refer to the Trading Partner chapter in the *Oracle e-Commerce Implementation Manual* for recommendations on selecting the correct trading partner EDI Location Code for the control record 0010 for the transaction in the transaction interface file.

Oracle e-Commerce Gateway Required Fields

The following is a list of the Oracle e-Commerce Gateway required fields. These fields are required to authenticate the trading partner and transaction. If the required data is not provided in the transaction, the Oracle e-Commerce Gateway

import process fails the transaction. Then an exception message is displayed in the View Staged Documents window.

If the trading partner is valid and the transaction is enabled, the import process proceeds to validate the transaction using the user-defined column rules. If no process or column rule exceptions are detected, the Oracle e-Commerce Gateway import program will write the transaction to the Order Management Open Interface tables to be processed by the Order Management Open Interface API.

Table 6–2 Control Record Details

Oracle e-Commerce Gateway			
Column Name for Required Fields	Record Number	Position Number	Note
TEST_INDICATOR	0010	20	'T' or 'P'
TP_DOCUMENT_ID	0010	30	POI
TP_TRANSLATOR_CODE	0010	70	Translator identifier for this trading partner
TP_LOCATION_CODE	0010	80	The EDI Location Code

Control Record 0010

TEST_INDICATOR

This column represents the test or production indicator from the Trading Partner. If this value does not match the test or production indicator associated with the trading partner defined in Oracle e-Commerce Gateway, a process rule exception is detected. Then an exception message is displayed in the View Staged Documents window.

The valid values are T for test and P for production.

TP_DOCUMENT_ID

This column identifies the type of document being sent by the Trading Partner. If this document type is not enabled for the trading partner defined in Oracle e-Commerce Gateway, a process rule exception is detected. Then an exception message is displayed in the View Staged Documents window.

The valid value is POI for standard Purchase Order.

TP_TRANSLATOR_CODE, TP_LOCATION_CODE (EDI Location Code)

The two columns in combination uniquely identify a Trading Partner in Oracle e-Commerce Gateway. Once the trading partner definition is accessed, Oracle e-Commerce Gateway can verify whether the transaction is enabled for the Trading Partner.

If this trading partner is not defined in Oracle e-Commerce Gateway, a process rule exception is detected. Then an exception message is displayed in the View Staged Documents window.

Refer to the Trading Partner section to properly define your trading partners and get a better understanding of how these fields are used in the process.

Review Oracle e-Commerce Gateway Exceptions

Use the Oracle e-Commerce Gateway View Staged Documents window to review the Oracle e-Commerce Gateway transaction exceptions as specified by the process and column rules in the e-Commerce Gateway. Once the exceptions are identified and resolved, you can submit the transaction for reprocessing, ignore the exception during reprocessing, or delete the transaction. Select the preferred option in the View Staged Documents window.

Resolve Oracle e-Commerce Gateway Exceptions

To resolve Oracle e-Commerce Gateway exceptions, you can either correct the set up data in Oracle e-Commerce Gateway or Oracle Applications, or ask the Trading Partner to send a corrected transaction.

Note: You cannot change the actual content of the inbound transaction via the View Staged Document window.

If the Trading Partner sends a corrected transaction, be sure to delete the erroneous transaction from Oracle e-Commerce Gateway's staging tables using the View Staged Documents window. The duplicate transaction may cause confusion.

Relevant Oracle Order Management Profiles and Setup Steps

The required Order Management setup steps are the same whether the order is entered on-line or through Order Import. The following is a sample list of the Order Management setup steps needed to process orders through the e-Commerce Gateway or entered on-line.

- Customer and Customer Sites must be defined in Order Management/ Oracle Receivables.
- Order Types must be defined in Order Management.
- Item and appropriate customer items must be defined in Oracle Inventory.

Oracle Order Management Defaulting Functionality

Oracle Order Management provides powerful Defaulting functionality that is used by the Order Import process, as well as by orders entered manually using the Order Management forms. If a column is left null in the interface tables, then the Defaulting Rules are invoked, to provide a value for those columns. Many of the attributes can be defaulted per customer or customer site, by populating columns in the Customer tables (Order Management tab). Examples of those attributes are salesperson, shipping warehouse, payment terms, order type, freight terms, ship method, price list, and many others. There are many other sources for data that can be used in defaulting, including Pricing Agreements, Order Type, Line Type, Item, etc.

You can create your own Defaulting Rules by navigating to Setup, Rules, Defaults from the OM Superuser Menu. There is a white paper on Using Defaulting Rules in Oracle Order Management that gives an explanation of the Defaulting Rules forms with examples of their use. With appropriate use of Defaulting Rules, you can significantly reduce the amount of data that has to be entered manually or populated in the Order Import tables.

Order Import Open Interface Data Levels

The EDI standard transaction has three (3) key levels of data: Header, Line (item), and Schedules (Shipments).

The Order Import open interface tables have a two level structure where the order line and a single shipment are combined into one table entry for each combination. Order Import requires that line and shipment data be consolidated into the LINE level table. For example, if an order in a standard EDI transaction has an order line with five (5) shipments, then the order line data is combined with each shipment detail in the transaction interface file. Subsequently, there will be five (5) detail table entries in the application Order Import open interface tables.

Table 6–3 Standard Purchase Order EDI Transactions

Standard purchase order EDI transactions have a three level structure:				
Order Header				
Order Line				
Order Shipment (Schedules)				
The Order Import open interface tables have a two level structure				
Order Header				
Order Line (Combined Order Line and Order Shipment)				

The e-Commerce Gateway transaction accommodated the three level transactions so that the item data is written only once on Record 2000 (also records 2010/2020 if needed) followed by all the shipment (schedules) in records 3000-4900. The process will combine the item data to each set of its associated shipment data before writing the data to the Order Import LINE level table.

Though price is often set at the line level in a standard transaction, it was necessary to have the price at the shipment level in this transaction to allow different prices at the Order Import LINE level; hence all price data is on record 3000.

The following table has a simple example of the X12 850 Purchase Order segments mapping to the e-Commerce Gateway transaction.

Table 6–4 Example of the X12 850

X12 - 850 Transaction (3 Levels of Data)		e-Commerce Gateway POI File (Two Levels of Data)		Order Import Level (Two Levels of Data)
X12 Level	X12 Segment	Transaction Record	Data Content	LEVEL
Header	BEG	1000	Basic PO header	HEADER
Line (item 1)	PO1	2000	Basic PO item 1 (for first shipment)	
Shipment (first for Item 1)	SCH	3000/4000	Shipment Data (with first shipment)	LINE (Item 1/shipment 1)

Table 6–4 Example of the X12 850

X12 - 850 Transaction (3 Levels of Data)		e-Commerce Gateway POI File (Two Levels of Data)	Order Import Level (Two Levels of Data)
Shipment (second for Item 1)	SCH	3000/4000	Shipment Data (with second shipment)
			LINE (Item 1/shipment 2)
Shipment (third for Item 1)	SCH	3000/4000	Shipment Data (with third shipment)
			LINE (Item 1/shipment 3)
Line (item 2)	PO1	2000	Basic PO item 2 (for first shipment)
Shipment (first for Item 2)	SCH	3000/4000	Shipment Data (with first shipment)
			LINE (Item 2/shipment 1)
Shipment (second for Item 2)	SCH	3000/4000	Shipment Data (with second shipment)
			LINE (Item 2/shipment 2)
Shipment (third for Item 2)	SCH	3000/4000	Shipment Data (with third shipment)
			LINE (Item 2/shipment 3)
Shipment (fourth for Item 2)	SCH	3000/4000	Shipment Data (with fourth shipment)
			LINE (Item 2/shipment 4)

Note: An X12 Schedule (SCH) Segment contains Order Management Shipment data.

The following table illustrates the transaction structure with more detail.

Table 6–5 Sample key records for the Order Import LINE table

e-Commerce Gateway Inbound Purchase Order Transaction (Three Levels of Data)			Order Import Level (Two Levels of Data: HEADER and LINE)
Record	Data Content	LEVEL	
1000	Schedule header	HEADER	
2000	Item 1: line numbers, customer item, supplier item	(Copy all item data to each of its shipment below)	
2010			
2020	Item 1: INVENTORY_ITEM_SEGMENT_1 - 10		
	Item 1: INVENTORY_ITEM_SEGMENT_11 - 20		
3000	(1 st) Prices, configuration/options references	LINE (Item1/Schedule1 data)	
4000	(1 st) Schedule1 dates, etc.		
3000	(2 nd) Prices, configuration/options references	LINE (Item1/Schedule2 data)	
4000	(2 nd) Schedule2 dates, etc.		
3000	(3 rd) Prices, configuration/options references	LINE (Item1/Schedule3 data)	
3000	(3 rd) Schedule3 dates, etc.		
2000	Item 2: line numbers, customer item, supplier item	(Copy all item data to each of its shipment below)	
2010			
2020	Item 2: INVENTORY_ITEM_SEGMENT_1 - 10		
	Item 2: INVENTORY_ITEM_SEGMENT_11 - 20		

Table 6–5 Sample key records for the Order Import LINE table

e-Commerce Gateway Inbound Purchase Order Transaction		(Three Levels of Data)	Order Import Level (Two Levels of Data: HEADER and LINE)
3000	(1 st)	Prices, configuration/options references	LINE (Item2/Schedule1 data)
4000	(1 st)	Schedule1 dates, etc.	
3000	(2 nd)	Prices, configuration/options references	LINE (Item2/Schedule2 data)
4000	(2 nd)	Schedule1 dates, etc.	
2000	Item 3: line numbers, customer item, supplier item		(Copy all item data to each of its shipment below)
2010			
2020	Item 3: INVENTORY_ITEM_SEGMENT_11 - 20		
3000	(1 st)	Prices, configuration/options references	LINE (Item3/Schedule1 data)
4000	(1 st)	Schedule1 dates, etc.	
3000	(2 nd)	Prices, configuration/options references	LINE (Item3/Schedule2 data)
4000	(2 nd)	Schedule1 dates, etc.	

Line and Shipment Records

The following table illustrates key data on records 2000-4900 that contain the line and shipment detail required by Order Import. Refer to the transaction record layout for the position of the data on the record.

Table 6–6 Sample key data for the Order Import LINE table

Case	Record 2000	Customer Line	Customer Item	Item-ID-Type	Inventory Item
	3000	Operation	Price	Quantity	UOM
	4000	Shipment Number	Customer Request Date		
1	2000	1	ABC	CUST	XYZ
2	3000	INSERT	10.00		
3	4000	1	20000101 230000		
4	3000	INSERT	10.00		
5	4000	2	20000202 000000		
6	3000	INSERT	10.00		
7	4000	3	20000202 000000		
8	2000	2			WWW
9	3000	INSERT	20.00		
10	4000	1	20000101 230000		
11	3000	INSERT	20.00		
12	4000	2	20000102 220000		

Case 1: ITEM DETAIL

Record 2000 contains the customer line number, customer item, and/or supplier item.

Data for INVENTORY_SEGMENT1 through INVENTORY_SEGMENT20 may be written on records 2010 and 2020 if needed.

The item level data on records 2000-2020 is associated with each set of 3000-4900 records below it. They are combined to write an entry into the Order Import LINE table.

Case 2 and 3: SHIPMENT DETAIL

The first set of shipment level data where price and configuration data is on record 3000, and dates are on record 4000.

Case 4 and 5: SHIPMENT DETAIL

The second set of shipment level data where price and configuration data is on record 3000, and dates are on record 4000.

Case 6 and 7: SHIPMENT DETAIL

The third set of shipment level data where price and configuration data is on record 3000, and dates are on record 4000.

Case 8: ITEM DETAIL

Details for the second line item starts

Case 9 and 12: SHIPMENT DETAIL

Pairs of shipment level detail for the second line item.

Original System Reference Data in all Order Import Tables

Order Import requires a set of Original System reference fields. These are derived by the e-Commerce Gateway transaction process if one is not found in the transaction.

The POI transaction program derives the header and line level Original System reference fields, if data is not supplied in the transaction when the transaction was created.

The Original System reference fields are used to match change orders to original orders. Thus, these values must be unique within all orders for a customer and entered with the new orders.

The same algorithm that determines this data must be used to derive the data in the new order (POI) transaction and change order (POCI) transaction process to successfully matched change orders to new orders.

Table 6–7 Sample data that is derived

Table			
LEVEL	Data Element	Default Content	Sample
Header	Original System	Purchase Order Number	86420-03
	Document Reference	(-Release, if applicable)	
Line	Original System	Customer Line Number	001
	Line Reference		
Line	Original System	Current Request Date	19990308
	Shipment Reference	+(Ship-to) EDI Location Code	000000+CHIC-01

Higher level Original System data is propagated to the lower data levels in the transaction. For example, the derived ORIG_SYS_DOCUMENT REFERENCE at the header level is copied to the same field at the LINE and all other used levels for that transaction.

Data must be unique only within the given customer. This is illustrated in the following table.

Table 6–8 Unique Data

Table			
LEVEL	Data Element	Customer A	Customer B
Header	Original System	86420-03	86420-03
	Document Reference		
Line 1 (First Shipment)	Original System	86420-03	86420-03
	Document Reference (Copy from header)		
	Original System	001	001
	Line Reference (1)		
	Original System	19990308	19990308
	Shipment Reference (1)	000000+CHIC-01	000000+CHIC-01

Table 6–8 Unique Data

Table			
LEVEL	Data Element	Customer A	Customer B
Line 1 (Second Shipment)	Original System	86420-03	86420-03
	Document Reference (Copy from header)		
	Original System	001	001
	Line Reference (1)		
	Original System	19990308 000000+ATL-01	19990308 000000+ATL-01
	Shipment Reference (2)		
	Sold_to_Customer	1230	1240

ORIG_SYS_DOCUMENT_REF**Original System Document Reference**

Original System Document Reference is an identifier for the order within Order Management to guarantee its uniqueness within one operating unit.

Most often, it is the purchase order number, plus release number (if appropriate). A character such as the dash sign between each component improves readability.

The ORIG_SYS_DOCUMENT_REF is derived by the POI program or generated by a process such as the EDI Translator before the order is imported into the Order Import tables. It is not sent by the customer in the transaction as a field known as Original System Document Reference.

Sample Original System Document Reference

Original System Document Reference

86420

Purchase Order Number

86420

Note: The user may enter data in this field or use the purchase order number default.

Table 6–9 Sample Original System Document Reference

Original System Document Reference	Original System Document Reference
86420-03	86420-03
(Default)	(Default)
Purchase Order Number	Purchase Order Release Number
86420	03

Refer to CUSTOMER_PO_NUMBER under OE_HEADERS_INTERFACE table below for more detail.

ORIG_SYS_LINE_REF

Original System Line Reference

Original System Line Reference is an identifier for the line number in the order. This field is used for matching lines and schedules in subsequent transactions to the current order in Order Management.

The ORIG_SYS_LINE_REF and the ORIG_SYS_SHIPMENT_REF are stored at the line level in the base order tables. The combination of the ORIG_SYS_LINE_REF and the ORIG_SYS_SHIPMENT_REF (see below) must be unique within the ORIG_SYS_DOCUMENT_REFERENCE specified at the header.

The ORIG_SYS_LINE is derived by the POI program or generated by a process such as the EDI Translator before the order is imported into the Order Import tables. It is not sent by the customer in the transaction as a field known as Original System Line Reference.

Sample Original System Line Reference

Original System Line Reference

001

(Default)

Customer Line Number

001

Note: The user may enter data in this field or use the customer line number default.

ORIG_SYS_SHIPMENT_REF

Original System Shipment Reference

Original System Schedule Reference is an identifier for the shipment within a line in the order. This field is used for matching the line's schedules in subsequent transactions to the current order's schedules in Order Management. Often the current request date is the value used.

ORIG_SYS_SHIPMENT_REF (shipment indicator) must be unique within the ORIG_SYS_LINE_REF (line indicator) within the purchase order that is identified by the ORIG_SYS_DOCUMENT_REF (header indicator).

The same customer request data may be entered for several ship-to locations. To guarantee uniqueness concatenate the ship-to EDI LOCATION code to the date. For example, the customer request date (in the transaction) + ship-to EDI location code would be 120000308+CHIC-02. A character such as the plus sign between each component improves readability.

The ORIG_SYS_LINE is derived by the POI program or generated by a process such as the EDI Translator before the order is imported into the Order Import tables. The customer does not send it in the transaction, as the Original System Shipment Reference field.

Sample Original System Shipment Reference

Table 6–10 Sample Original System Shipment Reference

Original System Shipment Reference	Original System Shipment Reference
19990308 000000+CHIC-01	19990308 000000+CHIC-01
(Default)	(Default)
Current Request Date (includes time)	Ship To EDI_LOCATION CODE
19990308	CHIC-01

Note: The user may enter data in this field or use the default that is the CUSTOMER_REQUEST_SYSDATE+ (Ship-to) EDI_LOCATION CODE.

Note: Use a plus sign between these fields to improve readability.

Alternative Original System Shipment Reference

If your trading partner can retain unique shipment numbers to identify the shipment level (shipment/schedule dates, ship-to locations) data in their purchasing application, the shipment number may be copied into the ORIG_SYS_SHIPMENT_REF field in the transaction file.

If you maintain the same coding convention for the three original reference fields for both the inbound purchase order and the inbound purchase order change, the change order process can match the order data in the base Order Management tables. It is assumed that you will only modify the shipment records that changed, and add shipment records that were not already present.

In another scenario, the trading partner may send all the shipment records, even shipment records with no changes for your review, if you cannot accurately match the shipment records from the purchasing application.

Sample Original System Shipment Reference

Original System Shipment Reference

6

Shipment Number

(You copied from the transaction)

6

Table 6–11 Sample of all Original System Data using Shipment Numbers

LEVEL	Data Element	Content	Sample	Note
Header	Original System Document Reference	Purchase Order Number (-Release, if applicable)	86420-03	the default
Line	Original System Line Reference	Customer Line Number	001	the default
Line	Original System Shipment Reference	Shipment Number	6	copied into the transaction

Table 6–12 Matching Change Order Data to base Order Using the Original System Reference Data

Order in base Order tables			Change Order Data in Order Import			
Original System			Original System			
Document Reference (PO)	Line Reference (line number)	Shipment Reference (shipment number)	Document Reference (PO)	Line Reference (line number)	Shipment Reference (shipment number)	Note on
86420-03	001	1	86420-03	001	1	Matching change order transaction to the order in the base order tables.
86420-03	001	2				Change the order data match to Line 001, shipment 1 in the base order.
86420-03	001	3	86420-03	001	3	No change in the order data found for Line 001, shipment 2. Do not update the record in the base order.
						Change the order data match to Line 001, shipment 3 in the base order.

Table 6–12 Matching Change Order Data to base Order Using the Original System Reference Data

Order in base Order tables	Change Order Data in Order Import			
Original System	Original System			
	86420-03	001	4	Change the order data not matching to base order line 001, shipment number 4 so add line 1, shipment number 4.
	86420-03	001	5	Change the order data not matching to base order line 001, shipment number 5 so add line 1, shipment number 5.
	86420-03	002	1	Change the order data not matching to base order line 002, shipment number 1. so add line 2, shipment number 1.
	86420-03	002	2	Change the order data not matching to base order line 002, shipment number 2 so add line 002, shipment number 2.

If the customer does not send a unique, non-changeable shipment level identifier that is assigned to the **Original System Shipment Reference**, such as the Shipment Number, then any subsequent change order processing will require human intervention to do the record matching and determine what shipment data will be updated.

- ORIG_SYS_CREDIT_REF (Original System Credit Reference),
- ORIG_SYS_DISCOUNT_REF (Original System Discount Reference),
- ORIG_SYS_LOTSERIAL_REF (Original System Lot-Serial Reference),
- ORIG_SYS_RESERVATION_REF (Original System Reservation Reference)**

Refer the *Oracle Manufacturing APIs and Open Interface Manual* for details.

DEFAULT:

There are no defaults to the Credit, Discount, Lot-Serial, or Reservation reference fields. You need to supply this data in the record if you are using those records.

*When testing this transaction be sure to change all values of the Original System References for each test, otherwise, the transaction will fail in the Order Import validation due to duplicate entries.

Customer Shipment Number

CUSTOMER_SHIPMENT_NUMBER is a counter for the set of shipment data within an item that is identified by the same CUSTOMER_LINE_NUMBER. The customer in the transaction sends this number. It can be used to match change order shipment records to the base order line record if you copy it to the ORIG_SYS_SHIPMENT_REF (not use the default). It has no purpose if the inbound purchase order process generates this number. Order Management has its own shipment number counter that is used in Order Import.

Table 6–13 Customer Shipment Number

Sample X12 Segment	X12 Data Element	Order Import Data Element for This Customer Generated Data	Order Import Data
First PO1	Assigned Identification on PO1	CUSTOMER_LINE_NUMBER	001
First SCH within last PO1	Assigned Identification on SCH	CUSTOMER_SHIPMENT NUMBER	1
Second SCH within last PO1	Assigned Identification on SCH	CUSTOMER_SHIPMENT NUMBER	2
Third SCH within last PO1	Assigned Identification on SCH	CUSTOMER_SHIPMENT NUMBER	3
Second PO1	Assigned Identification on PO1	CUSTOMER_LINE_NUMBER	002
First SCH within last PO1	Assigned Identification on SCH	CUSTOMER_SHIPMENT NUMBER	1
Second SCH within last PO1	Assigned Identification on SCH	CUSTOMER_SHIPMENT NUMBER	2

Comment Text

The e-Commerce Gateway allows only a maximum length of 1024 characters per field. To import text more than 1024 characters long, write a custom procedure to concatenate several fields to import them to the longer field allowed in the open interface tables.

Flags

There are several flags in the interface tables of Order Management that affect the Order Import processing. Valid values of these flags are Y, N, and null. Null means different things depending on the particular flag. The Order Import process sets the defaults if a code is not found in the transaction.

Most of the flag data elements are not activated on the transaction, since the defaults are usually adequate. Their data elements are listed at the end of the activated data elements in the Transaction Definition window.

Table 6–14 *Flags*

Flag	Description	Default	Possible Values
Booked	This flag can be used instead of populating the oe_actions_ interface table with the BOOK_ORDER action, and it has the same effect.		Y, N, Null
Cancelled	The Cancelled flag is typically used to indicate that the line or order being imported should be imported in a Cancelled state.	N, or Null	N, Null
Closed	The Closed flag is typically used to indicate that the line or order being imported should be imported in a Closed state.	N, or Null	N, Null

Table 6–14 Flags

Flag	Description	Default	Possible Values
Error	The Order Import process sets on the error flag whenever an error is encountered during the validation process.	N, or Null	N, Null
Ready	The ready flag indicates that the record will be processed in the Order Import Process.	Y, or Null	Y, N, Null
Rejected	Rejected orders or rejected order lines are <i>DELETED</i> during the next execution of the Order Import program.	N, or Null	N, Null

Validate Mode Parameter in Concurrent Manager

There is a validate mode parameter in Concurrent Manager for executing Order Import. This parameter tells the process to only validate the record, and not to process valid records any further. Base Order Management tables will not have records inserted, updated, or deleted.

Table 6–15 Validate Mode Parameter

(Only)		
Validate		
READY_FLAG	Parameter	Processing
N	Y	Record is not processed
N	N	Record is not processed
Y or NULL	Y	Process to Validate Only
Y or NULL	N	Process to Insert/Update/Delete in Base Table

Order Import Open Interface Data

The Order Management Order Import Open Interface is used by the Inbound Purchase Order transaction. This validates the incoming data entered into the Order Management Open Interface tables by the Oracle e-Commerce Gateway import program or any other loading program.

The data for the Order Import tables have the following sources:

- Data in the transaction from the trading partner or determined by the EDI Translator.
- Data derived or hard codes by the POI Program given the trading partner or the presence of data in the transaction.
- Data derived by the Order Import process when it calls APIs to the Oracle tables.

The following is a list of the Order Management Open Interface required fields or other notations about the data elements. These fields are required for the Order Management Open Interface Import program to successfully process and move the data from the Order Management Open Interface tables into the Oracle Order Management base application tables.

Required fields noted as derived or hard coded do not require a value in the transaction on the transaction interface file since the values are determined by the Oracle e-Commerce Gateway process.

Refer to the *Oracle Order Management User's Guide* and the *Oracle Manufacturing APIs and Open Interface Manual* for more detail.

Table 6–16 Sample Columns of OE_HEADERS_INTERFACE

Oracle Applications Column Name for Fields	Oracle e-Commerce Gateway Column Name	Hardcoded/ Derived	Record Number	Position Number
OPERATION_CODE	OPERATION_CODE(_EXT1)	If Null, default to 'INSERT'	1000	030/040
CUSTOMER_PO_NUMBER	CUSTOMER_PO_NUMBER	(Required by Order Import)	1000	050
ORDER_TYPE	ORDER_TYPE(_EXT1)	This can be defaulted. Values must match those defined in Order Management. Use code conversion for the standard code in the transaction if necessary.	1000	080/090
ORDER_CATEGORY	ORDER_CATEGORY(_EXT1)	If Null, default to 'ORDER'	1000	100/110
ORIG_SYS_DOCUMENT_REF	ORIG_SYS_DOCUMENT_REF	If Null, derived	1000	120
ORDER_DATE_TYPE	ORDER_DATE_TYPE(_EXT1)	If Null, default to 'SHIP' so the dates are interpreted as ship dates and not delivery dates.	1000	130/140
CUSTOMER_NAME	CUSTOMER_NAME (not required)	Derived given the EDI Location code and Translator code on Control Record 0010. Data on Record 1400 will be not be imported unless the customer was not found.	1400	010
CUSTOMER_NUMBER	CUSTOMER_NUMBER (not required)	Derived given the EDI Location code and Translator code on Control Record 0010. Data on Record 1400 will not be imported unless the customer was not found.	1400	020

Table 6–16 Sample Columns of OE_HEADERS_INTERFACE

Oracle Applications Column Name for Fields	Oracle e-Commerce Gateway Column Name	Hardcoded/ Derived	Record Number	Position Number
(All Sold to Address Data)	SOLD TO Address	Derived given the EDI Location code and Translator code on Control Record 0010.	1410	
CREATED_BY	CREATED_BY	Derived from the Requester ID for the concurrent manager request that.		
CREATION_DATE	CREATION_DATE	Set to SYSDATE		
LAST_UPDATE_DATE	LAST_UPDATE_DATE	Set to SYSDATE		
LAST_UPDATED_BY	LAST_UPDATED_BY	Derived from the Requester ID for the concurrent manager request that. Please check		
ORDER_SOURCE_ID	ORDER_SOURCE_ID	Hard code: 6 meaning EDI		
SOLD_TO_ORD_ID	SOLD_TO_ORD_ID	Derived during the Trading Partner lookup given the EDI Location code and Translator code on Control Record 0010		

Table 6–17 Sample Columns of OE_LINES_INTERFACE

Oracle Applications Column Name for Fields	Oracle e-Commerce Gateway Column Name	Hardcoded/ Derived	Record Number	Position Number
ORIG_SYS_DOCUMENT_REF	ORIG_SYS_DOCUMENT_REF	If Null, derived	2000	030
ORIG_SYS_LINE_REF	ORIG_SYS_LINE_REF	If Null, derived	2000	040
ORIG_SYS_SHIPMENT_REF	ORIG_SYS_SHIPMENT_REF	If Null, derived	2000	050
OPERATION_CODE	OPERATION_CODE(EXT1)	If Null, default to 'INSERT'	2000	060/070
TOP_LINE_MODEL_REF	TOP_LINE_MODEL_REF	(See Configurations above)	2000	080

Table 6–17 Sample Columns of OE_LINES_INTERFACE

Oracle Applications Column Name for Fields	Oracle e-Commerce Gateway Column Name	Hardcoded/ Derived	Record Number	Position Number
LINE_TO_LINE_REF	LINE_TO_LINE_REF	(See Configurations above)	2000	090
MODEL_GROUP_NUMBER	MODEL_GROUP_NUMBER	(See Configurations above)	2000	100
OPTION_NUMBER	OPTION_NUMBER	(See Configurations above)	2000	110
COMPONENT_CODE	COMPONENT_CODE	(See Configurations above)	2000	120
SORT_ORDER	SORT_ORDER	(See Configurations above)	2000	130
LINE_NUMBER	LINE_NUMBER	If Null, derived	2010	020
CUSTOMER_LINE_NUMBER	CUSTOMER_LINE_NUMBER	If Null, set to LINE_NUMBER.	2010	030
ITEM_TYPE_CODE	ITEM_TYPE_CODE	If Null, default to STANDARD	2010	080/090
CALCULATE_PRICE_FLAG	CALCULATE_PRICE_FLAG	If Null, default to 'Y' for yes.		
CREATED_BY	CREATED_BY	Derived		
CREATION_DATE	CREATION_DATE	Set to SYSDATE		
DELIVERY_ID	DELIVERY_ID	Derived		
LAST_UPDATE_DATE	LAST_UPDATE_DATE	Set to SYSDATE		
LAST_UPDATED_BY	LAST_UPDATED_BY	Derived		
ORDER_SOURCE_ID	ORDER_SOURCE_ID	Hard code: 6 meaning EDI		
READY_FLAG	READY_FLAG	Set to 'Y' for yes.		
Shipment Dates			3000	many
CUSTOMER_SHIPMENT_NUMBER	CUSTOMER_SHIPMENT_NUMBER	Increment by one for each set of shipment data within an item that is identified by the same CUSTOMER_LINE_NUMBER.	4000	040
Ship to Location (Line Level)			3800	many

Note: Column names denoted with (_EXT1) indicates that it can have code conversion for that data element. Either the internal value or the external value will be passed to the Order Import tables following the logic of the code conversion process. Refer to the Code Conversion section of the Oracle e-Commerce Gateway Implementation Guide.

The following table levels of data are included in the transaction file at both the Order header and Order line level. Additional Original System reference data is required at the Order line level. Data is only required if these tables are used.

- OE_CREDITS_INTERFACE
- OE_PRICE_ADJS_INTERFACE
- OE_RESERVTONS_INTERFACE
- OE_ACTIONS_INTERFACE
- OE_PRICE_ATTN_INTERFACE

The following table level of data is included in the transaction file at the Order line level only. Data is only required if the table is used.

- OE_LOTSERIALS_INTERFACE

Leave at least one column activated to appear in the transaction interface file.

Table 6–18 HEADER LEVEL RECORDS

Oracle Applications Column Name for Required Fields	Oracle e-Commerce Gateway Column Name	Hardcoded/ Derived	Record Number	Position Number
OE_CREDITS_INTERFACE				
ORIG_SYS_DOCUMENT_REF	ORIG_SYS_DOCUMENT_REF	Required	1700	010
ORIG_SYS_CREDIT_REF	ORIG_SYS_CREDIT_REF	Required	1700	030
OPERATION_CODE	OPERATION_CODE(_EXT1)	If Null, default to 'INSERT'	1700	040/050
ORDER_SOURCE_ID	ORDER_SOURCE_ID	Hard code: 6 meaning EDI		
CREATED_BY	CREATED_BY	Derived		

Table 6–18 HEADER LEVEL RECORDS

Oracle Applications Column Name for Required Fields	Oracle e-Commerce Gateway Column Name	Hardcoded/ Derived	Record Number	Position Number
LAST_UPDATED_BY	LAST_UPDATED_BY	Derived		
CREATION_DATE	CREATION_DATE	Set to SYSDATE		
LAST_UPDATE_DATE	LAST_UPDATE_DATE	Set to SYSDATE		
OE_PRICE_ADJS_INTERFACE				
ORIG_SYS_DOCUMENT_REF	ORIG_SYS_DOCUMENT_REF	Required	1800	010
ORIG_SYS_DISCOUNT_REF	ORIG_SYS_DISCOUNT_REF	Required	1800	030
OPERATION_CODE	OPERATION_CODE(_EXT1)	If Null, default to INSERT	1800	040/050
ORDER_SOURCE_ID	ORDER_SOURCE_ID	Hard code: 6 meaning EDI		
LAST_UPDATED_BY	LAST_UPDATED_BY	Derived		
LAST_UPDATE_DATE	LAST_UPDATE_DATE	Set to SYSDATE		
CREATION_DATE	CREATION_DATE	Set to SYSDATE		
CREATED_BY	CREATED_BY	Derived		
OE_RESERVTRANS_INTERFACE				
ORIG_SYS_DOCUMENT_REF	ORIG_SYS_DOCUMENT_REF	Required	1900	010
ORIG_SYS_RESERVATION_REF	ORIG_SYS_RESERVATION_REF	Required	1900	030
OPERATION_CODE	OPERATION_CODE(_EXT1)	If Null, default to INSERT	1900	040/050
ORDER_SOURCE_ID	ORDER_SOURCE_ID	Hard code: 6 meaning EDI		
OE_ACTIONS_INTERFACE				

Table 6–18 HEADER LEVEL RECORDS

Oracle Applications Column Name for Required Fields	Oracle e-Commerce Gateway Column Name	Hardcoded/ Derived	Record Number	Position Number
ORIG_SYS_DOCUMENT_REF	ORIG_SYS_DOCUMENT_REF	Required	1950	010
OPERATION_CODE	OPERATION_CODE(_EXT1)	If Null, default to INSERT	1950	030/040
ORDER_SOURCE_ID	ORDER_SOURCE_ID	Hard code: 6 meaning EDI		

Table 6–19 Line Level Records

Oracle Applications Column Name for Required Fields	Oracle e-Commerce Gateway Column Name	Hardcoded/ Derived	Record Number	Position Number
OE_CREDITS_INTERFACE				
ORIG_SYS_DOCUMENT_REF	ORIG_SYS_DOCUMENT_REF	Required	5000	010
ORIG_SYS_LINE_REF	ORIG_SYS_LINE_REF	Required	5000	030
ORIG_SYS_SHIPMENT_REF	ORIG_SYS_SHIPMENT_REF	Required	5000	040
ORIG_SYS_CREDIT_REF	ORIG_SYS_CREDIT_REF	Required	5000	050
OPERATION_CODE	OPERATION_CODE(_EXT1)	If Null, default to INSERT	5000	060/070
ORDER_SOURCE_ID	ORDER_SOURCE_ID	Hard code: 6 meaning EDI		
CREATED_BY	CREATED_BY	Derived		
LAST_UPDATED_BY	LAST_UPDATED_BY	Derived		
CREATION_DATE	CREATION_DATE	Set to SYSDATE		
LAST_UPDATE_DATE	LAST_UPDATE_DATE	Set to SYSDATE		
OE_PRICE_ADJS_INTERFACE				
ORIG_SYS_DOCUMENT_REF	ORIG_SYS_DOCUMENT_REF	Required	6000	010
ORIG_SYS_LINE_REF	ORIG_SYS_LINE_REF	Required	6000	030
ORIG_SYS_SHIPMENT_REF	ORIG_SYS_SHIPMENT_REF	Required	6000	040

Table 6–19 Line Level Records

Oracle Applications				
Column Name for Required Fields	Oracle e-Commerce Gateway Column Name	Hardcoded/ Derived	Record Number	Position Number
ORIG_SYS_DISCOUNT_REF	ORIG_SYS_DISCOUNT_REF	Required	6000	050
OPERATION_CODE	OPERATION_CODE(_EXT1)	If Null, default to 'INSERT	6000	060/070
ORDER_SOURCE_ID	ORDER_SOURCE_ID	Hard code: 6 meaning EDI		
LAST_UPDATED_BY	LAST_UPDATED_BY	Derived		
LAST_UPDATE_DATE	LAST_UPDATE_DATE	Set to SYSDATE		
CREATION_DATE	CREATION_DATE	Set to SYSDATE		
CREATED_BY	CREATED_BY	Derived		
OE_RESERVTRANS_INTERFACE				
ORIG_SYS_DOCUMENT_REF	ORIG_SYS_DOCUMENT_REF	Required	7000	010
ORIG_SYS_LINE_REF	ORIG_SYS_LINE_REF	Required	7000	030
ORIG_SYS_SHIPMENT_REF	ORIG_SYS_SHIPMENT_REF	Required	7000	040
ORIG_SYS_RESERVATION_REF	ORIG_SYS_RESERVATION_REF	Required	7000	050
OPERATION_CODE	OPERATION_CODE(_EXT1)	If Null, default to 'INSERT	7000	060/070
ORDER_SOURCE_ID	ORDER_SOURCE_ID	Hard code: 6 meaning EDI		
OE_ACTIONS_INTERFACE				
ORIG_SYS_DOCUMENT_REF	ORIG_SYS_DOCUMENT_REF	Required	8000	010
ORIG_SYS_LINE_REF	ORIG_SYS_LINE_REF	Required	8000	030
ORIG_SYS_SHIPMENT_REF	ORIG_SYS_SHIPMENT_REF	Required	8000	040
OPERATION_CODE	OPERATION_CODE(_EXT1)	If Null, default to 'INSERT	8000	050/060
ORDER_SOURCE_ID	ORDER_SOURCE_ID	Hard code: 6 meaning EDI		
OE_LOTSERIALS_INTERFACE				

Table 6–19 Line Level Records

Oracle Applications				
Column Name for Required Fields	Oracle e-Commerce Gateway Column Name	Hardcoded/ Derived	Record Number	Position Number
ORIG_SYS_DOCUMENT_REF	ORIG_SYS_DOCUMENT_REF	Required	9000	010
ORIG_SYS_LINE_REF	ORIG_SYS_LINE_REF	Required	9000	030
ORIG_SYS_SHIPMENT_REF	ORIG_SYS_SHIPMENT_REF	Required	9000	040
ORIG_SYS_LOTSERIAL_REF	ORIG_SYS_LOTSERIAL_REF	Required	9000	050
ORDER_SOURCE_ID	ORDER_SOURCE_ID	Hard code: 6 meaning EDI		
LAST_UPDATED_BY	LAST_UPDATED_BY	Derived		
LAST_UPDATE_DATE	LAST_UPDATE_DATE	Set to SYSDATE		
CREATION_DATE	CREATION_DATE	Set to SYSDATE		
CREATED_BY	CREATED_BY	Derived		

OE_HEADERS_INTERFACE Table

CUSTOMER_PO_NUMBER

This column represents the purchase order number for the customer purchase order being imported into Oracle Order Management.

The number need not be unique for the customer.

If the purchase order is a release, the release number may be concatenated to the purchase order number separated by a dash, for example, PO1234-01 for purchase order PO1234 with release 01.

If you concatenate the customer’s purchase order number and release number, you may impact the ORIG_SYS_DOCUMENT_REFERENCE.

Table 6–20 Construction of CUSTOMER_PO_NUMBER:

Modified Purchase Order Number	
86420-03	
(Constructed by the EDI Translator to load into Oracle Order Management)	
Purchase Order Number from Customer's Purchasing Application	Purchase Order Release Number from Customer's Purchasing Application
86420	03

Table 6–21 Construction of ORIG_SYS_DOCUMENT_REF:

Original System Document Reference	
86420-03	
(Defaulted by the POI Program to use CUSTOMER_PO_NUMBER)	
Your Modified Purchase Order Number moved into CUSTOMER_PO_NUMBER	
86420 -03	Note: You concatenated the fields.

Table 6–22 Construction of ORIG_SYS_DOCUMENT_REF:

Original System Document Reference	
86420-03	
(Constructed by the EDI Translator then copied to the file)	
Purchase Order Number from Customer's Purchasing Application	Purchase Order Release Number from Customer's Purchasing Application
86420	03

This concatenation needs to be reversed in purchase order acknowledgments so customers see the two data elements separately. You may use flexfields to store the data elements separately to facilitate the purchase order acknowledgment transaction that requires two separate components.

OPERATION_CODE

This code identifies that the transaction is to be inserted, changed, or deleted. For new orders, the appropriate code is INSERT. Code conversion is permitted for this data element if a code is placed in the OPERATION_CODE_EXT1 field and the code is set up through the three code conversion forms in the e-Commerce Gateway. The valid values are INSERT, UPDATE, or DELETE.

DEFAULT:

The code is set to INSERT for this transaction by the e-Commerce Gateway if a code is not provided in the transaction or not derived through code conversion. This code is written to all the Order Import tables for the given transaction.

ORDER_CATEGORY

This code identifies the category of the order as defined in Order Management. The valid values are ORDER, MIXED, or RETURN.

DEFAULT:

The code is set to ORDER.

ORDER_DATE_TYPE

The expected type of date is defaulted in Order Import to the value set up for the customer or customer site. The value will apply to all dates in the transaction. You may override the customer default by entering one of the following codes in the transaction or derive it through code conversion. For code conversion, you can copy the date qualifier found in the standard transaction to the e-Commerce Gateway transaction and set up code conversion to derive one of the following codes.

- ARRIVAL
- SHIP

ORDER_SOURCE_ID

This code identifies that the transaction was loaded into Order Import via the e-Commerce Gateway.

DEFAULT:

The code is set to 6 meaning EDI.

ORDER_TYPE

This code identifies the type of order such as a standard order. You define these codes in Order Management.

There is no default. You must enter a value.

READY_FLAG

This flag indicates if the transaction is ready to be moved to the Order Management base order tables if all validation is passed. If you set it to N, even the valid order would remain in the Order Import tables until the flag is set to Y or the transaction is deleted from the Order Import tables.

SOLD_TO_ORG_ID

(Not activated on the transaction interface file.)

This code identifies that the customer for all the ship-to sites in the transaction. It is derived given the EDI Location Code and Translator on the Control Record 0010. Refer to the Trading Partner Section in the *Oracle e-Commerce Gateway Implementation Manual* for details

Original System Reference Data:**ORIG_SYS_DOCUMENT_REF**

See Definition above.

Update Columns:

The transaction program in the e-Commerce Gateway updates the following fields.

CREATED_BY

This code identifies who loaded this transaction into the Order Import tables. The code is set to the Requester ID associated with the concurrent manager request that processed this transaction.

CREATION_DATE

This is the date that the e-Commerce Gateway loaded the transaction into the Order Import table entries. This date is set to the system date.

LAST_UPDATE_DATE

This date is equal to the CREATION_DATE for this transaction. This date is set to the system date.

LAST_UPDATED_BY

This code is equal to the CREATED_BY code for this transaction.

OE_LINES_INTERFACE Table

Configuration Data

The following data elements must be entered for Configured items to show the relationship of line items in the file. Refer to the *Oracle Manufacturing APIs and Open Interface Manual* for details.

- TOP_LINE_MODEL_REF
- LINE_TO_LINE_REF
- MODEL_GROUP_NUMBER
- OPTION_NUMBER
- COMPONENT_CODE
- SORT_ORDER

CALCULATE_PRICE_FLAG

This flag indicates to Order Import to calculate a price (flag is Y) or not calculate a price (flag is N).

DEFAULT:

The code is set to 'Y' meaning Order Import will calculate a price even if one is found in the transactions. The calculated price and the price in the transaction will be compared. There is an error if there is a discrepancy.

CUSTOMER_LINE_NUMBER

This is the customer's line number as defined in their procurement application. There is no default. This reference is passed from the transactions.

LINE_NUMBER

This is the order line number as calculated by the e-Commerce Gateway as required by Order Import. Do not enter data into this field for new orders.

DEFAULT:

Increment the line counter by one for each LINE level record that is read.

OPERATION_CODE

See OPERATION_CODE under the OE_HEADERS_INTERFACE table above.

ORDER_SOURCE_ID

This code identifies that the transaction was loaded into Order Import via the e-Commerce Gateway.

DEFAULT:

The code is set to '6' meaning EDI.

Item Data:**CUSTOMER_ITEM_NAME**

This column represents the customer's item number for the buyer item as defined in their purchasing application.

CUSTOMER_ITEM_REVISION

This column represents the customer's item's revision level. This field is for display only. This field is not used in the Inventory tables to look up the supplier item.

INVENTORY_ITEM

This column represents the supplier's item number corresponding to the buyer item as defined in Oracle Inventory.

ITEM_REVISION

This column represents the supplier's item revision level. This field is for display only. This field is not used in the Inventory tables to look up the supplier item.

Original System Reference Data:

ORIG_SYS_DOCUMENT_REF,

ORIG_SYS_LINE_REF,

ORIG_SYS_SHIPMENT_REF

See Definition above.

Update Columns:

CREATED_BY

This code identifies who loaded this transaction into the Order Import tables. The code is set to the Requester ID associated with the concurrent manager request that processed this transaction.

CREATION_DATE

This is the date that the e-Commerce Gateway loaded the transaction into the Order Import table entries. This date is set to the system date.

LAST_UPDATE_DATE

This date is equal to the CREATION_DATE for this transaction. This date is set to the system date.

LAST_UPDATED_BY

This code is equal to the CREATED_BY code for this transaction.

OE_CREDITS_INTERFACE Table

If you do not plan to use this feature, you should deactivate the records that are associated to this table. If you do not deactivate the records, errors will occur in the Order Import validation, because the records will be incomplete. Records are activated and deactivated in the Transaction Definition window in the e-Commerce Gateway.

ORDER_SOURCE_ID

This code identifies that the transaction was loaded into Order Import via the e-Commerce Gateway.

DEFAULT:

The code is set to 6 meaning EDI.

Original System Reference Data:

ORIG_SYS_DOCUMENT_REF,

ORIG_SYS_LINE_REF,

ORIG_SYS_SHIPMENT_REF

See Definition above.

ORIG_SYS_CREDITS_REF

See the definition in the section Order Import in the revised *Oracle Order Management Open Interfaces, API, & Electronic Messaging Guide*.

Update Columns:**CREATED_BY**

This code identifies who loaded this transaction into the Order Import tables. The code is set to the Requester ID associated with the concurrent manager request that processed this transaction.

CREATION_DATE

This is the date that the e-Commerce Gateway loaded the transaction into the Order Import table entries. This date is set to the system date.

LAST_UPDATE_DATE

This date is equal to the CREATION_DATE for this transaction. This date is set to the system date.

LAST_UPDATED_BY

This code is equal to the CREATED_BY code for this transaction.

OE_PRICE_ADJS_INTERFACE Table

If you do not plan to use this feature, you should deactivate the records that are associated to this table. If you do not deactivate the records, errors will occur in the Order Import validation, because the records will be incomplete.

ORDER_SOURCE_ID

This code identifies that the transaction was loaded into Order Import via the e-Commerce Gateway.

DEFAULT:

The code is set to 6 meaning EDI.

Original System Reference Data:

ORIG_SYS_DOCUMENT_REF,

ORIG_SYS_LINE_REF,

ORIG_SYS_SHIPMENT_REF

See Definition above.

ORIG_SYS_DISCOUNT_REF

See the definition in the section Order Import in the revised *Oracle Order Management Open Interfaces, API, & Electronic Messaging Guide*.

Update Columns:

CREATED_BY

This code identifies who loaded this transaction into the Order Import tables. The code is set to the Requester ID associated with the concurrent manager request that processed this transaction.

CREATION_DATE

This is the date that the e-Commerce Gateway loaded the transaction into the Order Import table entries. This date is set to the system date.

LAST_UPDATE_DATE

This date is equal to the CREATION_DATE for this transaction. This date is set to the system date.

LAST_UPDATED_BY

This code is equal to the CREATED_BY code for this transaction.

OE_LOTSERIALS_INTERFACE Table

If you do not plan to use this feature, you should deactivate the records that are associated to this table. If you do not deactivate the records, errors will occur in the Order Import validation, because the records will be incomplete.

ORDER_SOURCE_ID

This code identifies that the transaction was loaded into Order Import via the e-Commerce Gateway.

DEFAULT:

The code is set to 6 meaning EDI.

Original System Reference Data:

ORIG_SYS_DOCUMENT_REF,

ORIG_SYS_LINE_REF,

ORIG_SYS_SHIPMENT_REF

See Definition above.

ORIG_SYS_LOTSERIAL_REF

See the definition in the section Order Import in the revised *Oracle Order Management Open Interfaces, API, & Electronic Messaging Guide*.

Update Columns:**CREATED_BY**

This code identifies who loaded this transaction into the Order Import tables. The code is set to the Requester ID associated with the concurrent manager request that processed this transaction.

CREATION_DATE

This is the date that the e-Commerce Gateway loaded the transaction into the Order Import table entries. This date is set to the system date.

LAST_UPDATE_DATE

This date is equal to the CREATION_DATE for this transaction. This date is set to the system date.

LAST_UPDATED_BY

This code is equal to the CREATED_BY code for this transaction.

OE_RESERVTRANS_INTERFACE Table

If you do not plan to use this feature, you should deactivate the records that are associated to this table. If you do not deactivate the records, errors will occur in the Order Import validation, because the records will be incomplete.

ORDER_SOURCE_ID

This code identifies that the transaction was loaded into Order Import via the e-Commerce Gateway.

DEFAULT:

The code is set to 6 meaning EDI.

Original System Reference Data:

ORIG_SYS_DOCUMENT_REF,

ORIG_SYS_LINE_REF,

ORIG_SYS_SHIPMENT_REF

See Definition above.

ORIG_SYS_RESERVATION_REF

See the definition in the section Order Import in the revised *Oracle Order Management Open Interfaces, API & Electronic Messaging Guide*.

Update Columns:

CREATED_BY

This code identifies who loaded this transaction into the Order Import tables. The code is set to the Requester ID associated with the concurrent manager request that processed this transaction.

CREATION_DATE

This is the date that the e-Commerce Gateway loaded the transaction into the Order Import table entries. This date is set to the system date.

LAST_UPDATE_DATE

This date is equal to the CREATION_DATE for this transaction. This date is set to the system date.

LAST_UPDATED_BY

This code is equal to the CREATED_BY code for this transaction.

OE_ACTIONS_INTERFACE Table

If you do not plan to use this feature, you should deactivate the records that are associated to this table. If you do not deactivate the records, errors will occur in the Order Import validation, because the records will be incomplete.

ORDER_SOURCE_ID

This code identifies that the transaction was loaded into Order Import via the e-Commerce Gateway.

Original System Reference Data:

ORIG_SYS_DOCUMENT_REF,

ORIG_SYS_LINE_REF,

ORIG_SYS_SHIPMENT_REF

See Definition above.

Update Columns:

CREATED_BY

This code identifies who loaded this transaction into the Order Import tables. The code is set to the Requester ID associated with the concurrent manager request that processed this transaction.

CREATION_DATE

This is the date that the e-Commerce Gateway loaded the transaction into the Order Import table entries. This date is set to the system date.

LAST_UPDATE_DATE

This date is equal to the CREATION_DATE for this transaction. This date is set to the system date.

LAST_UPDATED_BY

This code is equal to the CREATED_BY code for this transaction.

Review Order Management Open Interface Exceptions

At the completion of the Order Management Open Interface Order Import program, the orders with exception can be viewed in the Order Import window.

Refer to the *Oracle Order Management User's Guide*, and the *Oracle Manufacturing APIs and Open Interface Manual* for more detail.

Resolve Order Management Open Interface Exceptions

There are three ways to resolve Order Management Open Interface exceptions as follows:

- Using the information given by the View Staging Documents window in the e-Commerce Gateway, correct the set up data in Oracle Applications.

- Correct erroneous entries in the Order Management Order Import window.

- Request the customer to send a corrected transaction.

If you chose to update Oracle Applications data or change the erroneous entries using the Order Import window in Order Management, you can resubmit the Order Management Open Interface Order Import process to revalidate the transaction.

If you chose to have the customer send a corrected transaction, you must first delete the rejected Order data in the Order Management Open Interface tables using the Order Import window and then re-import the updated transaction using the Oracle e-Commerce Gateway.

Order Import Item Cross-Referencing

Item cross-referencing and validation are not performed in the e-Commerce Gateway. The e-Commerce Gateway writes data to the Order Import tables. Order Import calls the Process Order API that then calls Oracle Inventory's item cross reference APIs. The API is given the customer item or generic item (and a cross reference type to identify which type it is), and optionally the vendor item to determine the following:

Derive the internal supplier's inventory item given the customer item or generic inventory item, e.g., customer item, UPC, ISBN, or EAN number.

If both the customer/generic items and the supplier inventory item are sent in the transaction, compare the supplier item in the transaction to the derived internal supplier inventory item to be equal. If the supplier item in the transaction and the derived vendor inventory item do not equal, display an error and use the derived internal supplier inventory item in the order. The discrepancy should be discussed with the customer to determine the accurate supplier time and have the customer update their tables if necessary.

If only the supplier inventory item is sent, validate it.

Inventory Tables

Three Oracle Inventory tables contain various item data: customer item cross reference data, generic inventory items cross reference data, and the (internal) supplier inventory items. The first two tables will be searched to cross reference the items in the transaction to the internal inventory item.

Define all the cross-reference items and the internal inventory item in Oracle Inventory before running the e-Commerce Gateway inbound purchase order process and Order Import.

Table 6–23 Inventory Tables

Table	Content	Navigation
Customer Cross Reference table: MTL_CUSTOMER_ITEM_XREFS MTL_CUSTOMER_ITEMS	Customer specific items to cross reference to internal supplier item	<i>In Oracle Inventory:</i> Items> Customer Items> Customer Items
Inventory Item Cross Reference (generic) table: MTL_CROSS_REFERENCES	Generic inventory items such as UPC codes, ISBN codes, and EAN codes to cross reference to internal supplier item.	<i>In Oracle Inventory:</i> Items> Cross Reference (Click Assign to add cross references to the cross reference type selected.)
Vendor's Inventory Item table: MTL_SYSTEM_ITEMS	Internal vendor inventory items	<i>In Oracle Inventory:</i> Items> Master Items

Cross Reference Types for the Customer Item Cross Reference Table

Cross Reference Types qualify the item that was entered. The corresponding field in the transaction is the CUSTOMER_ITEM_ID_TYPE, is explained below.

The Customer Item Cross Reference table has codes CUST and INT seeded by Oracle Inventory. One of these codes can be written to the Order Import table if you wish to use the Customer Item Cross Reference table during item cross-referencing.

Table 6–24 Customer Item Cross Reference Table

Seeded		
Cross Reference Types		
for Customer Items	Meaning	Use
CUST	Customer Specific Number	Identify the item as a customer defined item.
INT	Internal Item	The item is recognized as the internal supplier's inventory item, but it is also defined in the customer cross reference table so the item can also be displayed in the Ordered Item field in the order.

Cross Reference Types for the Inventory Item Cross Reference Table

The user can define many different cross reference type codes for generic inventory items while setting up the item cross reference types and their names. The types are stored in the MTL_CROSS_REFERENCE_TYPES table. The corresponding field in the transaction is the CUSTOMER_ITEM_ID_TYPE.

One of the user defined codes can be written to the Order Import table if you wish to use the Inventory Item Cross Reference table during item cross referencing; no values are seeded by Oracle Inventory.

The following table has samples of the CUSTOMER_ITEM_ID_TYPE that may be defined by the user.

Table 6–25 *CUSTOMER_ITEM_ID_TYPE*

Sample of User Defined Cross Reference Types for Inventory Item	
	Meaning
UPC	Uniform Product Code
EAN	European Article Number (Use any EAN format that you choose)
ISBN	International Standard Book Number

Refer the *Oracle Inventory User’s Guide* for details on how to set up the various items.

Data in the e-Commerce Gateway Transaction

There are three essential columns in the transaction in record 2000. Either one or both item fields may be populated. The customer must decide if the customer item or generic item is placed in the e-Commerce Gateway transaction since only one field is in the transaction record to hold either field.

Supplier Item (INVENTORY_ITEM)

The supplier's inventory item number as stored in the customer's application and found in the transaction. It should be an item defined as an inventory item in Oracle Inventory since it validated against that table.

Customer Item (CUSTOMER_ITEM_NAME)

This field contains the customer item or a generic item as defined in the customer's inventory application.

Item Type (CUSTOMER_ITEM_ID_TYPE)

This field contains a valid qualifier that identifies the type of item placed in CUSTOMER_ITEM_NAME.

CUSTOMER_ITEM_ID_TYPE

The following table has samples of the CUSTOMER_ITEM_ID_TYPE and its interpretation.

Table 6–26 *CUSTOMER_ITEM_ID_TYPE*

Sample			
CUSTOMER_ITEM_ID_TYPE			
CUSTOMER_ITEM_ID_TYPE	as defined in Oracle Inventory	Interpretation on CUSTOMER_ITEM_NAME	Search will be Directed to which Table
	(in Record 2000, position 70 or 80 depending on code conversion)	(in Record 2000, position 50)	
Seeded Code	CUST	Customer Specific item	Customer Item Cross Reference table

Table 6–26 CUSTOMER_ITEM_ID_TYPE

CUSTOMER_ITEM_ID_TYPE	Sample CUSTOMER_ITEM_ID_TYPE	Interpretation on CUSTOMER_ITEM_NAME	Search will be Directed to which Table
	as defined in Oracle Inventory		
Seeded Code	INT	Internal Item in the Customer Item Cross Reference table	Customer Item Cross Reference table
User Defined Code	UPC	UPC code	Inventory Item Cross Reference table
User Defined Code	EAN	EAN number	Inventory Item Cross Reference table

If the wrong code is passed to the Order Import tables, you may not get a match on the item. If a code is entered that does not exist, there will be an error on the item. When in doubt, leave it null. (Processing of a null CUSTOMER_ITEM_ID_TYPE a later feature.)

CUSTOMER_ITEM_ID_TYPE Code Conversion

Code conversion on the product qualifier in standard transactions is necessary to associate them to the cross reference type as defined in Oracle Inventory.

Table 6–27 Code Conversion

X12 Product Qualifier (Data Element 235)	Sample Cross Reference Type	Search will be Directed to which Table
	in Oracle Inventory	
BP	CUST	Customer Item Cross Reference table
EN	EAN	Inventory Item Cross Reference table
IB	ISBN	Inventory Item Cross Reference table

Table 6–27 Code Conversion

X12 Product Qualifier (Data Element 235)	Sample Cross Reference Type in Oracle Inventory	Search will be Directed to which Table
UP	UPC	Inventory Item Cross Reference table
VP	INT	Customer Item Cross Reference table if the item is moved into CUSTOMER_ITEM_NAME in the transaction.
VP	(not applicable if the associated item is moved to INVENTORY_ITEM)	Inventory Item table if the item is moved into INVENTORY_ITEM in the transaction.

CUSTOMER_ITEM_ID_TYPE can be set up for code conversion in the e-Commerce Gateway so the standard code in position 80 can be converted to the valid Oracle code. Alternately, the EDI translator may place the valid Oracle Inventory defined code in position 70.

Table Searches

The first item to be used in a search is the customer item or generic item in the CUSTOMER_ITEM_NAME if the field is populated. The customer's purchasing application may not be maintaining accurate supplier items, if they rely on their own customer items and generic items for accurate order fulfilment. If only the supplier item is sent, then they are more likely to be maintained accurately by the customer.

If CUSTOMER_ITEM_ID_TYPE is not null, then only the table corresponding to its value is searched. This enables you to restrict customers to ordering items that are set up using only the specific CUSTOMER_ITEM_ID_TYPE. If you want all the tables to be searched, then leave this field null.

For example, if you know that the customer sent a customer item, and then you should populate CUSTOMER_ITEM_ID_TYPE with CUST, to speed up processing. Only the customer cross reference item table will be accessed for validating the CUSTOMER_ITEM_NAME.

If you know the customer sent a UPC code, then you should populate CUSTOMER_ITEM_ID_TYPE with the UPC. Only the generic item table will be accessed for validating the CUSTOMER_ITEM_NAME.

If the INVENTORY_ITEM is also populated in the transaction, it will be compared to the derived inventory item from the item cross-referencing. If they are different, a warning is generated. The derived inventory items will be used in the sales order.

The following table summarizes the tables that are accessed, given the data on record 2000 in the transaction.

Table 6–28 Accessed Tables

Data in the e-Commerce Gateway Transaction	Table Searched
CUSTOMER_ITEM_NAME is populated and CUSTOMER_ITEM_ID_TYPE is 'CUST' or 'INT' If the Supplier Inventory Item is in the transaction, compare it to the derived Supplier Inventory Item. Use the derived supplier inventory item, and create an error message (showing the sent inventory item) if the vendor inventory items do not match.	Customer Cross Reference table
CUSTOMER_ITEM_NAME is populated and CUSTOMER_ITEM_ID_TYPE is a valid Cross Reference Type (other than CUST or INV). If the Supplier Inventory Item is in the transaction, compare it to the derived Supplier Inventory Item. Use the derived supplier inventory item, and create an error message (showing the sent inventory item) if the supplier inventory items do not match.	Inventory Item Cross Reference (Generic) table
CUSTOMER_ITEM_NAME is NOT populated and only the Supplier Inventory Item is populated.	Inventory Item table (for internal vendor items)

If the CUSTOMER_ITEM_ID_TYPE is null but data is in the CUSTOMER_ITEM_NAME, the search will span all three tables until the item is found.

You run the risk of getting multiple table hits in the inventory item cross reference (generic) table if the same item number is defined under multiple CUSTOMER_ITEM_ID_TYPES such as UPC and EAN. However, since the naming conventions

for multiple items are diverse, the probability may be slight. Multiple items found will cause an error on the item.

Table 6–29 *CUSTOMER_ITEM_ID_TYPE*

Data in the e-Commerce Gateway Transaction	Order of the Tables Searched
CUSTOMER_ITEM_NAME is populated and <i>CUSTOMER_ITEM_ID_TYPE</i> is null	Customer Cross Reference table
If the Supplier Inventory Item is sent, compare it to the derived Supplier Inventory Item.	Inventory Item Cross Reference (Generic) table
Use the derived supplier inventory item, and create an error message (showing the sent inventory item) if the vendor inventory items do not match.	Inventory Item table (internal supplier items)

Combination of Item Data in the Transaction Record 2000

The following table summarizes the results of item cross-referencing given the combination of data on the record 2000.

Table 6–30 Combination of Item Data in the Transaction Record 2000

INVENTORY_ITEM	Interpretation on CUSTOMER_ITEM_NAME (a.k.a Customer Item)	Sample CUSTOMER_ITEM_ID_ TYPE as defined in Oracle Inventory	Result of Item Cross Referencing in Oracle Inventory
in Record 2000, position 30	in Record 2000, position 50	in Record 2000, position 70 or 80 depending on code conversion	
Null	any Customer Specific item	CUST	Result 1: Customer Item Cross Reference table is accessed to validate the customer item and derive its corresponding internal inventory item.
Null	any Internal Item in the Customer Item Cross Reference table	INT	See Result 1.
Null	any UPC code	UPC	Result 2: Inventory Item Cross Reference table is accessed with the generic item to validate the generic item and derive its corresponding internal inventory item.
Null	any EAN number	EAN	See Result 2.
any Inventory Item	Null	Null	Result 3: Inventory Item table is accessed to validate the item in the transaction. If the inventory item is invalid, an error message is created.

Table 6–30 Combination of Item Data in the Transaction Record 2000

INVENTORY_ITEM	Interpretation on CUSTOMER_ITEM_NAME (a.k.a Customer Item)	Sample	Result of Item Cross Referencing in Oracle Inventory
		CUSTOMER_ITEM_ID_ TYPE as defined in Oracle Inventory	
any Inventory Item	any Customer Specific item	CUST	<p>Result 4:</p> <p>Customer Item Cross Reference table is accessed to validate the customer item and derive its corresponding internal inventory item.</p> <p>The derived Inventory Item is compared to the Inventory Item in the transaction.</p> <p>If the derived and the sent Inventory Item do not match, an error message is generated where the sent inventory item is displayed in the error.</p>

Table 6–30 Combination of Item Data in the Transaction Record 2000

INVENTORY_ITEM	Interpretation on CUSTOMER_ITEM_NAME (a.k.a Customer Item)	Sample	Result of Item Cross Referencing in Oracle Inventory
		CUSTOMER_ITEM_ID_ TYPE as defined in Oracle Inventory	
any Inventory Item	any Internal Item in the Customer Item Cross Reference table	INT	See Result 4.
any Inventory Item	any UPC code	UPC	Result 5: Inventory Item Cross Reference table is accessed to validate the generic item and derive its corresponding internal inventory item. The derived Inventory Item is compared to the Inventory Item in the transaction. If the derived and the sent Inventory Item do not match, an error message is generated where the sent inventory item is displayed in the error.
any Inventory Item	any EAN number	EAN	See Result 5

Error Situations:

1. In all cases where more than one table entry is retrieved during the table search, an error message is created.
2. In all cases where the derived inventory item from the table search and the inventory item found in the transaction do not match, the derived inventory item replaces the sent inventory item in the Order Import tables, and the inventory item from the transaction is shown in the error message.

Results of Item Cross Referencing on the Sales Order

If the customer item/generic inventory item to supplier inventory item cross referencing is successful and the supplier inventory item validation is successful, the proper item data is moved to the following fields in the sales order in Order Management.

Table 6–31 Cross Referenced Sales Order

Data in the transaction	In the Order passed to the base Orders tables
CUSTOMER_ITEM_NAME if sent (that has the customer item or generic item)	Ordered Item
Always a validated INVENTORY_ITEM. If only the supplier item was sent, it is the inventory item if it is valid; otherwise it is the derived supplier item.	Inventory Item (the internal item)

Be sure to modify any form's folder to display both the Ordered Items and the Inventory Item.

Miscellaneous Item Data

ITEM_REVISION on Record 2000

The following item revision fields are for display only. These fields are not used in the Inventory tables to look up the Supplier Item for the order in Oracle Order Management:

Customer Item Revision (CUSTOMER_ITEM_REVISION)

The customer's item revision level as defined in the customer's purchasing application.

Supplier Item Revision (ITEM_REVISION)

The supplier's item revision level as defined in the supplier's order management application.

If the customer has multiple revisions to the base item that translates to different supplier items, then the customer item defined in Oracle Inventory should consist of the base item reference concatenated with the revision number/reference.

Table 6–32 Items and Revisions

Base Customer Item in Purchasing	Customer Item Revision in Purchasing	Customer Item into Order Management (defined in Oracle Inventory)	Supplier Item in Oracle Inventory
86420	03	86420-03	98765
86420	04	86420-04	98799

To return the base customer item and their revisions to the customer in the purchase order acknowledgement transactions, consider storing the two components in flexfields.

INVENTORY_ITEM_SEGMENT_1 on Record 2010

The internal Oracle Inventory table ID for the valid internal inventory item is moved to INVENTORY_ITEM_SEGMENT_1 in the order in the base Order Management table. If the transaction has data in INVENTORY_ITEM_SEGMENT_1 in Record 2010, the item cross-referencing is not performed. The presence of data in that field assumes that the table ID is to be used in the process.

Inbound Purchase Order Change (POCI/860/ORDCHG)

Note the record layout difference between the Inbound Purchase Order and the Inbound Purchase Order Change.

Table 6–33 R11i POI and POCI Record Difference

Level	Order Import Table	New Record	Position	Add Fields Record	Add Fields Position	Insert Fields Record	Insert Fields Position
Header	OE_HEADERS_INTFACE	1010	all				
Header	OE_HEADERS_INTFACE	1020-1050	all				
Header	OE_CREDITS_INTERFACE			1700	100-110		
Header	OE_PRICE_ADJS_INTERFACE			1800	200-230		

Table 6–33 R11i POI and POCI Record Difference

Header	OE_RESERVTSN_INTERFACE	1900	100-110
Header	OE_ACTIONS_INTERFACE	1950	
Line	OE_LINE_INTERFACE	3010	all
Line	OE_LINE_INTERFACE	3012-3018	all
Line	OE_CREDITS_INTERFACE	5000	130-140
Line	OE_PRICE_ADJS_INTERFACE	6020-6050	all
Line	OE_RESERVTSN_INTERFACE	7000	120-130
Line	OE_ACTIONS_INTERFACE	8000	
Line	OE_LOTSERIAL_INTERFACE	9000	120-130

Outbound Purchase Order Acknowledgment (POAO/855/ORDRSP) (POCAO/865/ORDRSP)

When a new sales order is changed to a Booked status within Order Management, an API is called. This API determines if the customer and customer location are defined as a Trading Partner within e-Commerce Gateway and if the POAO transaction is enabled for the Trading Partner.

When an existing sales order is modified and changed to a Booked status within Order Management, an API is called. This API determines if the customer and customer location are defined as a Trading Partner within e-Commerce Gateway and if the POCAO transaction is enabled for the Trading Partner.

If either of the above mentioned conditions is met, then the sales order data is either added or updated in the OE_HEADERS_ACKS and OE_LINE_ACKS tables. The ACKNOWLEDGE_FLAG must be set to Y in the OE_HEADERS_ACKS table to make the acknowledgment eligible for extraction.

Order Management Transaction Summaries

The following Oracle Order Management transactions are summarized:

Table 6–34 Order Management Transactions

Transaction Name	Direction	Tran. Code	ASC X12	EDIFACT	Documentation Status
Purchase Order	Inbound	POI	850	ORDERS	Included here
Purchase Order Change	Inbound	POCI	860	ORDCHG	In Process
Purchase Order Acknowledgment	Outbound	POAO	855	ORDRSP	In Process
Purchase Order Change Acknowledgment	Outbound	POCAO	865	ORDRSP	In Process

Current

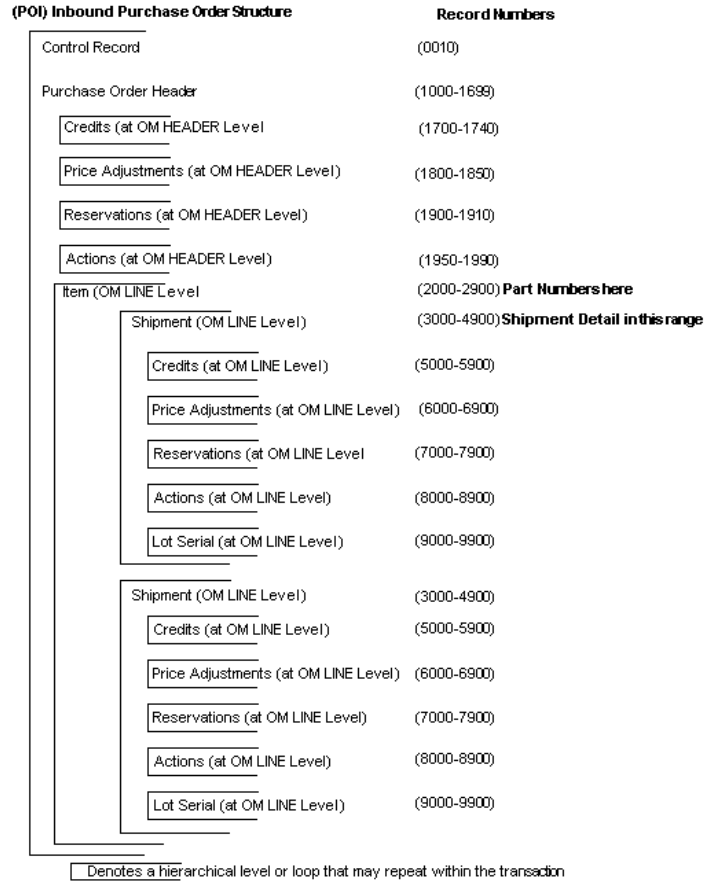
The transaction file may change when enhancements are made such as additional data added to the transaction.

Look for current transaction summaries on Oracle Support’s web site.

Current detail record layouts are reported via the Transaction Layout Report and the Interface File Data Report.

Inbound Purchase Order (POI/850/ORDERS)

Figure 6–1 POI Inbound Purchase Order Structure



NOTE: The customer and supplier item data in records 2000-2900 will be repeated on each shipment level data in records 3000-4900 to create entries in the Order Import LINE table.

Inbound Purchase Order (POI/850/ORDERS)

Table 6–35 Record occurrences within the transaction:

Records	CONTENT	OCCURENCES
0010	Control Record	Only one record occurrence per transaction
1000-1699	PO Header Records	Only one record occurrence per transaction
1700-1740	Credits	One set of records per header
1800-1850	Price Adjustments	One set of records per header
1900-1910	Reservations	One set of records per header
1950-1990	Actions	One set of records per header
2000-2900	PO Line	One set of records per item within the PO header
3000-4900	PO Line Shipment	One set of records per shipment within the PO Line
5000-5900	Credits	One set of records per shipment within the PO Line
6000-6900	Price Adjustments	One set of records per shipment within the PO Line
7000-7900	Lot Serial	One set of records per shipment within the PO Line
8000-8900	Reservations	One set of records per shipment within the PO Line
9000-9900	Actions	One set of records per shipment within the PO Line

Inbound Purchase Order (POI/850/ORDERS)

Table 6–36 Record Summary

Seq.	Data	Data Level	Record Number	Note
	Control Record	HEADER	0010	
	Basic Purchase Order Header	HEADER	1000	
	Currency Code & Conversion Rates	HEADER	1060	
	Payment Amount, Credit Card, Payment Type/Code, Tax Exempt	HEADER	1070	
	FOB Point, Freight Terms, Shipment Priority, Shipping Method	HEADER	1090	
	Shipping & Packing Instructions	HEADER	1100	

Table 6–36 Record Summary

Seq.	Data	Data Level	Record Number	Note
	Header Request Date, Demand Class	HEADER	1110	
	Header Flexfields	HEADER	1200-1230	Flexfields
	PO Header Flexfields	HEADER	1240-1270	Flexfields
	Global Header Flexfields	HEADER	1280-1320	Flexfields
	Customer Name & Number	HEADER	1400	Flexfields
	Sold-to Address	HEADER	1410	
	Sold-to Contact	HEADER	1420	
	Ship-to Address (Header Level)	HEADER	1500	
	Ship-to Contact (Header Level)	HEADER	1510	
	Invoice Address	HEADER	1520	
	Invoice Contact	HEADER	1530	
	Ordered By Name	HEADER	1540	
	Ship-from Address (Header Level)	HEADER	1600	
	Extension Tables: Purchase Order Header Data	HEADER		(custom)
	Credits Data	CREDIT	1700	
	Credit Flexfields	CREDIT	1710-1740	Flexfields
	Price Adjustments Data	PRICE ADJUSTMENT	1800	
	Discount/List Names (for price adjustments)	PRICE ADJUSTMENT	1810	
	Price Adjustment Flexfields	PRICE ADJUSTMENT	1820-1850	Flexfields
	Reservations Data	RESERVATION	1900	
	Reservations Flexfields	RESERVATION	1910-1940	Flexfields
	Action Data	ACTION	1950	
	Action Flexfields	ACTION	1960-1990	Flexfields

Table 6–36 Record Summary

Seq.	Data	Data Level	Record Number	Note
	Item (Original System Reference, Configuration)	LINE	2000	
	Inventory Item Segments	LINE DETAIL	2010-2020	Flexfields
	Item Detail (Items, Price)	LINE DETAIL	3000	
	Payment Type/Code, Tax Exempt	LINE DETAIL	3020	
	Project, Task, Contract Numbers	LINE DETAIL	3030	
	Service Comments	LINE DETAIL	3050	
	FOB Point, Freight Terms, Shipment Priority/Method, Freight Carrier	LINE DETAIL	3080	
	Demand Bucket	LINE DETAIL	3090	
	Shipment quantities, dates	LINE DETAIL	4000	
	Shipping & Packing Instructions	LINE DETAIL	4010	
	Item Flexfields	LINE DETAIL	4200-4230	Flexfields
	Global Header Flexfields	LINE DETAIL	4300-4340	Flexfields
	Pricing Flexfields	LINE DETAIL	4400-442	Flexfields
	Industry Header Flexfields	LINE DETAIL	4500-4560	Flexfields
	Service Flexfields	LINE DETAIL	4600-463	Flexfields
	Return Flexfields	LINE DETAIL	4700-473	Flexfields
	Ship-to Address (Line Level)	LINE DETAIL	4800	
	Ship-to Contact (Line Level)	LINE DETAIL	4810	
	Deliver to Contacts	LINE DETAIL	4820	
	Trader Partner Flexfields	LINE DETAIL	4830-4860	Flexfields
	Ship-from Address (Line Level)	LINE DETAIL	4870	
	Credits Data	CREDIT	5000	
	Credit Flexfields	CREDIT	5100-5130	Flexfields
	Price Adjustments Data	PRICE ADJUSTMENT	6000	

Table 6–36 Record Summary

Seq.	Data	Data Level	Record Number	Note
	Discount/List Names (for price adjustments)	PRICE ADJUSTMENT	6010	
	Price Adjustment Flexfields	PRICE ADJUSTMENT	6100-6130	Flexfields
	Reservations Data	RESERVATION	7000	
	Reservations Flexfields	RESERVATION	7100-7130	Flexfields
	Action Data	ACTION	8000	
	Action Flexfields	ACTION	8100-8130	Flexfields
	Lot Serial Data	LOT SERIAL	9000	
	Serial Lot Flexfields	LOT SERIAL	9100-9130	Flexfields

Inbound Purchase Order (POI/850/ORDERS)

Table 6–37 Transaction specific data in the Common Key positions 1-100:

Position	CODE	CONTENT
1-25	TP_CD	Trading Partner Code as defined in the EDI Translator
26-47	PO	Purchase order number
48-69	ITEM	Purchase order line number
70-91	(blank)	(Not needed)
92-95	(varies)	Record Number
96-97	(varies)	Record Layout
98-100	(varies)	Record Layout Qualifier

Table 6–38 Transaction specific data in the Common Key positions 1-100 per record:

Seq.	Data	Trading Partner	Ref 1	Ref 2	Reference 3	Record Number	Record Layout	Record
								Layout Qualifier
	Length	25	22	22	22	4	2	3
	Position	1-25	26-47	48-69	70-91	92-95	96-97	98-100
	Control Record	TP_CD	PO			0010	CT	CTL
	Basic Purchase Order Header	TP_CD	PO			1000	PO	PO1
	Currency Code & Conversion Rates	TP_CD	PO			1060	PY	CUR
	Payment Amount, Credit Card, Payment Type/Code, Tax Exempt	TP_CD	PO			1070	PY	TAX
	FOB Point, Freight Terms, Shipment Priority, Shipping Method	TP_CD	PO			1090	FB	SHP
	Shipping & Packing Instructions	TP_CD	PO			1100	NT	SHP
	Demand Class	TP_CD	PO			1110	MS	MIS
	Header Flexfields 1-4	TP_CD	PO			1200	A1	HD1
	Header Flexfields 5-9	TP_CD	PO			1210	A2	HD2
	Header Flexfields 10-14	TP_CD	PO			1220	A2	HD3
	Header Flexfield 15	TP_CD	PO			1230	A2	HD4
	Purchase Order Flexfields 1-4	TP_CD	PO			1240	A1	PO1
	Purchase Order Flexfields 5-9	TP_CD	PO			1250	A2	PO2
	Purchase Order Flexfields 1-14	TP_CD	PO			1260	A2	PO3
	Purchase Order Flexfield 15	TP_CD	PO			1270	A2	PO4
	Global Flexfields 1-4	TP_CD	PO			1280	A1	GL1

Table 6–38 Transaction specific data in the Common Key positions 1-100 per record:

Seq.	Data	Trading Partner	Ref 1	Ref 2	Reference 3	Record Number	Record Layout	Record
								Layout Qualifier
	Global Flexfields 5-9	TP_CD	PO			1290	A2	GL2
	Global Flexfields 10-14	TP_CD	PO			1300	A2	GL3
	Global Flexfields 15-19	TP_CD	PO			1310	A2	GL4
	Global Flexfield 20	TP_CD	PO			1320	A2	GL5
	Customer Name & Number	TP_CD	PO			1400	CU	CUS
	Sold-to Address (header level)	TP_CD	PO			1410	AD	SLD
	Sold-to Contact (header level)	TP_CD	PO			1420	CN	SLD
	Ship-to Address (header level)	TP_CD	PO			1500	AD	ST1
	Ship-to Contact (header level)	TP_CD	PO			1510	CN	ST1
	Invoice Address (header level)	TP_CD	PO			1520	AD	BT1
	Invoice Contact (header level)	TP_CD	PO			1530	CN	BT1
	Ordered By Name	TP_CD	PO			1540	CN	OBY
	Ship-from Address	TP_CD	PO			1600	AD	SFM
	Credits Data	TP_CD	PO			1700	CR	CRD
	Credit Flexfields 1-4	TP_CD	PO			1710	A1	CR1
	Credit Flexfields 5-9	TP_CD	PO			1720	A2	CR2
	Credit Flexfields 10-14	TP_CD	PO			1730	A2	CR3
	Credit Flexfield 15	TP_CD	PO			1740	A2	CR4
	Price Adjustments Data	TP_CD	PO			1800	PR	ADJ
	Discount/List Names	TP_CD	PO			1810	PR	AD1

Table 6–38 Transaction specific data in the Common Key positions 1-100 per record:

Seq.	Data	Trading Partner	Ref 1	Ref 2	Reference 3	Record Number	Record Layout	Record Layout Qualifier
	Price Adjustment Flexfields 1-4	TP_CD	PO			1820	A1	AD1
	Price Adjustment Flexfields 5-9	TP_CD	PO			1830	A2	AD2
	Price Adjustment Flexfields 10-14	TP_CD	PO			1840	A2	AD3
	Price Adjustment Flexfield 15	TP_CD	PO			1850	A2	AD4
	Reservations Data	TP_CD	PO			1900	RS	RES
	Reservation Flexfields 1-4	TP_CD	PO			1910	A1	RS1
	Reservation Flexfields 5-9	TP_CD	PO			1920	A2	RS2
	Reservation Flexfields 10-14	TP_CD	PO			1930	A2	RS3
	Reservation Flexfield 15	TP_CD	PO			1940	A2	RS4
	Action Data	TP_CD	PO			1950	AC	ACT
	Actions Flexfields 1-4	TP_CD	PO			1960	A1	AC1
	Actions Flexfields 5-9	TP_CD	PO			1970	A2	AC2
	Actions Flexfields 10-14	TP_CD	PO			1980	A2	AC3
	Actions Flexfield 15	TP_CD	PO			1990	A2	AC4
	Item Data (Original System Reference, Configuration Data)	TP_CD	PO	LINE		2000	IT	ITM
	Inventory Item Segments 1-10	TP_CD	PO	LINE		2010	IT	IS1
	Inventory Item Segments 11-20	TP_CD	PO	LINE		2020	IT	IS2
	Operation Code, Price Quantities, List Price	TP_CD	PO	LINE		3000	LN	LN1

Table 6–38 Transaction specific data in the Common Key positions 1-100 per record:

Seq.	Data	Trading Partner	Ref 1	Ref 2	Reference 3	Record Number	Record Layout	Record
								Layout Qualifier
	Customer Payment Terms, Tax Exempt	TP_CD	PO	LINE		3020	PY	TAX
	Project, Task, Contract Numbers	TP_CD	PO	LINE		3030	PJ	SRV
	Service Transaction Comments	TP_CD	PO	LINE		3050	SV	CM1
	FOB Point, Freight Terms, Shipment Priority/Method, Freight Carrier	TP_CD	PO	LINE		3080	FB	SHP
	Demand Bucket, Class, Stream	TP_CD	PO	LINE		3090	DM	DMD
	Shipment Data (UOM, Ship/Promise Dates)	TP_CD	PO	LINE		4000	DT	QTY
	Shipping & Packing Instructions	TP_CD	PO	LINE		4010	NT	NTE
	Item Flexfields 1-4	TP_CD	PO	LINE		4200	A1	IT1
	Item Flexfields 5-9	TP_CD	PO	LINE		4210	A2	IT2
	Item Flexfields 10-14	TP_CD	PO	LINE		4220	A2	IT3
	Item Flexfield 15	TP_CD	PO	LINE		4230	A2	IT4
	Global Header Flexfields 1-4	TP_CD	PO	LINE		4300	A1	GL1
	Global Header Flexfields 5-9	TP_CD	PO	LINE		4310	A2	GL2
	Global Header Flexfields 10-14	TP_CD	PO	LINE		4320	A2	GL3
	Global Header Flexfields 15-19	TP_CD	PO	LINE		4330	A2	GL4
	Global Header Flexfield 20	TP_CD	PO	LINE		4340	A2	GL5

Table 6–38 Transaction specific data in the Common Key positions 1-100 per record:

Seq.	Data	Trading Partner	Ref 1	Ref 2	Reference 3	Record Number	Record Layout	Record
								Layout Qualifier
	Pricing Flexfields 1-4	TP_CD	PO	LINE		4400	A1	PC1
	Pricing Flexfields 5-9	TP_CD	PO	LINE		4410	A2	PC2
	Pricing Flexfield 10	TP_CD	PO	LINE		4420	A2	PC3
	Industry Header Flexfields 1-4	TP_CD	PO	LINE		4500	A1	IN1
	Industry Header Flexfields 5-9	TP_CD	PO	LINE		4510	A2	IN2
	Industry Header Flexfields 10-14	TP_CD	PO	LINE		4520	A2	IN3
	Industry Header Flexfields 15-19	TP_CD	PO	LINE		4540	A2	IN4
	Industry Header Flexfields 20-24	TP_CD	PO	LINE		4540	A2	IN5
	Industry Header Flexfields 25-29	TP_CD	PO	LINE		4550	A2	IN6
	Industry Header Flexfield 30	TP_CD	PO	LINE		4560	A2	IN7
	Service Flexfields 1-4	TP_CD	PO	LINE		4600	A1	SV1
	Service Flexfields 5-9	TP_CD	PO	LINE		4610	A2	SV2
	Service Flexfields 10-14	TP_CD	PO	LINE		4620	A2	SV3
	Service Flexfield 15	TP_CD	PO	LINE		4630	A2	SV4
	Return Flexfields 1-4	TP_CD	PO	LINE		4700	A1	RT1
	Return Flexfields 5-9	TP_CD	PO	LINE		4710	A2	RT2
	Return Flexfields 10-14	TP_CD	PO	LINE		4720	A2	RT3
	Return Flexfield 15	TP_CD	PO	LINE		4730	A2	RT4
	Ship-to Address (Item Level)	TP_CD	PO	LINE		4800	AD	ST1

Table 6–38 Transaction specific data in the Common Key positions 1-100 per record:

Seq.	Data	Trading Partner	Ref 1	Ref 2	Reference 3	Record Number	Record Layout	Record
								Layout Qualifier
	Ship-to Contact (Item Level)	TP_CD	PO	LINE		4810	CN	ST1
	Invoice to Contacts	TP_CD	PO	LINE		4820	CN	IND
	Trading Partner Flexfields 1-4	TP_CD	PO	LINE		4830	A1	TP1
	Trading Partner Flexfields 5-9	TP_CD	PO	LINE		4840	A2	TP2
	Trading Partner Flexfields 10-14	TP_CD	PO	LINE		4850	A2	TP3
	Trading Partner Flexfield 15	TP_CD	PO	LINE		4860	A2	TP4
	Ship-from Address (Item Level)	TP_CD	PO	LINE		4870	AD	SFR
	Credits Data	TP_CD	PO	LINE		5000	CR	CRD
	Credit Flexfields 1-4	TP_CD	PO	LINE		5100	A1	CR1
	Credit Flexfields 5-9	TP_CD	PO	LINE		5110	A2	CR2
	Credit Flexfields 10-14	TP_CD	PO	LINE		5120	A2	CR3
	Credit Flexfield 15	TP_CD	PO	LINE		5130	A2	CR4
	Price Adjustments Data	TP_CD	PO	LINE		6000	PR	ADJ
	Discount/List Names	TP_CD	PO	LINE		6010	PR	AD1
	Price Adjustment Flexfields 1-4	TP_CD	PO	LINE		6110	A1	AD1
	Price Adjustment Flexfields 5-9	TP_CD	PO	LINE		6110	A2	AD2
	Price Adjustment Flexfields 10-14	TP_CD	PO	LINE		6120	A2	AD3
	Price Adjustment Flexfield 15	TP_CD	PO	LINE		6130	A2	AD4
	Reservations Data	TP_CD	PO	LINE		7000	RS	RES

Table 6–38 Transaction specific data in the Common Key positions 1-100 per record:

Seq.	Data	Trading Partner	Ref 1	Ref 2	Reference 3	Record Number	Record Layout	Record
								Layout Qualifier
	Reservation Flexfields 1-4	TP_CD	PO	LINE		7100	A1	RS1
	Reservation Flexfields 5-9	TP_CD	PO	LINE		7110	A2	RS2
	Reservation Flexfields 10-14	TP_CD	PO	LINE		7120	A2	RS3
	Reservation Flexfield 15	TP_CD	PO	LINE		7130	A2	RS4
	Action Data	TP_CD	PO	LINE		8000	AC	ACT
	Actions Flexfields 1-4	TP_CD	PO	LINE		8100	A1	AC1
	Actions Flexfields 5-9	TP_CD	PO	LINE		8110	A2	AC2
	Actions Flexfields 10-14	TP_CD	PO	LINE		8120	A2	AC3
	Actions Flexfield 15	TP_CD	PO	LINE		8130	A2	AC4
	Lot Serials Data	TP_CD	PO	LINE		9000	LT	SER
	Serial Lot Flexfields 1-4	TP_CD	PO	LINE		9100	A1	LT1
	Serial Lot Flexfields 5-9	TP_CD	PO	LINE		9110	A2	LT2
	Serial Lot Flexfields 10-14	TP_CD	PO	LINE		9120	A2	LT3
	Serial Lot Flexfield 15	TP_CD	PO	LINE		9130	A2	LT4

Inbound Purchase Order Change (POCI/860/ORDCHG)

Figure 6–2 POCI Inbound Purchase Order Change Structure

(POCI) Inbound Purchase Order Change Structure	Record Numbers
Control Record	(0010)
Purchase Order Header	(1000-1699)
Credits (at OM HEADER Level)	(1700-1740)
Price Adjustments (at OM HEADER Level)	(1800-1850)
Reservations (at OM HEADER Level)	(1900-1910)
Actions (at OM HEADER Level)	(1950-1990)
Item (OM LINE Level)	(2000-2900) Part Numbers here
Shipment (OM LINE Level)	(3000-4900) Shipment Detail in this range
Credits (at OM LINE Level)	(5000-5900)
Price Adjustments (at OM LINE Level)	(6000-6900)
Reservations (at OM LINE Level)	(7000-7900)
Actions (at OM LINE Level)	(8000-8900)
Lot Serial (at OM LINE Level)	(9000-9900)
Shipment (OM LINE Level)	(3000-4900)
Credits (at OM LINE Level)	(5000-5900)
Price Adjustments (at OM LINE Level)	(6000-6900)
Reservations (at OM LINE Level)	(7000-7900)
Actions (at OM LINE Level)	(8000-8900)
Lot Serial (at OM LINE Level)	(9000-9900)

Denotes a hierarchical level or loop that may repeat within the transaction

NOTE: The customer and supplier item data in records 2000-2900 will be repeated on each shipment level data in records 3000-4900 to create entries in the Order Import LINE table.

Inbound Purchase Order Change (POCI/860/ORDCHG)

Table 6–39 Record occurrences within the transaction:

Records	CONTENT	OCCURENCES
0010	Control Record	Only one record occurrence per transaction
1000-1699	PO Header Records	Only one record occurrence per transaction
1700-1740	Credits	One set of records per header
1800-1850	Price Adjustments	One set of records per header
1900-1910	Reservations	One set of records per header
1950-1990	Actions	One set of records per header
2000-2900	PO Line	One set of records per item within the PO header
3000-4900	PO Line Shipment	One set of records per shipment within the PO Line
5000-5900	Credits	One set of records per shipment within the PO Line
6000-6900	Price Adjustments	One set of records per shipment within the PO Line
7000-7900	Lot Serial	One set of records per shipment within the PO Line
8000-8900	Reservations	One set of records per shipment within the PO Line
9000-9900	Actions	One set of records per shipment within the PO Line

Inbound Purchase Order Change (POCI/860/ORDCHG)

Table 6–40 Record Summary

Seq.	Data	Data Level	Record Number	Note
	Control Record	HEADER	0010	
	Basic Purchase Order Header	HEADER	1000	
	Change Data	HEADER	1010	
	Change Comments	HEADER	1020-1050	
	Currency Code & Conversion Rates	HEADER	1060	
	Payment Amount, Credit Card, Payment Type/Code, Tax Exempt	HEADER	1070	

Table 6–40 Record Summary

Seq.	Data	Data Level	Record Number	Note
	FOB Point, Freight Terms, Shipment Priority, Shipping Method	HEADER	1090	
	Shipping & Packing Instructions	HEADER	1100	
	Header Request Date, Demand Class	HEADER	1110	
	Header Flexfields	HEADER	1200-1230	Flexfields
	PO Header Flexfields	HEADER	1240-1270	Flexfields
	Global Header Flexfields	HEADER	1280-1320	Flexfields
	Customer Name & Number	HEADER	1400	Flexfields
	Sold-to Address	HEADER	1410	
	Sold-to Contact	HEADER	1420	
	Ship-to Address (Header Level)	HEADER	1500	
	Ship-to Contact (Header Level)	HEADER	1510	
	Invoice Address	HEADER	1520	
	Invoice Contact	HEADER	1530	
	Ordered By Name	HEADER	1540	
	Ship-from Address (Header Level)	HEADER	1600	
	Extension Tables: Purchase Order Header Data	HEADER	(custom)	(custom)
	Credits Data	CREDIT	1700	
	Credit Flexfields	CREDIT	1710-1740	Flexfields
	Price Adjustments Data	PRICE ADJUSTMENT	1800	
	Discount/List Names (for price adjustments)	PRICE ADJUSTMENT	1810	
	Price Adjustment Flexfields	PRICE ADJUSTMENT	1820-1850	Flexfields
	Reservations Data	RESERVATION	1900	
	Reservations Flexfields	RESERVATION	1910-1940	Flexfields
	Action Data	ACTION	1950	

Table 6–40 Record Summary

Seq.	Data	Data Level	Record Number	Note
	Action Flexfields	ACTION	1960-1990	Flexfields
	Item (Original System Reference, Configuration)	LINE	2000	
	Inventory Item Segments	LINE DETAIL	2010-2020	Flexfields
	Item Detail (Items, Price)	LINE DETAIL	3000	
	Change Data	LINE DETAIL	3010	
	Change Comments	LINE DETAIL	3012-3018	
	Payment Type/Code, Tax Exempt	LINE DETAIL	3020	
	Project, Task, Contract Numbers	LINE DETAIL	3030	
	Service Comments	LINE DETAIL	3050	
	FOB Point, Freight Terms, Shipment Priority/Method, Freight Carrier	LINE DETAIL	3080	
	Demand Bucket	LINE DETAIL	3090	
	Shipment quantities, dates	LINE DETAIL	4000	
	Shipping & Packing Instructions	LINE DETAIL	4010	
	Item Flexfields	LINE DETAIL	4200-4230	Flexfields
	Global Header Flexfields	LINE DETAIL	4300-4340	Flexfields
	Pricing Flexfields	LINE DETAIL	4400-442	Flexfields
	Industry Header Flexfields	LINE DETAIL	4500-4560	Flexfields
	Service Flexfields	LINE DETAIL	4600-463	Flexfields
	Return Flexfields	LINE DETAIL	4700-473	Flexfields
	Ship-to Address (Line Level)	LINE DETAIL	4800	
	Ship-to Contact (Line Level)	LINE DETAIL	4810	
	Deliver to Contacts	LINE DETAIL	4820	
	Trader Partner Flexfields	LINE DETAIL	4830-4860	Flexfields
	Ship-from Address (Line Level)	LINE DETAIL	4870	

Table 6–40 Record Summary

Seq.	Data	Data Level	Record Number	Note
	Credits Data	CREDIT	5000	
	Credit Flexfields	CREDIT	5100-5130	Flexfields
	Price Adjustments Data	PRICE ADJUSTMENT	6000	
	Discount/List Names (for price adjustments)	PRICE ADJUSTMENT	6010	
	Change Reason Text	PRICE ADJUSTMENT	6020	
	Price Adjustment Flexfields	PRICE ADJUSTMENT	6100-6130	Flexfields
	Reservations Data	RESERVATION	7000	
	Reservations Flexfields	RESERVATION	7100-7130	Flexfields
	Action Data	ACTION	8000	
	Action Flexfields	ACTION	8100-8130	Flexfields
	Lot Serial Data	LOT SERIAL	9000	
	Serial Lot Flexfields	LOT SERIAL	9100-9130	Flexfields

Inbound Purchase Order Change (POCI/860/ORDCHG)

Table 6–41 Transaction specific data in the Common Key positions 1-100:

Position	CODE	CONTENT
1-25	TP_CD	Trading Partner Code as defined in the EDI Translator
26-47	PO	Purchase order number
48-69	ITEM	Purchase order line number
70-91	(blank)	(Not needed)
92-95	(varies)	Record Number
96-97	(varies)	Record Layout
98-100	(varies)	Record Layout Qualifier

Table 6–42 Transaction specific data in the Common Key positions 1-100 per record:

Seq.	Data	Trading Partner	Ref 1	Ref 2	Ref 3	Record Number	Record Layout	Record Layout
								Qualifier
	Length	25	22	22	22	4	2	3
	Position	1-25	26-47	48-69	70-91	92-95	96-97	98-100
	Control Record	TP_CD	PO			0010	CT	CTL
	Basic Purchase Order Header	TP_CD	PO			1000	PO	PO1
	Change Data	TP_CD	PO			1010	PO	CHG
	Change Comment 1	TP_CD	PO			1020	PO	CC1
	Change Comment 2	TP_CD	PO			1030	PO	CC2
	Change Comment 3	TP_CD	PO			1040	PO	CC3
	Change Comment 4	TP_CD	PO			1050	PO	CC4
	Currency Code & Conversion Rates	TP_CD	PO			1060	PY	CUR
	Payment Amount, Credit Card, Payment Type/Code, Tax Exempt	TP_CD	PO			1070	PY	TAX
	FOB Point, Freight Terms, Shipment Priority, Shipping Method	TP_CD	PO			1090	FB	SHP
	Shipping & Packing Instructions	TP_CD	PO			1100	NT	SHP
	Demand Class	TP_CD	PO			1110	MS	MIS
	Header Flexfields 1-4	TP_CD	PO			1200	A1	HD1
	Header Flexfields 5-9	TP_CD	PO			1210	A2	HD2
	Header Flexfields 10-14	TP_CD	PO			1220	A2	HD3
	Header Flexfield 15	TP_CD	PO			1230	A2	HD4
	Purchase Order Flexfields 1-4	TP_CD	PO			1240	A1	PO1
	Purchase Order Flexfields 5-9	TP_CD	PO			1250	A2	PO2

Table 6–42 Transaction specific data in the Common Key positions 1-100 per record:

Seq.	Data	Trading Partner	Ref 1	Ref 2	Ref 3	Record Number	Record Layout	Record Layout Qualifier
	Purchase Order Flexfields 1-14	TP_CD	PO			1260	A2	PO3
	Purchase Order Flexfield 15	TP_CD	PO			1270	A2	PO4
	Global Flexfields 1-4	TP_CD	PO			1280	A1	GL1
	Global Flexfields 5-9	TP_CD	PO			1290	A2	GL2
	Global Flexfields 10-14	TP_CD	PO			1300	A2	GL3
	Global Flexfields 15-19	TP_CD	PO			1310	A2	GL4
	Global Flexfield 20	TP_CD	PO			1320	A2	GL5
	Customer Name & Number	TP_CD	PO			1400	CU	CUS
	Sold-to Address (header level)	TP_CD	PO			1410	AD	SLD
	Sold-to Contact (header level)	TP_CD	PO			1420	CN	SLD
	Ship-to Address (header level)	TP_CD	PO			1500	AD	ST1
	Ship-to Contact (header level)	TP_CD	PO			1510	CN	ST1
	Invoice Address (header level)	TP_CD	PO			1520	AD	BT1
	Invoice Contact (header level)	TP_CD	PO			1530	CN	BT1
	Ordered By Name	TP_CD	PO			1540	CN	OBY
	Ship-from Address	TP_CD	PO			1600	AD	SFM
	Credits Data	TP_CD	PO			1700	CR	CRD
	Credit Flexfields 1-4	TP_CD	PO			1710	A1	CR1
	Credit Flexfields 5-9	TP_CD	PO			1720	A2	CR2
	Credit Flexfields 10-14	TP_CD	PO			1730	A2	CR3

Table 6–42 Transaction specific data in the Common Key positions 1-100 per record:

Seq.	Data	Trading Partner	Ref 1	Ref 2	Ref 3	Record Number	Record Layout	Record Layout
								Qualifier
	Credit Flexfield 15	TP_CD	PO			1740	A2	CR4
	Price Adjustments Data	TP_CD	PO			1800	PR	ADJ
	Discount/List Names	TP_CD	PO			1810	PR	AD1
	Price Adjustment Flexfields 1-4	TP_CD	PO			1820	A1	AD1
	Price Adjustment Flexfields 5-9	TP_CD	PO			1830	A2	AD2
	Price Adjustment Flexfields 10-14	TP_CD	PO			1840	A2	AD3
	Price Adjustment Flexfield 15	TP_CD	PO			1850	A2	AD4
	Reservations Data	TP_CD	PO			1900	RS	RES
	Reservation Flexfields 1-4	TP_CD	PO			1910	A1	RS1
	Reservation Flexfields 5-9	TP_CD	PO			1920	A2	RS2
	Reservation Flexfields 10-14	TP_CD	PO			1930	A2	RS3
	Reservation Flexfield 15	TP_CD	PO			1940	A2	RS4
	Action Data	TP_CD	PO			1950	AC	ACT
	Actions Flexfields 1-4	TP_CD	PO			1960	A1	AC1
	Actions Flexfields 5-9	TP_CD	PO			1970	A2	AC2
	Actions Flexfields 10-14	TP_CD	PO			1980	A2	AC3
	Actions Flexfield 15	TP_CD	PO			1990	A2	AC4
	Item Data (Original System Reference, Configuration Data)	TP_CD	PO	LINE		2000	IT	ITM
	Inventory Item Segments 1-10	TP_CD	PO	LINE		2010	IT	IS1

Table 6–42 Transaction specific data in the Common Key positions 1-100 per record:

Seq.	Data	Trading Partner	Ref 1	Ref 2	Ref 3	Record Number	Record Layout	Record Layout Qualifier
	Inventory Item Segments 11-20	TP_CD	PO	LINE		2020	IT	IS2
	Operation Code, Price Quantities, List Price	TP_CD	PO	LINE		3000	LN	LN1
	Change Data	TP_CD	PO	LINE		3010	LN	CHG
	Change Comment 1	TP_CD	PO	LINE		3012	LN	CM1
	Change Comment 2	TP_CD	PO	LINE		3014	LN	CM2
	Change Comment 3	TP_CD	PO	LINE		3016	LN	CM3
	Change Comment 4	TP_CD	PO	LINE		3018	LN	CM4
	Customer Payment Terms, Tax Exempt	TP_CD	PO	LINE		3020	PY	TAX
	Project, Task, Contract Numbers	TP_CD	PO	LINE		3030	PJ	SRV
	Service Transaction Comments	TP_CD	PO	LINE		3050	SV	CM1
	FOB Point, Freight Terms, Shipment Priority/Method, Freight Carrier	TP_CD	PO	LINE		3080	FB	SHP
	Demand Bucket, Class, Stream	TP_CD	PO	LINE		3090	DM	DMD
	Shipment Data (UOM, Ship/Promise Dates)	TP_CD	PO	LINE		4000	DT	QTY
	Shipping & Packing Instructions	TP_CD	PO	LINE		4010	NT	NTE
	Item Flexfields 1-4	TP_CD	PO	LINE		4200	A1	IT1
	Item Flexfields 5-9	TP_CD	PO	LINE		4210	A2	IT2
	Item Flexfields 10-14	TP_CD	PO	LINE		4220	A2	IT3
	Item Flexfield 15	TP_CD	PO	LINE		4230	A2	IT4

Table 6–42 Transaction specific data in the Common Key positions 1-100 per record:

Seq.	Data	Trading Partner	Ref 1	Ref 2	Ref 3	Record Number	Record Layout	Record Layout
								Qualifier
	Global Header Flexfields 1-4	TP_CD	PO	LINE		4300	A1	GL1
	Global Header Flexfields 5-9	TP_CD	PO	LINE		4310	A2	GL2
	Global Header Flexfields 10-14	TP_CD	PO	LINE		4320	A2	GL3
	Global Header Flexfields 15-19	TP_CD	PO	LINE		4330	A2	GL4
	Global Header Flexfield 20	TP_CD	PO	LINE		4340	A2	GL5
	Pricing Flexfields 1-4	TP_CD	PO	LINE		4400	A1	PC1
	Pricing Flexfields 5-9	TP_CD	PO	LINE		4410	A2	PC2
	Pricing Flexfield 10	TP_CD	PO	LINE		4420	A2	PC3
	Industry Header Flexfields 1-4	TP_CD	PO	LINE		4500	A1	IN1
	Industry Header Flexfields 5-9	TP_CD	PO	LINE		4510	A2	IN2
	Industry Header Flexfields 10-14	TP_CD	PO	LINE		4520	A2	IN3
	Industry Header Flexfields 15-19	TP_CD	PO	LINE		4540	A2	IN4
	Industry Header Flexfields 20-24	TP_CD	PO	LINE		4540	A2	IN5
	Industry Header Flexfields 25-29	TP_CD	PO	LINE		4550	A2	IN6
	Industry Header Flexfield 30	TP_CD	PO	LINE		4560	A2	IN7
	Service Flexfields 1-4	TP_CD	PO	LINE		4600	A1	SV1
	Service Flexfields 5-9	TP_CD	PO	LINE		4610	A2	SV2
	Service Flexfields 10-14	TP_CD	PO	LINE		4620	A2	SV3

Table 6–42 Transaction specific data in the Common Key positions 1-100 per record:

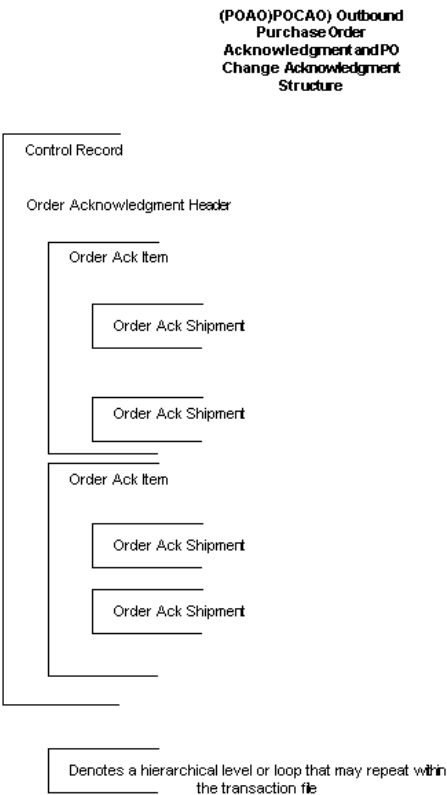
Seq.	Data	Trading Partner	Ref 1	Ref 2	Ref 3	Record Number	Record Layout	Record
								Layout Qualifier
	Service Flexfield 15	TP_CD	PO	LINE		4630	A2	SV4
	Return Flexfields 1-4	TP_CD	PO	LINE		4700	A1	RT1
	Return Flexfields 5-9	TP_CD	PO	LINE		4710	A2	RT2
	Return Flexfields 10-14	TP_CD	PO	LINE		4720	A2	RT3
	Return Flexfield 15	TP_CD	PO	LINE		4730	A2	RT4
	Ship-to Address (Item Level)	TP_CD	PO	LINE		4800	AD	ST1
	Ship-to Contact (Item Level)	TP_CD	PO	LINE		4810	CN	ST1
	Invoice to Contacts	TP_CD	PO	LINE		4820	CN	IND
	Trading Partner Flexfields 1-4	TP_CD	PO	LINE		4830	A1	TP1
	Trading Partner Flexfields 5-9	TP_CD	PO	LINE		4840	A2	TP2
	Trading Partner Flexfields 10-14	TP_CD	PO	LINE		4850	A2	TP3
	Trading Partner Flexfield 15	TP_CD	PO	LINE		4860	A2	TP4
	Ship-from Address (Item Level)	TP_CD	PO	LINE		4870	AD	SFR
	Credits Data	TP_CD	PO	LINE		5000	CR	CRD
	Credit Flexfields 1-4	TP_CD	PO	LINE		5100	A1	CR1
	Credit Flexfields 5-9	TP_CD	PO	LINE		5110	A2	CR2
	Credit Flexfields 10-14	TP_CD	PO	LINE		5120	A2	CR3
	Credit Flexfield 15	TP_CD	PO	LINE		5130	A2	CR4
	Price Adjustments Data	TP_CD	PO	LINE		6000	PR	ADJ
	Discount/List Names	TP_CD	PO	LINE		6010	PR	AD1
	Change Reason Text 1	TP_CD	PO	LINE		6020	PR	CM1

Table 6–42 Transaction specific data in the Common Key positions 1-100 per record:

Seq.	Data	Trading Partner	Ref 1	Ref 2	Ref 3	Record Number	Record Layout	Record Layout
								Qualifier
	Change Reason Text 2	TP_CD	PO	LINE		6030	PR	CM2
	Change Reason Text 3	TP_CD	PO	LINE		6040	PR	CM3
	Change Reason Text 4	TP_CD	PO	LINE		6050	PR	CM4
	Price Adjustment Flexfields 1-4	TP_CD	PO	LINE		6110	A1	AD1
	Price Adjustment Flexfields 5-9	TP_CD	PO	LINE		6110	A2	AD2
	Price Adjustment Flexfields 10-14	TP_CD	PO	LINE		6120	A2	AD3
	Price Adjustment Flexfield 15	TP_CD	PO	LINE		6130	A2	AD4
	Reservations Data	TP_CD	PO	LINE		7000	RS	RES
	Reservation Flexfields 1-4	TP_CD	PO	LINE		7100	A1	RS1
	Reservation Flexfields 5-9	TP_CD	PO	LINE		7110	A2	RS2
	Reservation Flexfields 10-14	TP_CD	PO	LINE		7120	A2	RS3
	Reservation Flexfield 15	TP_CD	PO	LINE		7130	A2	RS4
	Action Data	TP_CD	PO	LINE		8000	AC	ACT
	Actions Flexfields 1-4	TP_CD	PO	LINE		8100	A1	AC1
	Actions Flexfields 5-9	TP_CD	PO	LINE		8110	A2	AC2
	Actions Flexfields 10-14	TP_CD	PO	LINE		8120	A2	AC3
	Actions Flexfield 15	TP_CD	PO	LINE		8130	A2	AC4
	Lot Serials Data	TP_CD	PO	LINE		9000	LT	SER
	Serial Lot Flexfields 1-4	TP_CD	PO	LINE		9100	A1	LT1
	Serial Lot Flexfields 5-9	TP_CD	PO	LINE		9110	A2	LT2
	Serial Lot Flexfields 10-14	TP_CD	PO	LINE		9120	A2	LT3
	Serial Lot Flexfield 15	TP_CD	PO	LINE		9130	A2	LT4

Outbound Purchase Order Acknowledgment (POAO/855/ORDRSP)(POCAO/865/ORDRSP)

Figure 6–3 POAO POCAO Outbound Purchase Order Acknowledgment and PO Change Acknowledgment Structure



Outbound Purchase Order Acknowledgement (POAO/855/ORDRSP) (POCAO/865/ORDRSP)

Table 6–43 Record occurrences within the transaction:

Records	Content	Occurrences
0010	e-Commerce Gateway Control Record	Only one record occurrence per transaction
1000-1540	Acknowledgment Header Records	Only one record occurrence per transaction
2000-4030	Acknowledgment Item Records	One set of records per item within the Acknowledgment header

Table 6–44 Record Summary:

Seq.	Data	Data Level	Record Number	Note
	Control Record	HEADER	0010	
	Basic Purchase Order Header	HEADER	1000	
	Currency Code & Conversion Rates	HEADER	1060	
	Tax Exempt Information	HEADER	1080	
	Customer Payment Terms, List Price	HEADER	1090	
	FOB Point, Freight Terms	HEADER	1100	
	Shipment Priority, Freight Carrier	HEADER	1110	
	Shipping & Packing Instructions	HEADER	1120	
	Request Dates, Schedule Limits	HEADER	1130	
	Header Flexfields	HEADER	1200-1230	Flexfields
	PO Flexfields	HEADER	1240-1270	Flexfields
	Global Flexfields	HEADER	1280-1320	Flexfields
	Customer Name & Number	HEADER	1400	
	Sold_to Organization	HEADER	1410	
	Sold_to Contact Information	HEADER	1420	
	Ship_to Address Information	HEADER	1500	
	Ship_to Contact Information	HEADER	1510	

Table 6–44 Record Summary:

Seq.	Data	Data Level	Record Number	Note
	Invoice Address Information	HEADER	1520	
	Invoice Contact Information	HEADER	1530	
	Ordered By Name	HEADER	1540	
	Original System References	LINE	2000	
	Item Type Codes	LINE	2010	
	Item Segments 1-10	LINE	2020	
	Item Segments 11-20	LINE	2030	
	Demand Class, Bucket Type	LINE	2040	
	Customer Payment Terms, Tax Exempt Information	LINE	2050	
	Project, Task, Contract Numbers	LINE	2060	
	FOB Point, Freight Terms	LINE	2070	
	Shipment Priority/Method, Freight Carrier	LINE	2080	
	Sales representative, Reference Ids	LINE	2090	
	Order/Ship Quantity UOM	LINE	3000	
	Item Flexfields	LINE	3100-3130	Flexfields
	Global Flexfields	LINE	3200-3240	Flexfields
	Pricing Flexfields	LINE	3300-3320	Flexfields
	Industry Flexfields	LINE	3400-3460	Flexfields
	Return Flexfields	LINE	3470-3500	Flexfields
	Ship_to Address (Item Level)	LINE	4000	
	Ship_to Contacts (Item Level)	LINE	4010	
	Ship_from Address (Item Level)	LINE	4020	
	Deliver_to Contacts (Item Level)	LINE	4030	

Outbound Purchase Order Acknowledgement (POAO/855/ORDRSP)
(POCAO/865/ORDRSP)

Table 6–45 Transaction specific Data in the Common Key positions 1-100:

Position	CODE	CONTENT
1-25	TP_CD	Trading Partner Code as defined in the EDI Translator
26-47	PO	Purchase Order Number
48-69	ITEM	Purchase Order Line Number
70-91	(blank)	(Not needed)
92-95	(varies)	Record Number
96-97	(varies)	Record Layout
98-100	(varies)	Record Layout Qualifier

Table 6–46 Transaction specific Data in the Common Key positions 1-100 per record:

Seq.	Data							Record
		Trading Partner	Ref 1	Ref 2	Ref 3	Record Number	Record Layout	Layout Qualifier
	Length	25	22	22	22	4	2	3
	Position	1-25	26-47	48-69	70-91	92-95	96-97	98-100
	Control Record	TP_CD	PO			0010	CT	CTL
	Basic Purchase Order Header	TP_CD	PO			1000	PO	PO1
	Currency Code & Conversion Rates	TP_CD	PO			1060	PO	CUR
	Tax Exempt Information	TP_CD	PO			1080	TX	TAX
	Customer Payment Terms, List Price	TP_CD	PO			1090	TR	TRM
	FOB Point, Freight Terms	TP_CD	PO			1100	FB	FOB
	Shipment Priority, Freight Carrier	TP_CD	PO			1110	SH	SHP
	Shipping & Packing Instructions	TP_CD	PO			1120	IN	NTE
	Request Dates, Schedule Limits	TP_CD	PO			1130	MS	HDR

Table 6–46 Transaction specific Data in the Common Key positions 1-100 per record:

Seq.	Data	Trading Partner	Ref 1	Ref 2	Ref 3	Record Number	Record Layout	Record
								Layout Qualifier
	Header Flexfields 1-4	TP_CD	PO			1200	A1	HD1
	Header Flexfields 5-9	TP_CD	PO			1210	A2	HD2
	Header Flexfields 10-14	TP_CD	PO			1220	A2	HD3
	Header Flexfields 15	TP_CD	PO			1230	A2	HD4
	Purchase Order Flexfields 1-4	TP_CD	PO			1240	A1	PO1
	Purchase Order Flexfields 5-9	TP_CD	PO			1250	A2	PO2
	Purchase Order Flexfields 1-14	TP_CD	PO			1260	A2	PO3
	Purchase Order Flexfields 15	TP_CD	PO			1270	A2	PO4
	Global Flexfields 1-4	TP_CD	PO			1280	A1	GL1
	Global Flexfields 5-9	TP_CD	PO			1290	A2	GL2
	Global Flexfields 10-14	TP_CD	PO			1300	A2	GL3
	Global Flexfields 15-19	TP_CD	PO			1310	A2	GL4
	Global Flexfields 20	TP_CD	PO			1320	A2	GL5
	Customer Name & Number	TP_CD	PO			1400	CU	CUS
	Sold_to Address Information	TP_CD	PO			1410	AD	SLD
	Sold_to Contact Information	TP_CD	PO			1420	CN	SLD
	Ship_to Address Information	TP_CD	PO			1500	AD	ST1
	Ship_to Contact Information	TP_CD	PO			1510	CN	ST1
	Invoice Address Information	TP_CD	PO			1520	AD	BT1
	Invoice Contact Information	TP_CD	PO			1530	CN	BT1
	Ordered By Name	TP_CD	PO			1540	CN	OBY
	Original System References	TP_CD	PO	LINE		2000	IT	IT1
	Item Type Codes	TP_CD	PO	LINE		2010	IT	IT2
	Item Segments 1-10	TP_CD	PO	LINE		2020	IT	SG1

Table 6–46 Transaction specific Data in the Common Key positions 1-100 per record:

Seq.	Data	Trading Partner	Ref 1	Ref 2	Ref 3	Record Number	Record Layout	Record
								Layout Qualifier
	Item Segments 11-20	TP_CD	PO	LINE		2030	IT	SG2
	Demand Class, Bucket Type	TP_CD	PO	LINE		2040	DM	DMD
	Customer Payment Terms, Tax Exempt Information	TP_CD	PO	LINE		2050	IT	PTT
	Project, Task, Contract Numbers	TP_CD	PO	LINE		2060	PD	PRJ
	FOB Point, Freight Terms	TP_CD	PO	LINE		2070	FB	FOB
	Shipment Priority/Method, Freight Carrier	TP_CD	PO	LINE		2080	SH	SHP
	Sales representative, Reference Ids	TP_CD	PO	LINE		2090	MS	LNE
	UOM, Ship/Promise Dates, Lead Time	TP_CD	PO	LINE		3000	DT	DTL
	Item Flexfields 1-4	TP_CD	PO	LINE		3100	A1	IT1
	Item Flexfields 5-9	TP_CD	PO	LINE		3110	A2	IT2
	Item Flexfields 10-14	TP_CD	PO	LINE		3120	A2	IT3
	Item Flexfields 15	TP_CD	PO	LINE		3130	A2	IT4
	Global Flexfields 1-4	TP_CD	PO	LINE		3200	A1	GL1
	Global Flexfields 5-9	TP_CD	PO	LINE		3210	A2	GL2
	Global Flexfields 10-14	TP_CD	PO	LINE		3220	A2	GL3
	Global Flexfields 15-19	TP_CD	PO	LINE		3230	A2	GL4
	Global Flexfields 20	TP_CD	PO	LINE		3240	A2	GL5
	Pricing Flexfields 1-4	TP_CD	PO	LINE		3300	A1	PC1
	Pricing Flexfields 5-9	TP_CD	PO	LINE		3310	A2	PC2
	Pricing Flexfield 10	TP_CD	PO	LINE		3320	A2	PC3
	Industry Flexfields 1-4	TP_CD	PO	LINE		3400	A1	IN1
	Industry Flexfields 5-9	TP_CD	PO	LINE		3410	A2	IN2

Table 6–46 Transaction specific Data in the Common Key positions 1-100 per record:

Seq.	Data	Trading Partner	Ref 1	Ref 2	Ref 3	Record Number	Record Layout	Record
								Layout Qualifier
	Industry Flexfields 10-14	TP_CD	PO	LINE		3420	A2	IN3
	Industry Flexfields 15-19	TP_CD	PO	LINE		3430	A2	IN4
	Industry Flexfields 20-24	TP_CD	PO	LINE		3440	A2	IN5
	Industry Flexfields 25-29	TP_CD	PO	LINE		3450	A2	IN6
	Industry Flexfields 30	TP_CD	PO	LINE		3460	A2	IN7
	Return Flexfields 1-4	TP_CD	PO	LINE		3470	A1	RT1
	Return Flexfields 5-9	TP_CD	PO	LINE		3480	A2	RT2
	Return Flexfields 10-14	TP_CD	PO	LINE		3490	A2	RT3
	Return Flexfields 15	TP_CD	PO	LINE		3500	A2	RT4
	Ship-to Address (Item Level)	TP_CD	PO	LINE		4000	AD	ST1
	Ship-to Contact (Item Level)	TP_CD	PO	LINE		4010	CN	ST1
	Ship-from Address (Item Level)	TP_CD	PO	LINE		4020	AD	ST1
	Invoice Contact (Item Level)	TP_CD	PO	LINE		4030	CN	ST1

User Guide Update for Order Management Transactions

Purchase Order Inbound (POI) Program

Use this transaction to import customer purchase orders into your Oracle Order Management system. Using the Order Management Order Import Open Interface, you can import a high-volume of purchase orders, complete with all the standard online purchase order entry features, including customer item to supplier item cross referencing.

Prerequisite Setup in Oracle Order Management

The required Order Management setup steps are the same whether the order is entered on-line or through Order Import. The following is a sample list of the Order Management setup steps needed to process orders through the e-Commerce Gateway or entered on-line.

1. Define Customers and Customer Sites in Order Management/Oracle Receivables.
2. Define Order Types in Order Management.
3. Define items and appropriate customer items in Oracle Inventory

Prerequisites

- Create the inbound directory and update the INIT.ORA file. See: Defining Data File Directories, *Oracle e-Commerce Gateway Implementation Manual, Release 11i*.
- Define the ECE: Inbound file path profile option. See: e-Commerce Gateway Profile Options, *Oracle e-Commerce Gateway Implementation Manual, Release 11i*.
- Define trading partner relationships and enable EDI transactions for the trading partner. See: Defining Trading Partner Data in the *Oracle e-Commerce Gateway User Guide, Release 11i*.
- Define code conversions. See: Defining Code Conversion Categories, Assigning Categories, and Defining Code Conversion Values in the *Oracle e-Commerce Gateway User's Guide*.
- Customize interface data file layout, if necessary. See: Changing the Interface Data File Record Layout in the *Oracle e-Commerce Gateway User's Guide*.

Purchase Order Inbound Program

To run the Purchase Order Inbound program:

1. Navigate to the Import Program window.
2. Select Request to submit an individual request.
3. Select the Purchase Order Inbound request.
4. Open the Parameters window.
5. Enter the inbound file path or accept the default.
6. Enter the inbound data file or accept the default.
7. Enter the following parameters:

Debug Mode: Set the debug mode to report on debug information. Debug information is written to the concurrent manager log file:

0 = OFF: No debug information is generated.

1 = LOW: Summary debug information is written to the concurrent manager log file.

2 = MEDIUM: Medium level debug information is written to the concurrent manager log file.

3 = HIGH: High level debug information is written to the concurrent manager log file. (

8. In the Run Import field:

Select Yes to initiate the Order Management Order Import Open Interface Import program using the default parameter values.

Enter No to cancel the Order Management Order Import Open Interface.

9. The default Map Code for the POI transaction is ECE_POI_FF.
10. When finished, click OK in the Parameters window.
 - a. Enter the completion options.
 - b. Enter any schedule options to schedule the request.
 - c. Choose Submit and note the Request ID returned.

See Also

Oracle e-Commerce Gateway Implementation Manual, Release 11i.

Inbound Purchase Order, Oracle e-Commerce Gateway Implementation Manual, Release 11i Submitting a Request, *Oracle Applications User's Guide*

Viewing the Status of Concurrent Programs in the *Oracle e-Commerce Gateway User's Guide*

Purchase Order Change Inbound (POCI) Program

Use this transaction to import customer purchase order changes into your Oracle Order Management system. Using the Order Management Order Import Open Interface, you can import purchase orders with all standard online purchase order entry features, including the customer item to supplier item cross referencing.

Prerequisite Setup in Oracle Order Management

The required Order Management setup steps are the same whether the order is entered on-line or through Order Import. The following is a sample list of the Order Management setup steps to process orders through the e-Commerce Gateway or entered on-line.

1. Define Customers and Customer Sites in Order Management/Oracle Receivables.
2. Define Order Types in Order Management.
3. Define items and appropriate customer items in Oracle Inventory.

Prerequisites

- Create the inbound directory and update the INIT.ORA file. See: Defining Data File Directories, *Oracle e-Commerce Gateway Implementation Manual, Release 11i*.
- Define the ECE: Inbound file path profile option. See: e-Commerce Gateway Profile Options, *Oracle e-Commerce Gateway Implementation Manual, Release 11i*.
- Define trading partner relationships and enable EDI transactions for the trading partner. See: Defining Trading Partner Data in the *Oracle e-Commerce Gateway User Guide, Release 11i*.
- Define code conversions. See: Defining Code Conversion Categories, Assigning Categories, and Defining Code Conversion Values in the *Oracle e-Commerce Gateway User's Guide*.
- Customize interface data file layout, if necessary. See: Changing the Interface Data File Record Layout in the *Oracle e-Commerce Gateway User's Guide*.

Purchase Order Change Inbound

To run the Purchase Order Change Inbound program:

1. Navigate to the Import Program window.

2. Select Single Request to submit an individual request.
3. Select the IN: Purchase Order Changes request.
4. Open the Parameters window.
Enter the File Path or accept the default.
Enter the File Name or accept the default.
5. Enter the following parameters:
Debug Mode: Set the debug mode to report on debug information. Debug information is written to the concurrent manager log file:
0 = OFF: No debug information is generated.
1 = LOW: Summary debug information is written to the concurrent manager log file.
2 = MEDIUM: Medium level debug information is written to the concurrent manager log file.
3 = HIGH: High level debug information is written to the concurrent manager log file. (In the Run Import field:
Select Yes to initiate the Order Management Order Import Open Interface Import program using the default parameter values.
Enter No to cancel the Order Management Order Import Open Interface.
6. The default Map Code for the POI transaction is ECE_POI_FF.
7. When finished, click OK in the Parameters window.
Enter completion options.
Enter schedule options to schedule the request.
Choose Submit and note the Request ID returned.

See Also

Oracle e-Commerce Gateway Implementation Manual, Release 11i.

Inbound Purchase Order Changes, Oracle e-Commerce Gateway Implementation Manual, Release 11i Submitting a Request, *Oracle Applications User's Guide*

Viewing the Status of Concurrent Programs in the Oracle e-Commerce Gateway User's Guide

Purchase Order Acknowledgment Outbound (POAO) Program

Use this transaction to export customer purchase orders acknowledgment from your Oracle Order Management system.

Prerequisite Setup in Oracle Order Management

The required Order Management setup steps for the purchase order acknowledgments are the following:

1. Define Customers and Customer Sites in Order Management/Oracle Receivables.
2. Define Order Types in Order Management.
3. Define items and appropriate customer items in Oracle Inventory.

Prerequisites

- Create the Outbound directory and update the INIT.ORA file. See: Defining Data File Directories, *Oracle e-Commerce Gateway Implementation Guide, Release 11i*.
- Define the ECE: Outbound file path profile option. See: e-Commerce Gateway Profile Options, *Oracle e-Commerce Gateway Implementation Guide, Release 11i*.
- Define trading partner relationships and enable EDI transactions for the trading partner. See: Defining Trading Partner Data in the *Oracle e-Commerce Gateway Implementation Guide, Release 11i*.
- Define code conversions. See: Defining Code Conversion Categories, Assigning Categories, and Defining Code Conversion Values in the *Oracle e-Commerce Gateway User's Guide*.
- Customize interface data file layout, if necessary. See: Changing the Interface Data File Record Layout in the *Oracle e-Commerce Gateway User's Guide*.

Purchase Order Acknowledgement Outbound Program

To run the Purchase Order Acknowledgment Outbound program:

1. Navigate to the Extract Program window.
2. Select Request to submit an individual request.
3. Select the Out: Purchase Order Acknowledgments request.

4. Open the Parameters window.
Enter the Outbound file path or accept the default.
Enter the Outbound file name or accept the default.
5. Enter the following parameters, as appropriate:
Debug Mode: Set the debug mode to report on debug information. Debug information is written to the concurrent manager log file:
0 = OFF: No debug information is generated.
1 = LOW: Summary debug information is written to the concurrent manager log file.
2 = MEDIUM: Medium level debug information is written to the concurrent manager log file.
3 = HIGH: High level debug information is written to the concurrent manager log file.
Customer Number
Customer Name
Sales Order Reference From
Sales Order Reference To
Sales Order Number From
Sales Order Number To
Purchase Order Number From
Purchase Order Number To
Date From
Date To
6. When finished, click OK in the Parameters window.
7. Enter completion options.
8. Enter schedule options to schedule the request.
9. Choose Submit and note the Request ID returned.

See Also

Oracle e-Commerce Gateway Implementation Guide, Release 11i.

Outbound Purchase Order Acknowledgments, *Oracle e-Commerce Gateway Implementation Guide, Release 11i* Submitting a Request, *Oracle Applications User's Guide*

Viewing the Status of Concurrent Programs in the *Oracle e-Commerce Gateway User's Guide*

Purchase Order Change Acknowledgment Outbound (POCAO) Program

Use this transaction to export customer purchase order change acknowledgment from your Oracle Order Management system.

Prerequisite Setup in Oracle Order Management

The required Order Management setup steps for the purchase order change acknowledgments are the following:

1. Define Customers and Customer Sites in Order Management/Oracle Receivables.
2. Define Order Types in Order Management.
3. Define Items and appropriate customer items in Oracle Inventory.

Prerequisites

- Create the Outbound directory and update the INIT.ORA file. See: Defining Data File Directories, *Oracle e-Commerce Gateway Implementation Guide, Release 11i*.
- Define the ECE: Outbound file path profile option. See: e-Commerce Gateway Profile Options, *Oracle e-Commerce Gateway Implementation Guide, Release 11i*.
- Define trading partner relationships and enable EDI transactions for the trading partner. See: Defining Trading Partner Data in the *Oracle e-Commerce Gateway Implementation Guide, Release 11i*.
- Define code conversions. See: Defining Code Conversion Categories, Assigning Categories, and Defining Code Conversion Values in the *Oracle e-Commerce Gateway User's Guide*.
- Customize interface data file layout, if necessary. See: Changing the Interface Data File Record Layout in the *Oracle e-Commerce Gateway User's Guide*.

To run the Purchase Order Change Acknowledgment Outbound program:

1. Navigate to the Extract Program window.
2. Select Single Request to submit an individual request.
3. Select the Out: Purchase Order Change Acknowledgments request.
4. Open the Parameters window.

Enter the File Path or accept the default.

Enter the File Name or accept the default.

Enter the following parameters, as appropriate:

Debug Mode: Set the debug mode to report on debug information. Debug information is written to the concurrent manager log file:

0 = OFF: No debug information is generated.

1 = LOW: Summary debug information is written to the concurrent manager log file.

2 = MEDIUM: Medium level debug information is written to the concurrent manager log file.

3 = HIGH: High level debug information is written to the concurrent manager log file. (

Customer Number

Customer Name

Sales Order Reference From

Sales Order Reference To

Sales Order Number From

Sales Order Number To

Purchase Order Number From

Purchase Order Number To

Date From

Date To

5. When finished, click OK in the Parameters window.

6. Enter completion options.
7. Enter schedule options to schedule the request.
8. Choose Submit and note the Request ID returned.

See Also

Oracle e-Commerce Gateway Implementation Guide, Release 11i.

Outbound Purchase Order Change Acknowledgments, Oracle e-Commerce Gateway Implementation Guide, Release 11i Submitting a Request, Oracle Applications User's Guide

Viewing the Status of Concurrent Programs in the Oracle e-Commerce Gateway User's Guide

ReadMe for November, 2000 Patch

Purchase Order Inbound (POI) Patch Readme

README for the Inbound Purchase Order Update: (November 2000)

NOTE: This code is carried forward in patch 1621172 in March 2001.

The POI transaction record layout changed substantially from the POI transaction originally released with R11i.1. There was too much movement of data across records to list the full detail, so the record summaries in the detail document listed below may guide you.

The following changes were made:

1. The file will consist of fewer records at the header and line/shipment levels, since more data was placed on records. The records can have a maximum length of 1024 so data from a few previous records were combined to create fewer records.
 2. (Major Change) the supplier item and customer item (and other item data) are written on a single set of record 2000 series before its set of shipment records in the record 3000 series. Previously, all item data had to be reentered on a record with each set shipment data. These changes are illustrated in the detail document mentioned below. This was an important change so the file record hierarchy reflected the standard transaction/message segment hierarchy. The price remains at the shipment level, because the Oracle tables can accept different prices at the shipment level.
 3. The Credit, Price Adjustment, Reservation, and Actions tables were added to the header level.
 4. The Credit, Price Adjustment, Reservation, Lot/Serial, and Actions tables were already specified at the line/shipment level in the original R11i.1 POI transactions. They apply to the shipment level in the new POI.
- Remember, the Item in the Record 2000 series is just a second level representation in the file so you do you have to reenter the item with every shipment in the third level. The open interface table is still the line/shipment combination, so these five tables of data are associated with that same line/shipment combination above it in the file.

Major Data Map Changes Needed:

Major data map changes are required after this patch is installed. Major data map changes were already required from Release 10.7 and Release 11 to map the new Order Management data to standard transactions. If you have started on R11i.1 data maps, adjustments are also needed to accommodate the change of this patch.

Must Run Seed Data Reconciliation in the Gateway:

After this patch is installed, run the Oracle e-Commerce Gateway Seed Data Reconciliation program for the POI transaction. Enter No for the four questions on preserving the layout, process rules, column rules, and code conversion assignments. You do not want to preserve the old record layouts for the inbound purchase order and purchase order change transactions. (If you enter Yes to any of the four questions, it is unknown what becomes of the separation of the item data to the record 2000 series and the shipment data in the record 3000 series.)

Detail Document

Refer to the R11i.1 *Oracle e-Commerce Gateway Implementation Manual* update for Order Management Transactions in Oracle Support's Metalink web site under the e-Commerce Gateway for details.

Additional Information for the March 2001 Patch

(Not found in the README)

R11i Order Management Transaction

POI - Inbound Purchase Order Transaction

PATCH:

Patch 1621172

POI: Address Derivation.

Note: Other Order Management transactions may have transaction modules updated by this patch for some seed data.

IMPORTANT:

Perform the following steps:

1. Load the patch.
2. Run the **Seed Data Reconciliation Process** with the option to **NOT PRESERVE** the old record.

You need the new record layout for the inbound purchase order (POI) transaction to get the updated address records and new record 4000. There may be other record changes also.

If you accidentally preserved the old layout, run the patch again.

3. Run the Transaction Layout Report to get the new POI record layout.
4. Change the Transaction Map in your EDI Translator so it can produce the POI records defined by this patch.

Old transaction files will not process successfully after this patch.

ADDRESS DETAIL

All addresses in the inbound PO transaction detail should have their address site table IDS derived by the Address derivation in the Gateway by using either:

1. The EDI LOCATION CODE in the record position 20.

2. The full address if it is also on the record. There is an exception for the SOLD TO address (record 1400) that is described below.

Which address derivation method is used first is defined in the Transaction Profile set up where the address precedence is defined.

Table 6–47 Address Records in the Inbound Purchase Order (POI)

Record	Position	Address Type	Note
1400	20 for EDI Location Code	SOLD TO data	
1500	20 for EDI Location Code	SHIP TO address (header level)	
1520	20 for EDI Location Code	Invoice/bill to address (header level)	
1600	20 for EDI Location Code	Ship from address (header level)	This is your warehouse. It is not likely to be sent by Trading Partner
4800	20 for EDI Location Code	Ship to address (line level)	
4870	20 for EDI Location Code	Ship from address (line level)	This is your warehouse. It is not likely to be sent by Trading Partner

COMBINED RECORDS 1400 and 1410 (SOLD TO ADDRESS):

1. The SOLD TO data from record 1400 and 1410 were combined into a new record 1400.

Record 1410 no longer exists.

2. Record 1400 CUSTOMER_NUMBER (position 10) and CUSTOMER_NAME (position 30) are optional.

If the CUSTOMER ID was not successfully derived from the SOLD TO address derivation, then Order IMPORT could use the CUSTOMER_NUMBER or CUSTOMER_NAME (if it is sent on the file) in an attempt to find the CUSTOMER_ID during Order Import processing.

If the Gateway derives the address site table ID, it does not write the derived CUSTOMER_NAME and CUSTOMER_NUMBER to the OrderImport tables.

3. The data on record 1400 must successfully derive the CUSTOMER_ID given the SOLD TO data from one of the two methods mentioned above; otherwise the order cannot be loaded into the base order tables.

The methods are:

- From address derivation in the Gateway using the EDI Location code or the full address, or
- Customer lookup in Order Import using the CUSTOMER_NUMBER (position 10) or CUSTOMER_NAME (position 30).
- Hence, record 1400 is a REQUIRED RECORD. All other address records are optional including the ship to records that may be defaulted by Order Import.

RECORDS 4000: (Shipment level data)

This record was substantially modified to have mostly only the dates, quantities, and UOMs, plus a few requested data elements.

You may activate the other data that was on record 4000 if you need it. Scroll to the end of the transaction level that has the record 4000 data to see the other fields. Assign a record number, record position, and data length to the desired fields. They may be added to record 4000.

Electronic Messaging Technical Information

- [What is RosettaNet?](#) on page 7-2
- [Process_PO](#) on page 7-4
- [Acknowledge_PO](#) on page 7-34
- [Show_SalesOrder](#) on page 7-50
- [Change_salesorder](#) on page 7-68
- [Inbound Change PO Request](#) on page 7-83
- [Cancel Purchase Order](#) on page 7-117
- [Open Interface Tracking](#) on page 7-140

What is RosettaNet?

RosettaNet is a consortium of more than 400 of the world's leading Electronic Components (EC), Information Technology (IT), Semiconductor Manufacturing (SM) and Solution Provider (SP) companies. It is a self-funded, non-profit organization dedicated to creating, implementing and promoting open e-business standards. These standards form a common e-business language, aligning processes between trading partners on a global basis.

RosettaNet standards offer a robust non proprietary solution, encompassing data dictionaries, implementation framework, and business message schemas and process specifications, for e-business standardization.

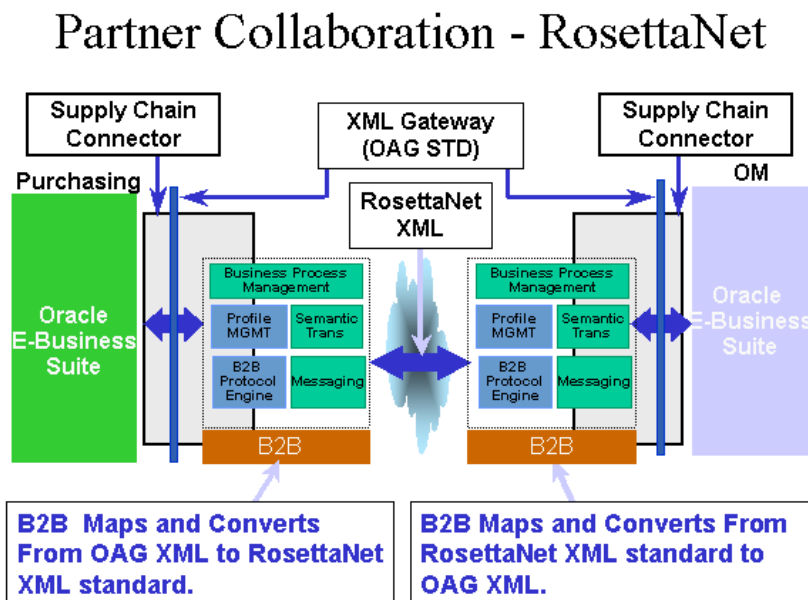
RosettaNet Partner Interface Processes "PIPs" are specialized system-to-system XML-based dialogs that define business processes between trading partners. The message dialog PIPs apply to the following core processes: Administration; Partner, Product and Service Review; Product Introduction; Order Management; Inventory Management; Marketing Information Management; Service and Support; and Manufacturing.

Oracle Order Management supports OAG transactions which correspond to Rosettanet transactions. The following table shows the relation between Rosettanet transactions and OAG transactions.

Table 7–1 Rosettanet and OAG Transactions

Rosettanet	OAG	Description
PIP 3A4 -Purchase Order Request	Process_PO	New purchase order from the buyer to the seller
PIP 3A4- Purchase Order confirmation	Acknowledge_PO	Acknowledgement for the new Purchase order from Seller to the buyer
PIP 3A9 - Request Purchase Order Cancellation	Cancel_PO	Request for canceling a PO from buyer to the seller
PIP 3A6 - Distribute Order Status	Show_Salesorder	Send Sales order status to the buyer from seller.

Following diagram describes how the Rosettanet transaction can be supported by Order Management in conjunction with B2B server.

Figure 7-1 Partner Collaboration - RosettaNet

Process_PO

Process_PO is used between seller and buyer to exchange information including pricing, availability and status of the order. Process_PO consists of the Inbound PO request and the outbound Acknowledge PO message, which reports the status of the order to the buyer. Typical users of this collaboration would be the high order volume industries such as High Tech and electronics.

Major Features

Order Management 11i.9 supports the Process_PO transaction for Open Applications Group (OAG) XML standard. The Process_PO transaction is between the buyer and seller. The buyer can send a new purchase order using this transaction. The following are the major features of this transaction within Order Management:

Consume the Process_PO Document

Order management consumes the Process_PO (OAG) document from the buyer and creates an order in Order Management. The internal control number (which is unique for each message) of the incoming message is stored for later use to retrieve the sender information of the original message. This information is mapped on the outbound Acknowledgement message. Order management consumes the Process_PO (OAG) document from the buyer and creates an order in Order Management.

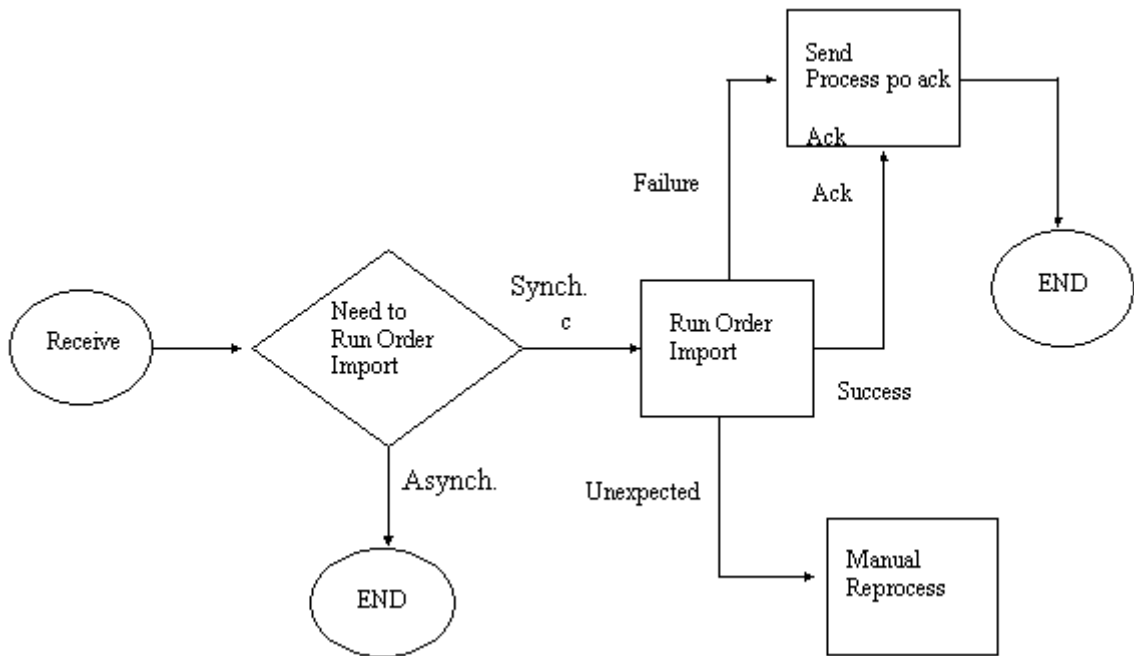
Setup the Order Import Process in Synchronous or Asynchronous Mode

You can setup the profile option to initiate Order Import for every order or wait and process multiple orders via concurrent processing. The profile option is called OM: Run Order Import for XML, that can be set up on the user or site level. The possible values for this profile option will be Synchronous or Asynchronous. The default is Synchronous.

On successful completion of Order Import (run either in synchronous or asynchronous mode), the sales order will be created in Order Management system.

Error Handling

The error handling process is below:

Figure 7–2 Error Handling Process

There are three types of outcomes from running Order Import process:

1. **Successful processing** - The confirmation API is called to send the Acknowledgment to the sender with the Accept status. If the user chooses to book the order before sending out acknowledgment message out, the Accept with Changes status may be sent if there are changes that happened at the booking.
2. **Failed processing** - This status is when there are validation errors or data setup errors. The acknowledgment is sent to the sender with the Reject status. The Order Import Concurrent program will complete normally.
3. **Unexpected Errors** - There can be some unexpected errors due to resource constraints that will require reprocessing of Order Import upon fixing those issues. This process is manual. The Order Import Concurrent program will not complete normally.

Figure 7–3 Corrections Form

Order S	Orig Sys Docu	Change	Request ID	Operation Code	Sold To Org ID	Sold To Org
20	syp1008-11			INSERT		
20	ARDOCREF771			INSERT	1006	
20	syp1008-9		2198646			
20	syp1008-7		2198608			
20	syp1008-5		2198424			
20	ar-16		2193104		1006	
20	syp1008-4		2198343			
20	syp1007-5		2198222		1006	
20	syp1008-4		2198331			
20	syp1008-3		2198330			

Buttons: Lines, Discounts, Validate, Import, Actions, Sales Credits, Add Customers, Errors, Pricing Attributes

CONFIRM BOD

Upon receipt of an inbound XML document such as a Process_PO or Cancel_PO, Order Management has implemented the outbound OAG Confirm BOD message to signal the successful receipt of data.

Major features of this transaction are as follows:

- This message contains information about the inbound XML message received, as well as the status and description of the inbound message received.
- Unlike the Acknowledge PO message, the Confirm BOD is sent before Order Import is run – it is purely a confirmation.

Generate Confirm BOD (CBOD)

Upon receipt of the inbound Process_PO, Order Management can send out the Confirm BOD to the buyer indicating that the data has reached the seller. The confirm bod will not be sent out every time. It will be sent depending on the value of the <CONFIRMATION> tag in the inbound document. The CBOD will not be

sent if the value of that tag is 0 or 1. It will be sent if the value of that tag is 2. Confirm BOD should be generated on the successful insertion of the data in open interface tables. It is part of the workflow that begins after XML gateway has finished the insertion of the data successfully. At this point only the Successful status will be sent back on the CBOD as its unlikely that failure will occur after Order Management controls the process. Any failure before this point will be trapped by the XML gateway and a notification is sent to the buyer. Confirm BOD is only generated if the inbound document has required confirmation.

Message Details

Message Map 'ONT_3A4R_OAG72_IN.xgm'

The message map 'ONT_3A4R_OAG72_IN' is created using the Oracle XML Gateway Message Designer tool.

Table 7-2 Message Map Details

Name	Description
Message Map Name:	ONT_3A4R_OAG72_IN
Direction:	Inbound
(Internal) Transaction Type:	ONT
(Internal) Transaction Subtype:	POI
External Transaction Type:	PO
External Transaction Subtype:	PROCESS
DTD Directory:	xml/oag72
Map Directory:	patch/115/xml/US
Message Maps XGM File Name:	ONT_3A4R_OAG72_IN.xgm
Standard:	OAG
Release:	7.2
Format:	DTD
DTD Name:	PROCESS_PO_007

Table 7–3 Columns Enabled for Code Conversion

Defaulted Column	Default Value, and Condition (if any)
<UOM> (in schedule level <QUANTITY qualifier="ORDERED">)	Code Category – UOM, mapped to OE_LINES_ INTERFACE.ORDER_QUANTITY_UOM

Key: H => OE_HEADER_INTERFACE, L => OE_LINES_INTERFACE

Table 7-4 Defaulted Columns

Defaulted Columns	Default Value, and Condition (if any)
H.ORDER_SOURCE_ID	20
H.ORDER_TYPE	'Standard'
H.XML_TRANSACTION_TYPE_CODE	'POI'
H.HEADER DEFAULT	1-4
L.LINE DEFAULT	1-4
L.ORDER_SOURCE_ID	20
L.ORIG_SYS_DOCUMENT_REF	H.ORIG_SYS_DOCUMENT_REF
L.XML_TRANSACTION_TYPE_CODE	'POI'

Key: H => OE_HEADER_INTERFACE, L => OE_LINES_INTERFACE

Table 7-5 Derived Columns

Derived Columns	Source of Data, and Condition (if any)
H.CREATED_BY	FND_GLOBAL.USER_ID
H.CREATION_DATE	SYSDATE
H.INVOICE_TO_ORG_ID	PARTNRIDX of BillTo PARTNER
H.LAST_UPDATED_BY	FND_GLOBAL.LOGIN_ID
H.LAST_UPDATE_DATE	SYSDATE
H.LAST_UPDATE_LOGIN	FND_GLOBAL.USER_ID
H.ORG_ID	PARTNRIDX of SoldTo PARTNER, if this value is not null
H.SOLD_TO_ORG_ID	PARTNRIDX of SoldTo PARTNER
H.XML_MESSAGE_ID	XML Gateway Internal Control Number
L.CREATED_BY	FND_GLOBAL.USER_ID
L.CREATION_DATE	SYSDATE
L.LAST_UPDATED_BY	FND_GLOBAL.LOGIN_ID
L.LAST_UPDATE_DATE	SYSDATE
L.LAST_UPDATE_LOGIN	FND_GLOBAL.LOGIN_ID

Table 7–5 Derived Columns

Derived Columns	Source of Data, and Condition (if any)
L.ORDERED_QUANTITY, L.ORDER_QUANTITY_UOM	OAG Derivation from schedule level <QUANTITY qualifier="ORDERED"> tag
L.REQUEST_DATE	OAG Derivation from schedule level <DATETIME qualifier="NEEDELV"> tag
L.SHIP_TO_ORG_ID	PARTNRIDX of ShipTo PARTNER

Table 7–6 Workflow Event Setup

Detail	Value
Event Name	oracle.apps.ont.oi.po_inbound.create
Event Description	Order Management Generic Inbound Event
Subscription	R_OEOI_ORDER_IMPORT
Subscription Description	Oracle Order Management subscription to the event oracle.apps.ont.oi.po_inbound.create raised by the post process of the Process_PO XML mapping

Key: H => OE_HEADER_INTERFACE, L => OE_LINES_INTERFACE

Table 7-7 Process_PO Message Map

OAG Element	Description/Comment	Oracle OM Table/Column	OAG Required	Code conversion Needed Y/N
CNTROLAREA	The fields included in this area provide information about the XML document i.e. BSR, SENDER and DATETIME described below.	XMLG	Y	
<BSR>	Shows the Business Service Request name per OAGI:	XMLG	Y	
<VERB>	Value is 'PROCESS.'		Y	
<NOUN>	Value is 'PO.'		Y	
<REVISION>	Value is '007.'		Y	
<SENDER>	Provides information on the system that sends the document:	XMLG	Y	
<LOGICALID>	Sender system identifier. Value is '1'.		Y	
<COMPONENT>	Sender application name. Value is 'PURCHASING'		Y	
<TASK>	Event or Action. Value is 'POISSUE.'		Y	
<REFERENCEID>	Unique reference id for this doc.		Y	
<CONFIRMATION>	Confirmation when doc is received. Value is '0' means none required.		Y	
<LANGUAGE>	ISO language in which the text fields are transmitted.		Y	
<CODEPAGE>	Character set used in this XML doc. Value is 'US7ASCII.'		Y	
<AUTHID>	System id of sender. Value is 'APPS.'		Y	
<DATETIME>	Creation date and time of the XML document.	XMLG	Y	
DATAAREA	The fields included in this area provide information about the data included in the XML document.		Y	
POORDERHDR	This data type provides header level PO information. One PO Header data type is required per document.		Y	

Table 7-7 Process_PO Message Map

OAG Element	Description/Comment	Oracle OM Table/Column	OAG Required	Code conversion Needed Y/N
<POID>	This identifies the unique id for the purchase order. For Standard POs, the PO number from Oracle Purchasing is entered here. For Blanket Releases, it should be 'Blanket PO# - Release. It's the customer PO number for order management.	OE_HEADERS_INTERFACE.Orig_SYS_DOCUMENT_REF OE_HEADERS_INTERFACE.CUSTOMER_PO_NUMBER	Y	
<POTYPE>	This identifies various types of PO. "Standard" or "Blanket" is used here. Order management does not have any matching data for the PO type. The Order Type on Process PO will no longer automatically be set to 'Standard.' Order Management does not need map.		Y	
<ACKREQUEST>	Specifies to the supplier if an Acknowledgment is expected or not Oracle does not support consuming Ack PO so this field is only used to indicate if acceptance for the PO is required or not 0 - Not Required 2 – Required Order Management does not need map.		N	
<CONTRACTS>	Seller's Contract document number, to be used only if this is a release from the blanket order (as it is only available for blanket orders and can only be loaded through PDOI) If its available it should be populated in Sales Order Management does not need map.		N	
<DATETIME(DOCUMENT)>	Timestamp for Purchase Order (Standard or Release) creation. Order Management does not need map.		N	

Table 7-7 Process_PO Message Map

OAG Element	Description/Comment	Oracle OM Table/Column	OAG Required	Code conversion Needed Y/N
<DESCRIPTION>	Description for PO Header. Order Management does not need map.		N	
<OPERAMT(EXTE NTED(T)>	Total amount of the PO. Following are the fields included in this segment. Order Management does not need map.		N	
<VALUE>	Monetary amount of the PO.		N	
<NUMOFDEC>	Indicates the number of decimals in the value field.		N	
<SIGN>	'+' or '-' indicates whether the amount is positive or negative.		N	
<CURRENCY>	Three character ISO currency code.		N	
<UOMVALUE>	Numeric value indicates the value of the factor when amount is expressed in terms of multiples of UOM.		N	Y
<UOMNUMDEC>	Represents number of decimals in the UOMVALUE field.		N	
<UOM>	Unit of Measure indicates the units of the quantitative amount.		N	Y
<PORELEASE>	Indicates a new Release number; used only when PO Type is Blanket. Order Management does not need map.		N	

Table 7-7 Process_PO Message Map

OAG Element	Description/Comment	Oracle OM Table/Column	OAG Required	Code conversion Needed Y/N
<USERAREA>	The following fields are provided by Oracle in this Userarea:		N	
<DATETIME>	Start Active Date for Blanket PO			
<DATETIME>	Not applicable for Order Management			
<FOB>	End Active Date for Blanket PO. Not applicable for Order Management.			
<DESCRIPTN>	FOB shipping terms. This is the FOB point code.			
<TERMID>	FOB Description.			
<FTTERM>	FOB Terms.			
<DESCRIPTN>	Freight payment terms.			
<TERMID>	Freight Description.	OE_HEADERS_INTERFACE.FOB_POINT_CODE		
<EXCHRATE>	Freight Terms.			
<DATETIME>	Currency Exchange Rate.			
<DATETIME>	Date for the Exchange Rate			
	Acceptance Due By date (or Ack by date when PO Ack functionality is supported in Oracle Purchasing)			
<CONFIRM>	Not applicable for Order Management.	OE_HEADERS_INTERFACE.FREIGHT_TERMS_CODE		
	Indicates if the PO is confirmed by the supplier. 'Y' - Confirmed			
DFFPOHEADER	'N' - Not Confirmed. Not applicable for order Management.			
<ATTR1 - ATTR16>	PO header level DFF attributes (16)			
PCARDHDR	Not applicable for order Management			
	This segment contains PCARD detail Not applicable for order Management			
<MEMBERNAME>	Member Name on the Pcard			
>	Pcard Number			
<PCARDNUM>	Expiration Date of the Pcard			
<DATETIME>	Brand of the Pcard (EDI does the mapping of the PCard brands, it will not be possible in XML:)			
<PCARDBRAND>				

Table 7-7 Process_PO Message Map

OAG Element	Description/Comment	Oracle OM Table/Column	OAG Required	Code conversion Needed Y/N
PARTNER	This data type provides information about the trading partner. Two occurrences of Partner data type are required - Supplier and SoldTo. Oracle ERP provides two additional Partner occurrences - BillTo and Carrier which are optional per DTD.		Y	
PARTNER-Supplier	Order Management does not need to Map this segment. It's the supplier information.		Y	
<NAME1>	Name of Trading Partner.		Y	
<ONETIME>	Indicates if this partner is established for this transaction only.		Y	
<PARTNRID>	Uniquely identifies the partner in ERP.		Y	
<PARTNERTYPE>	Identifies the type of Partner. Value is 'Supplier'.		Y	
<CURRENCY>	Preferred operating currency of Partner.		N	
<DESCRIPTION>	Not Used.		N	
<NAME2 - NAME9>	Not Used.		N	
<PARTNRIDX>	Unique identifier of the Partner Supplier		N	
<TAXEXEMPT>	Not Used.		N	
<TAXID>	Tax identifier of the Partner.		N	
<USERAREA>			N	
DFFVENDOR <ATTR1 - ATTR16>	PO Vendor level DFF attributes (16)			
<CustomerNum>	Buyer's identifier in the supplier's system (vendor's customer number).			

Table 7-7 Process_PO Message Map

OAG Element	Description/Comment	Oracle OM Table/Column	OAG Required	Code conversion Needed Y/N
ADDRESS	This data type provides address information for this partner. The following rows list fields for Address data type related to Partner Supplier Site		Y	
<ADDRLINE1 - ADDRLINE9>	Lines of site address.		Y	
<ADDRTYPE>	Not Used.		N	
<CITY>	City within the address.		Y	
<COUNTRY>	Country within the address.		N	
<COUNTY>	Not Used.		N	
<DESCRIPTN>	Supplier Site Name		N	
<FAX1 - FAX9>	Fax numbers of supplier site		N	
<POSTALCODE>	Postal code within the address.		N	
<REGION>	Not Used.		N	
<STATEPROVN>	State of Province within the address.		Y	
<TAXJRS DCTN>	Not Used.		N	
<TELEPHONE1 - TELEPHONE9>	Telephone numbers for this address.		N	
<URL>	Not Used.		N	
<USERAREA> DFFVENDORSITE <ATTR1 - ATTR16>	PO Vendor Site level DFF attributes (16)		N	
CONTACT	This data type provides contact information for this supplier.		Y	
<NAME1>	Contact name for Supplier. The contact is entered for each PO at the header, that contact should be used as the supplier contact. The contact name and the telephone numbers of this contact should be used. See closed issue #3		Y	
<DESCRIPTN>	Not Used.		N	

Table 7-7 Process_PO Message Map

OAG Element	Description/Comment	Oracle OM Table/Column	OAG Required	Code conversion Needed Y/N
<EMAIL>	Email address for the Contact		N	
<FAX1 - FAX9>	Fax No. of the contact		N	
<NAME2 - NAME9>	Not Used.		N	
<TELEPHONE1 - TELEPHONE9>	Telephone number of the Contact.		N	
PARTNER-SoldTo			Y	
<NAME1>	Name of the buyer company name Sold to name.		Y	
<ONETIME>	Indicates if this partner is established for this transaction only.		Y	
<PARTNRID>	Uniquely identifies the partner in ERP.This is the Sold_to customer id which is internal to the buyers application.		Y	
<PARTNERTYPE>	Identifies the type of Partner. Value is 'SoldTo'.		Y	
<CURRENCY>	Preferred operating currency of Partner.		N	
<PARTNRIDX>	Unique identifier of the Partner. This is the EDI location code for the Buyer.This code should be used to derive the information about this buyer.	OE_HEADERS_INTERFACE.SOLD_TO_ORG_ID	N	
ADDRESS	The following rows list fields for Address data type related to Partner SoldTo. This is the Sold to address information.		N	
<ADDRLINE1 - ADDRLINE9>	Lines of address.		N	
<CITY>	City within the address.		N	
<COUNTRY>	Country within the address.		N	
<FAX1 - FAX9>	Fax Number		N	
<POSTALCODE>	Postal code within the address.		N	

Table 7-7 Process_PO Message Map

OAG Element	Description/Comment	Oracle OM Table/Column	OAG Required	Code conversion Needed Y/N
<STATEPROVN>	State of Province within the address.		N	
<TELEPHONE1 - TELEPHONE9>	Telephone numbers for this address.		N	
CONTACT	The following rows list fields for Contact data type related to Partner SoldTo.		Y	
<NAME1>	Name of Buyer e.g. Pat Stock		Y	
<DESCRIPTN>	Not Used.		N	
<EMAIL>	Email address for the Contact.		N	
<FAX1 - FAX9>	Fax Number of the contact		N	
<NAME2 - NAME9>	Alternate Contact Names		N	
<TELEPHONE1 - TELEPHONE9>	Telephone number of the Contact.		N	
<USERAREA>	Not Used.		N	
PARTNER-BillTo	Bill To location in ERP.		N	
<NAME>	Name of Trading Partner. Bill to customer name.	OE_HEADERS_INTERFACE.INVOICE_CUSTOMER	Y	
<ONETIME>	Indicates if this partner is established for this transaction only.		Y	
<PARTNRID>	Uniquely identifies the Bill To Location Id in ERP. This is the EDI location code for the bill to customer site.		Y	
<PARTNERTYPE>	Identifies the type of Partner. Value is 'BillTo'.		Y	
<CURRENCY>	Preferred operating currency of Partner.		N	
<PARTNRIDX>	Unique identifier of the Partner.	OE_HEADERS_INTERFACE.INVOICE_TO_ORG_ID	N	

Table 7-7 Process_PO Message Map

OAG Element	Description/Comment	Oracle OM Table/Column	OAG Required	Code conversion Needed Y/N
ADDRESS	The following rows list fields for Address data type related to Partner Bill_To.		N	
<ADDRLINE1 - ADDRLINE9>	Lines of address.	OE_HEADERS_INTERFACE.INVOICE_ADDRESS1/2/3	N	
<ADDRTYPE>	Not Used.		N	
<CITY>	City within the address.	OE_HEADERS_INTERFACE.INVOICE_CITY	N	
<COUNTRY>	Country within the address.	OE_HEADERS_INTERFACE.INVOICE_COUNTRY	N	
<COUNTY>	Not Used.	OE_HEADERS_INTERFACE.INVOICE_COUNTY	N	
<POSTALCODE>	Postal code within the address.	OE_HEADERS_INTERFACE.INVOICE_POSTAL_CODE	N	
<STATEPROVN>	State of Province within the address.	OE_HEADERS_INTERFACE.INVOICE_STATE	N	
<TELEPHONE1 - TELEPHONE9>	Telephone numbers for this address.		N	
PARTNER-CARRIER	Carrier information is passed in this segment. The carrier may become a supplier in Oracle Purchasing in patchset release 'H' or 'I.' That may impact how the carrier tags are derived and the address / contact segments may be used then. This segment is not applicable to Order Management.		N	
<NAME1>	Name of Trading Partner.		Y	
<ONETIME>	Indicates if this partner is established for this transaction only.		Y	
<PARTNRID>	Uniquely identifies the partner in ERP.		Y	

Table 7-7 Process_PO Message Map

OAG Element	Description/Comment	Oracle OM Table/Column	OAG Required	Code conversion Needed Y/N
<PARTNERTYPE>	Identifies the type of Partner. Value is 'Carrier'.		Y	
ADDRESS	The Address data type is not used for Partner Carrier.		N	
CONTACT	The Contact data type is not used for Partner Carrier.		N	
POTERM	POTERM data type represents payment due dates and discounts.		Y	
<DESCRIPTN>	Description of payment terms.		Y	
<TERMID>	Identifier for payment terms. This is the payment terms code for example 'Net30.' Can be mapped to payment_terms in order management on the header level.		Y	Y
ATTCHREF	Attachments at the header level. This will support attachments 'TO SUPPLIER' only.		N	
<FILENAME>	File name of the file attached if the data type is file.		Y	
<URI>	URL, if the data type is URL.		Y	
<DATETIME (CREATION)>	Creation Date of the document.		N	
<DESCRIPTN>	Description of the document.		N	
<NOTES1-NOTES9>	If the data type is 'Long text' or short text' then pass the notes to the supplier in these elements.		N	
<USERAREA>	The following fields are provided by Oracle in the USERAREA.		Y	
<SEQNUM>	Sequence Number of the attachment.		Y	
<DATATYPE>	Datatype Name - Short Text, Long Text, Image or OLE Object.			
POORDERLIN	This data type provides details of a PO line. At least one Purchase Order Line data type is required. This data type will occur one or more times.		Y	

Table 7-7 Process_PO Message Map

OAG Element	Description/Comment	Oracle OM Table/Column	OAG Required	Code conversion Needed Y/N
<POLINENUM>	Line number of the PO. Original PO line number reference	OE_LINES_INTERFACE.Orig_SYS_LINE_REF	Y	
<QUANTITY(ORDERED)>	Indicates quantity of item ordered. Numeric only, stores the value of quantity.		Y	
<VALUE>	One character numeric only; indicates the no. of decimals in the value field.		Y	
<NUMOFDEC>	+ 'or ' - ' indicates whether the quantity is positive or negative.		Y	
<SIGN>	Unit of Measure indicates the units of the quantity.		Y	
<UOM>			Y	Y
<ITEM>	Identifier of the product. Concatenate all the segments to display the item This is the customer item number.	OE_LINES_INTERFACE.CUSTOMER_ITEM_NAME	Y	
<DESCRIPTN>	Description of the item.		N	
<ITEMX>	Vendor's Item Number. This is the item number internal to the supplier (Order management)side.		N	
<USERAREA>				
<USERITEM DESCRIPTN>	User Item Description	OE_LINES_INTERFACE.USER_ITEM_DESCRIPTION	N	
<DATETIME(NEE DDELV)>	Need By date for the Item.		N	
<HAZRDMATL>	Hazardous material class description.		N	
<ITEMRV>	Item Revision Number. This is the customer item revision.	OE_LINES_INTERFACE.CUSTOMER_ITEM_REVISION	N	
<ITEMRVX>	Not Used		N	
<NOTES1 - NOTES9>	Note to supplier.		N	

Table 7-7 Process_PO Message Map

OAG Element	Description/Comment	Oracle OM Table/Column	OAG Required	Code conversion Needed Y/N
<OPERAMT(UNIT)(T)>	Unit price of the item. Following are the fields included in this segment.		N	
<VALUE>	Monetary unit amount of PO line.		N	
<NUMOFDEC>	Indicates the number of decimals in the value field.		N	
<SIGN>	'+' or '-' indicates whether the amount is positive or negative.		N	
<CURRENCY>	Three character ISO currency code.		N	
<UOMVALUE>	Numeric value indicates the value of the factor when amount is expressed in terms of multiples of UOM.		N	
<UOMNUMDEC>	Represents number of decimals in the UOMVALUE field.			
<UOM>	Unit of Measure indicates the units of the quantitative amount.		N	
<USERAREA>	The following fields are provided by Oracle ERP within this userarea: No applicable for OM.			
<REQUESTOR>	Requestor of this line. The requester is at the distribution level and not at the line level. Need to find how EDI is deriving it as they have it at the line level.			
<CATEGORYID>	Item Category unique identifier.			
<CONTRACTPONUM>	Contract PO Number for this line			
<CONTRACTPOLINENUM>	Contract PO line Number for this line. The contract line numbers do not exist in Purchasing. Need to find EDI is deriving them as they have it at the line level.			

Table 7-7 Process_PO Message Map

OAG Element	Description/Comment	Oracle OM Table/Column	OAG Required	Code conversion Needed Y/N
<VENDORQUOTE ENUM>	Vendor's Quote number for this line		N	
<LISTPRICE>	List Price of the item			
<MARKETPRICE>	Market Price of the item			
<PRICENOTTOEX CEED>	Unit Price not to exceed this amount.			
<NEGPRICE>	Negotiable Price Flag. Y/N. Only applicable to Blanket and is called 'Price Override' in Oracle Purchasing.			
<TAXABLE>				
<TXNREASONCO DE>	If this item is taxable or not Y/N			
<TYPE1099>	Transaction Reason Code, this is used to group requisition lines for autocreating POs			
<LINEORDERTYP E>	Type 1099 Y/N			
	Line Order Type e.g. Goods, Services etc.			
<HAZRDUNNUM >	UN Hazard number			
<HAZRDUNDES C>	UN Hazard Description			
	Desc Flex Fields at the line level			
DFFLINE				
<ATTR1-ATTR16> DFFITEM	Desc Flex Fields at the item level			
<ATTR1-ATTR16>	Key Flex Fields at the item level			
KFFITEM				
<ATTR1-ATTR16>	Item Level attachments from the master org			
ATTCHREF-Item Level from Master Org				
	Item Level attachments from the inventory org			
ATTCHREF-Item Level from Inventory Org				

Table 7–7 Process_PO Message Map

OAG Element	Description/Comment	Oracle OM Table/Column	OAG Required	Code conversion Needed Y/N
<USERAREA- CONTD..> <ATTCHREF>	Each occurrence of attachments in the userarea above at the item level will be treated as the ATTCHREF segment and will have the following elements. This will support attachments 'TO SUPPLIER' only.		N	
<FILENAME>	File name of the file attached if the data type is file.		Y	
<URI>	URL, if the data type is URL.		Y	
<DATETIME (CREATION)>	Creation Date of the document.		N	
<DESCRIPTN>	Description of the document		N	
<NOTES1- NOTES9>	If the data type is 'Long text' or short text' then pass the notes to the supplier in these elements.		N	
<SEQNUM> <DATATYPE>	Sequence Number of the attachment Datatype Name - Short Text, Long Text, File, URL, Image or OLE Object.		Y Y	
PARTNER	Not Used at line level.		N	
ATTCHREF	Attachments at the line level. This will support attachments 'TO SUPPLIER' only.		N	
<FILENAME>	File name of the file attached if the data type is file		Y	
<URI>	URL, if the data type is URL.		Y	
<DATETIME (CREATION)>	Creation Date of the document.		N	
<DESCRIPTN>	Description of the document.		N	
<NOTES1- NOTES9>	If the data type is 'Long text' or short text' then pass the notes to the supplier in these elements.		N	

Table 7-7 Process_PO Message Map

OAG Element	Description/Comment	Oracle OM Table/Column	OAG Required	Code conversion Needed Y/N
<USERAREA>	The following fields are provided by Oracle in the USERAREA.		Y	
<SEQNUM>	Sequence Number of the attachment.		Y	
<DATATYPE>	Datatype Name - Short Text, Long Text, Image or OLE Object.			
POLINESCHD	Data type for requested ship date information for this PO Line.		N	
<<DATETIME(NEEDEDDELV)>>	Needed-By delivery date. This is the request date for the line.	OE_LINES_INTERFACE.REQUEST_DATE	Y	
<QUANTITY(ORDERED)>	Indicates quantity of item ordered.	OE_LINES_INTERFACE.ORDERED_QUANTITY	Y	
<VALUE>	Numeric only, stores the value of quantity.		Y	
<NUMOFDEC>	One character numeric only; indicates the no. of decimals in the value field.		Y	
<SIGN>	+ or '-' indicates whether the quantity is positive or negative.		Y	
<UOM>	Unit of Measure indicates the units of the quantity.		Y	
		OE_LINES_INTERFACE.ORDER_QUANTITY_UOM		Y
<PSCLINENUM>	Identifies the line number on the delivery schedule of the PO. Shipment line reference from the original PO Line.	OE_LINES_INTERFACE.ORIG_SYS_SHIPMENT_REF	N	

Table 7-7 Process_PO Message Map

OAG Element	Description/Comment	Oracle OM Table/Column	OAG Required	Code conversion Needed Y/N
<USERAREA>				
<DATETIME>	Promise Date		N	
<DATETIME>	Last Acceptance Date		N	
<PRICEOVRD>	Price Override, this is the new price. Note this will be a complete Amount field as per OAG specs		N	
<TAXABLE>	Taxable Y/N		N	
<TAXCODE>	Tax Code if Taxable is 'Y'		N	
<REQUESTOR>	Requestor for this shipment. See Open issue#8		N	
PARTNER-ShipTo	The following fields related to ShipTo Partner data type are provided by Oracle ERP within this userarea:			
<NAME1>	Name of the ShipTo partner.		N	
<ONETIME>	Indicates if this partner is established for this transaction only. This is always '0'.		N	
<PARTNRID>	Uniquely identifies the partner in ERP.		N	
<PARTNERTYPE>	Identifies the type of Partner. Value is 'ShipTo'.		N	
<CURRENCY>	Preferred operating currency of Partner.		N	
<PARTNRIDX>	Unique identifier of the Partner.	_LINES_		
ADDRESS	Lines of address for Partner ShipTo.	INTERFACE.SHIP_TO_ORG_ID	N	
<ADDRLINE1 - ADDRLINE9>	City within the address.	OE_LINES_		
<CITY>	Country within the address.	INTERFACE.SHIP_TO_ADDRESS1/2/3	N	
<COUNTRY>	Postal code within the address.	OE_LINES_		
<POSTALCODE>	State or Province within the address.	INTERFACE.SHIP_TO_CITY	N	
<STATEPROVN>	Telephone numbers for this address.	OE_LINES_		
<TELEPHONE1 - TELEPHONE9>		INTERFACE.SHIP_TO_COUNTRY	N	
		OE_LINES_		
		INTERFACE.SHIP_TO_POSTAL_CODE	N	
		OE_LINES_		
		INTERFACE.SHIP_TO_STATE	N	

Table 7-7 Process_PO Message Map

OAG Element	Description/Comment	Oracle OM Table/Column	OAG Required	Code conversion Needed Y/N
<USERAREA-CONTD..> <ATTCHREF>	Attachments at the Shipment level. This will support attachments 'TO SUPPLIER' only. There could be multiple occurrences of this segment for each shipment schedule.		N	
<FILENAME>	File name of the file attached if the data type is file.		Y	
<URI>	URL, if the data type is URL.		Y	
<DATETIME (CREATION)>	Creation Date of the document.		N	
<DESCRIPTN>	Description of the document		N	
<NOTES1-NOTES9>	If the data type is 'Long text' or short text' then pass the notes to the supplier in these elements.		N	
<SEQNUM> <DATATYPE>	Sequence Number of the attachment. Datatype Name - Short Text, Long Text, Image or OLE Object.		Y Y	

Oracle XML Gateway Message Designer - Properties

The source DTD used is 003_process_po_007.dtd, revision 7.2.1 of the Open Application Group. The other associated external reference DTD files are;

- oagis_domains.dtd
- oagis_resources.dtd
- oagis_fields.dtd
- oagis_segments.dtd
- oagis_extensions.dtd
- oagis_entity_extensions.dtd

All the DTD's will be checked in ONT source area under \$ont/xml/oag72

The target for the Inbound XML Message are the Order Management Open Interface tables OE_HEADERS_INTERFACE & OE_LINES_INTERFACE.

The PROCESS_PO DTD is a three level hierarchy with order lines split into one or more shipment lines. The Order Management architecture is however a two level one with Order Header and one or more Lines. The message map will collapse the three level XML Message into a two level structure when the data is inserted in the Order Management Open Interface tables.

Please refer to [Table 7-7, "Process_PO Message Map"](#) for a detail analysis of the elements mapped, actions and derivation rules used by the Message Map.

Both the message map created using the Message Designer and its associated DTD's are stored in the database. The following Java programs are available to Load/Delete Maps or Load/Delete DTD's into/from the XML Gateway repository. Please refer to the *Oracle XML Gateway Manual* for more information.

Load/Delete Maps, Load/Delete DTD's

Note: The following process is used only for customizations.

1. `java LoadMap <DB username> <DB password> <Hostname>:<Port>:<SID>
<mymap.xgm>`

Example: `java oracle.apps.ecx.loader.LoadMap apps apps
ap505dbs:1521:dev115 ONT_3A4R_OAG72_IN.xgm`

2. `java DeleteMap <DB username> <DB password> <Hostname>:<Port>:<SID>
<mapname>`

Example: `java oracle.apps.ecx.loader.DeleteMap apps apps
ap505dbs:1521:dev115 ONT_3A4R_OAG72_IN.xgm`

The Message Map is a .xgm file which will be stored in the Order Management Source area under \$ont/patch/115/xml/US/ ONT_3A4R_OAG72_IN.xgm

Note: Maps and DTD's must be kept in sync between the Message Designer and the XML Gateway repository. When in doubt, always reload the map and DTD as a pair.

Extension Tags

Order Management added the following tags as extensions on the Process_PO XML documents sent to Order Management:

\$ont/xml/oag72/oagis_extensions.dtd

```
<!ELEMENT FTTERM (DESCRIPTN?, TERMID?)>
```

```
<!ELEMENT FOB (DESCRIPTN?, TERMID?)>
```

```
<!ELEMENT USERITEMDESCRIPTN %STRDOM;>
```

The following changes were made by Procurement as extensions on the XML documents sent to Order Management – as a result, Order Management must also modify the same OAG dtds to prevent parse errors at XML gateway:

\$ont/xml/oag72/oagis_entity_extensions.dtd

Add the following tags:

```
<!ENTITY % DATETIME.EXCHRATE DATE "DATETIME">
```

Change <!ENTITY % SEG_DATETIME_QUALIFIERS_EXTENSION "OTHER"> to <!ENTITY % SEG_DATETIME_QUALIFIERS_EXTENSION "OTHER | EXCHRATE DATE">

\$ont/xml/oag72/oagis_extensions.dtd

Add the following tags:

```
<!ELEMENT PAYMMETHOD (DESCRIPTN?, TERMID?)>
```

```
<!ELEMENT CREDITCRD  
(CARDID?, NAME?, (%DATETIME.EXPIRATION;)?)>
```

```
<!ELEMENT STARTACTIVEDATE %STRDOM;>
```

```
<!ELEMENT ENDACTIVEDATE %STRDOM;>
```

```
<!ELEMENT CATEGORYID %STRDOM;>
```

```
<!ELEMENT REVISIONNUM %STRDOM;>
```

```
<!ELEMENT ATTACHMENT (TEXT?)>
```

```
<!ELEMENT EXCHRATE %STRDOM;>
```

```
<!ELEMENT CONFIRM %STRDOM;>
```

```
<!ELEMENT PCARDHDR  
(MEMBERNAME?, PCARDNUM?, (%DATETIME.EXPIRATION;)?, PCARDBRAND?)>
```

```
<!ELEMENT MEMBERNAME %STRDOM;>
```

```
<!ELEMENT PCARDNUM %STRDOM;>
```

```
<!ELEMENT PCARDBRAND %STRDOM;>
<!ELEMENT CUSTOMERNUM %STRDOM;>
<!ELEMENT REQUESTOR %STRDOM;>
<!ELEMENT CONTRACTPONUM %STRDOM;>
<!ELEMENT CONTRACTPOLINENUM %STRDOM;>
<!ELEMENT VENDORQUOTENUM %STRDOM;>
<!ELEMENT LISTPRICE %STRDOM;>
<!ELEMENT MARKETPRICE %STRDOM;>
<!ELEMENT PRICENOTTOEXCEED %STRDOM;>
<!ELEMENT NEGPRICE %STRDOM;>
<!ELEMENT TAXABLE %STRDOM;>
<!ELEMENT TXNREASONCODE %STRDOM;>
<!ELEMENT TYPE1099 %STRDOM;>
<!ELEMENT LINEORDERTYPE %STRDOM;>
<!ELEMENT HAZRDUNNUM %STRDOM;>
<!ELEMENT HAZRDUNDESC %STRDOM;>
<!ELEMENT PRICEOVRD %STRDOM;>
<!ELEMENT DISTPROJECT
(REQUESTOR?,DISTNUM?,PROJECTNUM?,PROJECTTYPE?,TASKNUM?,(%
QUANTITY.ORDERED;)?,CONVRATE,(%DATETIME.EXCHRATEDATE;)?,D
ESTTYPE?,DFFDISTRIBUTN?)>
<!ELEMENT PROJECTNUM %STRDOM;>
<!ELEMENT DISTNUM %STRDOM;>
<!ELEMENT PROJECTTYPE %STRDOM;>
<!ELEMENT TASKNUM %STRDOM;>
<!ELEMENT CONVRATE %STRDOM;>
<!ELEMENT DESTTYPE %STRDOM;>
<!ELEMENT DFFPOHEADER
```

(ATTRIBUTE1?,ATTRIBUTE2?,ATTRIBUTE3?,ATTRIBUTE4?,ATTRIBUTE5?,ATTRIBUTE6?,ATTRIBUTE7?,ATTRIBUTE8?,ATTRIBUTE9?,ATTRIBUTE10?,ATTRIBUTE11?,ATTRIBUTE12?,ATTRIBUTE13?,ATTRIBUTE14?,ATTRIBUTE15?,ATTRIBUTE16?)>

<!ELEMENT DFFVENDORSITE

(ATTRIBUTE1?,ATTRIBUTE2?,ATTRIBUTE3?,ATTRIBUTE4?,ATTRIBUTE5?,ATTRIBUTE6?,ATTRIBUTE7?,ATTRIBUTE8?,ATTRIBUTE9?,ATTRIBUTE10?,ATTRIBUTE11?,ATTRIBUTE12?,ATTRIBUTE13?,ATTRIBUTE14?,ATTRIBUTE15?,ATTRIBUTE16?)>

<!ELEMENT DFFVENDOR

(ATTRIBUTE1?,ATTRIBUTE2?,ATTRIBUTE3?,ATTRIBUTE4?,ATTRIBUTE5?,ATTRIBUTE6?,ATTRIBUTE7?,ATTRIBUTE8?,ATTRIBUTE9?,ATTRIBUTE10?,ATTRIBUTE11?,ATTRIBUTE12?,ATTRIBUTE13?,ATTRIBUTE14?,ATTRIBUTE15?,ATTRIBUTE16?)>

<!ELEMENT DFFLINE

(ATTRIBUTE1?,ATTRIBUTE2?,ATTRIBUTE3?,ATTRIBUTE4?,ATTRIBUTE5?,ATTRIBUTE6?,ATTRIBUTE7?,ATTRIBUTE8?,ATTRIBUTE9?,ATTRIBUTE10?,ATTRIBUTE11?,ATTRIBUTE12?,ATTRIBUTE13?,ATTRIBUTE14?,ATTRIBUTE15?,ATTRIBUTE16?)>

<!ELEMENT DFFITEM

(ATTRIBUTE1?,ATTRIBUTE2?,ATTRIBUTE3?,ATTRIBUTE4?,ATTRIBUTE5?,ATTRIBUTE6?,ATTRIBUTE7?,ATTRIBUTE8?,ATTRIBUTE9?,ATTRIBUTE10?,ATTRIBUTE11?,ATTRIBUTE12?,ATTRIBUTE13?,ATTRIBUTE14?,ATTRIBUTE15?,ATTRIBUTE16?)>

<!ELEMENT KFFITEM

(ATTRIBUTE1?,ATTRIBUTE2?,ATTRIBUTE3?,ATTRIBUTE4?,ATTRIBUTE5?,ATTRIBUTE6?,ATTRIBUTE7?,ATTRIBUTE8?,ATTRIBUTE9?,ATTRIBUTE10?,ATTRIBUTE11?,ATTRIBUTE12?,ATTRIBUTE13?,ATTRIBUTE14?,ATTRIBUTE15?,ATTRIBUTE16?,ATTRIBUTE17?,ATTRIBUTE18?,ATTRIBUTE19?,ATTRIBUTE20?)>

<!ELEMENT DFFDISTRIBUTN

(ATTRIBUTE1?,ATTRIBUTE2?,ATTRIBUTE3?,ATTRIBUTE4?,ATTRIBUTE5?,ATTRIBUTE6?,ATTRIBUTE7?,ATTRIBUTE8?,ATTRIBUTE9?,ATTRIBUTE10?,ATTRIBUTE11?,ATTRIBUTE12?,ATTRIBUTE13?,ATTRIBUTE14?,ATTRIBUTE15?,ATTRIBUTE16?)>

<!ELEMENT ATTRIBUTE1 %STRDOM;>

```
<!ELEMENT ATTRIBUTE2 %STRDOM;>
<!ELEMENT ATTRIBUTE3 %STRDOM;>
<!ELEMENT ATTRIBUTE4 %STRDOM;>
<!ELEMENT ATTRIBUTE5 %STRDOM;>
<!ELEMENT ATTRIBUTE6 %STRDOM;>
<!ELEMENT ATTRIBUTE7 %STRDOM;>
<!ELEMENT ATTRIBUTE8 %STRDOM;>
<!ELEMENT ATTRIBUTE9 %STRDOM;>
<!ELEMENT ATTRIBUTE10 %STRDOM;>
<!ELEMENT ATTRIBUTE11 %STRDOM;>
<!ELEMENT ATTRIBUTE12 %STRDOM;>
<!ELEMENT ATTRIBUTE13 %STRDOM;>
<!ELEMENT ATTRIBUTE14 %STRDOM;>
<!ELEMENT ATTRIBUTE15 %STRDOM;>
<!ELEMENT ATTRIBUTE16 %STRDOM;>
<!ELEMENT ATTRIBUTE17 %STRDOM;>
<!ELEMENT ATTRIBUTE18 %STRDOM;>
<!ELEMENT ATTRIBUTE19 %STRDOM;>
<!ELEMENT ATTRIBUTE20 %STRDOM;>
<!ELEMENT TANDC %STRDOM;>
<!ELEMENT GLOBALCONTRACT %STRDOM;>
<!ELEMENT GLOBALCONTRACTLIN %STRDOM;>
<!ELEMENT CONSIGNEDINV %STRDOM;>
<!ELEMENT DROPSHIPDETAILS (DROPSHIPMENT?, DROPSHIPCUSTNAME?,
SHIPINSTR?,
PACKINSTR?, SHIPMETHOD?, CUSTOMERPONUM?, CUSTOMERLINENUM?,
CUSTOMERSHIPNUM?,
CUSTOMERDESC?)>
```

```
<!ELEMENT DROPSHIPMENT %STRDOM;>
<!ELEMENT DROPSHIPCUSTNAME %STRDOM;>
<!ELEMENT SHIPINSTR %STRDOM;>
<!ELEMENT PACKINSTR %STRDOM;>
<!ELEMENT SHIPMETHOD %STRDOM;>
<!ELEMENT CUSTOMERPONUM %STRDOM;>
<!ELEMENT CUSTOMERLINENUM %STRDOM;>
<!ELEMENT CUSTOMERSHIPNUM %STRDOM;>
<!ELEMENT CUSTOMERDESC %STRDOM;>
<!ELEMENT SHIPPINGCONTROL %STRDOM;>
<!ELEMENT CONTRACTNUM %STRDOM;>
<!ELEMENT CONFIGID %STRDOM;>
<!ELEMENT PSCLNSTATUS %STRDOM;>
<!ELEMENT JOBTITLE %STRDOM;> -- HR Job title
<!ELEMENT CONTRACTORFIRSTNAME %STRDOM;> -- Contractor first name
<!ELEMENT CONTRACTORLASTNAME %STRDOM;> -- Contractor last name
<!ELEMENT PRICEDIFFERENTIAL
(NUMBER?,ENTITYTYPE?,MULTIPLIER?,PRICETYPE?)>
--repeatable segment
<!ELEMENT NUMBER %STRDOM;> --price differential line num
<!ELEMENT ENTITYTYPE %STRDOM;> -- Entity type
<!ELEMENT MULTIPLIER %STRDOM;> -- multiplier for the standard PO line
```

Acknowledge_PO

Overview

Order Management 11i.9 supports the Acknowledge_PO transaction for the OAG XML standard.

Oracle Order Management generates the Acknowledge_PO message upon consuming the Process_PO inbound message and sends an Acknowledge_PO message to the buyer. The Acknowledge_PO message is sent in both the Synchronous and Asynchronous processing of the incoming orders.

This outbound message carries the status of the order received via the inbound PO request to the buyer. The message also contains the reason code for the rejected orders.

Major Features

Major features of this transaction are as follows:

Generate and Send Acknowledge_PO Message Out

Oracle Order Management generates the Acknowledge_PO message upon consuming the Process_PO inbound message (by triggering the Order Import and entering the order in Order Management tables). At this point Order Management must trigger the workflow to send Acknowledge_PO message to the buyer. This message should be generated within the timeframe agreed upon in the Trading Partner Agreement.

The Acknowledge_PO message is generated in both the Synchronous and Asynchronous processing of the incoming orders.

Acknowledge_PO message is populated with the information about the incoming Process_PO request. The message id of the incoming PO is stored and retrieved later during the creation of the Acknowledge_PO.

For the details of this message and mapping to Order Management refer to the [Table 7–10, "Acknowledge_PO OAG Message Map Details"](#).

Oracle Order Management generates the Acknowledge_PO message upon consuming the Process_PO inbound message and sends an Acknowledge_PO message to the buyer.

The Acknowledge_PO message is sent in both the Synchronous and Asynchronous processing of the incoming orders.

Acknowledge_PO message has information about the incoming Process_PO request.

Possible Trigger Points for Acknowledge_PO Message

Note : In the case of failure at the XML gateway processing BEFORE the order reaches the Order Import tables there is no action required by Order Management as the XML gateway notification process will handle the exception by notifying the System Administrator on the Buyer's side.

Possible Trigger Points

The following are the possible trigger points for the Acknowledge_PO message. These points are exclusive to each other, which means that for an order only one Acknowledge_PO message is generated at one of these points in the process.

After Order import runs if there is a validation error and Order import is not able to create an order in the base order table, the Acknowledge_PO message is sent to the buyer with the **Reject** status. Otherwise, if Order import is successful, the Acknowledge_PO message is sent to the buyer with Accepted status. This is at order entry.

Message Map

Table 7–8 Message Map

Name	Description
Message Map Name:	ONT_3A4A_OAG72_OUT_PO
Direction:	Outbound
(Internal) Transaction Type:	ONT
(Internal) Transaction Subtype:	POA
External Transaction Type:	PO
External Transaction Subtype:	ACKNOWLEDGE
DTD Directory:	xml/oag72
Map Directory:	patch/115/xml/US
Message Maps XGM File Name:	ONT_3A4A_OAG72_OUT_PO.xgm

Table 7–8 Message Map

Name	Description
Standard:	OAG
Release:	7.2
Format:	DTD
DTD Name:	004_acknowledge_po_008.dtd

Table 7–9 Workflow Event Setup

Detail	Value
Event Name	oracle.apps.ont.oi.po_ack.create
Event Description	Event for Process_PO Outbound Acknowledgment
Subscription	R_OEOA_SEND_ACKNOWLEDGMENT
Subscription Description	Oracle Order Management subscription to the event oracle.apps.ont.oi.po_ack.create raised for sending acknowledgment for Process_PO.

Key: H => OE_HEADER_ACKS and L => OE_LINE_ACKS

Table 7–10 Acknowledge_PO OAG Message Map Details

OAG Element	Description /Comment	Oracle OM Table /Column	OAG Req Y/N	Code Conversion Needed?
<ACKNOWLEDGE_PO_008>				
<CNTROLAREA>				
<BSR>				
<VERB>ACKNOWLEDGE</VERB>	XMLG		Y	
<NOUN>PO</NOUN>	XMLG		Y	
<REVISION>007</REVISION>	XMLG		Y	
</BSR>				
<SENDER>				

Table 7-10 Acknowledge_PO OAG Message Map Details

OAG Element	Description /Comment	Oracle OM Table /Column	OAG Req Y/N	Code Conversion Needed?
<LOGICALID>CPAG<LOGICALID>	XMLG		Y	
<COMPONENT>ORDERMGMT<COMPONENT>	XMLG		Y	
<TASK>ACKPO<TASK>	XMLG		Y	
<REFERENCEID>CPAGPOBERLIN02<REFERENCEID>	XMLG		Y	
<CONFIRMATION>0</CONFIRMATION>	XMLG		Y	
<LANGUAGE>ENG</LANGUAGE>	XMLG		Y	
<CODEPAGE>CP000111</CODEPAGE>	XMLG		Y	
<AUTHID>RSCHULTE</AUTHID>	XMLG		Y	
<SENDER>				
<DATETIME qualifier="CREATION">	XMLG		Y	
<YEAR>1996</YEAR>	XMLG		Y	
<MONTH>06</MONTH>	XMLG		Y	
<DAY>30</DAY>	XMLG		Y	
<HOUR>23</HOUR>	XMLG		Y	
<MINUTE>59</MINUTE>	XMLG		Y	
<SECOND>59</SECOND>	XMLG		Y	
<SUBSECOND>0000</SUBSECOND>	XMLG		Y	
</DATETIME>				
</CNTROLAREA>				
<DATAAREA>				
<ACKNOWLEDGE_PO>				
<POORDERHDR>				

Table 7–10 Acknowledge_PO OAG Message Map Details

OAG Element	Description /Comment	Oracle OM Table /Column	OAG Req Y/N	Code Conversion Needed?
<DATETIME qualifier="DOCUMENT">	Date time stamp of the Sales order creation. Derived using Convert To OAG Datetime.	H.ORDERED_ DATE	Y	
<YEAR>1996</YEAR>			Y	
<MONTH>06</MONTH>			Y	
<DAY>30</DAY>			Y	
<HOUR>23</HOUR>			Y	
<MINUTE>59</MINUTE>			Y	
<SECOND>59</SECOND>			Y	
<SUBSECOND>0000</SUBSECON D>			Y	
</DATETIME>				
<OPERAMT qualifier="EXTENDED" type="T">	Total amount for the Order. This is not stored but calculated using Convert To OAG Amount.	H.ORDER_ TOTAL	Y	
<VALUE>670</VALUE>	Total monetary amount of the order.		Y	
<NUMOFDEC>2</NUMOFDEC>	Indicates the number of decimals in the value field.		Y	
<SIGN>+</SIGN>	'+' or '-' indicates whether the amount is positive or negative.		Y	
<CURRENCY>USD</CURRENCY>	Three character ISO currency code.		Y	
<UOMVALUE>1</UOMVALUE>	Numeric value indicates the value of the factor when amount is expressed in terms of multiples of UOM.		Y	

Table 7–10 Acknowledge_PO OAG Message Map Details

OAG Element	Description /Comment	Oracle OM Table /Column	OAG Req Y/N	Code Conversion Needed?
<UOMNUMDEC>0</UOMNUMDEC>	Represents number of decimals in the UOMVALUE field.		Y	
<UOM>EACH</UOM>	Unit of Measure indicates the units of the quantitative amount.		Y	
</OPERAMT>			Y	
<POID>12345678</POID>	ORIG_SYS_DOCUMENT_REF Original Customer PO number from the inbound process_PO should be populated here.	H.ORIG_SYS_DOCUMENT_REF	Y	
<POTYPE>224</POTYPE>	'STANDARD' Indicates the type of the PO. At the moment only Standard PO will be the only type supported. The Order Type on Process PO will no longer automatically be set to 'Standard.'		Y	
<PORELEASE>1</PORELEASE>	Po release number. Is not populated by OM. Purchasing will split the Po release number from the PO number which has a format of Ponumber release number *See Notes		N	
<ACKHEADER>				
<SENDER>	* See Notes		Y	
<LOGICALID>CPAG</LOGICALID>	XML Gateway API	XML Gateway API	Y	
<COMPONENT>PURCHASING</COMPONENT>	XML Gateway API	XML Gateway API	Y	

Table 7–10 Acknowledge_PO OAG Message Map Details

OAG Element	Description /Comment	Oracle OM Table /Column	OAG Req Y/N	Code Conversion Needed?
<TASK>POCREATE</TASK>	XML Gateway API	XML Gateway API	Y	
<REFERENCEID>CPAGPOBERLI N02</REFERENCEID>	XML Gateway API	XML Gateway API	Y	
<CONFIRMATION>0</CONFIRMATION>	XML Gateway API	XML Gateway API	Y	
<LANGUAGE>ENG</LANGUAGE>	XML Gateway API	XML Gateway API	Y	
<CODEPAGE>CP000111</CODEPAGE>	XML Gateway API	XML Gateway API	Y	
<AUTHID>CMKURT</AUTHID>	XML Gateway API	XML Gateway API	Y	
</SENDER>				
<ACKCODE>1</ACKCODE>	Status Code * See Notes	H.FIRST_ACK_CODE	Y	
<NOTES index="1">This is a test message</NOTES>	Reason Code * See Notes		N	
<SALESORDID>32325</SALESORDID>	Sales Order Number	H.ORDER_NUMBER	N	
</ACKHEADER>			N	
<PARTNER>			N	
<NAME index="1">CPAG</NAME>	Trading partner name . In this case Buyers name. Derived from OE_XML_PROCESS_UTIL.GET_SOLD_TO_EDI_LOC			
<ONETIME>0</ONETIME>	0			
<PARTNERID>A123</PARTNERID>	Internal customer number in the sellers system (OM).	H.SOLD_TO_ORG_ID		

Table 7-10 Acknowledge_PO OAG Message Map Details

OAG Element	Description /Comment	Oracle OM Table /Column	OAG Req Y/N	Code Conversion Needed?
<PARTNERIDX>A123</PARTNERIDX>	Unique identifier for the sold to partner. Value should be EDI Location code for the sold-to partner. Derived from OE_XML_PROCESS_UTIL.GET_SOLD_TO_EDI_LOC.			
<PARTNERTYPE>SoldTo</PARTNERTYPE>	Partner Type. Value should be 'Sold To'			
</PARTNER>				
<PARTNER>			N	
<NAME index="1">CPAG</NAME>	Trading partner name. In this case Suppliers name.	H.SHIP_FROM_ORG		
<ONETIME>0</ONETIME>	0			
<PARTNERID>A123</PARTNERID>	Supplier number This will be the internal location number for the seller.	H.SHIP_FROM_ORG_ID		
<PARTNERIDX>A123</PARTNERIDX>	Unique identifier for the supplier. EDI location code for the supplier. Setup in Inventory/setup/Org/Location. This code is used by the buyer (PO) to identify the supplier.			
<PARTNERTYPE>Supplier</PARTNERTYPE>	Partner type. The value should be 'Supplier'			
</PARTNER>				
<POTERM>			N	
<TERMID>XXX</TERMID>	Payment term code.	H.PAYMENT_TERM_ID	Y	

Table 7–10 Acknowledge_PO OAG Message Map Details

OAG Element	Description /Comment	Oracle OM Table /Column	OAG Req Y/N	Code Conversion Needed?
<DESCRIPTN>XXX</DESCRIPTI ON>	Description of the payment term		N	
</POTERM>				
</POORDERHDR>				
<POORDERLIN>				
<QUANTITY qualifier="ORDERED">	This is the original ordered Qty. Required by OAG but will be ignored by the Purchasing. This qty would need to be stored from the Original PO (Process_PO inbound). Derived Using Convert to OAG Quantity.	L.ORDERED_ QUANTITY	Y	
<VALUE>1</VALUE>			Y	
<NUMOFDEC>0</NUMOFDEC>			Y	
<SIGN>+</SIGN>			Y	
<UOM>EACH</UOM>			Y	
</QUANTITY>			Y	
<OPERAMT qualifier="UNIT" type="T">	Changed Unit selling price. This the new price if any changes are made to the price. If price is not changed this can be left null. Derived using Convert to OAG Operating Amount.	L.UNIT_ SELLING_ PRICE	N	
<VALUE>6500000</VALUE>			Y	
<NUMOFDEC>2</NUMOFDEC>			Y	
<SIGN>+</SIGN>			Y	
<CURRENCY>USD</CURRENCY >			Y	
<UOMVALUE>1</UOMVALUE>			Y	

Table 7-10 Acknowledge_PO OAG Message Map Details

OAG Element	Description /Comment	Oracle OM Table /Column	OAG Req Y/N	Code Conversion Needed?
<UOMNUMDEC>0</UOMNUMDEC>			Y	
<UOM>EACH</UOM>			Y	
</OPERAMT>				
<POLINENUM>1</POLINENUM>	Po Line number of the original PO	L.ORIG_SYS_LINE_REF	Y	
<ITEMX>XXXXXX</ITEMX>	Customer Item number.	L.CUSTOMER_ITEM	N	
<USERAREA>				
<USERITEMDESCRIPTN>item desc </USERITEMDESCRIPTN>		L.USER_ITEM_DESCRIPTION		
</USERAREA>				
<ACKLINE>				
<ACKCODE>1</ACKCODE>	Status Code * See Notes	L.FIRST_ACK_CODE	Y	
<NOTES index="1">This is a test message</NOTES>	Reason Code * See Notes		N	
</ACKLINE>				
<POLINESCHD>				
<DATETIME qualifier="NEEDELV">	Needed by delivery date as from the Original PO. Required by OAG and will be ignored by Purchasing. This should be mapped to the request date in OM. Derived using Convert To OAG Datetime.	L.REQUEST_DATE	Y	
<YEAR>1996</YEAR>			Y	
<MONTH>07</MONTH>			Y	
<DAY>01</DAY>			Y	
<HOUR>23</HOUR>			Y	

Table 7–10 Acknowledge_PO OAG Message Map Details

OAG Element	Description /Comment	Oracle OM Table /Column	OAG Req Y/N	Code Conversion Needed?
<MINUTE>59</MINUTE>			Y	
<SECOND>59</SECOND>			Y	
<SUBSECOND>0000</SUBSECOND>			Y	
</DATETIME>				
<QUANTITY qualifier="ORDERED">	Original Order Quantity. This qty would need to be stored from the Original PO (Process_PO inbound). Derived using Convert To OAG Quantity.	L.ORDERED_QUANTITY	Y	
<VALUE>1</VALUE>			Y	
<NUMOFDEC>0</NUMOFDEC>			Y	
<SIGN>+</SIGN>			Y	
<UOM>EACH</UOM>			Y	Y
</QUANTITY>				
<PSCLINENUM>1</PSCLINENUM>	Original Schedule shipment line number from the original PO.	L.ORIG_SYS_SHIPMENT_REF	N	
</USERAREA>				
<ACKCODE>X</ACKCODE>	* See Notes	L.FIRST_ACK_CODE	N	
<DATETIME qualifier = "PROMDELV">	Promised delivery date from OM. Derived using Convert To OAG Datetime.	L.SCHEDULE_ARRIVAL_DATE	N	
<YEAR>1996</YEAR>			Y	
<MONTH>07</MONTH>			Y	
<DAY>01</DAY>			Y	
<HOUR>23</HOUR>			Y	
<MINUTE>59</MINUTE>			Y	

Table 7-10 Acknowledge_PO OAG Message Map Details

OAG Element	Description /Comment	Oracle OM Table /Column	OAG Req Y/N	Code Conversion Needed?
<SECOND>59</SECOND>			Y	
<SUBSECOND>0000</SUBSECON D>			Y	
</DATETIME>				
<SALESORDID>123</SALESORDI D>	Sales Order number for this order line.	H.ORDER_ NUMBER	N	
<SOLINENUM>1</SOLINENUM >	Sales order line number * See Notes	L.LINE_ NUMBER	N	
</USERAREA>				
</POLINESCHD>				
</POORDERLIN>				
</ACKNOWLEDGE_PO>				
</DATAAREA>				
</ACKNOWLEDGE_PO_008>				
** XMLG – XML gateway				

Table 7-11 Message Map Notes

Name	Description
POHEADER / PORELASE	This is PO release number which Order Management doesn't store. For the interim , the solution is provided by Oracle Application Server 10g where its concatenated to the PO number and sent in as a Customer PO number. On the Purchasing side it will be separated again by Oracle Application Server 10g and populated in to the release number columns for Purchasing. Order Management does not need to do any action item related to this at the moment but it remains as an open issue for the future enhancement where Order Management will need to handle Blanket POs with the release number stored.
ACKHEADER / ACKCODE	Status code – <ul style="list-style-type: none"> • 0- Accepted • 2- Reject
ACKHEADER / NOTES	Reason code – Depending on the Ackcode populate the text – Accept or Reject When ACK PO is used as a confirmation for Change PO or Cancel_PO the Reason code is not required.
POLINE / ACKLINE / ACKCODE	Status Code – <ul style="list-style-type: none"> • Accept if all the shipment lines are accepted (even with changes) • Rejected if any of the shipment lines is rejected.
POLINE / POLNSTATUS	Line Status – <ul style="list-style-type: none"> • Open - PO line is open to receiving. • Closed - PO line has completed normally. It is no longer available to receiving. • Cancelled – PO line has completed abnormally or has been deleted. The PO is no longer open to receiving.
USERAREA / SOLINENUM	This element contains sales order line number. Its been added here to allow 3A4 transaction to create multiple Sales orders from one PO.

Table 7–11 Message Map Notes

Name	Description
POLINESCHD/ USERAREA/ ACKLINE / ACKCODE	Status code – <ul style="list-style-type: none"> • 0- Accept • 2- Reject
POLINESCHD / DATETIME(NEEDEDELV)	This date element will contain the request_date.
POLINESCHD / QUANTITY (ORDERED)	This quantity element will contain the ordered_quantity.

Message Map 'ONT_3A4A_OAG72_OUT.xgm'

The message map 'ONT_3A4A_OAG72_OUT' is created using the Oracle XML Gateway Message Designer tool. Please refer to [Table 7–8, "Message Map"](#) for the detailed map analysis.

The source DTD used is 004_acknowledge_po_008.dtd, revision 7.2.1 of the Open Application Group. The other associated external reference DTD files are;

oagis_domains.dtd

oagis_resources.dtd

oagis_fields.dtd

oagis_segments.dtd

All the DTD's will be checked in ONT source area under \$ont/xml/oag72

The target for the Outbound XML Message are the Order Management Acknowledgment tables OE_HEADER_ACKS & OE_LINE_ACKS.

The ACKNOWLEDGE_PO_008 DTD is a three level hierarchy with Order Lines split into one or more Shipment Lines. The Order Management architecture is however a two level one with the order header and one or more lines. The message map will expand the two level structure of the Order Management tables to three level XML Message.

Please refer to [Table 7–8, "Message Map"](#) for a detail analysis of the elements mapped, actions and derivation rules used by the Message Map.

Both the message map created using the Message Designer and its associated DTD's are stored in the database. The following Java programs are available to

Load/Delete Maps or Load/Delete DTD's into/from the XML Gateway repository. Please refer to the *Oracle XML Gateway Manual* for more information.

Load/Delete Maps, Load/Delete DTD's

Note: The following process is used only for customizations.

1. `java LoadMap <DB username> <DB password> <Hostname>:<Port>:<SID>
<mymap.xgm>`

Example: `java oracle.apps.ecx.loader.LoadMap apps apps
ap505dbs:1521:dev115 ONT_3A4A_OAG72_OUT_PO.xgm`

2. `java DeleteMap <DB username> <DB password> <Hostname>:<Port>:<SID>
<mapname>`

Example: `java oracle.apps.ecx.loader.DeleteMap apps apps
ap505dbs:1521:dev115 ONT_3A4A_OAG72_OUT_PO.xgm`

The Message Map is a .xgm file which will be stored in the Order Management Source area under \$ont/patch/115/map/ ONT_3A4A_OAG72_OUT_PO.xgm

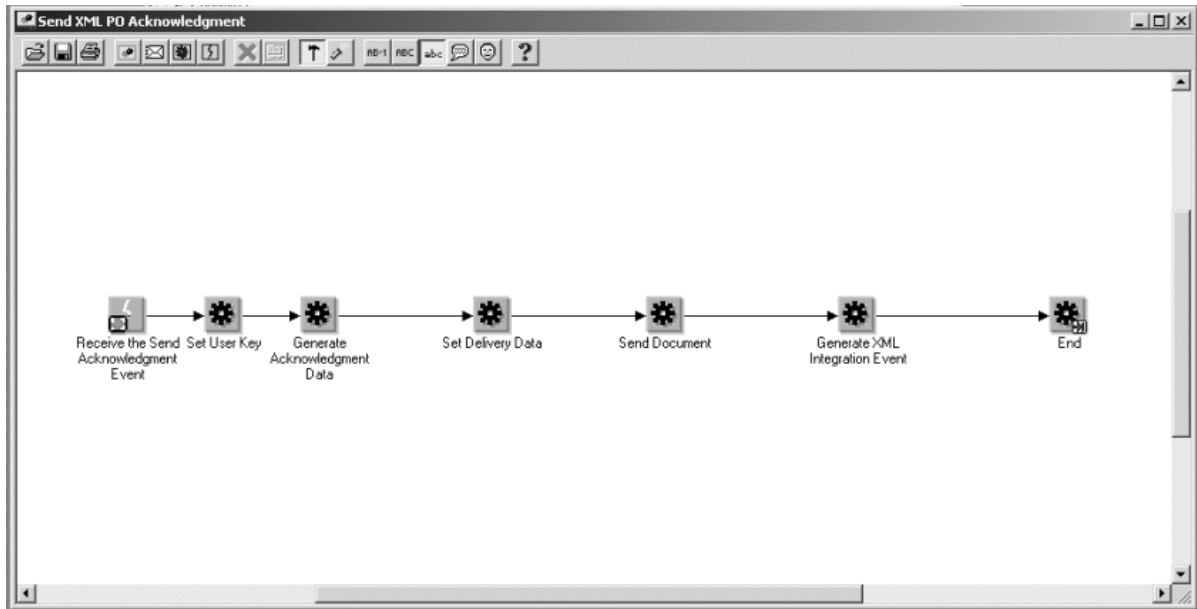
Note: Maps and DTD's must be kept in sync between the Message Designer and the XML Gateway repository. When in doubt, always reload the map and DTD as a pair.

Pre-Defined Workflow Event

If the Profile Option OM: Run Order Import for XML is set to 'Synchronous,' a pre-defined Workflow event is seeded to run Order Import if the Profile Option OM: Process Transaction is set to 'ONLINE.' This Workflow subscribes to the event raised by the Post Process of the XML Gateway. As part of the Workflow Process, it will run Order Import for the XML Message, which is currently being consumed. After the Run Order Import functional finishes (Successfully or With Error), Process to Raise the Event is Called.

Seeded Workflow

Figure 7–4 Send XML PO Acknowledgment



Show_SalesOrder

Show_salesorder Overview

The Show_SalesOrder message is used by the seller to send order status information, by line item for an order to a buyer. This outbound message carries the status of the order line.

Major Features

Major features of this transaction include:

Generate and send Show_salesorder message out

Oracle Order Management generates the outbound Show SO message at various trigger points described in following sections. The Show SO message contains status information about the order at the line item level. For the details of this message and mappings refer to [Table 7-14, "Message Map"](#).

Periodical

The Show_salesorder message can be sent periodically depending on the period set in the trading partner agreement between the buyer and the seller. Examples of this would be daily, weekly, or at a particular time of the day.

Order Management provides the concurrent program called by the triggering mechanism, scheduling the process of the concurrent program to generate the show_salesorder message. This API has following parameters:

1. **Trading partner id (EDI Location code)** – Order Management created the Show_salesorder message for this trading partner.
2. **Open Orders Only** – This is the order status that Order Management creates the show_salesorder message for. It chooses between open or closed orders. If Open Orders Only is set to Yes, only open orders will be picked up. If Open Orders Only is set to No, only closed orders will be picked up.
3. **Closed for Days** – Order Management is currently not supporting the use of this parameter. This would be number of days. Order Management would create a show_salesorder message for all closed orders, which are closed within last 'N' number of days.
4. **Sales Order No From** – Starting at sales order number.
5. **Sales Order No To** – End at sales order number.

6. **Sales Order date from-** From sales order creation date.
7. **Sales Order date to** – To sales order date.
8. **Customer PO No From** – Starting at Customer PO number.
9. **Customer PO No To** – End at Customer PO number.

Order Changes

Changes in the attributes of the order that occur during order processing. This could be shipping or scheduling for example. If changes to any of these attributes occur in a booked order, the show SO message is sent.

Changes in following attributes are supported at this point:

- Unit selling Price,
- Ordered Qty
- Scheduled arrival date
- Shipped Qty
- Scheduled Ship Date

Any time any of these attributes change for a booked order Order Management generates a Show_Salesorder message.

Adding a new line to a booked order also generates a Show_Salesorder message. Consequently, splitting a line also generates a Show_Salesorder message, due to the quantity change and the generation of the new line.

- Status change due to business events – Booking the order, shipping, or scheduling of lines - the Show_SalesOrder is generated when the order is booked. It will also, due to detection of the corresponding attribute changes, be generated for scheduling and shipping of lines.

Business Scenarios and Process Flow

OUTBOUND

1. The Show_SalesOrder message is created by the Order Management in response to various trigger points. The triggering of show_salesorder could be result of any of the following scenarios or combination of them:
 - Determined from the period set by the customer, when the Show_SalesOrder is triggered

- The order is booked
- If the Unit selling price, Schedule delivery date, or ordered quantity changed systematically (for booked orders)
- The line is scheduled
- The line is shipped
- The line is cancelled
- The line is added to a booked order

The XML Gateway forms the SHOW_SO message.

Message Map

Show_SalesOrder Message Map

Key: H => OE_HEADER_ACKS, L => OE_LINE_ACKS

Table 7-12 Show_SalesOrder Message Map

OAG Element	Desc/Comment	Oracle OM Table/Column	OAG reqd Y/N	Code Conversion Needed Y/N
<SHOW_SALESORDER_006>				
<CNTROLAREA>			Y	
<BSR>				
<VERB>SHOW</VERB>	XMLG			
<NOUN>SALESORDER</NOUN>	XMLG			
<REVISION>006</REVISION>	XMLG			
</BSR>				
<SENDER>			N	
<LOGICALID>XGRB1109</LOGICALID>	XMLG			
<COMPONENT>OM</COMPONENT>	XMLG			
<TASK>SO</TASK>	XMLG			

Table 7–12 Show_SalesOrder Message Map

OAG Element	Desc/Comment	Oracle OM Table/Column	OAG reqd Y/N	Code Conversion Needed Y/N
<REFERENCEID>9534223449</REFERENCEID>	XMLG			
<CONFIRMATION>1</CONFIRMATION>	XMLG			
<LANGUAGE>ENG</LANGUAGE>	XMLG			
<CODEPAGE>CP001001</CODEPAGE>	XMLG			
<AUTHID>SMITHJ</AUTHID>	XMLG			
</SENDER>				
<DATETIME qualifier="CREATION"> <YEAR>1998</YEAR>	XMLG		Y	
<MONTH>12</MONTH>				
<DAY>29</DAY>				
<HOUR>00</HOUR>				
<MINUTE>25</MINUTE>				
<SECOND>45</SECOND>				
<SUBSECOND>0000</SUBSECOND>				
</DATETIME>				
</CNTROLAREA>				
<DATAAREA>				
<SHOW_SALESORDER>				
<SOHEADER>				

Table 7-12 Show_SalesOrder Message Map

OAG Element	Desc/Comment	Oracle OM Table/Column	OAG reqd Y/N	Code Conversion Needed Y/N
<DATETIME qualifier="DOCUMENT"> <YEAR>1998</YEAR>	Date time stamp of sales order creation. Derived using Convert To OAG Datetime	H.ORDERED_DATE		
<MONTH>12</MONTH>				
<DAY>29</DAY>				
<HOUR>00</HOUR>				
<MINUTE>25</MINUTE>				
<SECOND>45</SECOND>				
<SUBSECOND>0000</SUBSECOND>				
</DATETIME>				
<SALESORDID>S0001</SALESORDID>	Sales order number	H.ORDER_NUMBER	Y	
<POID>123<POID>	Customer Po number	H.ORIG_SYS_DOCUMENT_REF	N	
<SOSTATUS>XXX<SOSTATUS>	Sales order status (see notes)	H.FIRST_ACK_CODE	N	
<PARTNER>			N	
<NAME index="1">ACME</NAME>	Trading partner name . In this case Buyers name. Derived from OE_XML_PROCESS_UTIL.GET_SOLD_TO_EDI_LOC			
<ONETIME>0</ONETIME>	0			
<PARTNRID>AC01234</PARTNRID>	Internal customer number in the sellers system (OM)	H.SOLD_TO_ORG_ID		

Table 7-12 Show_SalesOrder Message Map

OAG Element	Desc/Comment	Oracle OM Table/Column	OAG reqd Y/N	Code Conversion Needed Y/N
<PARTNRIDX>1234</PARTNRIDX>	Unique identifier for the sold to partner. Value should be EDI Location code for the sold-to partner. Derived from OE_XML_PROCESS_UTIL.GET_SOLD_TO_EDI_LOC			
<PARTNRTYPE>sold-to</PARTNRTYPE>	Partner Type. Value should be 'Sold To'			
</PARTNER>				
<PARTNER>			N	
<NAME index="1">ACME</NAME>	Trading partner name. In this case Suppliers name			
<ONETIME>0</ONETIME>	0			
<PARTNRID>AC01234</PARTNRID>	Supplier number This will be the internal location number for the seller.	H.SHIP_FROM_ORG_ID		
<PARTNRIDX>1234</PARTNRIDX>	Unique identifier for the supplier. EDI location code for the supplier. Setup in Inventory/setup/Org/Location. This code is used by the buyer (PO) to identify the supplier.	ECX_OAG_CONTROLAREA_TP_VSOURCE_TP_LOCATION_CODE (XMLG)		
<PARTNRTYPE>Supplier</PARTNRTYPE>	Partner type. The value should be 'Supplier'			

Table 7–12 Show_SalesOrder Message Map

OAG Element	Desc/Comment	Oracle OM Table/Column	OAG reqd Y/N	Code Conversion Needed Y/N
</PARTNER>				
<SALESINFO>			N	
<SALESPERSN>BOB</SALESPERSN>	Sales person name. Derived from OE_XML_PROCESS_UTIL.GET_SALES_PERSON			
</SALESINFO>				
</SOHEADER>				
<SOLINE>				
<OPERAMT qualifier="UNIT" type="T">	Unit selling price. Derived using Convert to OAG Operating Amount.	L.UNIT_SELLING_PRICE	N	
<VALUE>6500000</VALUE>				
<NUMOFDEC>2</NUMOFDEC>				
<SIGN>+</SIGN>				
<CURRENCY>USD</CURRENCY>				
<UOMVALUE>1</UOMVALUE>				
<UOMNUMDEC>0</UOMNUMDEC>				
<UOM>EACH</UOM>				
</OPERAMT>				
<SOLINENUM>1</SOLINENUM>	Sales order line number	L.LINE_NUMBER	Y	
<ITEMX>XXX</ITEMX>	Buyer's item number	L.CUSTOMER_ITEM	N	
<SOLNSTATUS>OPEN</SOLNSTATUS>	Sales Order Line status (see notes)	L.FIRST_ACK_CODE	N	

Table 7–12 Show_SalesOrder Message Map

OAG Element	Desc/Comment	Oracle OM Table/Column	OAG reqd Y/N	Code Conversion Needed Y/N
<USERAREA>				
<USERITEMDESCRIPTN>itemdesc</USERITEMDESCRIPTN>		L.USER_ITEM_DESCRIPTION		
</USERAREA>				
<SOSCHEDULE>				
<DATETIME qualifier="DELIVSCHED"> <YEAR>1998</YEAR>	Date time on which the goods are scheduled to be delivered. Derived using Convert To OAG Datetime	L.SCHEDULE_ARRIVAL_DATE	N	
<MONTH>12</MONTH>				
<DAY>29</DAY>				
<HOUR>00</HOUR>				
<MINUTE>25</MINUTE>				
<SECOND>45</SECOND>				
<SUBSECOND>0000</SUBSECOND>				
</DATETIME>				
<DATETIME qualifier="NEEDELV"> <YEAR>1998</YEAR>	Needed by delivery date as from the Original PO. This should be mapped to the request date in OM. Derived using Convert To OAG Datetime	L.REQUEST_DATE	N	
<MONTH>12</MONTH>				
<DAY>29</DAY>				
<HOUR>00</HOUR>				
<MINUTE>25</MINUTE>				
<SECOND>45</SECOND>				

Table 7-12 Show_SalesOrder Message Map

OAG Element	Desc/Comment	Oracle OM Table/Column	OAG reqd Y/N	Code Conversion Needed Y/N
<SUBSECOND>0000</SUBSECON D>				
</DATETIME>				
<QUANTITY qualifier="ORDERED">	Quantity ordered. Derived using Convert To OAG Quantity.	L.ORDERED_ QUANTITY	N	
<VALUE>1</VALUE>				
<NUMOFDEC>0</NUMOFDEC>				
<SIGN>+</SIGN>				
<UOM>EACH</UOM>				
</QUANTITY>				Y
<POLINENUM>123<POLINENUM>	Customer po line number	L.ORIG_SYS_ LINE_REF	N	
<PSCLINENUM>123<PSCLINENU M>	Customer PO schedule line number.	L.ORIG_SYS_ SHIPMENT_REF	N	
<SOSLINENUM>123<SOSLINENU M>	Schedule line number.	L.SHIPMENT_ NUMBER	N	
</SOSCHEDULE>				
</SOLINE>				
</SHOW_SALESORDER>				
</DATAAREA>				
</SHOW_SALESORDER_006>				

Table 7-13 Message Map Notes

SO Status	SO Line Status
Open- Any status other than the following ones in Order Management.	Open- Any status other than the following ones in Order Management.
Closed- Closed status in Order Management	Shipped – Shipped status in Order Management
Cancelled- Cancelled status in Order Management	Closed- Closed status in Order Management
	Cancelled- Cancelled status in Order Management

Message Map

Table 7-14 Message Map

Name	Description
Message Map Name:	ONT_3A6_OAG72_OUT_SSO
Direction:	Outbound
(Internal) Transaction Type:	ONT
(Internal) Transaction Subtype:	SSO
External Transaction Type:	SO
External Transaction Subtype:	SHOW
DTD Directory:	xml/oag72
Map Directory:	patch/115/xml/US
Message Maps XGM File Name:	ONT_3A6_OAG72_OUT_SSO.xgm
Standard:	OAG
Release:	7.2
Format:	DTD
DTD Name:	SHOW_SALESORDER_006

Table 7–15 Workflow Event Setup

Detail	Value
Event Name	oracle.apps.ont.oi.show_so.create
Event Description	Event for Outbound Show_SalesOrder
Subscription	R_SHOW_SALES_ORDER
Subscription Description	Event subscription for the oracle.apps.ont.oi.show_so.create event for Outbound Show_SalesOrder.

Message Set Up

To implement a message with a trading partner, use the XML Gateway message set up to define the trading partner or hub, code conversion values, and internal to external transaction name cross references. In addition, you may identify the XML Gateway system administrator to relate system or process errors.

Message Map ‘ONT_3A6_OAG72_OUT_SSO.xgm’

The message map ONT_3A6_OAG72_OUT_SSO is created using the Oracle XML Gateway Message Designer tool. Please refer to [Table 7–14, "Message Map"](#) for the detailed map analysis.

The source DTD used is 091_show_salesorder_006.dtd, revision 7.2.1 of the Open Application Group. The other associated external reference DTD files are:

- oagis_domains.dtd
- oagis_resources.dtd
- oagis_fields.dtd
- oagis_segments.dtd

All the DTD’s are checked in ONT source area under \$ont/xml/oag72.

The Source for the Outbound XML Show SO Message is the Order Management Acknowledgment tables OE_HEADER_ACKS & OE_LINE_ACKS.

The 091_show_salesorder_006 DTD is a three level hierarchy with order lines split into one or more shipment lines. The Order Management architecture is however a two level one with the order header and one or more lines. The message map will expand the two level structure of the Order Management tables to a three level XML Message.

Please refer to [Table 7-14, "Message Map"](#) for a detail analysis of the elements mapped, actions and derivation rules used by the Message Map.

Both the message map created using the Message Designer and its associated DTD's are stored in the database. The following Java programs are available to Load/Delete Maps or Load/Delete DTD's into/from the XML Gateway repository. Make sure you chenv to your development environment and execute "source /ecxdev/ecx/utills/ javaenv_wfidc.sh" to set up your Java environment.

Load/Delete Maps, Load/Delete DTD's

Note: The following process is used only for customizations.

1. 1. java LoadMap <DB username> <DB password> <Hostname>:<Port>:<SID>
<mymap.xgm>

Example: java oracle.apps.ecx.loader.LoadMap apps apps
ap505dbs:1521:dev115 ONT_3A6_OAG72_OUT_SSO.xgm

2. 2. java DeleteMap <DB username> <DB password> <Hostname>:<Port>:<SID>
<mapname>

Example: java oracle.apps.ecx.loader.DeleteMap apps apps
ap505dbs:1521:dev115 ONT_3A6_OAG72_OUT_SSO.xgm

The Message Map is a .xgm file which will be stored in the Order Management Source area under \$ont/patch/115/map/ ONT_3A6_OAG72_OUT_SSO.xgm

Note: Maps and DTD's must be kept in sync between the Message Designer and the XML Gateway repository. When in doubt, always reload the map and DTD as a pair.

Pre-Defined Workflow: Order Management Show_SalesOrder

A pre-defined Workflow Process Show_SalesOrder is provided as part of OESO Workflow, which generates the Outbound Show_SalesOrder.

This Workflow receives the event to Show_SalesOrder, then generates the data using the Acknowledgment tables, and then use XML Gateway's SEND DOCUMENT function to deliver the Show_SalesOrder XML to the trading partner.

Show_SalesOrder - Concurrent Program

Use these steps to create the following:

Table 7–16 Show_SalesOrder - Concurrent Program

Step	Description
1	To Create Executable for Show_SalesOrder concurrent program:
2	To Create a Concurrent program for Show Sales Order concurrent request:
3	To add parameters to the program:
4	To add the concurrent program to the Order Management Concurrent Programs Request Group:
5	Value sets for the concurrent program:

To Create Executable for Show_SalesOrder concurrent program:

1. From Sys Admin / Application Developer Responsibility choose: Concurrent > Program > Executable.
2. Create new executable **Show SalesOrder**:
Executable: Show SalesOrder
Short Name: OEXSSOCP
Application: Oracle Order Management
Description: Show Sales Order Concurrent Program Executable
Execution Method: PL/SQL Stored Procedure
Execution File: OE_Acknowledgment_Pub.Process_SSO_CONC_PGM
3. Save your work.

To Create a Concurrent program for Show Sales Order concurrent request:

1. From Sys Admin / Application Developer Responsibility choose: Concurrent ->Program -> Define.
2. Create new program **Show Sales Order**:
Program: Show Sales Order
Short Name: OEXSSOCP
Application: Oracle Order Management

Description: Show Sales Order Concurrent program for Periodical Support of Show_SalesOrder

Under the Executable section:

Name: OEXSSOCP

Method: PL/SQL Stored Procedure

Note: This concurrent program is seeded as disabled. It will be enabled for Pack I or above.

3. Save your work.

To add parameters to the program:

1. Click Parameters and add the following:

Table 7–17 Parameters to Add

Seq	Parameters	Description	Value Set	Range	Prompt	Default value	Disp Size	Req	Disp
10	P_Customer_Id	Customer_id for the Trading Partner Location Code selected	ONT_TP_LOCATION_CODE		Trading Partner Location Code		40	Y	Y
20	P_Open_Orders_Only	Open Orders Only	Yes_No		Open Orders Only	SQL Statement: select meaning from fnd_lookups where lookup_type='YES_NO' and lookup_code='Y'	20	N	Y
30	P_Closed_for_days	Closed for days	OM: Number		Closed for days		22	N	N

Table 7-17 Parameters to Add

Seq	Parameters	Description	Value Set	Range	Prompt	Default value	Disp Size	Req	Disp
40	P_So_Number_From	Sales Order Number From	ONT_CUST_ORDER_NUMBER	Low	Sales Order Number From		22	N	Y
50	P_So_Number_To	Sales Order Number To	ONT_CUST_ORDER_NUMBER	High	Sales Order Number To	SQL Statement: Select :\$FLEX\$.ONT_CUST_ORDER_NUMBER From Dual	22	N	Y
60	P_So_Date_From	Sales Order Date From	FND_STANDAR D_DATE	Low	Sales Order Date From		11	N	Y
70	P_So_Date_To	Sales Order Date To	FND_STANDAR D_DATE	High	Sales Order Date To	SQL Statement: select fnd_date.date_to_ chardate(fnd_date.canonical_to_date(:\$FLEX\$.FND_STANDARD_DATE)) from DUAL	11	N	Y
80	P_Customer_PO_No_From	Customer PO Number From	ONT_CUST_PO_NUMBER	Low	Customer PO Number From		50	N	Y
90	P_Customer_PO_No_To	Customer PO Number To	ONT_CUST_PO_NUMBER	High	Customer PO Number To	SQL Statement: Select :\$FLEX\$.ONT_CUST_PO_NUMBER From Dual	50	N	Y

1. To add the concurrent program to the Order Management Concurrent Programs Request Group:
2. From Sys Admin Responsibility choose: Security -> Responsibility-> Request.
3. Query for: OM Concurrent Programs.
4. Add a new record:
Requests: Program
Name: Show Sales Order
Application: Oracle Order Management
5. Save your work.

Value sets for the concurrent program:

1. From Sys Admin / Application Developer Responsibility choose: Application -> Validation -> Sets.

Valueset Name: ONT_TP_LOCATION_CODE

Description: EDI Location Codes

Format Type: Char

Maximum Size: 40

Validation Type: Table

Table Application: Oracle Receivables

Table Name: HZ_CUST_ACCT_SITES hz

Value: hz.ece_tp_location_code CHAR(40)

ID: hz.cust_account_id NUMBER(15)

Where/Order By:

```
where hz.ece_tp_location_code is not null
and exists (select d.source_tp_location_code
from ecx_tp_headers h
,ecx_tp_details d
where h.tp_header_id = d.tp_header_id
and d.source_tp_location_code = hz.ece_tp_location_code
and h.party_id = hz.cust_account_id)
```

Valueset Name: ONT_CUST_ORDER_NUMBER

Description: Order Numbers restricted by sold to org

Format Type: Number

Maximum Size: 22

Validation Type: Table

Table Application: Oracle Order Management

Table Name: OE_ORDER_HEADERS

Value: order_number Number(22)

ID: order_number Number(22)

Where/Order By:

```
where sold_to_org_id = :$FLEX$.ONT_TP_LOCATION_CODE  
and sold_to_org_id is not null  
ORDER BY order_number DESC
```

Valueset Name: ONT_CUST_PO_NUMBER

Description: Customer Po Numbers restricted by sold to org

Format Type: Char

Maximum Size: 50

Validation Type: Table

Table Application: Oracle Order Management

Table Name: OE_ORDER_HEADERS

Value: CUST_PO_NUMBER CHAR2(50)

ID: CUST_PO_NUMBER CHAR2(50)

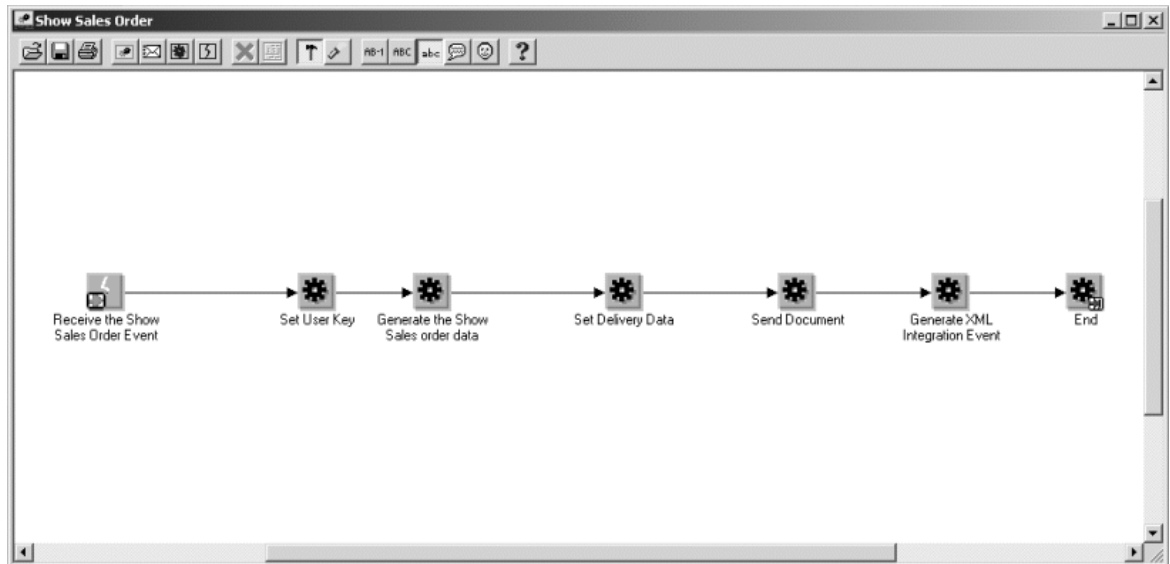
Where/Order By:

```
WHERE sold_to_org_id = :$FLEX$.ONT_TP_LOCATION_CODE  
and CUST_PO_NUMBER IS NOT NULL  
ORDER BY cust_po_number DESC
```

2. Save your work.

Seeded Workflow

Figure 7-5 Show_SalesOrder Workflow



Change_salesorder

Overview

Using the Change_salesorder, the seller can:

- Acknowledge acceptance of a pending purchase order line
- Initiate changes to a purchase order

You can set up a requirement for a response document for seller-initiated changes within the Trading Partner Agreement. For example, the buyer must respond with PIP3A8, Request Purchase Order Change, to accept the proposed changes. If the changes are not accepted, PIP3A9, request Purchase Order Cancel must be sent by the buyer.

Both Order Import of closed orders and HVOP are currently not supported for XML orders.

Major Features

Generate a Change_salesorder Message Outbound From Order Management

Oracle Order management generates a Change_salesorder message when the sales order is changed either manually or by the process_order API. The change_salesorder may also be generated in case of change of status of the sales order from Pending to Accept or Reject, and notify to the buyer of the change.

The following are the assumed definitions for Accept, Reject, and Pending:

- Accept - Booked status
- Reject - Order cannot be Entered and is rejected.
- Pending - Entered status.

Note: While the RosettaNet standard allows Pending lines to go to Reject, in Order Management, this implies that 3A7 would be sent when an Entered order is rejected. However, since currently all of the 3A7 (and 3A6) processing requires an Order to be Booked, triggering 3A7 on the transition from Pending to Reject is not supported.

Place an Order on Hold and Clear Holds

When changes occur, a Change_salesorder message notifies the buyer to put the order on hold. This hold can be cleared as follows:

Changes are made to the order after booking and the order is put on hold before sending 3A7 to the buyer

1. Purchasing Accepts Proposed Changes

Purchasing sends a Request Purchase Order Change (3A8) transaction verifying the acceptance of the proposed changes.

The 3A8 Response is consumed and the order line attributes mentioned below are compared. If any of these attributes do not match the current data on the order line, the 3A8 Response is rejected.

If the attributes do match, then the 3A8 Response is accepted and the hold is removed. The order progresses through the processing.

NOTE: If there is a 3A8 that comes in as an independent request before 3A7 response then the hold is removed and the 3A8 request will be acted upon. If there is a subsequent 3A8 that arrives as a response for the same 3A7, it is processed and depending if the response has values that match the order attributes mentioned below or it will be accepted or rejected.

2. Purchasing Rejects Proposed Changes

If purchasing rejects the 3A7 proposed changes, then purchasing sends a Request Purchase Order Cancel (3A9) transaction, canceling the original purchase order.

Order Management receives and consumes the Request Purchase Order Cancel (3A9) and cancels the purchase order within Order Management.

If Cancel PO is received for the lines on hold the lines are removed from hold and cancelled.

If purchasing rejects the 3A7 proposed changes, then purchasing can also send back a 3A8 Response using the ACKCODE tag with a value of 2 to communicate 'Reject.' Note that no value or a value of 0 in this tag implies 'Accept.'

If the ACKCODE tag has a value of 2 at either Line or Shipment level for a Change PO Response, Order Management does not consume that line. However, Order

Management does process the Accepted lines (i.e. ACKCODE tag has no value or a value of 2).

If the ACKCODE tag has a value of 2 at Header level, Order Management doesn't consume the entire document.

This feature of Accepting and Rejecting changes is not required - if the ACKCODE tag is not populated, Order Management will automatically process every line to check if the attributes match the values currently on the Sales Order.

For the implementations where the seller chooses not to require a response for 3A7 there is a setup to state that the response is not required for the 3A7 created by the seller. In this case the order is put on hold. A profile option 'OM: Change SO Response Required' controls the response at site level with values Y or N.

Ability to Match Incoming Change_PO (3A8) or Cancel_PO (3A9) to The Original Change_salesorder (3A7)

Order Management must be able to match the incoming 3A8 (which can be a response to the 3A7 originated by Order Management) or 3A9. The match can be done using the following two criteria together:

- Customer purchase order number + Trading partner (buyer)id
- Inbound 3A8 will have a flag, that indicates if the specific 3A8 is a response (to 3A7) or is a new request. This flag separates the responses from the independent 3A8 that may occur for the same order for which 3A7 was generated.
- 3A9 is processed and results in the removal of the hold and cancellation (partial or full) of the lines.

Integration With the Collaboration History Module (CLN)

The Collaboration History module records the transaction history when the trading partners exchange messages. Order Management incorporates the API calls during the processing of the XML messages.

The following point within the transaction dialog is recorded in the CLN module:

- Change sales order message sent
- One or more buyer rejected lines are received in the 3A8 Change PO response to an earlier 3A7 Change Sales Order message.
- 3A8 Change PO response to an earlier 3A7 Change Sales Order message is rejected by the buyer at the header level.

Message Map

3A7 Message Map

The message map is created using Oracle XML Gateway's message designer.

Map Name - ONT_3A7_OAG72_OUT_SO

DTD Name - 087_change_salesorder_008.dtd (Change Sales Order)

Note: Most data values in the message map are seeded.

Load/Delete Maps, Load/Delete DTD's

Note: The following process is used only for customizations.

1. `java LoadMap DB username> DB password> Hostname>:Port>:SID> mymap.xgm>`

Example: `java oracle.apps.ecx.loader.LoadMap apps apps ap505dbs:1521:dev115 ONT_3A7R_OAG72_OUT_SO.xgm`

2. `java DeleteMap DB username> DB password> Hostname>:Port>:SID> mapname>`

Example: `java oracle.apps.ecx.loader.DeleteMap apps apps ap505dbs:1521:dev115 ONT_3A7R_OAG72_OUT_SO.xgm`

The Message Map is a .xgm file which will be stored in the Order Management Source area under \$ont/patch/115/map/ ONT_3A7R_OAG72_OUT_SO.xgm.

Note: Maps and DTD's must be kept in sync between the Message Designer and the XML Gateway repository. When in doubt, always reload the map and DTD as a pair.

Transaction

The following is the seeded data for the 3A7 transaction:

- Transaction Type – ONT
- Transaction Sub Type – CSO

- External Transaction Type – SALESORDER
 - External Transaction Sub Type – CHANGE

Table 7–18 ONT_3A7_OAG72_OUT_SO

OAG	Comment/Description	OAG Reqd Y/N	OM table/Column
<CHANGE_SALESORDER_008>			
<CNTROLAREA>		Y	
<BSR>		Y	
<VERB>CHANGE</VERB>		Y	
<NOUN>SALESORDER</NOUN>		Y	
<REVISION>008</REVISION>		Y	
</BSR>		Y	
<SENDER>		Y	
<LOGICALID>XGRB1109</LOGICALID>	XMLG	Y	
<COMPONENT>SALES</COMPONENT>	XMLG	Y	
<TASK>OM</TASK>	XMLG	Y	
<REFERENCEID>9534223449</REFEREN CEID>	XMLG	Y	
<CONFIRMATION>1</CONFIRMATIO N>	XMLG	Y	
<LANGUAGE>ENG</LANGUAGE>	XMLG	Y	
<CODEPAGE>CP001001</CODEPAGE>	XMLG	Y	
<AUTHID>SMITHJ</AUTHID>	XMLG	Y	
</SENDER>		Y	
<DATETIME qualifier="CREATION">	XMLG	Y	
<YEAR>1998</YEAR>		Y	

Table 7-18 ONT_3A7_OAG72_OUT_SO

OAG	Comment/Description	OAG Reqd Y/N	OM table/Column
<MONTH>11</MONTH>		Y	
<DAY>21</DAY>		Y	
<HOUR>00</HOUR>		Y	
<MINUTE>25</MINUTE>		Y	
<SECOND>45</SECOND>		Y	
<SUBSECOND>0000</SUBSECOND>		Y	
<TIMEZONE>-0600</TIMEZONE>		Y	
</DATETIME>		Y	
</CNTROLAREA>		Y	
<DATAAREA>		Y	
<CHANGE_SALESORDER>		Y	
<SOHEADER>		Y	
<SALESORDID>S0001</SALESORDID>	Sales order number	Y	Oe_Header_ Acks.ORDER_ NUMBER
<SALESORG index="1">ORG12345</SALESORG>	Sales Organization code	Y	
<NOTES1>Invalid ID</NOTES1>	Accept or Reject? (See notes)	N	"Accepted"
<POID>1234</POID>	Customer PO number	Y	Oe_Header_ Acks.CUSTO MER_PO_ NUMBER
<SOSTATUS>Open</SOSTATUS>	Sales Order Status (see notes)	N	Oe_Header_ Acks.FIRST_ ACK_CODE
<USERAREA>		N	
<PORELEASE>1</PORELEASE>	PO release number. No map to OM column	N	
<REVISIONNUM>2</REVISIONNUM>	PO revision number. No map to OM column	N	

Table 7-18 ONT_3A7_OAG72_OUT_SO

OAG	Comment/Description	OAG Req'd Y/N	OM table/Column
<ORACLE.SUPPLIERDOCREF>123</ORACLE.SUPPLIERDOCREF>	Suppliers reference for the PO	N	Oe_header_Acks. ORIG_SYS_DOCUMENT_REF
</USERAREA>		N	
<PARTNER>		N	
<NAME index="1">ACME</NAME>	Name of Sold To Trading Partner	Y	Derived from Oe_Header_Acks.SOLD_TO_ORG_ID
<ONETIME>0</ONETIME>		Y	
<PARTNRID>YYYY</PARTNRID>	Unique ID for Sold To Trading Partner	Y	Oe_Header_Acks.SOLD_TO_ORG_ID
<PARTNRTYPE>SoldTo</PARTNRTYPE>		Y	
<PARTNRIDX>XXXX</PARTNRIDX>	EDI Location Code for Sold To Trading Partner	N	Derived from Oe_Header_Acks.SOLD_TO_ORG_ID
</PARTNER>			
<PARTNER>			
<NAME index="1">ACME</NAME>	Name of supplier		
<ONETIME>0</ONETIME>		N	
<PARTNRID>AC01234</PARTNRID>	Unique id for the supplier	Y	Oe_Header_Acks.SHIP_FROM_ORG_ID

Table 7-18 ONT_3A7_OAG72_OUT_SO

OAG	Comment/Description	OAG Reqd Y/N	OM table/Column
<PARTNRIDX>AC01234</PARTNRIDX>	Location code of the supplier	N	EcX_Oag_ ControlArea_ Tp_ V.SOURCE_ TP_ LOCATION_ CODE
<PARTNRTYPE>SUPPLIER</PARTNRTYPE>		Y	
</PARTNER>		Y	
</SOHEADER>		Y	
<SOLINE>		Y	
<OPERAMT qualifier="UNIT" type="T">	Unit price	N	Derived from Oe_Line_ Acks.UNIT_ SELLING_ PRICE
<VALUE>1</VALUE>		N	
<NUMOFDEC>0</NUMOFDEC>		N	
<SIGN>+</SIGN>		N	
<CURRENCY>USD</CURRENCY>			
<UOMVALUE>1</UOMVALUE>			
<UOMNUMDEC>0</UOMNUMDEC>		N	
<UOM>EACH</UOM>		Y	
</OPERAMT>		Y	
<QUANTITY qualifier="ORDERED">	Ordered quantity	Y	
<VALUE>1</VALUE>		Y	
<NUMOFDEC>0</NUMOFDEC>		Y	
<SIGN>+</SIGN>		Y	
<UOM>EACH</UOM>		Y	

Table 7-18 ONT_3A7_OAG72_OUT_SO

OAG	Comment/Description	OAG Reqd Y/N	OM table/Column
</QUANTITY>		Y	
<SOLINENUM>1</SOLINENUM>	Sales order line number		Oe_Line_Acks.LINE_NUMBER
<NOTES1/>	Accept or Reject? (See notes)	N	"Accepted"
<POLINENUM>1</POLINENUM>	Customer po line number	Y	Oe_Line_Ack.CUSTOMER_LINE_NUMBER
<SOLNSTATUS>Open</SOLNSTATUS>	Sales Order Line status (see notes)	Y	Oe_Line_Acks.FIRST_ACK_CODE
<ITEM>123456</ITEM>	Supplier Item number	Y	
<ITEMX>123456</ITEMX>	Buyer item number	Y	Oe_Line_Acks.CUSTOMER_ITEM
<USERAREA>			
<ORACLE.SUPPLIERLINEREF>1</ORACLE.SUPPLIERLINEREF>	Suppliers reference for customer PO line	N	Oe_Line_Acks.ORIG_SYS_LINE_REF
</USERAREA>			
<SOSCHEDULE>		N	
<DATETIME qualifier="DELIVSCHED">	Scheduled arrival date	N	Derived from Oe_Line_Acks.SCHEDULE_ARRIVAL_DATE
<YEAR>1998</YEAR>		N	
<MONTH>11</MONTH>		N	
<DAY>21</DAY>			
<HOUR>00</HOUR>			

Table 7-18 ONT_3A7_OAG72_OUT_SO

OAG	Comment/Description	OAG Reqd Y/N	OM table/Column
<MINUTE>25</MINUTE>			
<SECOND>45</SECOND>			
<SUBSECOND>0000</SUBSECOND>		Y	
<TIMEZONE>-0600</TIMEZONE>		Y	
</DATETIME>			
<DATETIME qualifier="NEEDEDELV">	Requested ship date	Y	Derived from Oe_Line_ Acks.REQUES T_DATE
<YEAR>1998</YEAR>			
<MONTH>11</MONTH>		N	
<DAY>21</DAY>		Y	
<HOUR>00</HOUR>		Y	
<MINUTE>25</MINUTE>		Y	
<SECOND>45</SECOND>		Y	
<SUBSECOND>0000</SUBSECOND>			
<TIMEZONE>-0600</TIMEZONE>		Y	
</DATETIME>		N	
<QUANTITY qualifier="ORDERED">	Scheduled quantity	N	Derived from Oe_Line_ Acks.ORDERE D_ QUANTITY, ORDER_ QUANTITY_ UOM
<VALUE>1</VALUE>		Y	
<NUMOFDEC>0</NUMOFDEC>		Y	
<SIGN>+</SIGN>		Y	

Table 7-18 ONT_3A7_OAG72_OUT_SO

OAG	Comment/Description	OAG Reqd Y/N	OM table/Column
<UOM>EACH</UOM>		Y	
</QUANTITY>			
<SOSLINENUM>1</SOSLINENUM>	Sales Order shipment number	Y	Oe_Line_Acks.SHIPMENT_NUMBER
<PSCLINENUM>1</PSCLINENUM>	Customer schedule line number	N	Oe_Line_Acks.CUSTOMER_SHIPMENT_NUMBER
<USERAREA>			
<ORACLE.SUPPLIERSHIPMENTREF>1 </ORACLE.SUPPLIERSHIPMENTREF>	Suppliers reference for customer shipment	N	Oe_line_Acks.ORIG_SYS_SHIPMENT_REF
<ORACLE.SPLITLINE>N</ORACLE.SPLITLINE>	Indication that this line is split	N	Derived
<ORACLE.SPLITFROMLINEREF>1 <ORACLE.SPLITFROMLINEREF>	Split from line reference	N	Oe_line_Acks.SPLIT_FROM_LINE_ID
<DATETIME qualifier="SHIPSCHED">	Schedule ship date	N	Derived from Oe_Line_Acks.SCHEDULE_SHIP_DATE
<YEAR>1998</YEAR>		N	
<MONTH>11</MONTH>		N	
<DAY>21</DAY>		N	
<HOUR>00</HOUR>		N	
<MINUTE>25</MINUTE>		N	
<SECOND>45</SECOND>		N	

Table 7-18 ONT_3A7_OAG72_OUT_SO

OAG	Comment/Description	OAG Req'd Y/N	OM table/Column
<SUBSECOND>0000</SUBSECOND>		N	
<TIMEZONE>-0600</TIMEZONE>		N	
</DATETIME>		N	
</USERAREA>			
</SOSCHEDULE>			
</SOLINE>			
</CHANGE_SALESORDER>			

Table 7-19 Statuses and Descriptions

Status	Description
So Status	Open- Any status other than the following ones in Order Management. Cancelled- Cancelled status in OM
So Line Status	Open- Any status other than the following ones in Order Management. On Hold – Line is on 3A7 Change Notification Hold Cancelled- Cancelled status in Order Management
NOTES (Header and Line level)	The Notes can be used in case of 3A7 being sent in order to change Pending status to either Accept or Reject. In case of Reject it is necessary to populate this element. Possible values - <ul style="list-style-type: none"> Accepted

Message Details

This table describes the data types and fields in the DTD used by the message map. The message map may remove used fields so that empty data tags are neither generated for outbound transactions nor examined by inbound transactions.

Message Map Details

Table 7–20 Message Map Details

Message Map Name:	ONT_3A7_OAG72_OUT_SO
Direction:	OUT
(Internal) Transaction Type:	ONT
(Internal) Transaction Subtype:	CSO
External Transaction Type:	SALESORDER
External Transaction Subtype:	CHANGE
DTD Directory:	ont/xml/oag72
Map Directory:	patch/115/xml/US
Message Maps XGM File Name:	ONT_3A7_OAG72_OUT_SO.xgm
Standard:	OAG
Release:	7.2
Format:	DTD
DTD Name:	087_change_salesorder_008.dtd

Workflow Event Setup

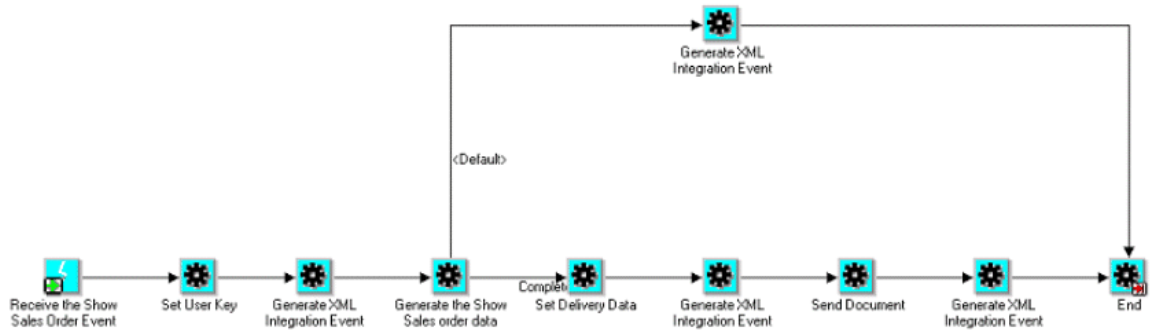
NOTE: The same event/subscription/workflow that was developed for Show sales order is reused for Change SO.

Table 7–21 Workflow Event Setup

Detail	Value
Event Name	oracle.apps.ont.oi.show_so.create
Event Description	Event for 3A6 Outbound Show sales order
Subscription	R_SHOW_SALES_ORDER
Subscription Description	Event subscription for the oracle.apps.ont.oi.show_so.create event for Outbound Show sales order.

Workflow (oexwfoa.wft)

Figure 7–6 Process R_SHOW_SALES_ORDER



Lookups

Table 7–22 Lookups

Lookup Type	Code	Meaning	Enabled
HOLD_TYPE	CHANGE_NOTIFICATION_HOLD	Change Notification Hold for 3A7	No
RELEASE_REASON	3A7_RESPONSE_RECEIVED	Response to 3A7 is received. Hold released automatically.	No

Hold Definition

Hold is for the 3A7. This hold is at the order line level.

Table 7–23 3A7 Hold

Hold Name	Hold Type	Description	Hold ID
3A7 Change Notification Hold	CHANGE_ NOTIFICATION_ HOLD	Line level hold - Waiting for a response on 3A7	56

Inbound Change PO Request

Overview

The Inbound Change PO Request enables a buyer to change a purchase order and the seller to acknowledge if the changes are accepted, rejected, or pending. This supports a process for trading partners, based on acknowledgment responses, to do the following:

- Change purchase orders
- Acknowledge whether the buyer accepts the changes made by the seller

Major Features

Consume the Change_PO Message Inbound From Purchasing

Oracle Order Management consumes the Change_PO message inbound from Purchasing, thereby updating the purchase order previously entered.

Changes to the purchase order follow the same rules for changing a purchase order that currently exist in Order Management.

Scenario 1: Redistribution of the quantity over shipments (Oracle Purchasing)

Original PO

Table 7–24 *Redistribution - Before*

Line	Shipment	Qty
1	1	5
1	2	5
1	3	5

If the quantity of the 3 shipments in above example is redistributed over 2 shipments Change PO looks as follows:

Table 7–25 *Redistribution - Two Shipments*

Line	Shipment	Qty
1	1	7

Table 7–25 Redistribution - Two Shipments

Line	Shipment	Qty
1	2	8
1	3	0

Scenario 2: Cancellation of a shipment (Oracle Purchasing)

Cancellation of one of the shipments can cause a Change PO to start (Oracle Purchasing).

Original PO

Table 7–26 Cancellation of a Shipment

Line	Shipment	Qty
1	1	6
1	2	5
1	3	5

If Shipment 3 is cancelled Change PO would look as follows:

Table 7–27 After Cancellation

Line	Shipment	Qty
1	3	0

Changes supported

Change PO supports change in the following elements:

- Bill to location
- Ship to location (Schedule level)
- Quantity (schedule level)
- Need by date (Schedule level)
- Price
- Payment terms

Set up the Order Import Process to be Run in Synchronous or Asynchronous Mode

You can setup a profile option to start Order Import for every order or wait and process multiple orders via concurrent processing. The Profile Option is called OM: Run Order Import for XML, and can be set up at the site level. The possible values for this profile option are Asynchronous and Synchronous.

Integration With Collaboration History Module (CLN)

The Collaboration History module records the transaction history when the trading partners exchange messages. To enable this Order Management has incorporated the API calls during the processing of the XML messages. In the case of 3A8 Change PO request message collaboration history is recorded at the key points in processing Change PO.

- When Change PO comes in and is consumed by Order Management by entering the data into Order Import tables (and before running the Order Import Process)
- Before running Order Import (if Order Import is going to be run immediately)
OR
- If Order Import is not going to be run and is going to be run in batch mode later on
- Order Import completed successfully /Failed

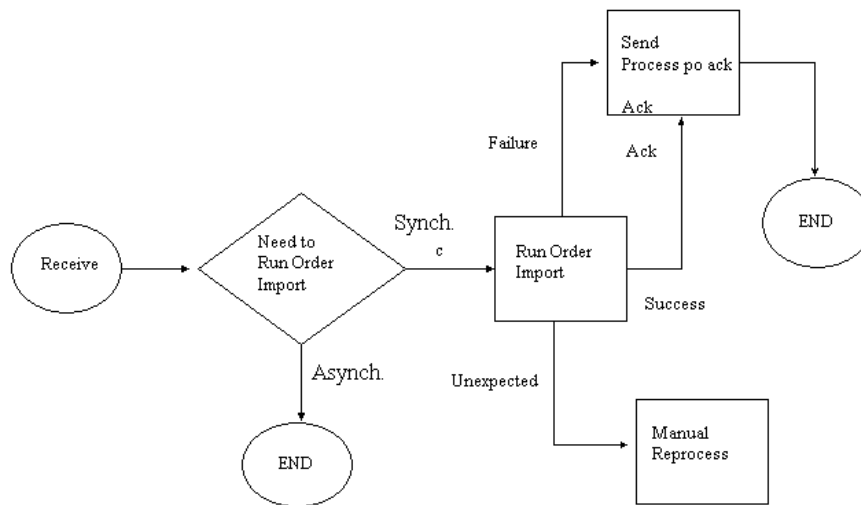
Support For Split Lines

Order management supports splitting of lines for multiple reasons. When the line is split, new shipment information is created for the line and stored in Order Management. In this case the 3A7 is sent to the buyer notifying of this change. Upon receiving the information about the new line buyer's system sends the acknowledgement either accepting or rejecting this new line. If the new line is accepted by the buyer, the buyer sends the 3A8 (change PO) as a response with the new buyer information as well as the reference for the new line sent by the seller.

If the line is rejected by the buyer, 3A8 can be sent by the buyer to cancel the split or original line.

Error Handling

The error handling process is described here:

Figure 7-7 Error Handling

There are 3 types of outcomes from running order import process:

- **Successful processing** - In this case the confirmation API is called to send the Acknowledgment out to the sender with the Accept status.
- **Failed processing** - This is the status when there are validation errors or data setup errors. In this case the acknowledgment is sent out to the sender with the Reject Status.
- **Unexpected Errors** - There can be some unexpected errors due to resource constraints and will require reprocessing of the Order Import upon fixing those issues. This process is manual.

Message Map

Table 7–28 *ONT_3A8R_OAG72_IN.xgm*

OAG	Comment/Desc	Re qd Y/N	Oracle OM Table/Column	Oracle PO Table /Column
<CHANGE_PO_006>				
<CNTROLAREA>				
<BSR>		Y		
<VERB value="CHANGE">CHANGE</ VERB>		Y		
<NOUN value="PO">PO</NOUN>		Y		
<REVISION value="006">006</REVISION>		Y		
</BSR>		Y		
<SENDER>		Y		
<LOGICALID>XX141HG09</LO GICALID>		Y		
<COMPONENT>PUR</COMPO NENT>		Y		
<TASK>MAINT</TASK>		Y		
<REFERENCEID>95129945823449 </REFERENCEID>		Y		
<CONFIRMATION>1</CONFIR MATION>		Y		
<LANGUAGE>ENG</LANGUA GE>		Y		
<CODEPAGE>test</CODEPAGE >		Y		
<AUTHID>CMKURT</AUTHID >		Y		
</SENDER>				
<DATETIME qualifier="CREATION">	Timestamp for PO (Standard or Release) creation	Y		

Table 7–28 ONT_3A8R_OAG72_IN.xgm

OAG	Comment/Desc	Re qd Y/N	Oracle OM Table/Column	Oracle PO Table /Column
<YEAR>1998</YEAR>		Y		
<MONTH>11</MONTH>		Y		
<DAY>20</DAY>		Y		
<HOUR>17</HOUR>		Y		
<MINUTE>06</MINUTE>		Y		
<SECOND>45</SECOND>		Y		
<SUBSECOND>0000</SUBSEC OND>		Y		
<TIMEZONE>-0600</TIMEZONE >		Y		
</DATETIME>				
</CNTROLAREA>				
<DATAAREA>				
<CHANGE_PO>				
<POHEADER>				
<POID>a</POID>	Unique ID for the PO. For Standard POs, the PO number from Oracle Purchasing is entered here. For Blanket Releases, it should be 'Blanket PO#-Release#.	Y	OE_HEADERS_ INTERFACE.Cust_ po_number	PO_ECX_HEADER_ ARCH_ V.contractb PHA.SEGMENT1
<NOTES>Notes</NOTES>		N		
<REASONCODE>Admin Error</REASONCODE>	Reason for the change	N	OE_HEADERS_ INTERFACE.change _reason	
<USERAREA>				
<ORGID>	Org Id		OE_HEADERS_ INTERFACE.org_id	

Table 7-28 ONT_3A8R_OAG72_IN.xgm

OAG	Comment/Desc	Re qd Y/N	Oracle OM Table/Column	Oracle PO Table /Column
<ORACLE.CHANGESEQUENCE >12 </ORACLE.CHANGESEQUENC E>	Change Sequence		OE_HEADERS_ INTERFACE.change _sequence	
<ORACLE.BOOKEDFLAG>Y </ORACLE.BOOKEDFLAG>	Booked Flag		OE_HEADERS_ INTERFACE.booked _flag	
<MSGTYPE>Response</MSGTY PE>	Response Flag		OE_HEADERS_ INTERFACE.respon se_flag	
<ACKCODE>	A value of 2 for a 3A8 Change PO response document implies a header level buyer rejection and will result in the entire document not being processed.			
<ORACLE.SUPPLIERDOCREF>1 22 </ORACLE.SUPPLIERDOCREF>	Suppliers reference for the Customer PO. Needs to be sent back by the buyer as a reference for the newly created line on 3A8 response as well as 3A9. Also needed to be sent on every new change request for the new line.	N	OE_HEADERS_ INTERFACE.ORIG_ SYS_DOCUMENT_ REF	
</USERAREA>				

Table 7–28 ONT_3A8R_OAG72_IN.xgm

OAG	Comment/Desc	Re qd Y/N	Oracle OM Table/Column	Oracle PO Table /Column
<PARTNER>	Supplier This data type provides information about the trading partner. No need to map this segment for OM	Y		
<NAME1>abcd</NAME1>	Name of the trading partner			PO_ECX_HEADER_ARCH_V.supp_org_name PO_VENDORS.VENDOR_NAME
<ONETIME>a</ONETIME>	Indicator of whether this partner is Established for this transaction only.	Y		PO_ECX_HEADER_ARCH_V.supp_of DECODE (PO_VENDORS.ONE_ TIME_ FLAG,'N','0','1')
<PARTNRID>a</PARTNRID>	Unique identifier for the partner in Oracle Applications (Supplier Number).	Y		PO_ECX_HEADER_ARCH_V.supplier_partner_id PO_VENDORS.SEGMENT1
<PARTNRTYPE>Supplier</PARTNRTYPE>		Y		
<CURRENCY>a</CURRENCY>	Preferred operating currency of the partner.			PO_ECX_HEADER_ARCH_V.supp_Currency PO_VENDORS.PAYMENT_CURRENCY_CODE

Table 7–28 ONT_3A8R_OAG72_IN.xgm

OAG	Comment/Desc	Re qd Y/N	Oracle OM Table/Column	Oracle PO Table /Column
<PARTNRIDX>a</PARTNRIDX>	Unique identifier of the partner Supplier.	Y		PO_ECX_HEADER_ARCH_ V.supplier_partner_id_x NVL(PO_VENDOR_SITES_ALL_ECE_TP_LOCATIN_CODE,PHA.VENDOR_ID
<ADDRESS>	Address of the partner			
<ADDRLINE index="1">a</ADDRLINE>				
<CITY>a</CITY>				
<COUNTRY>a</COUNTRY>				
<DESCRIPTN>a</DESCRIPTN>				
<FAX index="1">a</FAX>				
<POSTALCODE>a</POSTALCODE>				
<STATEPROVN>a</STATEPROVN>				
<TELEPHONE index="1">a</TELEPHONE>				
</ADDRESS>				
</PARTNER>				
<PARTNER>	Sold To This data type provides information about the trading partner	Y		

Table 7–28 ONT_3A8R_OAG72_IN.xgm

OAG	Comment/Desc	Re qd Y/N	Oracle OM Table/Column	Oracle PO Table /Column
<NAME1>abcd</NAME1>	Name of the buyer company			PO_ECX_HEADER_ARCH_V.buying_org_name HR_ALL_ORGANIZATION_UNITS_TL.NAME
<ONETIME>a</ONETIME>	Indicator of whether this partner is Established for this transaction only.			PO_ECX_HEADER_ARCH_V.buying_org_otf '0
<PARTNRID>a</PARTNRID>	Unique identifier for the partner in Oracle Applications (Supplier Number).			
<PARTNRTYPE>Sold To</PARTNRTYPE>	Sold To			
<CURRENCY>a</CURRENCY>	Preferred operating currency of the partner.			PO_ECX_HEADER_ARCH_V.buying_org_currency GL_SETS_OF_BOOKS.CURRENCY – CODE
<PARTNRIDX>a</PARTNRIDX>	Unique identifier of the partner.		OE_HEADERS_INTERFACE.sold_to_org_id	PO_ECX_HEADER_ARCH_V.partner_id_x HR_LOCATIONS_ALL.ECE_TP_LOCATION_CODE
<TAXID>a</TAXID>	Tax identifier			
<ADDRESS>	Address of the partner. This information needs to be mapped for the case where EDI location code is not sent.			

Table 7-28 ONT_3A8R_OAG72_IN.xgm

OAG	Comment/Desc	Re qd Y/N	Oracle OM Table/Column	Oracle PO Table /Column
<ADDRLINE index="1">a</ADDRLINE>				HR_ ORGANIZATION_ UNITS_ V.ADDRESS_LINE_ 1-3
<CITY>a</CITY>				HR_ ORGANIZATION_ UNITS_ V.TOWN_OR_CI T Y
<COUNTRY>a</COUNTRY>				PO_ECX_HEADER_ ARCH_V.buying_ org_country
<COUNTY>a</COUNTY>				OE_HEADERS_ INTERFACE.INVOIC E_COUNTY
<POSTALCODE>a</POSTALCO DE>				HR_ ORGANIZATION_ UNITS_ V.POS TAL _CODE
<STATEPROVN>a</STATEPRO VN>				HR_ ORGANIZATION_ UNITS_ V.RE GI ON2
<TELEPHONE index="1">a</TELEPHONE>				HR_ ORGANIZATION_ UNITS_ V.TELEPHONE_ NUMBER_1-3
</ADDRESS>				
</PARTNER>				
<PARTNER>	Bill to partner- This is an optional segment			

Table 7–28 ONT_3A8R_OAG72_IN.xgm

OAG	Comment/Desc	Re qd Y/N	Oracle OM Table/Column	Oracle PO Table /Column
<NAME1>abcd</NAME1>	Name of the bill to partner			PO_ECX_HEADER_ARCH_V.buying_org_name HR_ALL_ORGANIZATION_UNITS_TL.NAME
<ONETIME>a</ONETIME>				PO_ECX_HEADER_ARCH_V.billto_org_otf '0
<PARTNRID>a</PARTNRID>				PO_ECX_HEADER_ARCH_V.partner_id PHA.ORG_ID
<PARTNRTYPE>Bill to</PARTNRTYPE>				
<CURRENCY>a</CURRENCY>				FINANCIALS_SYSTEM_PARAMS_ALL.PAYMENT_CURRENCY_CODE
<PARTNRIDX>a</PARTNRIDX>			OE_HEADERS_INTERFACE.invoice_to_org_id	HR_LOCATIONS_ALL.ECE_TP_LOCATION_CODE
<ADDRESS>	The address information needs to be mapped for the case where EDI Location code is not sent.			
<ADDRLINE index="1">a</ADDRLINE>			OE_HEADERS_INTERFACE.INVOICE_ADDRESS1-3	HR_LOCATIONS_ALL.ADDRESS_LINE_1-3
<CITY>a</CITY>			OE_HEADERS_INTERFACE.INVOICE_CITY	HR_LOCATIONS_ALL.TOWN_OR_CITY

Table 7-28 ONT_3A8R_OAG72_IN.xgm

OAG	Comment/Desc	Re qd Y/N	Oracle OM Table/Column	Oracle PO Table /Column
<COUNTRY>a</COUNTRY>			OE_HEADERS_ INTERFACE.INVOI CE_COUNTRY	HR_LOCATIONS_ ALL.COUNTRY
<COUNTY>a</COUNTY>				OE_HEADERS_ INTERFACE.SHIP_ TO_COUNTY
<POSTALCODE>a</POSTALCO DE>			OE_HEADERS_ INTERFACE.INVOI CE_POSTAL_CODE	HR_LOCATIONS_ ALL.POSTAL_CODE
<STATEPROVN>a</STATEPRO VN>			OE_HEADERS_ INTERFACE.INVOI CE_STATE	HR_LOCATIONS_ ALL.REGION_2
<TELEPHONE index="1">a</TELEPHONE>				HR_LOCATIONS_ ALL.TELEPHONE_ NUMBER_1-3
</ADDRESS>				
</PARTNER>				
<PARTNER>	Ship to partner- This is an optional segment			
<NAME1>abcd</NAME1>	Name of the Ship to partner			
<ONETIME>a</ONETIME>				
<PARTNRID>a</PARTNRID>				
<PARTNRTYPE>Ship To</PARTNRTYPE>				
<CURRENCY>a</CURRENCY>				
<PARTNRIDX>a</PARTNRIDX>			OE_HEADERS_ INTERFACE.ship_ to_org_id	

Table 7–28 ONT_3A8R_OAG72_IN.xgm

OAG	Comment/Desc	Re qd Y/N	Oracle OM Table/Column	Oracle PO Table /Column
<ADDRESS>	The address information needs to be mapped for the case where EDI Location code is not sent.			
<ADDRLINE index="1">a</ADDRLINE>			OE_HEADERS_ INTERFACE.SHIP_ TO_ADDRESS1-3	
<CITY>a</CITY>			OE_HEADERS_ INTERFACE.SHIP_ TO_CITY	
<COUNTRY>a</COUNTRY>			OE_HEADERS_ INTERFACE.SHIP_ TO_COUNTRY	
<POSTALCODE>a</POSTALCO DE>			OE_HEADERS_ INTERFACE.SHIP_ TO_POSTAL_CODE	
<STATEPROVN>a</STATEPRO VN>			OE_HEADERS_ INTERFACE.SHIP_ TO_STATE	
<TELEPHONE index="1">a</TELEPHONE>				
</ADDRESS>				
<POTERM>	Payment term information			
<DESCRIPTN>a</DESCRIPTN>	Description of payment terms.			PO_ECX_HEADER_ ARCH_V.paymet_ terms_description AP_ TERMS.DESCRPTIN
<TERMID>Immediate</TERMID >	Identifier for payment terms.		OE_HEADERS_ INTERFACE. payment_term	PO_ECX_HEADER_ ARCH_V.payme t_ terms_name AP_ TERMS.NAME

Table 7-28 ONT_3A8R_OAG72_IN.xgm

OAG	Comment/Desc	Re qd Y/N	Oracle OM Table/Column	Oracle PO Table /Column
<DAYSNUM>a</DAYSNUM>				No mapping provided by PO although generated XML has these elements.
</POTERM>				
</POHEADER>				
<POLINE>	This data type provides details of a POline. At least one PO line data type is required. This data type will occur one or more times.			
<OPERAMT qualifier="UNIT" type="T">	Unit price of the item		OE_LINES_INTERFACE.customer_item_net_price	
<VALUE>a</VALUE>	Monetary unit amount of the PO line.			PO_ECX_LINE_ARCH_V.price PO_LINES_ARCHIVE_ALL.UNIT_PRICE
<NUMOFDEC>a</NUMOFDEC>				
<SIGN>a</SIGN>				
<CURRENCY>a</CURRENCY>	Three-character ISO currency code.			PO_ECX_HEADER_ARCH_V.po_currency PHA.CURRENCY_CODE
<UOMVALUE>a</UOMVALUE>				
<UOMNUMDEC>a</UOMNUMDEC>				
<UOM>a</UOM>				
</OPERAMT>				

Table 7–28 ONT_3A8R_OAG72_IN.xgm

OAG	Comment/Desc	Re qd Y/N	Oracle OM Table/Column	Oracle PO Table /Column
<QUANTITY qualifier="ORDERED">	Quantity of the item ordered. This will be the new qty if there is a change in the qty on schedule level.			
<VALUE>a</VALUE>	Numeric Value of the qty			PO_ECX_LINE_ ARCH_V.quantity PO_LINES_ ARCHIVE_ ALL.QUANTITY
<NUMOFDEC>a</NUMOFDEC >				
<SIGN>a</SIGN>				
<UOM>a</UOM>				
</QUANTITY>				
<REASONCODE>abc</REASON CODE>	Reason Code. PO does not use this tag. If blank this should be defaulted as 'No Reason Provided'	N		
<POLINENUM>a</POLINENU M>	Po Line number	Y	OE_LINES_ INTERFACE.custom er_line_number	PO_ECX_LINE_ ARCH_V.line_num PO_LINES_ ARCHIVE_ ALL.LINE_NUM
<DESCRIPTN>a</DESCRIPTN>	Item Description. No need to map for OM.			PO_ECX_LINE_ ARCH_V.description PO_LINES_ ARCHIVE_ ALL.ITEM_ DESCRIPTION

Table 7-28 ONT_3A8R_OAG72_IN.xgm

OAG	Comment/Desc	Re qd Y/N	Oracle OM Table/Column	Oracle PO Table /Column
<ITEMRV>a</ITEMRV>	Customer item revision			PO_ECX_LINE_ ARCH_V.itemrv PO_LINES_ ARCHIVE_ ALL.ITEM_ REVISION
<ITEMRVX>a</ITEMRVX>	Not used			
<ITEM>a</ITEM>	Identifier of the product. This is the customer item number		OE_LINES_ INTERFACE.custom er_item_name	PO_ECX_LINE_ ARCH_V.item MTL_SYSTEM_ ITEMS_B_ KJV.SEGMENT1
<ITEMX>a</ITEMX>	Supplier item number. This is the item number in OM			PO_ECX_LINE_ ARCH_V.itemx PO_LINES_ ARCHIVE_ ALL.VENDOR_ PRODUCT_NUM
<ITEMTYPE>a</ITEMTYPE>				
<NOTES index="1">a</NOTES>	Notes to supplier.		OE_LINES_ INTERFACE.change _comments	PO_ECX_LINE_ ARCH_V.note_to_ vendor PO_LINES_ ARCHIVE_ ALL.NOTE_TO_ VENDOR
<USERAREA>				
<ACKCODE>	A value of 2 for a 3A8 Change PO response document implies a line level buyer rejection and will result in that line not being processed.			

Table 7–28 ONT_3A8R_OAG72_IN.xgm

OAG	Comment/Desc	Re qd Y/N	Oracle OM Table/Column	Oracle PO Table /Column
<ORACLE.SUPPLIER_LINE_REF>1 </ORACLE.SUPPLIER_LINE_REF>	Suppliers reference for the Customer PO Line. Needs to be sent back by the buyer as a reference for the newly created line on 3A8 response as well as 3A9. Also needed to be sent on every new change request for the new line.	N	OE_LINES_INTERFACE.orig_sys_line_ref	
</USERAREA>				
<SCHEDULE>				
<USERAREA>				
<PARTNER>	Bill to partner- This is an optional segment			
<NAME1>abcd</NAME1>	Name of the bill to partner			
<ONETIME>a</ONETIME>				
<PARTNRID>a</PARTNRID>				
<PARTNRTYPE>Bill to</PARTNRTYPE>				
<CURRENCY>a</CURRENCY>				
<PARTNRIDX>a</PARTNRIDX>			OE_LINES_INTERFACE.invoice_to_org_id	
<ADDRESS>	The address information needs to be mapped for the case where EDI Location code is not sent.			

Table 7-28 ONT_3A8R_OAG72_IN.xgm

OAG	Comment/Desc	Re qd Y/N	Oracle OM Table/Column	Oracle PO Table /Column
<ADDRLINE index="1">a</ADDRLINE>			OE_LINES_ INTERFACE.INVOI CE_TO_ ADDRESS1-3	
<CITY>a</CITY>			OE_LINES_ INTERFACE.INVOI CE_TO_CITY	
<COUNTRY>a</COUNTRY>			OE_LINES_ INTERFACE.INVOI CE_TO_COUNTRY	
<COUNTY>a</COUNTY>			OE_LINES_ INTERFACE.INVOI CE_TO_COUNTY	
<POSTALCODE>a</POSTALCO DE>			OE_LINES_ INTERFACE.INVOI CE_TO_POSTAL_ CODE	
<STATEPROVN>a</STATEPRO VN>			OE_LINES_ INTERFACE.INVOI CE_TO_STATE	
<TELEPHONE index="1">a</TELEPHONE>				
</ADDRESS>				
<PARTNER>	Ship to partner			
<NAME1>abcd</NAME1>	Name of the Ship to partner			PO_ECX_LINE_ LOC_ARCH_ V.Shipto_org_name HR_ ORGANIZATION_ UNITS_V .NAME
<ONETIME>a</ONETIME>				

Table 7–28 ONT_3A8R_OAG72_IN.xgm

OAG	Comment/Desc	Re qd Y/N	Oracle OM Table/Column	Oracle PO Table /Column
<PARTNRID>a</PARTNRID>				PO_ECX_LINE_ LOC_ARCH_ V.partner_id PO_LINE_ LOCATIONS_ ARCHIVE_ ALL.ORG_ID
<PARTNRTYPE>Ship to</PARTNRTYPE>	Value ShipTo			PO_ECX_LINE_ LOC_ARCH_ V.shipto_org_ partner_type 'ShipTo'
<CURRENCY>a</CURRENCY>				
<PARTNRIDX>a</PARTNRIDX >	EDI Location code		OE_HEADERS_ INTERFACE.ship_ to_org_id	HR_LOCATIONS_ ALL.ECE_TP_ LOCATION_CODE
<ADDRESS>				
<ADDRLINE index="1">a</ADDRLINE>	Address of the ship to		OE_HEADERS_ INTERFACE.SHIP_ TO_ADDRESS1-3	HR_LOCATIONS_ ALL.ADDRESS_ LINE_1,2,3
<CITY>a</CITY>	City		OE_HEADERS_ INTERFACE.SHIP_ TO_CITY	HR_LOCATIONS_ ALL.TOWN_OR_ CITY
<COUNTRY>a</COUNTRY>	Country		OE_HEADERS_ INTERFACE.SHIP_ TO_COUNTRY	HR_LOCATIONS_ ALL.COUNTRY
<COUNTY>a</COUNTY>			OE_LINES_ INTERFACE.SHIP_ TO_COUNTY	
<POSTALCODE>a</POSTALCO DE>	Postal Code		OE_HEADERS_ INTERFACE.SHIP_ TO_POSTAL_CODE	HR_LOCATIONS_ ALL.POSTAL_CODE
<STATEPROVN>a</STATEPRO VN>	State		OE_HEADERS_ INTERFACE.SHIP_ TO_STATE	HR_LOCATIONS_ ALL.REGION_2
<TELEPHONE index="1">a</TELEPHONE>	Telephone			HR_LOCATIONS_ ALL.TELEPHONE_ NUMBER 1,2,3

Table 7-28 ONT_3A8R_OAG72_IN.xgm

OAG	Comment/Desc	Re qd Y/N	Oracle OM Table/Column	Oracle PO Table /Column
</ADDRESS>				
</PARTNER>				
<OPERATION>	Operation Code		OE_LINES_ INTERFACE.operati on_code	
<ACKCODE>	A value of 2 for a 3A8 Change PO Response document implies a line level buyer rejection and will result in that line not being processed.			
<ORACLE.SUPPLIER_SHIPMENT_REF>1 </ORACLE.SUPPLIER_SHIPMENT_REF>	Supplier's reference for the customer PO shipment line. Needs to be sent back by the buyer as a reference for the newly created line on 3A8 response as well as 3A9. Also needed to be sent on every new change request for the new line.	N	OE_LINES_ INTERFACE.orig_ sys_shipment_ref	
</USERAREA>				
<DATETIME qualifier="NEEDELV">	Need by Delivery date. This will be the new date if there are any changes.		OE_LINES_ INTERFACE.request _date	PO_ECX_LINE_ LOC_ARCH_V.need_ by_date PO_LINE_ LOCATIONS_ ARCHIVE_ ALL.NEED_BY_ DATE
<YEAR>1998</YEAR>				
<MONTH>11</MONTH>				
<DAY>01</DAY>				

Table 7-28 ONT_3A8R_OAG72_IN.xgm

OAG	Comment/Desc	Re qd Y/N	Oracle OM Table/Column	Oracle PO Table /Column
<HOUR>00</HOUR>				
<MINUTE>00</MINUTE>				
<SECOND>00</SECOND>				
<SUBSECOND>0000</SUBSECON D>				
<TIMEZONE>-0600</TIMEZON E>				
</DATETIME>				
<QUANTITY qualifier="ORDERED">	Quantity of the item ordered. This will be the new qty.		OE_LINES_ INTERFACE.ordere d_quantity OE_LINES_ INTERFACE.order_ quantity_uom	PO_ECX_LINE_ LOC_ARCH_ V.ordered_quantity (PO_LINE_ LOCATIONS_ ARCHIVE_ ALL.QUANTITY -PO_LINE_ LOCATIONS_ ARCHIVE_ ALL.QUANTITY_ CANCELLED)
<VALUE>a</VALUE>				
<NUMOFDEC>a</NUMOFDEC >				
<SIGN>a</SIGN>				
<UOM>a</UOM>				
</QUANTITY>				
<PSCLINENUM>a</PSCLINEN UM>	Line number of the delivery schedule for the PO.	Y	OE_LINES_ INTERFACE.custom er_shipment_ number	PO_ECX_LINE_ LOC_ARCH_ V.shipment_um PO_LINE_ LOCATIONS_ ARCHIVE_ ALL.SHIPMENT_ NUM

Table 7–28 *ONT_3A8R_OAG72_IN.xgm*

OAG	Comment/Desc	Re qd Y/N	Oracle OM Table/Column	Oracle PO Table /Column
<DESCRIPTN>				
<REASONCODE>	Reason for cancellation		OE_LINES_ INTERFACE.change _reason	
</SCHEDULE>				
</POLINE>				
</CHANGE_PO>				
</DATAAREA>				
</CHANGE_PO_006>				

Seeded Workflow for Inbound XML Messages

A pre-defined Workflow event is seeded to run Order Import immediately if the Profile Option OM: Run Order Import for XML is set to 'Synchronous.' This Workflow subscribes to the event raised by the Post Process of the XML Gateway. As part of the Workflow Process it will run Order Import for the XML message, that is currently being consumed. The following Workflow is designed using Workflow Builder 2.6

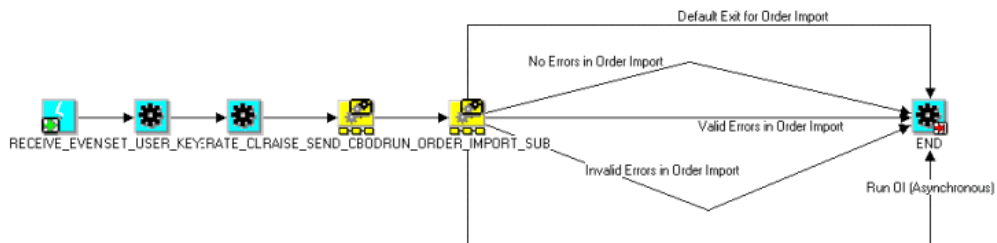
Figure 7–8 *Order Import Flow - Generic*

Figure 7–9 Raise Send CBOD Out Sub Process

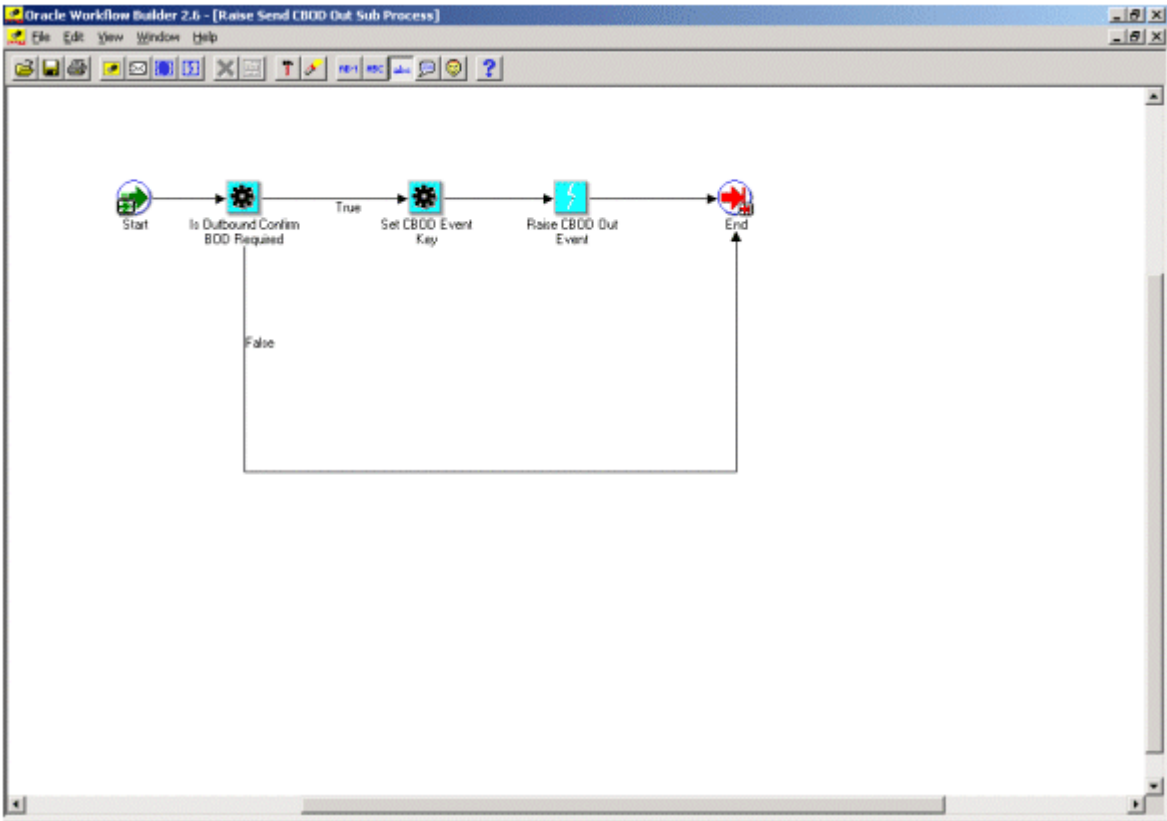


Figure 7–10 Run Order Import Sub Process

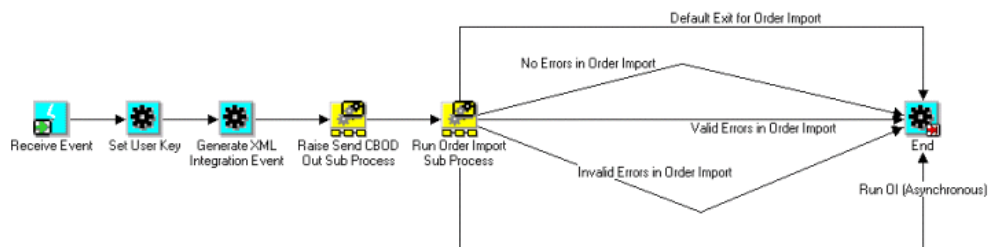
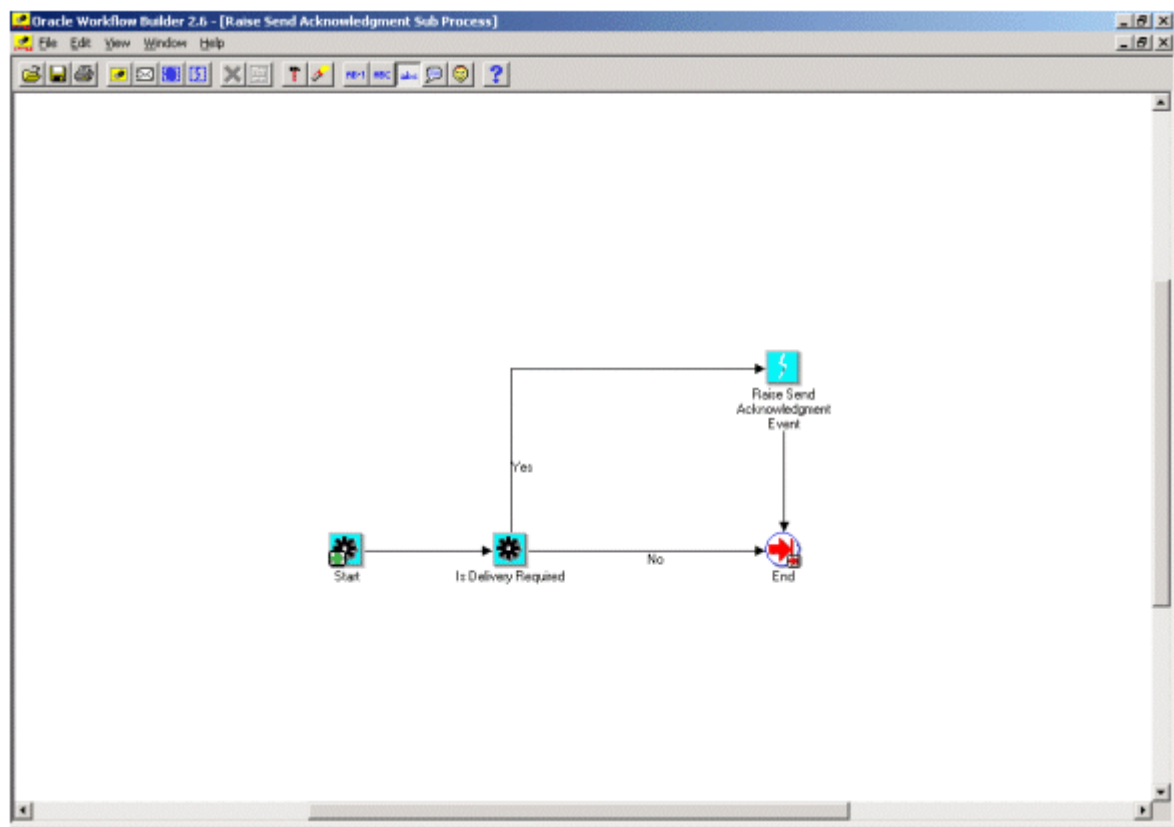


Figure 7–11 Raise Send Acknowledgement Sub Process



Message Details

This table describes the data types and fields in the DTD used by the message map. The message map may remove used fields so that empty data tags are neither generated for outbound transactions nor examined by inbound transactions.

Table 7–29 ONT_3A8R_OAG72_IN.xgm

Message Map Name:	ONT_3A8R_OAG72_IN.xgm
Direction:	Inbound
(Internal) Transaction Type:	ONT

Table 7–29 *ONT_3A8R_OAG72_IN.xgm*

(Internal) Transaction Subtype:	CHO
External Transaction Type:	PO
External Transaction Subtype:	CHANGE
DTD Directory:	\$ont/xml/oag72
Map Directory:	\$ont/patch/115/xml/US
Message Maps XGM File Name:	ONT_3A8R_OAG72_IN
Standard:	OAG
Release:	7.2
Format:	DTD
DTD Name:	057_change_po_006.dtd

Columns Enabled for Code Conversion

Table 7–30 *Columns Enabled for Code Conversion*

Tag	Category
<TERMid> (in header level <POTERM>)	PAYMENT_TERMS
<MSGTYPE> (in header level <USERAREA>)	DOC_PURPOSE_CODE
<UOM> (in schedule level <QUANTITY qualifier="ORDERED">)	UOM
<OPERATION> (in schedule level USERAREA)	ACTION_CODE

Defaulted Columns

Table 7–31 *Defaulted Columns*

Defaulted Columns	Default Value, and Condition (if any)
H.ORDER_SOURCE_ID	20
H.XML_TRANSACTION_TYPE_CODE	'CHO'
H.CHANGE_REASON	'Not provided'
L.ORDER_SOURCE_ID	20

Table 7–31 Defaulted Columns

Defaulted Columns	Default Value, and Condition (if any)
L.Orig_Sys_Document_Ref	H.Orig_Sys_Document_Ref
L.XML_Transaction_Type_Code	'CHO'
L.Change_Reason	'Not provided'

Derived Columns

Table 7–32 Derived Columns

Derived Columns	Source of Data, and Condition (if any)
H.CREATED_BY	FND_GLOBAL.USER_ID
H.CREATION_DATE	SYSDATE
H.INVOICE_TO_ORG_ID	PARTNRIDX of BillTo PARTNER
H.LAST_UPDATED_BY	FND_GLOBAL.LOGIN_ID
H.LAST_UPDATE_DATE	SYSDATE
H.LAST_UPDATE_LOGIN	FND_GLOBAL.USER_ID
H.ORG_ID	PARTNRIDX of SoldTo PARTNER, if this value is not null
H.SOLD_TO_ORG_ID	PARTNRIDX of SoldTo PARTNER
H.SHIP_TO_ORG_ID	PARTNRIDX of ShipTo PARTNER
H.XML_MESSAGE_ID	XML Gateway Internal Control Number
L.CALCULATE_PRICE_FLAG	'Y' if L.CUSTOMER_ITEM_NET_PRICE is NOT NULL
L.CHANGE_SEQUENCE	H.CHANGE_SEQUENCE
L.CREATED_BY	FND_GLOBAL.USER_ID
L.CREATION_DATE	SYSDATE
L.INVOICE_TO_ORG_ID	PARTNRIDX of BillTo PARTNER
L.LAST_UPDATED_BY	FND_GLOBAL.LOGIN_ID
L.LAST_UPDATE_DATE	SYSDATE
L.LAST_UPDATE_LOGIN	FND_GLOBAL.LOGIN_ID

Table 7–32 Derived Columns

Derived Columns	Source of Data, and Condition (if any)
L.ORDERED_QUANTITY, L.ORDER_QUANTITY_UOM	OAG Derivation from schedule level <QUANTITY qualifier="ORDERED"> tag
L.OPERATION_CODE	Derived based on stored procedure if <OPERATION> tag at schedule level is NULL
L.REQUEST_DATE	OAG Derivation from schedule level <DATETIME qualifier="NEEDEDLY"> tag
L.SHIP_TO_ORG_ID	PARTNRIDX of ShipTo PARTNER

Workflow Event Setup

Table 7–33 Workflow Event Setup

Detail	Value
Event Name	oracle.apps.ont.oi.po_inbound.create
Event Description	OM Generic Inbound Event
Subscription	R_OEOI_ORDER_IMPORT
Subscription Description	Oracle Order Management subscription to the event oracle.apps.ont.oi.po_inbound.create raised by the post process of the Change PO XML mapping

Extension Tags

Order Management added the following tags as extensions on the Process PO XML documents sent to OM:

```
$ont/xml/oag72/oagis_extensions.dtd
```

```
<!ELEMENT FTTERM (DESCRIPTN?, TERMID?)>
```

```
<!ELEMENT FOB (DESCRIPTN?, TERMID?)>
```

```
<!ELEMENT OPERATION %STRDOM;>
```

```
<!ELEMENT ORGID %STRDOM;>
```

```
<!ELEMENT USERITEMDESCRIPTN %STRDOM;>
```

```
<!ELEMENT ORACLE.CHANGESEQUENCE %STRDOM;>
<!ELEMENT ORACLE.BOOKEDFLAG %STRDOM;>
<!ELEMENT ORACLE.SUPPLIERDOCREF %STRDOM;>
<!ELEMENT ORACLE.SUPPLIERLINEREF %STRDOM;>
<!ELEMENT ORACLE.SUPPLIERSHIPMENTREF %STRDOM;>
<!ELEMENT ORACLE.SPLITFROMLINEREF %STRDOM;>
<!ELEMENT ORACLE.SPLITLINE %STRDOM;>
```

\$ont/xml/oag72/oagis_entity_extensions.dtd

Added the following tags:

```
<!ENTITY % DATETIME.EXCHRATE DATE "DATETIME">
```

Change `<!ENTITY % SEG_DATETIME_QUALIFIERS_EXTENSION "OTHER">` to
`<!ENTITY % SEG_DATETIME_QUALIFIERS_EXTENSION "OTHER | EXCHRATE">`

\$ont/xml/oag72/oagis_extensions.dtd

Added the following tags:

```
<!ELEMENT PAYMMETHOD (DESCRIPTN?, TERMID?)>
<!ELEMENT CREDITCRD (CARDID?, NAME?, (%DATETIME.EXPIRATION;?)>
<!ELEMENT STARTACTIVEDATE %STRDOM;>
<!ELEMENT ENDACTIVEDATE %STRDOM;>
<!ELEMENT CATEGORYID %STRDOM;>
<!ELEMENT REVISIONNUM %STRDOM;>
<!ELEMENT ATTACHMENT (TEXT?)>
<!ELEMENT EXCHRATE %STRDOM;>
<!ELEMENT CONFIRM %STRDOM;>
<!ELEMENT PCARDHDR
(MEMBERNAME?, PCARDNUM?, (%DATETIME.EXPIRATION;?), PCARDBRAND
?)><!ELEMENT MEMBERNAME %STRDOM;>
<!ELEMENT PCARDNUM %STRDOM;>
<!ELEMENT PCARDBRAND %STRDOM;>
```

```

<!ELEMENT CUSTOMERNUM %STRDOM;>
<!ELEMENT REQUESTOR %STRDOM;>
<!ELEMENT CONTRACTPONUM %STRDOM;>
<!ELEMENT CONTRACTPOLINENUM %STRDOM;>
<!ELEMENT VENDORQUOTENUM %STRDOM;>
<!ELEMENT LISTPRICE %STRDOM;>
<!ELEMENT MARKETPRICE %STRDOM;>
<!ELEMENT PRICENOTTOEXCEED %STRDOM;>
<!ELEMENT NEGPRICE %STRDOM;>
<!ELEMENT TAXABLE %STRDOM;>
<!ELEMENT TXNREASONCODE %STRDOM;>
<!ELEMENT TYPE1099 %STRDOM;>
<!ELEMENT LINEORDERTYPE %STRDOM;>
<!ELEMENT HAZRDUNNUM %STRDOM;>
<!ELEMENT HAZRDUNDESC %STRDOM;>
<!ELEMENT PRICEOVRD %STRDOM;>
<!ELEMENT DISTPROJECT
(REQUESTOR?,DISTNUM?,PROJECTNUM?,PROJECTTYPE?,TASKNUM?,(%QUA
NTITY.ORDERED;)?,CONVRATE,(%DATETIME.EXCHRATE,DATE;)?,DESTTYPE?
,DFFDISTRIBUTN?)>
<!ELEMENT PROJECTNUM %STRDOM;>
<!ELEMENT DISTNUM %STRDOM;>
<!ELEMENT PROJECTTYPE %STRDOM;>
<!ELEMENT TASKNUM %STRDOM;>
<!ELEMENT CONVRATE %STRDOM;>
<!ELEMENT DESTTYPE %STRDOM;>
<!ELEMENT
DFFPOHEADER(ATTRIBUTE1?,ATTRIBUTE2?,ATTRIBUTE3?,ATTRIBUTE4?,ATT
RIBUTE5?,ATTRIBUTE6?,ATTRIBUTE7?,ATTRIBUTE8?,ATTRIBUTE9?,ATTRIBUT
E10?,ATTRIBUTE11?,ATTRIBUTE12?,ATTRIBUTE13?,ATTRIBUTE14?,ATTRIBUTE

```

```
15?,ATTRIBUTE16?)><!ELEMENT
DFFVENDOR SITE(ATTRIBUTE1?,ATTRIBUTE2?,ATTRIBUTE3?,ATTRIBUTE4?,A
TTRIBUTE5?,ATTRIBUTE6?,ATTRIBUTE7?,ATTRIBUTE8?,ATTRIBUTE9?,ATTRIB
UTE10?,ATTRIBUTE11?,ATTRIBUTE12?,ATTRIBUTE13?,ATTRIBUTE14?,ATTRIB
UTE15?,ATTRIBUTE16?)>
```

```
<!ELEMENT
DFFVENDOR(ATTRIBUTE1?,ATTRIBUTE2?,ATTRIBUTE3?,ATTRIBUTE4?,ATTRI
BUTE5?,ATTRIBUTE6?,ATTRIBUTE7?,ATTRIBUTE8?,ATTRIBUTE9?,ATTRIBUTE1
0?,ATTRIBUTE11?,ATTRIBUTE12?,ATTRIBUTE13?,ATTRIBUTE14?,ATTRIBUTE15
?,ATTRIBUTE16?)>
```

```
<!ELEMENT
DFFLINE(ATTRIBUTE1?,ATTRIBUTE2?,ATTRIBUTE3?,ATTRIBUTE4?,ATTRIBUT
E5?,ATTRIBUTE6?,ATTRIBUTE7?,ATTRIBUTE8?,ATTRIBUTE9?,ATTRIBUTE10?,A
TTRIBUTE11?,ATTRIBUTE12?,ATTRIBUTE13?,ATTRIBUTE14?,ATTRIBUTE15?,AT
TRIBUTE16?)>
```

```
<!ELEMENT
DFFITEM(ATTRIBUTE1?,ATTRIBUTE2?,ATTRIBUTE3?,ATTRIBUTE4?,ATTRIBUT
E5?,ATTRIBUTE6?,ATTRIBUTE7?,ATTRIBUTE8?,ATTRIBUTE9?,ATTRIBUTE10?,A
TTRIBUTE11?,ATTRIBUTE12?,ATTRIBUTE13?,ATTRIBUTE14?,ATTRIBUTE15?,A
TTRIBUTE16?)>
```

```
<!ELEMENT
KFFITEM(ATTRIBUTE1?,ATTRIBUTE2?,ATTRIBUTE3?,ATTRIBUTE4?,ATTRIBUT
E5?,ATTRIBUTE6?,ATTRIBUTE7?,ATTRIBUTE8?,ATTRIBUTE9?,ATTRIBUTE10?,A
TTRIBUTE11?,ATTRIBUTE12?,ATTRIBUTE13?,ATTRIBUTE14?,ATTRIBUTE15?,AT
TRIBUTE16?,ATTRIBUTE17?,ATTRIBUTE18?,ATTRIBUTE19?,ATTRIBUTE20?)>
```

```
<!ELEMENT
DFFDISTRIBUTN(ATTRIBUTE1?,ATTRIBUTE2?,ATTRIBUTE3?,ATTRIBUTE4?,AT
TRIBUTE5?,ATTRIBUTE6?,ATTRIBUTE7?,ATTRIBUTE8?,ATTRIBUTE9?
,ATTRIBUTE10?,ATTRIBUTE11?,ATTRIBUTE12?,ATTRIBUTE13?,ATTRIBUTE14?,
ATTRIBUTE15?,ATTRIBUTE16?)>
```

```
<!ELEMENT ATTRIBUTE1 %STRDOM; >
```

```
<!ELEMENT ATTRIBUTE2 %STRDOM; >
```

```
<!ELEMENT ATTRIBUTE3 %STRDOM; >
```

```
<!ELEMENT ATTRIBUTE4 %STRDOM; >
```

```
<!ELEMENT ATTRIBUTE5 %STRDOM; >
```

```

<!ELEMENT ATTRIBUTE6 %STRDOM;>
<!ELEMENT ATTRIBUTE7 %STRDOM;>
<!ELEMENT ATTRIBUTE8 %STRDOM;>
<!ELEMENT ATTRIBUTE9 %STRDOM;>
<!ELEMENT ATTRIBUTE10 %STRDOM;>
<!ELEMENT ATTRIBUTE11 %STRDOM;>
<!ELEMENT ATTRIBUTE12 %STRDOM;>
<!ELEMENT ATTRIBUTE13 %STRDOM;>
<!ELEMENT ATTRIBUTE14 %STRDOM;>
<!ELEMENT ATTRIBUTE15 %STRDOM;>
<!ELEMENT ATTRIBUTE16 %STRDOM;>
<!ELEMENT ATTRIBUTE17 %STRDOM;>
<!ELEMENT ATTRIBUTE18 %STRDOM;>
<!ELEMENT ATTRIBUTE19 %STRDOM;>
<!ELEMENT ATTRIBUTE20 %STRDOM;>
<!ELEMENT TANDC %STRDOM;>
<!ELEMENT GLOBALCONTRACT %STRDOM;>
<!ELEMENT GLOBALCONTRACTLIN %STRDOM;>
<!ELEMENT CONSIGNEDINV %STRDOM;>
<!ELEMENT DROPSHIPDETAILS (DROPSHIPMENT?, DROPSHIPCUSTNAME?,
SHIPINSTR?, PACKINSTR?, SHIPMETHOD?, CUSTOMERPONUM?,
CUSTOMERLINENUM?, CUSTOMERSHIPNUM?, CUSTOMERDESC?)>
<!ELEMENT DROPSHIPMENT %STRDOM;>
<!ELEMENT DROPSHIPCUSTNAME %STRDOM;>
<!ELEMENT SHIPINSTR %STRDOM;>
<!ELEMENT PACKINSTR %STRDOM;>
<!ELEMENT SHIPMETHOD %STRDOM;>
<!ELEMENT CUSTOMERPONUM %STRDOM;>

```

```
<!ELEMENT CUSTOMERLINENUM %STRDOM;>  
<!ELEMENT CUSTOMERSHIPNUM %STRDOM;>  
<!ELEMENT CUSTOMERDESC %STRDOM;>  
<!ELEMENT SHIPPINGCONTROL %STRDOM;>  
<!ELEMENT CONTRACTNUM %STRDOM;>  
<!ELEMENT CONFIGID %STRDOM;>  
<!ELEMENT PSCLNSTATUS %STRDOM;>
```


Cancel Purchase Order

Overview

The Cancel Purchase Order message is used by the buyer to cancel a purchase order that was previously created by the buyer. This is an inbound message from buyer to the seller. Cancellation of the individual lines is allowed using this message. Cancellation of the PO follows same rules for canceling a purchase order that currently exist in Order Management. Partial cancellations of the line is allowed.

An Acknowledgement is sent to the buyer when the Cancel_PO message is consumed by Order management. Either an Accept or Reject status for the Cancel_PO request by the buyer will be sent. The Acknowledge PO message is used for the acknowledgement for the inbound Cancel_PO message.

Major Features

Consume the Cancel_PO Message Coming Inbound From Purchasing

Oracle Order Management consumes the Cancel_PO message inbound from Purchasing, thereby canceling the Purchase order previously entered. The whole purchase order is cancelled due to this inbound message. Cancellation of the individual lines is allowed.

Cancellation of the PO should follow same rules for canceling a purchase order that currently exist in Order Management. Partial cancellations of the line are allowed.

For example – If the order Qty is 10 on the original order and Cancel_PO comes in with the order Qty of 6. After processing the transaction, the Order Qty of the order will become 6 thereby cancelling 4 out of original Qty.

See [Table 7–35, "Message Map Details"](#) for the details and mapping of this message.

Acknowledge Cancellation of Purchase Order

You can confirm the deletion of the purchase order back to the buyer. Confirmation back to the buyer is done via an Acknowledge_PO message outbound from Order Management and will indicate that the purchase order has been canceled.

If the cancellation request cannot be fulfilled, in other words if the request falls outside of the allowable rules for cancellation, then an Acknowledge_PO message outbound from Order Management must be sent indicating that the purchase order deletion cannot be completed at this time and has been rejected.

Generate Confirm BOD (CBOD)

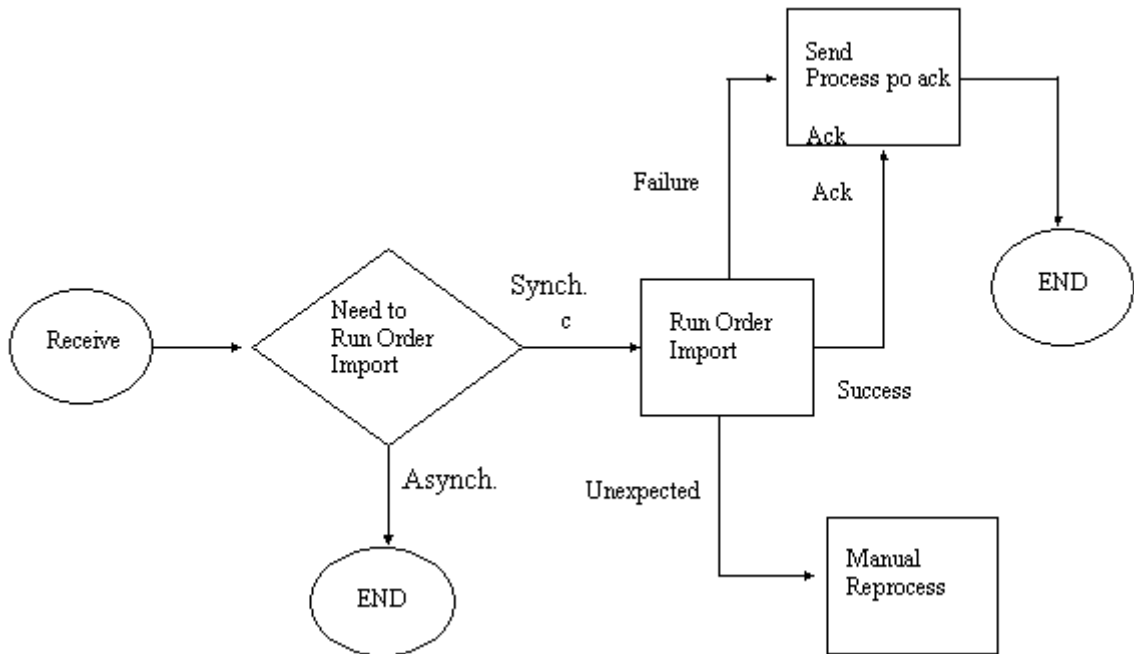
Upon receipt of the inbound Cancel_PO, Order Management can send out the Confirm BOD to the buyer indicating that the data has reached the seller. The confirm bod will not be sent out every time. It will be sent depending on the value of the <CONFIRMATION> tag in the inbound document. The CBOD will not be sent if the value of that tag is 0 or 1. It will be sent if the value of that tag is 2. Confirm BOD should be generated on the successful insertion of the data in open interface tables. It can be part of the workflow that begins after XML Gateway has finished the insertion of the data successfully. At this point only the 'Successful' status will be sent back on the CBOD as its unlikely that failure will occur after Order Management receives the control of the process. Any failure before this point will be trapped by XML gateway and notification will be sent to the buyer. Confirm BOD is only generated if the inbound document has required confirmation.

Set up the Order Import Process to Run in Synchronous or Asynchronous Mode

You can set the profile option to initiate Order Import for every order or wait and process multiple orders via concurrent processing. The profile option is OM: Run Order Import for XML, that is set at the user or site level. The possible values for this profile option will be Synchronous or Asynchronous.

Error Handling

Figure 7-12 Error Handling Process



The error handling process will be as below:

There are 3 types of outcomes from running order import process:

- **Successful processing** - In this case the confirmation API will be called to send the Acknowledgment out to the sender with the 'Accept' status.
- **Failed processing** - This will be the status when there are validation errors or data setup errors. In this case the acknowledgment will be sent out to the sender with the 'Reject' status.
- **Unexpected errors** - There can be some unexpected errors due to resource constraints and such which will require reprocessing of the Order Import upon fixing those issues. This process is manual.

Business and Process Flow

INBOUND

1. The CANCEL_PO OAG XML Message is placed on an AQ present in the E-Business Suite Database.
2. The CANCEL_PO Message is consumed using XML Gateway.
3. The XML Gateway will raise the appropriate event.
4. A CBOD is generated based on the Confirm flag in the message or the trading partner setup. See: [Generate Confirm BOD \(CBOD\)](#) on page 7-118
5. The seeded Order Management Workflow will receive this event and start the Cancel Flow.
6. Order import is used to process the message. The Order Management Module accepts the message based on set rules and tolerances previously agreed upon between the trading partners. The customer order is Cancelled.
7. The Order Management Inbound Workflow will then raise an event that will be subscribed by the Outbound Workflow, and in turn will generate the data for the OAG Acknowledge PO.
8. The XML Gateway forms the ACKNOWLEDGE_PO message.
9. This message is placed on an AQ present on the e-Business Suite Database.

CONFIRM BOD

Upon receipt of an inbound XML document such as a Process_PO or Cancel_PO, Order Management has implemented the outbound OAG Confirm BOD message to signal the successful receipt of data.

Major features of this transaction are as follows:

- This message contains information about the inbound XML message received, as well as the status and description of the inbound message received.
- Unlike the Acknowledge PO message, the Confirm BOD is sent before Order Import is run – it is purely a confirmation.

Message Map

Note: Most data values in the message map are seeded.

Message Map 'ONT_3A9R_OAG72_IN.xgm'

The message map 'ONT_3A9R_OAG72_IN' is created using the Oracle XML Gateway Message Designer tool. Please refer to [Table 7-35, "Message Map Details"](#) for the detailed map analysis.

The source DTD used is 058_cancel_po_006.dtd, revision 7.2.1 of the Open Application Group. The other associated external reference DTD files are:

- oagis_domains.dtd
- oagis_resources.dtd
- oagis_fields.dtd
- oagis_segments.dtd
- oagis_extensions.dtd
- oagis_entity_extensions.dtd

All the DTD's will be checked in ONT source area under \$ont/xml/oag72.

The target for the Inbound XML Message are the Order Management Open Interface tables OE_HEADERS_INTERFACE & OE_LINES_INTERFACE.

The CANCEL_PO DTD is a three level hierarchy with Order Lines split into one or more Shipment Lines. The Order Management architecture is however a two level one with Order Header and one or more Lines. The message map will collapse the three level XML Message into a two level structure when the data is inserted in the Order Management Open Interface tables.

Please refer to [Table 7-35, "Message Map Details"](#) for a detail analysis of the elements mapped, actions and derivation rules used by the Message Map.

Both the message map created using the Message Designer and its associated DTD's are stored in the database. The following Java programs are available to Load/Delete Maps or Load/Delete DTD's into/from the XML Gateway repository. Please refer to the *Oracle XML Gateway Manual* for more information.

Load/Delete Maps, Load/Delete DTD's

Note: The following process is used only for customizations.

- 1. java LoadMap DB username> DB password> Hostname>:Port>:SID> mymap.xgm>

Example: java oracle.apps.ecx.loader.LoadMap apps apps
ap505dbs:1521:dev115 ONT_3A9R_OAG72_IN.xgm

- 2. java DeleteMap DB username> DB password> Hostname>:Port>:SID> mapname>

Example: java oracle.apps.ecx.loader.DeleteMap apps apps
ap505dbs:1521:dev115 ONT_3A9R_OAG72_IN.xgm

The Message Map is a .xgm file which will be stored in the Order Management Source area under \$ont/patch/115/map/ ONT_3A9R_OAG72_IN.xgm

For details on how the Message Map will be delivered to Customers please refer to the [Table , "Business and Process Flow"](#) Section.

Note: Maps and DTD's must be kept in sync between the Message Designer and the XML Gateway repository. When in doubt, always reload the map and DTD as a pair.

key: H => OE_HEADERS_INTERFACE, L => OE_LINES_INTERFACE

Table 7-34 Cancel_PO Message Map

OAG Element	Description/Comment	Oracle OM table/Column	OAG reqd Y/N	Code Conversion Needed Y/N
<CANCEL_PO_006>				
<CNTROLAREA>				
<BSR>				
<VERB>CANCEL</VERB>				
<NOUN>PO</NOUN>				

Table 7–34 *Cancel_PO Message Map*

OAG Element	Description/Comment	Oracle OM table/Column	OAG reqd Y/N	Code Conversion Needed Y/N
<REVISION>006</REVISION>				
</BSR>				
<SENDER>				
<LOGICALID>XX141HG09</LOGICALID>				
<COMPONENT>PUR</COMPONENT>				
<TASK>MAINT</TASK>				
<REFERENCEID>95129945823449</REFERENCEID>				
<CONFIRMATION>1</CONFIRMATION>				
<LANGUAGE>ENG</LANGUAGE>				
<CODEPAGE>test</CODEPAGE>				
<AUTHID>CMKURT</AUTHID>				
</SENDER>				
<DATETIME qualifier="CREATION">				
<YEAR>1998</YEAR>				
<MONTH>11</MONTH>				
<DAY>21</DAY>				

Table 7–34 Cancel_PO Message Map

OAG Element	Description/Comment	Oracle OM table/Column	OAG reqd Y/N	Code Conversion Needed Y/N
<HOUR>16</HOUR>				
<MINUTE>46</MINUTE>				
<SECOND>45</SECOND>				
<SUBSECOND>0000</SUBSECON D>				
</DATETIME>				
</CNTROLAREA>				
<DATAAREA>				
<CANCEL_PO>				
<POHEADER>				
<DATETIME qualifier="DOCUMENT">				
<YEAR>1998</YEAR>				
<MONTH>11</MONTH>				
<DAY>21</DAY>				
<HOUR>16</HOUR>				
<MINUTE>46</MINUTE>				
<SECOND>45</SECOND>				
<SUBSECOND>0000</SUBSECON D>				
</DATETIME>				

Table 7–34 Cancel_PO Message Map

OAG Element	Description/Comment	Oracle OM table/Column	OAG reqd Y/N	Code Conversion Needed Y/N
<POID>a</POID>	Customer PO number	OE_HEADERS_INTERFACE.CUSTOMER_PO_NUMBER	Y	
<SITELEVEL index="1">a</SITELEVEL>	EDI location code? See open issues		Y	
<DESCRIPTN>Admin Error</DESCRIPTN>	Reason for Cancellation * see notes	OE_HEADERS_INTERFACE.CHANGE_REASON		
<NOTES index="1">a</NOTES>	Reason for cancellation. Does this need to match the cancel quick code in om?	OE_HEADERS_INTERFACE.CHANGE_COMMENTS		
<SALESORDID>a</SALESORDID>	Sales Order Number			
<USERAREA>				
<PARTNER>				
<NAME>ABC</NAME>	Name of buyer			
<ONETIME>O</ONETIME>	Onetime Y/N Set to N			
<PARTNRID>1006</PARTNRID>	Partner Location code			
<PARTNRRTYPE>SoldTo</PARTNRRTYPE>	Sold-To			
<PARTNRIDX>123</PARTNRIDX>	EDI location code for buyer	OE_HEADERS_INTERFACE.SOLD_TO_ORG_ID	Y	
</PARTNER>				

Table 7–34 Cancel_PO Message Map

OAG Element	Description/Comment	Oracle OM table/Column	OAG reqd Y/N	Code Conversion Needed Y/N
</USERAREA>				
</POHEADER>				
<POLINE>				
<ITEMX>a<ITEMX>	Buyers item Number			
<NOTES index="1">a</NOTES>	Free form text	OE_LINES_ INTERFACE. CHANGE_ COMMENTS		
<OPERAMT qualifier="UNIT" type="T">				
<VALUE>a</VALUE>				
<NUMOFDEC>a</NUMOFDEC>				
<SIGN>a</SIGN>				
<CURRENCY>a</CURRENCY>				
<UOMVALUE>a</UOMVALUE>				
<UOMNUMDEC>a</UOMNUMDEC>				
<UOM>a</UOM>				
</OPERAMT>				
<QUANTITY qualifier="ORDERED">				
<VALUE>a</VALUE>				
<NUMOFDEC>a</NUMOFDEC>				

Table 7-34 Cancel_PO Message Map

OAG Element	Description/Comment	Oracle OM table/Column	OAG reqd Y/N	Code Conversion Needed Y/N
<SIGN>a</SIGN>				
<UOM>a</UOM>				
</QUANTITY>				
<POLINENUM>a</POLINENUM>	Customer PO line number	OE_LINES_INTERFACE. CUSTOMER_LINE_NUMBER	Y	
<SCHEDULE>				
<QUANTITY qualifier="ORDERED">	Ordered Quantity. This is the new order qty hence changing the qty of the order. New qty cannot be negative or exceed the original qty.	OE_LINES_INTERFACE. ORDERED_QUANTITY OE_LINES_INTERFACE. ORDER_QUANTITY_UOM		
<VALUE>a</VALUE>				
<NUMOFDEC>a</NUMOFDEC>				
<SIGN>a</SIGN>				
<UOM>a</UOM>				
</QUANTITY>				
<PSCLINENUM>a</PSCLINENUM>	Shipment reference number	OE_LINES_INTERFACE. CUSTOMER_SHIPMENT_NUMBER	Y	

Table 7–34 Cancel_PO Message Map

OAG Element	Description/Comment	Oracle OM table/Column	OAG reqd Y/N	Code Conversion Needed Y/N
<DESCRPTN >Admin Error</DESCRPTN >	Reason for cancellation. * see notes	OE_LINES_INTERFAFAC E.CHANGE_REASON		
</SCHEDULE>				
</POLINE>				
</CANCEL_PO>				
</DATAAREA>				
</CANCEL_PO_006>				

Notes

Cancel Reason Code – The cancel reason code is not mandatory for OAG while it is mandatory for Order Management’s column. If the reason code is not sent by customer Order Management defaults it to ‘No Reason Provided.’

Note: This is seeded in Cancel_code lookup codes (OE_LOOKUP) as part of installation on Cancel_PO.

Note: The reason codes populated by the customer must be valid lookup codes in Cancel_Code lookup code.

Seeded Workflow for Inbound XML Messages

A pre-defined Workflow event is seeded to run Order Import immediately if the Profile Option OM: Run Order Import for XML is set to ‘Synchronous.’ This Workflow subscribes to the event raised by the Post Process of the XML Gateway. As part of the Workflow Process it will run Order Import for the XML message, that is currently being consumed. The following Workflow is designed using Workflow Builder 2.6

Figure 7–13 Order Import Flow - Generic

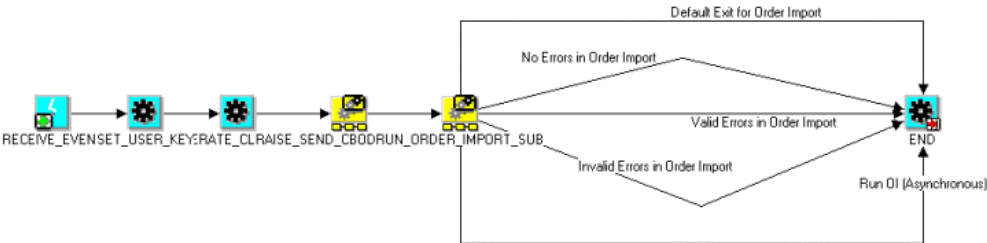


Figure 7-14 Raise Send CBOD Out Sub Process

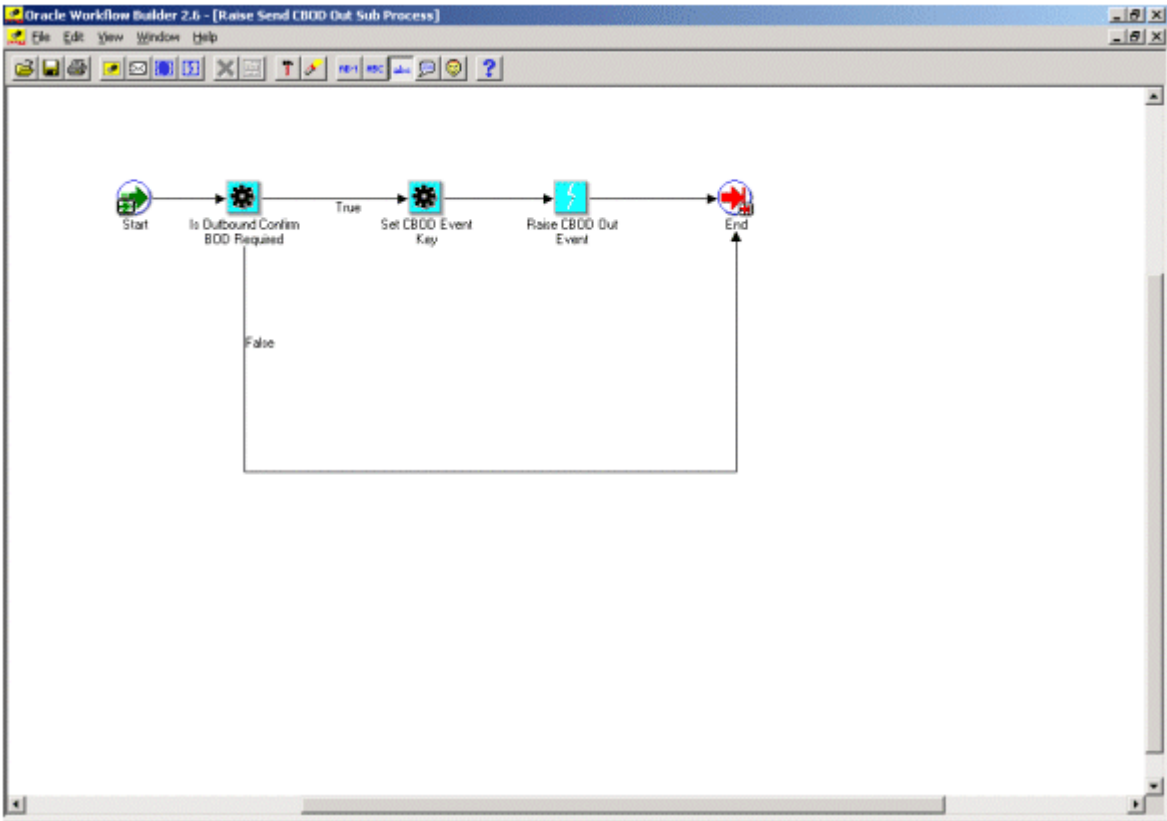


Figure 7–15 Run Order Import Sub Process

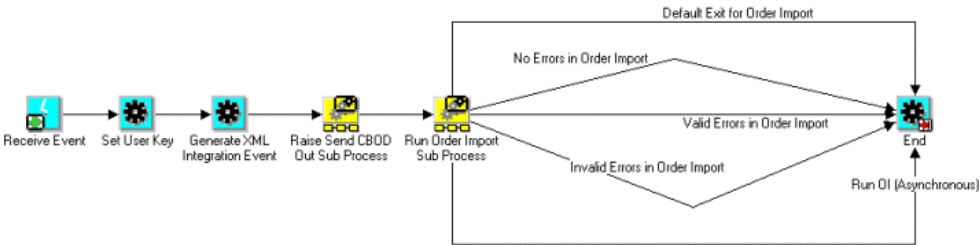
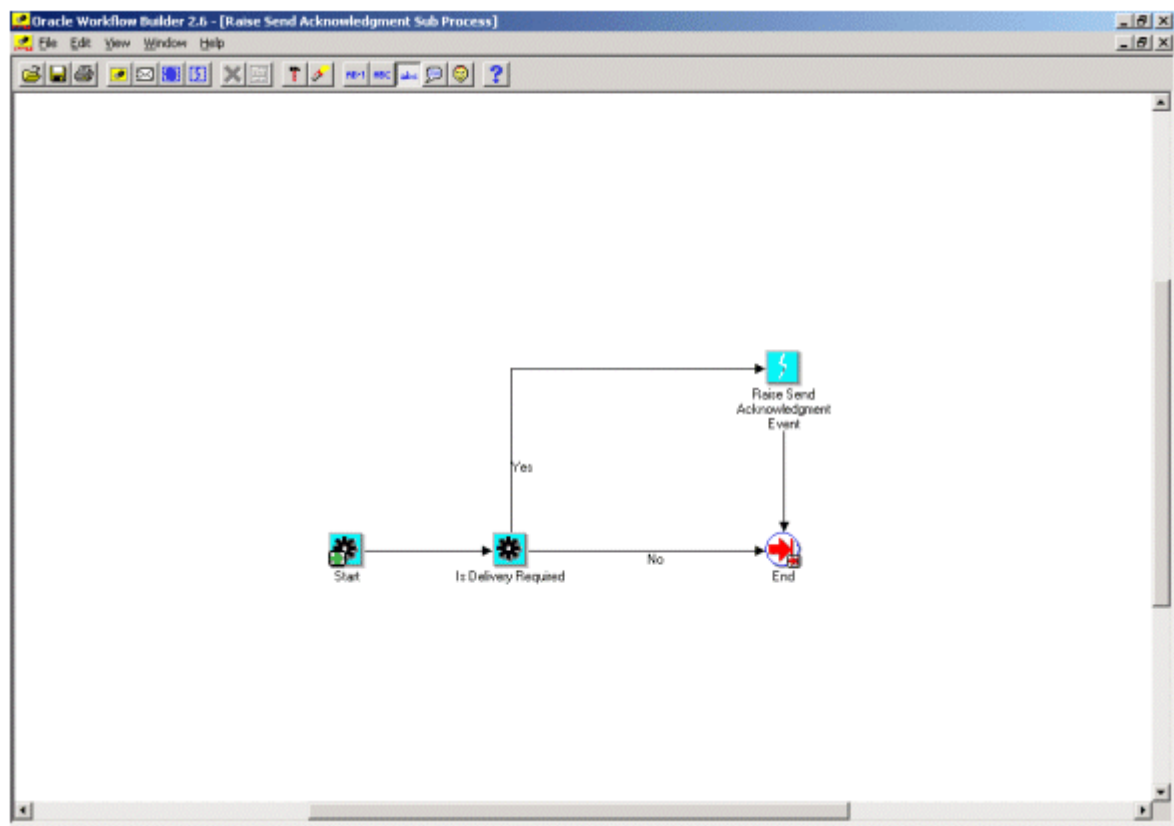


Figure 7–16 Raise Send Acknowledgement Sub Process



Message Details

This table describes the data types and fields in the DTD used by the message map.

The message map may remove used fields so that empty data tags are neither generated for outbound transactions nor examined by inbound transactions.

Table 7–35 Message Map Details

Name	Description
Message Map Name:	ONT_3A9R_OAG72_IN
Direction:	Inbound
(Internal) Transaction Type:	ONT
(Internal) Transaction Subtype:	CPO
External Transaction Type:	PO
External Transaction Subtype:	CANCEL
DTD Directory:	xml/oag72
Map Directory:	patch/115/xml/US
Message Maps XGM File Name:	ONT_3A9R_OAG72_IN.xgm
Standard:	OAG
Release:	7.2
Format:	DTD
DTD Name:	058_cancel_po_006.dtd

For the next three tables, refer to [Table 7–35, "Message Map Details"](#).

Table 7–36 Columns Enabled for Code Conversion

Column	Description
<UOM> (in schedule level <QUANTITY qualifier="ORDERED">)	Code Category – UOM, mapped to OE_ LINES_INTERFACE.ORDER_ QUANTITY_UOM

Key: H => OE_HEADER_INTERFACE, L => OE_LINES_INTERFACE

Table 7–37 Defaulted Columns

Defaulted Columns	Default Value, and Condition (if any)
H.CHANGE_REASON	'Not provided' (if Header-level DESCRIPTN tag is NULL>
H.OPERATION_CODE	UPDATE'
H.ORDER_SOURCE_ID	20
H.XML_TRANSACTION_TYPE_CODE	'CPO'
L.CHANGE_REASON	'Not provided' (if Line-level DESCRIPTN tag is NULL>
L.OPERATION_CODE	'UPDATE'
L.ORDERED_QUANTITY	0 (if derived ORDERED_QUANTITY is NULL)
L.ORDER_SOURCE_ID	20
L.ORIG_SYS_DOCUMENT_REF	H.ORIG_SYS_DOCUMENT_REF
L.XML_TRANSACTION_TYPE_CODE	'CPO'

Table 7–38 Derived Columns

Derived Columns	Source of Data, and Condition (if any)
H.CANCELLED_FLAG	Set to 'Y', if no lines are passed
H.CREATED_BY	FND_GLOBAL.LOGIN_ID
H.CREATION_DATE	SYSDATE
H.LAST_UPDATED_BY	FND_GLOBAL.LOGIN_ID
H.LAST_UPDATE_DATE	SYSDATE
H.ORG_ID	PARTNRIDX of SoldTo PARTNER, if this value is not null
H.SOLD_TO_ORG_ID	PARTNRIDX of SoldTo PARTNER
H.XML_MESSAGE_ID	XML Gateway Internal Control Number
L.CREATED_BY	FND_GLOBAL.LOGIN_ID
L.CREATION_DATE	SYSDATE

Table 7–38 Derived Columns

Derived Columns	Source of Data, and Condition (if any)
L.LAST_UPDATED_BY	FND_GLOBAL.LOGIN_ID
L.LAST_UPDATE_DATE	SYSDATE
L.ORDERED_QUANTITY, L.ORDER_QUANTITY_UOM	OAG Derivation from schedule level <QUANTITY qualifier="ORDERED"> tag

Table 7–39 Workflow Event Setup

Detail	Value
Event Name	oracle.apps.ont.oi.po_inbound.create
Event Description	Order Management Generic Inbound Event
Subscription	R_OEOI_ORDER_IMPORT
Subscription Description	Oracle Order Management subscription to the event oracle.apps.ont.oi.po_inbound.create raised by the post process of the Process_PO XML mapping

Extension Tags

The following changes were made by Procurement as extensions on the XML documents sent to Order Management.

\$ont/xml/oag72/oagis_entity_extensions.dtd

Add the following tags:

<!ENTITY % DATETIME.EXCHRATE DATE "DATETIME">

Change <!ENTITY % SEG_DATETIME_QUALIFIERS_EXTENSION "OTHER">
to <!ENTITY % SEG_DATETIME_QUALIFIERS_EXTENSION "OTHER |
EXCHRATE DATE">

\$ont/xml/oag72/oagis_extensions.dtd

Add the following tags:

<!ELEMENT PAYMMETHOD (DESCRIPTN?, TERMID?)>

<!ELEMENT CREDITCRD
(CARDID?, NAME?, (%DATETIME.EXPIRATION;?)>

<!ELEMENT STARTACTIVEDATE %STRDOM;>

```
<!ELEMENT ENDACTIVEDATE %STRDOM;>
<!ELEMENT CATEGORYID %STRDOM;>
<!ELEMENT REVISIONNUM %STRDOM;>
<!ELEMENT ATTACHMENT (TEXT?)>
<!ELEMENT EXCHRATE %STRDOM;>
<!ELEMENT CONFIRM %STRDOM;>
<!ELEMENT PCARDHDR
(MEMBERNAME?,PCARDNUM?,( (%DATETIME.EXPIRATION;)?,PCARDBRA
ND?)>
<!ELEMENT MEMBERNAME %STRDOM;>
<!ELEMENT PCARDNUM %STRDOM;>
<!ELEMENT PCARDBRAND %STRDOM;>
<!ELEMENT CUSTOMERNUM %STRDOM;>
<!ELEMENT REQUESTOR %STRDOM;>
<!ELEMENT CONTRACTPONUM %STRDOM;>
<!ELEMENT CONTRACTPOLINENUM %STRDOM;>
<!ELEMENT VENDORQUOTENUM %STRDOM;>
<!ELEMENT LISTPRICE %STRDOM;>
<!ELEMENT MARKETPRICE %STRDOM;>
<!ELEMENT PRICENOTTOEXCEED %STRDOM;>
<!ELEMENT NEGPRICE %STRDOM;>
<!ELEMENT TAXABLE %STRDOM;>
<!ELEMENT TXNREASONCODE %STRDOM;>
<!ELEMENT TYPE1099 %STRDOM;>
<!ELEMENT LINEORDERTYPE %STRDOM;>
<!ELEMENT HAZRDUNNUM %STRDOM;>
<!ELEMENT HAZRDUNDESC %STRDOM;>
<!ELEMENT PRICEOVRD %STRDOM;>
```

```

<!ELEMENT DISTPROJECT
(REQUESTOR?,DISTNUM?,PROJECTNUM?,PROJECTTYPE?,TASKNUM?,(%
Q
UANTITY.ORDERED;)?,CONVRATE,(%DATETIME.EXCHRATE;)?,DES
TTYPE?,DFFDISTRIBUTN?)>

<!ELEMENT PROJECTNUM %STRDOM;>

<!ELEMENT DISTNUM %STRDOM;>

<!ELEMENT PROJECTTYPE %STRDOM;>

<!ELEMENT TASKNUM %STRDOM;>

<!ELEMENT CONVRATE %STRDOM;>

<!ELEMENT DESTTYPE %STRDOM;>

<!ELEMENT DFFPOHEADER
(ATTRIBUTE1?,ATTRIBUTE2?,ATTRIBUTE3?,ATTRIBUTE4?,ATTRIBUTE5?,A
TTTRIBUTE6?,ATTRIBUTE7?,ATTRIBUTE8?,ATTRIBUTE9?,ATTRIBUTE10?,AT
TRIBUTE11?,ATTRIBUTE12?,ATTRIBUTE13?,ATTRIBUTE14?,ATTRIBUTE15?,
ATTRIBUTE16?)>

<!ELEMENT
DFFVENDORSITE(ATTRIBUTE1?,ATTRIBUTE2?,ATTRIBUTE3?,ATTRIBUTE4
?,ATTRIBUTE5?,ATTRIBUTE6?,ATTRIBUTE7?,ATTRIBUTE8?,ATTRIBUTE9?,
ATTRIBUTE10?,ATTRIBUTE11?,ATTRIBUTE12?,ATTRIBUTE13?
,ATTRIBUTE14?,ATTRIBUTE15?,ATTRIBUTE16?)>

<!ELEMENT DFFVENDOR
(ATTRIBUTE1?,ATTRIBUTE2?,ATTRIBUTE3?,ATTRIBUTE4?,ATTRIBUTE5?,A
TTTRIBUTE6?,ATTRIBUTE7?,ATTRIBUTE8?,ATTRIBUTE9?,ATTRIBUTE10?,AT
TRIBUTE11?,ATTRIBUTE12?,ATTRIBUTE13?,ATTRIBUTE14?,ATTRIBUTE15?,
ATTRIBUTE16?)>

<!ELEMENT DFFLINE
(ATTRIBUTE1?,ATTRIBUTE2?,ATTRIBUTE3?,ATTRIBUTE4?,ATTRIBUTE5?,A
TTTRIBUTE6?,ATTRIBUTE7?,ATTRIBUTE8?,ATTRIBUTE9?
,ATTRIBUTE10?,ATTRIBUTE11?,ATTRIBUTE12?,ATTRIBUTE13?,ATTRIBUTE
14?,ATTRIBUTE15?,ATTRIBUTE16?)>

<!ELEMENT DFFITEM

```

(ATTRIBUTE1?,ATTRIBUTE2?,ATTRIBUTE3?,ATTRIBUTE4?,ATTRIBUTE5?,ATTRIBUTE6?,ATTRIBUTE7?,ATTRIBUTE8?,ATTRIBUTE9?

,ATTRIBUTE10?,ATTRIBUTE11?,ATTRIBUTE12?,ATTRIBUTE13?,ATTRIBUTE14?,ATTRIBUTE15?,ATTRIBUTE16?)>

<!ELEMENT KFFITEM

(ATTRIBUTE1?,ATTRIBUTE2?,ATTRIBUTE3?,ATTRIBUTE4?,ATTRIBUTE5?,ATTRIBUTE6?,ATTRIBUTE7?,ATTRIBUTE8?,ATTRIBUTE9?

,ATTRIBUTE10?,ATTRIBUTE11?,ATTRIBUTE12?,ATTRIBUTE13?,ATTRIBUTE14?,ATTRIBUTE15?,ATTRIBUTE16?,ATTRIBUTE17?,ATTRIBUTE18?,ATTRIBUTE19?,ATTRIBUTE20?)>

<!ELEMENT

DDFDISTRIBUTN(ATTRIBUTE1?,ATTRIBUTE2?,ATTRIBUTE3?,ATTRIBUTE4?,ATTRIBUTE5?,ATTRIBUTE6?,ATTRIBUTE7?,ATTRIBUTE8?,ATTRIBUTE9?

,ATTRIBUTE10?,ATTRIBUTE11?,ATTRIBUTE12?,ATTRIBUTE13?,ATTRIBUTE14?,ATTRIBUTE15?,ATTRIBUTE16?)>

<!ELEMENT ATTRIBUTE1 %STRDOM;>

<!ELEMENT ATTRIBUTE2 %STRDOM;>

<!ELEMENT ATTRIBUTE3 %STRDOM;>

<!ELEMENT ATTRIBUTE4 %STRDOM;>

<!ELEMENT ATTRIBUTE5 %STRDOM;>

<!ELEMENT ATTRIBUTE6 %STRDOM;>

<!ELEMENT ATTRIBUTE7 %STRDOM;>

<!ELEMENT ATTRIBUTE8 %STRDOM;>

<!ELEMENT ATTRIBUTE9 %STRDOM;>

<!ELEMENT ATTRIBUTE10 %STRDOM;>

<!ELEMENT ATTRIBUTE11 %STRDOM;>

<!ELEMENT ATTRIBUTE12 %STRDOM;>

<!ELEMENT ATTRIBUTE13 %STRDOM;>

<!ELEMENT ATTRIBUTE14 %STRDOM;>

<!ELEMENT ATTRIBUTE15 %STRDOM;>

<!ELEMENT ATTRIBUTE16 %STRDOM;>
<!ELEMENT ATTRIBUTE17 %STRDOM;>
<!ELEMENT ATTRIBUTE18 %STRDOM;>
<!ELEMENT ATTRIBUTE19 %STRDOM;>
<!ELEMENT ATTRIBUTE20 %STRDOM;>
<!ELEMENT TANDC %STRDOM;>
<!ELEMENT GLOBALCONTRACT %STRDOM;>
<!ELEMENT GLOBALCONTRACTLIN %STRDOM;>
<!ELEMENT CONSIGNEDINV %STRDOM;>

Open Interface Tracking

Overview

The two main categories that comprise messages are Electronic Data Interchange (EDI) flat files and Open Applications Group (OAG) XML. The Open Interface Tracking form provides a unified view of Order Management's EDI and XML messages.

You can obtain the history of messages or delete historical records using a purge procedure. The form provides you with the current status and a textual message pertaining to processing of that message, or if using Oracle Workflow, you can view the diagram. Similarly, for XML messages, you can view the XML document. You can determine the specific outbound message (if any) triggered by a particular inbound message and vice versa.

This functionality will impact Order Import and the EDI/XML transaction processing. Both Order Import of closed orders and HVOP are not supported.

Order Management's Business Event System (BES)-based mechanism for other products (such as the Collaboration History module) captures and records the transaction history as XML messages exchanged between trading partners. The same mechanism has been extended to include other electronic messages and to make this information available via the form.

Major Features

Search For EDI/XML Electronic Messages

You can query up historical data for electronic messages received and processed by Order Management based upon a variety of search criteria such as Transaction Type, Order Source, etc.

View XML Data For a Particular Message

The XML document may convey important information, so you can navigate to the corresponding XML for a particular message.

View Workflow for XML Messages

You can navigate to the corresponding workflow for Order Management's XML messages.

Provides a Visual Cue To Associate Related Inbound and Outbound Messages

You can optionally group related Inbound and Outbound messages for a particular order. For instance, you can visually associate the inbound Cancel PO XML message with its corresponding outbound Confirm BOD and Acknowledgment. Similarly, an 850 Purchase Order Inbound EDI messages is associated with the corresponding 855 Purchase Order Acknowledgment Outbound.

Purge Procedure

You can purge the tracking data using a concurrent program. This concurrent program can also be used to purge the interface or acknowledgment tables.

Business Scenarios and Process Flow

INBOUND

1. For EDI/XML, the inbound Electronic Message is consumed using the appropriate Gateway and the Order Management Open Interface tables are populated. A record is created for this message with the information that data has been entered into the interface tables.
2. If the Electronic Message is processed using a workflow (i.e. for XML messages), then the record is updated with information regarding the current workflow status as the flow proceeds.
3. If Order Import (EDI/XML or standard) is run on the data, the record for the message is updated with information regarding the completion of Order Import.

OUTBOUND

1. Outbound EDI/XML Electronic Messages are generated via a concurrent program or when a business event is raised or when certain pre-determined attributes of the order are updated. When the triggering condition occurs, a record is created for this message with the information that the outbound Electronic Message will be generated.
2. If the Electronic Message is processed using a workflow (i.e. for XML messages), then the record is updated with information regarding the current workflow status as the flow proceeds.

- 3. When the appropriate Gateway generates and transmits the outbound Electronic Message, the record is updated with the information that the message has been sent.

The messages are logged as follow: a business event (**oracle.apps.ont.oi.xml_int.status**) is raised, and a seeded workflow subscription executes and writes the event information to a table.

Messages

This table is for any form-related messages that are seeded.

Table 7–40 Form Related Messages That Are Seeded

Code	Meaning
OE_OI_IFACE	&TRANSACTION - Data successfully entered into Open Interface Tables
OE_OI_IMPORT_MODE_SYNC	&TRANSACTION - Order Import will be run in synchronous mode
OE_OI_IMPORT_MODE_ASYNC	&TRANSACTION - Order Import will be run in asynchronous mode
OE_OI_IMPORT_SUCCESS_GEN	&TRANSACTION - Order Import successful
OE_OI_IMPORT_FAILURE	&TRANSACTION - Order Import failed
OE_OI_OUTBOUND_TRIGGERRED	OM is ready to send &TRANSACTION message
OE_OI_OUTBOUND_SETUP	&TRANSACTION - Setup Validation successful
OE_OI_OUTBOUND_SETUP_ERR	&TRANSACTION - Setup Validation failed

Profile Options

Table 7–41 Profile Options

Function	User Function Name	Description	Usage
ONT_EMINTEG_SOURCES	OM: Electronic Message Integration Event Sources	Order Sources enabled for Integration Business Event	Profile option to control Tracking

Electronic Message Status-Message Mapping

To provide a useful picture regarding the current processing status of an electronic message, the following messages will be written to the history table:

Table 7–42 Inbound XML Messages

Status	Message
Active	&TRANSACTION - Data successfully entered into Open Interface Tables
Active	&TRANSACTION - Order Import will be run in synchronous mode
Active	&TRANSACTION - Order Import will be run in asynchronous mode
Success/Error	&TRANSACTION - Order Import successful/failed

Table 7–43 Outbound XML Messages

Status	Message
Active	OM is ready to send &TRANSACTION message
Active/Error	&TRANSACTION - Setup Validation Successful/Failed
Success/Error	&TRANSACTION - message has been sent

Table 7–44 Inbound EDI Messages

Status	Message
Active	&TRANSACTION - Data successfully entered into Open Interface Tables
Success/Error	&TRANSACTION - Order Import successful/failed

Table 7–45 Outbound EDI Messages

Status	Message
Active	EDI Acknowledgment values derived
Success/Error	&TRANSACTION - message has been sent

Standard Order Import

Table 7–46 *Standard Order Import*

Status	Message
Success/Error	&TRANSACTION - Order Import successful/failed

Event/Event Subscription

The following event **oracle.apps.ont.oi.xml_int.status** is seeded

The following event subscription to the above event will also be seeded:

Seeded workflow OEM/Open Interface Tracking

The workflow (item type code OEM, name OM: Open Interface Tracking) will be seeded. This workflow will write the data from the event **oracle.apps.ont.oi.xml_int.status** to the table.

Event Parameters & Population

Event Parameters

The event **oracle.apps.ont.oi.xml_int.status** will contain the following parameters:

Table 7–47 *Event Parameters*

Parameter Name	Parameter Description	Comments
XMLG_INTERNAL_CONTROL_NUMBER	Unique document number generated by XML Gateway for inbound messages	Overloaded to provide a unique id for EDI inbounds and Order Import as well
XMLG_MESSAGE_ID	Unique message id generated by XML gateway for outbound messages	
XMLG_INTERNAL_TXN_TYPE	Typically 'ONT' except for Confirm BOD which is owned by 'ECX'	
XMLG_INTERNAL_TXN_SUBTYPE	Identifies XML transaction type	Overloaded to also contain EDI and Order Import transaction type, namely '850', '860', '855', '865', 'GENERIC'

Table 7-47 Event Parameters

Parameter Name	Parameter Description	Comments
DOCUMENT_DIRECTION	'IN' or 'OUT'	
XMLG_DOCUMENT_ID	Unique ID provided by OM for outbound messages	Overloaded to provide a unique id for outbound EDI transactions
TRADING_PARTNER_TYPE	'C'	
TRADING_PARTNER_ID	TP Party ID	
TRADING_PARTNER_SITE	TP Party Site ID	
DOCUMENT_NO	Sales Order Number	
ORG_ID	Organization ID	
PARTNER_DOCUMENT_NO	Purchase Order Number	
DOCUMENT_REVISION_NO	Change Sequence	
ONT_DOC_STATUS	'ACTIVE', 'SUCCESS', 'ERROR'	
MESSAGE_TEXT	FND MESSAGE detailing what has occurred.	
WF_ITEM_TYPE	Workflow Info for XML processing	
WF_ITEM_KEY	Workflow Info for XML processing	
ORDER_SOURCE_ID	Order Source	
SOLD_TO_ORG_ID	Customer ID	
ORDER_TYPE_ID	Order Type ID	
CONC_REQUEST_ID	Processing Concurrent Request ID	

Table 7–47 Event Parameters

Parameter Name	Parameter Description	Comments
PROCESSING_STAGE	Code to identify why this event was raised. Sample values include 'INBOUND_IFACE', 'OUTBOUND_SENT'.	
SUBSCRIBER_LIST	Comma delimited list of product short names of the intended subscribers e.g. 'ONT,CLN' <i>see Closed Issue #7</i>	
DOCUMENT_STATUS	'SUCCESS', 'ERROR' <i>See Closed Issue #7</i>	

Not every parameter will be populated in all cases. The following table contains a Y if a particular parameter is populated

Table 7–48 Populated Parameters

Parameter Name	INBOUND - Parameter Populated (Electronic Message Types)	OUTBOUND - Parameter Populated (Electronic Message Types)
XMLG_INTERNAL_CONTROL_NUMBER	Y (XML, EDI, Order Import)	
XMLG_MESSAGE_ID		Y (XML, if document sent successfully)
XMLG_INTERNAL_TXN_TYPE	Y (XML, EDI, Order Import)	Y (XML, EDI)
XMLG_INTERNAL_TXN_SUBTYPE	Y (XML, EDI, Order Import)	Y (XML, EDI)
DOCUMENT_DIRECTION		Y (XML, EDI)
XMLG_DOCUMENT_ID		Y (XML, EDI)
TRADING_PARTNER_TYPE		Y (XML, except CBOD)
TRADING_PARTNER_ID		Y (XML, except CBOD)
TRADING_PARTNER_SITE		Y (XML, except CBOD)
DOCUMENT_NO	Y (XML, EDI, Order Import) (if import succeeds)	Y (XML if import succeeds or sales order already exists, EDI)

Table 7–48 Populated Parameters

Parameter Name	INBOUND - Parameter Populated (Electronic Message Types)	OUTBOUND - Parameter Populated (Electronic Message Types)
ORG_ID	Y (XML, EDI, Order Import)	Y (XML, EDI)
PARTNER_DOCUMENT_NO	Y (XML, EDI, Order Import)	Y (XML, EDI)
DOCUMENT_REVISION_NO	Y (XML, EDI, Order Import)	Y (XML, EDI)
DOCUMENT_STATUS	Y (XML, EDI, Order Import)	Y (XML, EDI)
ONT_DOC_STATUS	Y (XML, EDI, Order Import)	Y (XML, EDI)
MESSAGE_TEXT	Y (XML, EDI, Order Import)	Y (XML, EDI)
WF_ITEM_TYPE	Y (XML)	Y (XML, EDI *)
WF_ITEM_KEY	Y (XML except when PROCESSING_STAGE is IMPORT_SUCCESS or IMPORT_FAILURE)	Y (XML, EDI *)
ORDER_SOURCE_ID	Y (XML, EDI, Order Import)	Y (XML, EDI)
SOLD_TO_ORG_ID	Y (XML, EDI, Order Import)	Y (XML, EDI)
ORDER_TYPE_ID	Y (XML, EDI, Order Import) (if import succeeds)	Y (XML if import succeeds or sales order already exists, EDI)
CONC_REQUEST_ID	Y (XML, EDI, Order Import)	Y (XML if Conc Show SO)
PROCESSING_STAGE	Y (XML, EDI, Order Import)	Y (XML, EDI)
SUBSCRIBER_LIST	Y (XML, EDI, Order Import)	Y (XML, EDI)
RESPONSE_FLAG	Y (XML only for Change PO response)	

The following table maps the value of the event parameter PROCESSING_STAGE to the message text for that stage:

Table 7–49 Processing_Stage

XML	EDI	Order Import	Processing Stage Description (CODE)	Sample Message Text
Y	Y		Data received (INBOUND_IFACE)	“Process PO – Data successfully entered into Open Interface tables” “850 POI – Data successfully entered into Open Interface tables”
Y			Import mode (PRE_IMPORT)	“Process PO – Order Import will be run in Asynchronous mode” “Process PO – Order Import will be run in Synchronous mode”
Y	Y	Y	Order Import result (IMPORT_SUCCESS, IMPORT_FAILURE)	“Process PO - Order Import successful” “Process PO – Order Import failed”
Y	Y		Outbound message triggered (OUTBOUND_TRIGGERED)	“OM is ready to send the Acknowledge PO message” “OM is ready to send the Show SO message”
Y	Y *		Outbound setup result (OUTBOUND_SETUP)	“Acknowledge PO – Setup validation successful” “Acknowledge PO – Setup validation failed” “EDI Acknowledgment values derived”
Y	Y		Outbound sent (OUTBOUND_SENT)	“Acknowledge PO message has been sent” “855 POAO message has been sent”

Note: For Outbound_Triggered and Outbound_Setup, the EDI support is only active when the Pack J EDI workflow is being used.

Electronic Messaging Implementation Considerations

- [Process_PO](#) on page 8-2
- [Acknowledge_PO](#) on page 8-23
- [Show_SalesOrder](#) on page 8-29
- [Change_SalesOrder](#) on page 8-43
- [Change_PO](#) on page 8-48
- [Cancel_PO](#) on page 8-63
- [Open Interface Tracking](#) on page 8-68

Process_PO

Setup

Customer Setup

Using the Oracle Order Management Super User responsibility, Customer Standard window (Customers-> Standard), query a customer to create XML orders for. On one of the customer's addresses, create a primary **Sold To** site usage and specify an EDI Location (e.g. Q7Q-1A1). Save your work. When provided on the inbound XML document, this information will be used to denote a particular customer site and also to determine the destination of the acknowledgment.

Figure 8–1 Customer Addresses Window

Customers - Standard

Customer Addresses - Computer Service and Rentals, 1006

Country: **United States** Site Number: **1034** ☒

Address: **301 Summit Hill Drive**

Alternate Name:

City: **Chattanooga** State: **TN**

Postal Code: **37401** Province: County: **Hamilton**

EDI Location: **Q7Q-1A1** ☐ Identifying Address ☒ Active

Addressee:

Business Purposes | Characteristics | Communication | Contacts | Contacts : Roles | Bank Accounts

Usage	Location	Bill To Location	Primary	Active
<input checked="" type="checkbox"/> Bill To	Chattanooga (OPS)		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
<input checked="" type="checkbox"/> Ship To	Chattanooga (OPS)	Chattanooga (OPS)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
<input checked="" type="checkbox"/> Sold To			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
<input type="checkbox"/>			<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>			<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>			<input type="checkbox"/>	<input type="checkbox"/>

New Open

The following table lists the setup steps for Process_PO:

Table 8–1 Process_PO Setup Steps

Step	Description
1	Define Transactions
2	Define Trading Partner/Hub
3	Define Code Conversion
4	Setup the Oracle Workflow Business Event System

Define Transactions

Use Oracle XML Gateway to define a cross reference between the Oracle transaction name and the external transaction name. The external transaction name will be based on what is meaningful per the XML standard used by the recipient. The external transaction name will appear on the message envelope to support message transport.

Figure 8–2 Define Transactions Window

Define Transactions

Party Type

Transaction Type

Transaction Subtype

Transaction Description

Customer

ONT

POI

Purchase Order Inbound

External Transactions

Standard Code	Direction	External Transaction Type	External Transaction Subtype	Queue
OAG	IN	PO	PROCESS	APPS.ECX_IN_OAG_Q

Define Trading Partner/Hub

e-Business may be conducted directly with a business partner commonly known as a trading partner or via a hub such as Oracle Exchange where many buyers and sellers converge to conduct electronic commerce.

With Oracle XML Gateway, you can define the hub or the actual business partner as a trading partner. If you define the hub as the trading partner, you can identify all

the buyers and sellers who are conducting business on the hub as trading partners to the hub.

Included in the trading partner/hub definition are the following:

- Trading Partner/Hub name
- Message enabled
- Message confirmation requested
- Message map to use for message creation or consumption
- E-mail address of trading partner contact to notify for data errors
- Trading partner specific code conversion values
- Transport protocol - SMTP, HTTP, and HTTPS with credential and username/password as necessary

The Trading Partner Setup window is used to:

Enable messages for the trading partner by identifying the internal and external transaction type and transaction subtype codes, and the XML standard associated with the message.

Access the Trading Partner Code Conversion window.

Select a message map for the trading partner.

Identify the communications protocol and address for a message. Optionally, the user can be selected from a hub.

To set up a trading partner:

1. Navigate to the Define Trading Partner Setup form from the XML Gateway Responsibility by selecting Setup > Define Trading Partners.

Figure 8–3 Trading Partner Setup Window

Trading Partner Setup

Trading Partner Type

Customer

Trading Partner Name

Computer Service and Rentals

Trading Partner Site

50 King Street Toronto M5H3Y2

Company Admin Email

arjun.rihan@oracle.com

Code Conversion

Trading Partner Details

Transaction Type	Transaction SubType	Standard Code	External Transaction Type	External Transaction SubType	Direction	Map	Connection/Hub	Protocol Type
ECX	CBODO	OAG	BOD	CONFIRM	OUT	ECX_CBODO	DIRECT	SMTP
ONT	POI	OAG	PO	PROCESS	IN	ONT_3A4R_O		
ONT	CPO	OAG	PO	CANCEL	IN	ONT_3A9R_O		
ONT	POA	OAG	PO	ACKNOWLED	OUT	ONT_3A4A_O	DIRECT	SMTP
ONT	SSO	OAG	SALESORDE	SHOW	OUT	ONT_3A6_OA	DIRECT	SMTP

- 2. Enter the Trading Partner Type as Customer.
- 3. Select the Trading Partner and the Trading Partner Site.
- 4. Select the Transaction Type as ONT.
- 5. Select 'ONT_3A4R_OAG72_IN' as the Map.
- 6. Save your work.

Define Code Conversion

With Oracle XML Gateway, you can identify the cross reference of an Oracle code to something that is meaningful to the recipient or vice versa. Common examples of

Oracle e-Business Suite codes requiring code conversion are units of measure and currency code.

Code conversion values can be defined to be applied universally for all trading partners and all messages. Additionally, code conversion values can be defined for a specific XML standard or specific to a trading partner.

The Oracle XML Gateway code conversion function provides a method to cross-reference the codes defined in Oracle Applications to codes used by trading partners, the XML standard, or other standard codes in the transactions.

The Code Conversion Standard code determines which transaction standard these values are applied to. A Standard Code of UNIVERSAL applies these values across all Standards. In this case use the Code Conversion window from the Navigator Setup menu.

Figure 8–4 Standard Code Conversion Window

Category Code	Description
UNIT_PRICE_BASIS	Basis of Unit Price Code
UOM	
UPC	Uniform Product Code (UPC) Code

Standard Code	Oracle Value	Description	From Trading Partner Value	To Trading Partner Value	Standard	Data Seeded
OAG	EACH		ea	ea	<input checked="" type="checkbox"/>	<input type="checkbox"/>
OAG	GAL		gallon	gallon	<input checked="" type="checkbox"/>	<input type="checkbox"/>
OAG	PIECE		pc	pc	<input checked="" type="checkbox"/>	<input type="checkbox"/>
RosettaNet	EACH		EA	EA	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
UNIVERSAL	EACH		EA	EA	<input checked="" type="checkbox"/>	<input type="checkbox"/>
UNIVERSAL	PIECE		PC	PC	<input checked="" type="checkbox"/>	<input type="checkbox"/>
UNIVERSAL	SLAB		SL	SL	<input checked="" type="checkbox"/>	<input type="checkbox"/>
					<input type="checkbox"/>	<input type="checkbox"/>

Define the Subscribing System for the Event Subscription for the Event 'oracle.apps.ont.oi.po_inbound.create'

Subscriber: System - This will usually be the local system, the system where the subscription code is to run.

Note: This data will be available as the initial setup, and comes as an XML file and will be loaded to the customer system. The user of the system can change it to work for their system settings.

Setup the Oracle Workflow Business Event System

Oracle XML Gateway leverages the Oracle Workflow Business Event System to publish and subscribe to application business events of interest to automatically trigger message creation or consumption.

Seeded business events and event subscriptions to send outbound or consume inbound messages are delivered for all Oracle pre-built messages.

Note: The seeded event subscriptions may be configured during implementation to perform activities to address specific business requirements.

The XML Gateway Execution Engine interfaces with Oracle Workflow to actively notify the XML Gateway system administrator regarding system or process errors, or the trading partner contact for data errors.

The XML Gateway system administrator has the option to “retry” failed outbound processes, or “reprocess” failed inbound processes.

The following function is Internal to the XML Gateway Execution Engine:

Validate Trading Partner/Hub

Verify that the trading partner is defined and the required documents are enabled.

Message Set Up

To implement a message with a trading partner, use XML Gateway message set up to define the trading partner or hub, code conversion values, and internal to external transaction name cross references. In addition, you may identify the XML Gateway system administrator to notify for system or process errors.

Installation

This functionality is available as part of the Order Management 11.5.9

Loading of Message Map and DTD

Oracle message maps are delivered and installed as a part of Oracle Order Management file structure and schema. They are automatically loaded into the XML Gateway repository using the LoadMap program. The reports are available to verify the success of the loading process.

Enabling and Consuming the Business Event

The necessary business events and subscriptions are also delivered and installed as part of the schema, they are loaded by the driver.

Integrations

- Oracle XML Gateway
- Oracle Workflow
- Oracle Advanced Queuing

Message Map

Seeded Workflow for Inbound XML Messages

A pre-defined Workflow event is seeded to run Order Import immediately if the Profile Option OM: Run Order Import for XML is set to 'Synchronous.' This Workflow subscribes to the event raised by the Post Process of the XML Gateway. As part of the Workflow Process it will run Order Import for the XML message, that is currently being consumed. The following Workflow is designed using Workflow Builder 2.6

Figure 8–5 Order Import Flow - Generic

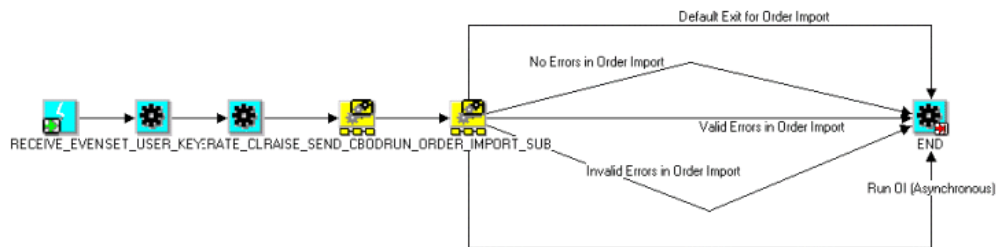


Figure 8–6 *Raise Send CBOD Out Sub Process*

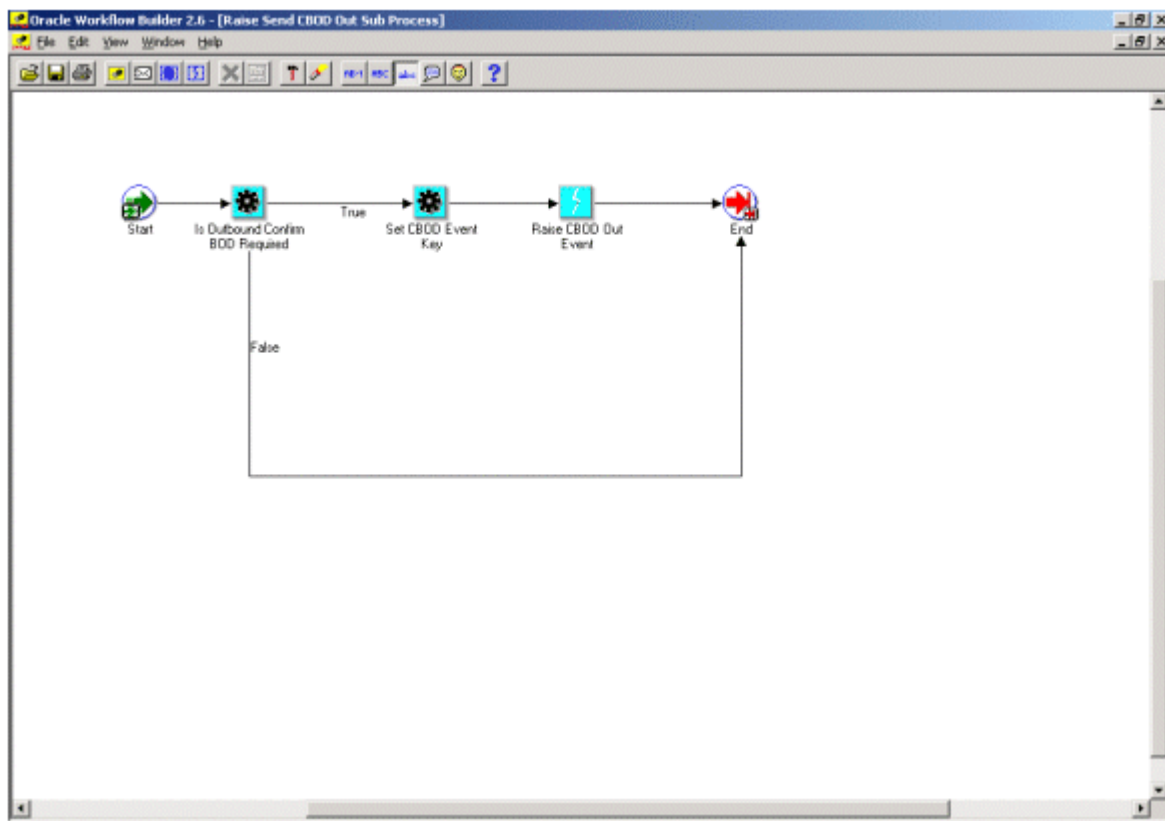


Figure 8-7 Run Order Import Sub Process

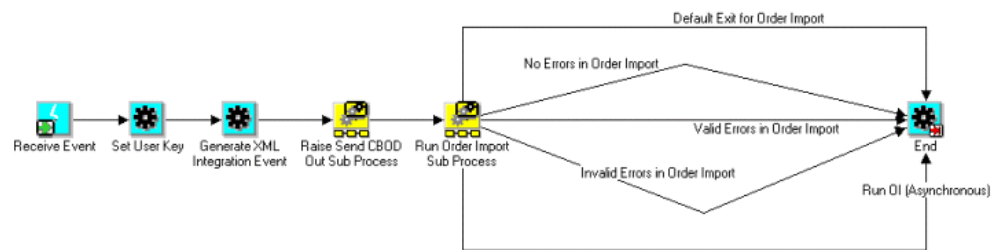
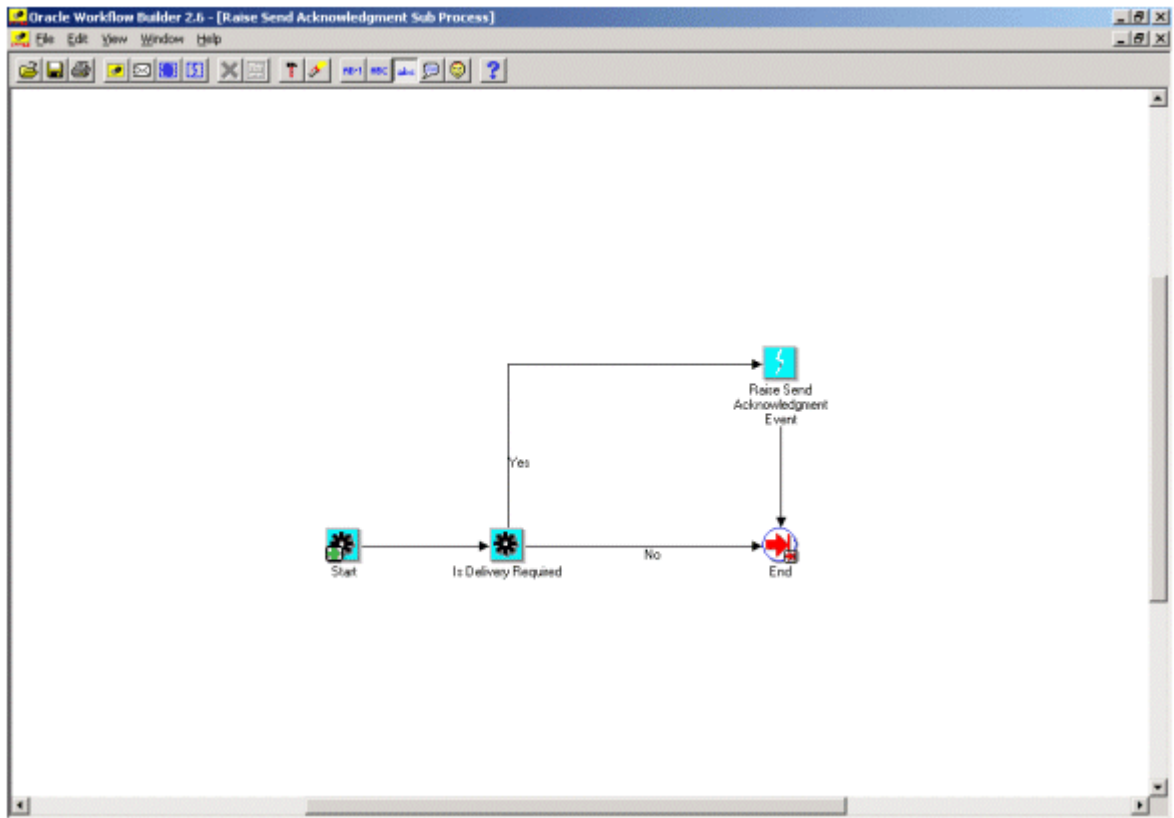


Figure 8–8 Raise Send Acknowledgement Sub Process



Sample Business Flow for 3A4 Process PO/3A4 Acknowledge PO

Table 8–2 Sample Business Flow for 3A4 Process PO/3A4 Acknowledge PO

Step	Buyer System E.G. Oracle Purchasing	Oracle Order Management
0		<p>Figure 8–9, "Trading Partner Setup completed for Oracle Order Management via XML Gateway forms"</p> <p>Figure 8–10, "Profile Option 'OM: Run Order Import for XML' set to 'SYNCHRONOUS' at site-level"</p>
1	Figure 8–11, "Buyer approves a Purchase Order"	
2		<p>Figure 8–12, "Order Management receives the Purchase Order electronically as a Process PO XML document"</p> <p>Figure 8–13, "Open Interface Tracking Details Main Tab"</p> <p>Figure 8–14, "Results Summary Inbound Message Details"</p>
3		<p>Figure 8–15, "Seeded inbound workflow is automatically triggered and runs Order Import"</p> <p>Figure 8–16, "Corresponding XML Sales Order is created Header"</p> <p>Figure 8–17, "Corresponding XML Sales Order is created Lines"</p>
4		<p>Figure 8–18, "Seeded outbound workflow is automatically triggered. This flow generates and sends Acknowledge PO to the buyer"</p> <p>Figure 8–19, "Status Diagram"</p> <p>Figure 8–20, "Transaction Monitor"</p> <p>Figure 8–21, "Results Summary"</p> <p>Figure 8–22, "Results Summary Outbound Message Details"</p>

Figure 8–9 Trading Partner Setup completed for Oracle Order Management via XML Gateway forms

Trading Partner Setup

Trading Partner Type: **Customer**

Trading Partner Name: **XMLCUST02**

Trading Partner Site: **Cyber XMLCUST02A Chattanooga TN 37401**

Company Admin Email: **rparamat@oracle.com**

Code Conversion

—Trading Partner Details—

Transaction Type	Transaction SubType	Standard Code	External Transaction Type	External Transaction SubType	Direction	Map	Connection/Hub	Protocol Type
ECX	CBOD0	OAG	BOD	CONFIRM	OUT	ECX_CBOD0	DIRECT	SMTP
ONT	POA	OAG	PO	ACKNOWLED	OUT	ONT_3A4A_O	DIRECT	SMTP
ONT	CPO	OAG	PO	CANCEL	IN	ONT_3A9R_O		
ONT	CHO	OAG	PO	CHANGE	IN	ONT_3A8R_O		
ONT	POI	OAG	PO	PROCESS	IN	ONT_3A4R_O		
ONT	CSO	OAG	SALESORDE	CHANGE	OUT	ONT_3A7_OA	DIRECT	SMTP
ONT	SSO	OAG	SALESORDE	SHOW	OUT	ONT_3A6_OA	DIRECT	SMTP

Figure 8–10 Profile Option 'OM: Run Order Import for XML' set to 'SYNCHRONOUS' at site-level

System Profile Values

Profile	Site	Application	Responsibility	User
OM: Run Order Import for XML	SYNCHRONOUS			

Figure 8–11 Buyer approves a Purchase Order

Purchase Orders (Vision Operations) - [New]

PO, Rev121220

TypeStandard Purchase

Created18-JUN-2004 18:19

SupplierABC Order Manag

SiteSUPPLIER SITE

Contact

Ship-ToV1- New York City

Bill-ToV1- New York City

CurrencyUSD

BuyerStock, Ms. Pat

StatusIncomplete

Total27,696.25

Description

Transaction Code

P-Card

Lines

Price Reference

Reference Documents

More

Agreement

Temporary Labor

Num	Rev	Job	Category	Description	UOM	Quantity
1			PRODUCTN.FIN	Dimension 4550	Each	10
2			PRODUCTN.FIN	Dimension 4550	Each	15

ItemAS54888

Dimension 4550

Catalog...

Currency...

Terms

Shipments

Approve...

Figure 8–12 Order Management receives the Purchase Order electronically as a Process PO XML document

The screenshot shows the 'Open Interface Tracking' window with the 'Main' tab selected. It contains a table with the following data:

Order Source	Order Source Reference	Customer	Customer Number
XML	PM.XPO-0617A	XMLCUST02	3908

A 'Details' button is located at the bottom right of the window.

Figure 8–13 Open Interface Tracking Details Main Tab

The screenshot shows the 'Open Interface Tracking Details' window with the 'Main' tab selected. It displays the following details:

Customer Number: 3908
Customer: XMLCUST02
Order Source Reference: PM.XPO-0617A
Order Source: XML
Message Count: 6

Transaction Type	Status	Message Text
Process PO	SUCCESS	Process PO - Order Import successful

At the bottom, there are three buttons: 'WF Monitor', 'XML Monitor', and 'Corrections'.

Figure 8–14 Results Summary Inbound Message Details



Figure 8–15 Seeded inbound workflow is automatically triggered and runs Order Import

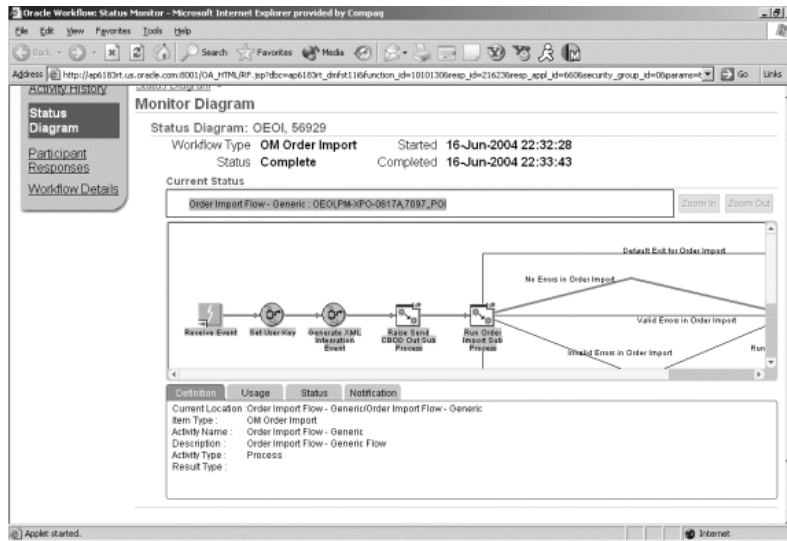


Figure 8–16 Corresponding XML Sales Order is created Header

Sales Orders (99014) - XMLCUST02

Order Information Line Items

Sales Quotes

Main Others

Customer	XMLCUST02	Order Number	99014
Customer Number	3908	Order Type	PM-Transaction01
Customer PO	PM-XPO-0617A	Date Ordered	16-JUN-2004 22:33:28
Customer Contact		Price List	Corporate
Ship To Location	7955	Salesperson	No Sales Credit
Quote Number		Status	Booked
	Chattanooga, TN, 37401, U	Currency	USD
Bill To Location	7953	Subtotal	1,200.00
	Cyber XMLCUST02A	Tax	0.00
	Chattanooga, TN, 37401, U	Charges	0.00
		Total	1,200.00

[] ..

Actions Related Items Configurator Availability Book Order

Figure 8–17 Corresponding XML Sales Order is created Lines

Sales Orders (70586) - Vision operations

Order Information Line Items

Firm

Order Total 16,767.76

Main Pricing Shipping Addresses Returns Services Others

Line	Ordered Item	Qty	Blanket Number	UOM	Unit Selling Price	Request Date	Sch
1.1	AS54888	5		Ea	1,117.85	20-JUN-2004 17:30:00	21-J
2.1	AS54888	5		Ea	1,117.85	20-JUN-2004 17:30:00	21-J

Line Total 5,589.25 Line Qty 5 Service Total 0.00

Description Dimension 4550

Actions Related Items Configurator Availability Book Order

Figure 8–18 Seeded outbound workflow is automatically triggered. This flow generates and sends Acknowledge PO to the buyer

Customer Number: 3908
 Customer: XMLCUST02
 Order Source Reference: PM-XP0-0617A
 Order Source: XML
 Message Count: 6

Main Others

Transaction Type	Status	Message Text
Process PO	SUCCESS	Process PO - Order Import successful
Acknowledge PO	SUCCESS	Acknowledge PO message has been sent.

WF Monitor XML Monitor Corrections

Figure 8–19 Status Diagram

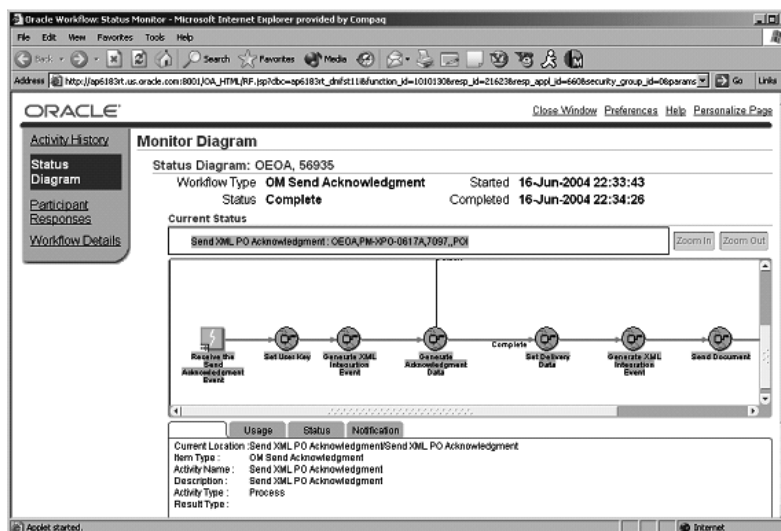


Figure 8–20 Transaction Monitor

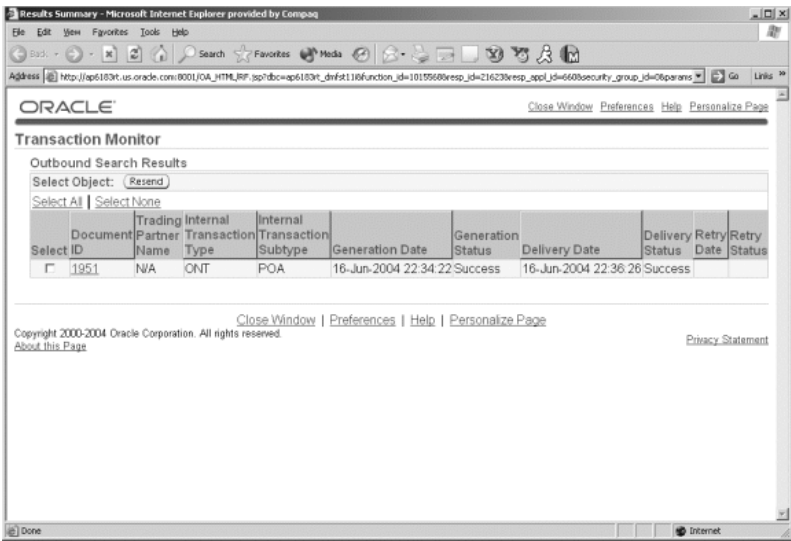


Figure 8–21 Results Summary

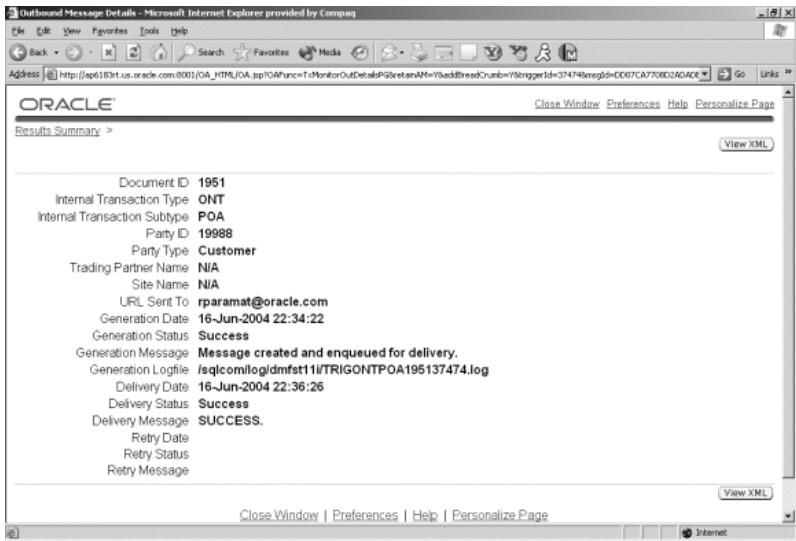
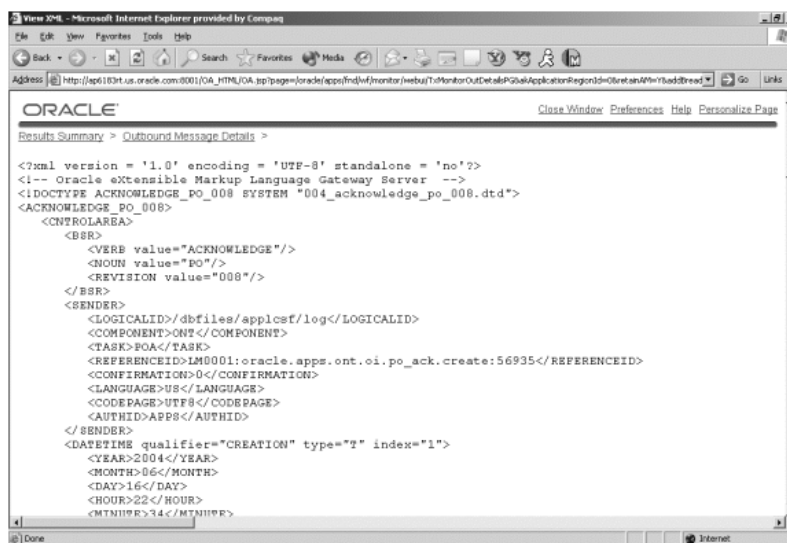


Figure 8–22 Results Summary Outbound Message Details

Acknowledge_PO

Loading of Message Map and DTD

Oracle message maps are delivered and installed as a part of the Oracle Order Management file structure and schema. They are automatically loaded into the XML Gateway repository using the LoadMap program. The reports are available to verify the success of the loading process.

Enabling and Consuming the Business Event

Create the Business Event and a Subscription for the event in the Application using the Workflow Administrator Responsibility. This event will be raised by the Post Process Action once the XML Message is consumed by Oracle XML Gateway and the data is loaded in the Open Interface Tables.

Setup

Customer Setup

Using the Oracle Order Management Super User responsibility, Customer Standard window (Customers-> Standard), query a customer to create XML orders for. On one of the customer’s addresses, create a primary **Sold To** site usage and specify an EDI Location (e.g. Q7Q-1A1). Save your work. When provided on the inbound XML document, this information will be used to denote a particular customer site and also to determine the destination of the acknowledgment.

Figure 8–23 Customer Addresses Window

Customers - Standard

Customer Addresses - Computer Service and Rentals, 1006

CountryUnited StatesSite Number1034

Address301 Summit Hill Drive

Alternate NameCityChattanoogaStateTN

Postal Code37401ProvinceCountyHamilton

EDI LocationQ7Q-1A1Identifying AddressActive

Addressee

Business Purposes

Characteristics

Communication

Contacts

Contacts : Roles

Bank Accounts

Usage	Location	Bill To Location	Primary	Active	
<input type="checkbox"/> Bill To	Chattanooga (OPS)		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
<input type="checkbox"/> Ship To	Chattanooga (OPS)	Chattanooga (OPS)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
<input checked="" type="checkbox"/> Sold To			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
<input type="checkbox"/>			<input type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/>			<input type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/>			<input type="checkbox"/>	<input type="checkbox"/>	

New

Open

Before performing these Order Management Suite XML setup steps, check the documentation for the following areas and verify that you have performed their set up:

- The Oracle Applications with which this transaction deals
- Oracle XML Gateway
- Oracle Workflow

Table 8–3 Setup Steps

Step	Description	New Transaction Development	Transaction Implementation
1	Define Transactions	Required	Optional
2	Define Trading Partner and Hub	NA	Required
3	Define Code Conversion	Required	Required
4	Setup Oracle Workflow Business Event System	NA	As Needed

Define Transactions

Use Oracle XML Gateway to define a cross reference between the Oracle transaction name and the external transaction name. The external transaction name will be based on what is meaningful per the XML standard used by the recipient. The external transaction name will appear on the message envelope to support message transport.

Define Trading Partner and Hub

e-Business may be conducted directly with a business partner commonly known as a trading partner or via a hub such as Oracle Exchange where many buyers and sellers converge to conduct electronic commerce.

With Oracle XML Gateway, you can define the hub or the actual business partner as a trading partner. If you define the hub as the trading partner, you can identify all the buyers and sellers who are conducting business on the hub as trading partners to the hub.

Included in the trading partner/hub definition are the following:

- Trading Partner/Hub name

- Message enabled
- Message confirmation requested
- Message map to use for message creation or consumption
- E-mail address of trading partner contact to notify for data errors
- Trading Partner specific code conversion values
- Transport protocol - SMTP, HTTP, or HTTPS with credential and username and password as necessary

Define Code Conversion

With Oracle XML Gateway, you may identify the cross reference of an Oracle code to something that is meaningful to the recipient or vice versa. Common examples of Oracle e-Business Suite codes requiring code conversion are units of measure and currency code.

Code conversion values may be defined to be applied universally for all trading partners and all messages. Additionally, code conversion values may be defined for a specific XML standard or specific to a trading partner.

Setup Oracle Workflow Business Event System

Oracle XML Gateway leverages the Oracle Workflow Business Event System to publish and subscribe to application business events of interest to automatically trigger message creation or consumption.

The following seeded event subscription, will listen to the event `oracle.apps.ont.oi.po_acknowledge.create` and will send the acknowledgment.

The XML Gateway Execution Engine interfaces with Oracle Workflow to actively notify the XML Gateway system administrator regarding system or process errors, or the trading partner contact for data errors.

The XML Gateway system administrator has the option to retry failed outbound processes, or reprocess failed inbound processes.

The raising of this event can be done, as part of the new workflow added for the Order Import Inbound flow. After Order Import (Run Order Import), finishes with the status complete we will Raise event, that will be received by above workflow for creating the acknowledgment.

The above workflow will be provided as standard flow (where the acknowledgment is sent when the synchronous flow of order import completes), the user can move the event and put it anywhere in their existing standard flow.

Message Set Up

To implement a message with a trading partner, use XML Gateway message set up to define the trading partner or hub, code conversion values, and internal to external transaction name cross references. In addition, you may identify the XML Gateway system administrator to notify for system or process errors.

The XML Gateway Execution Engine interfaces with Oracle Workflow to actively notify the XML Gateway system administrator regarding system or process errors, or the trading partner contact for data errors.

The XML Gateway system administrator has the option to “retry” failed outbound processes, or “reprocess” failed inbound processes.

The raising of this event can be done, as part of the new workflow added for the Order Import Inbound flow. After Order Import (Run Order Import), finish with status complete we will Raise event, which will be received by above workflow for creating the acknowledgment.

Oracle XML Gateway Details

This table lists the Oracle XML Gateway Details.

Table 8–4 Oracle XML Gateway Details

Detail	Value
Message Map Name:	ONT_3A4A_OAG72_OUT_PO
Direction:	Outbound
(Internal) Transaction Type:	ONT
(Internal) Transaction Subtype:	POA
External Transaction Type:	PO
External Transaction Subtype:	ACKNOWLEDGE
DTD Directory:	xml/oag72
Map Directory:	patch/115/xml/US
Message Maps XGM File Name:	ONT_3A4A_OAG72_OUT_PO.xgm
Standard:	OAG

Table 8–4 Oracle XML Gateway Details

Detail	Value
Release:	7.2
Format:	DTD
DTD Name:	004_acknowledge_po_008.dtd

USERAREAs

This table explains the data in the USERAREAs.

Table 8–5 USERAREA Data

Within Data Type	Data Element	Content
POORDERLIN	USERITEMDESCRIPTN	User Item Description
POLINESCHD	ACKCODE	Acknowledgment Code
POLINESCHD	DATETIME	Scheduled Arrival Date
POLINESCHD	SALESORDID	Sales Order Number
POLINESCHD	SOLINENUM	Line Number

Seed Codes for the Code Conversion

Following elements need to be enabled for code conversions:

- Currency
- UOM
- POType
- AckCode
- Termid
- Line Status

Show_SalesOrder

Setup

Customer Setup

Loading of Message Map and DTD

Oracle message maps are delivered and installed as a part of the Oracle Order Management file structure and schema. They are automatically loaded into the XML Gateway repository using the LoadMap program. The reports are available to verify the success of the loading process.

Enabling and Consuming the Business Event

Create the Business Event and a Subscription for the event in the application using the Workflow Administrator responsibility. This event will be raised by the either the WF process (WF Support), the Concurrent Program (Periodical Support), or through the Order Management internal processing (Attribute/Status Change Support).

Using the Oracle Order Management Super User responsibility, Customer Standard window (Customers-> Standard), query a customer to create XML orders for. On one of the customer's addresses, create a primary **Sold To** site usage and specify an EDI Location (e.g. Q7Q-1A1). Save your work. When provided on the inbound XML document, this information will be used to denote a particular customer site and also to determine the destination of the acknowledgment.

Figure 8–24 Customer Addresses Window

Customers - Standard

Customer Addresses - Computer Service and Rentals, 1006

Country: **United States** Site Number: **1034** ☒

Address: **301 Summit Hill Drive**

Alternate Name: City: **Chattanooga** State: **TN**

Postal Code: **37401** Province: County: **Hamilton**

EDI Location: **Q7Q-1A1** ☐ Identifying Address ☒ Active

Addressee:

Business Purposes Characteristics Communication Contacts Contacts : Roles Bank Accounts

Usage	Location	Bill To Location	Primary	Active
<input checked="" type="checkbox"/> Bill To	Chattanooga (OPS)		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
<input checked="" type="checkbox"/> Ship To	Chattanooga (OPS)	Chattanooga (OPS)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
<input checked="" type="checkbox"/> Sold To			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
<input type="checkbox"/>			<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>			<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>			<input type="checkbox"/>	<input type="checkbox"/>

New Open

Table 8–6 Setup Steps

Step	Description
1	Define Transactions
2	Define Trading Partner/Hub
3	Define Code Conversion
4	Setup Oracle Workflow Business Event System
5	Setting Up Constraints

Define Transactions

Use Oracle XML Gateway to define a cross reference between the Oracle transaction name and the external transaction name. The external transaction name will be based on what is meaningful per the XML standard used by the recipient. The external transaction name will appear on the message envelope to support message transport.

Define Trading Partner/Hub

e-Business may be conducted directly with a business partner commonly known as a trading partner or via a hub such as Oracle Exchange where many buyers and sellers converge to conduct electronic commerce.

With Oracle XML Gateway, you can define the hub or the actual business partner as a trading partner. If you define the hub as the trading partner, you can identify all the buyers and sellers who are conducting business on the hub as trading partners to the hub.

Included in the trading partner/hub definition are the following:

- Trading Partner/Hub name
- Message enabled
- Message confirmation requested
- Message map to use for message creation or consumption
- E-mail address of trading partner contact to notify for data errors
- Trading Partner specific code conversion values
- Transport protocol - SMTP, HTTP, or HTTPS with credential and username/password as necessary
- Source Location Code

Define Code Conversion

With Oracle XML Gateway, you may relate the cross reference of an Oracle code to something that is meaningful to the recipient or vice versa. Common examples of Oracle e-Business Suite codes requiring code conversion are units of measure and currency code.

Code conversion values may be defined to be applied universally for all trading partners and all messages. Additionally, code conversion values may be defined for a specific XML standard or specific to a trading partner.

Setup Oracle Workflow Business Event System

Oracle XML Gateway leverages the Oracle Workflow Business Event System to publish and subscribe to application business events of interest, to automatically trigger message creation or consumption.

The following seeded event subscription will listen to the event `oracle.apps.ont.oi.showso.create` and will send the acknowledgment.

The XML Gateway Execution Engine interfaces with Oracle Workflow to actively notify the XML Gateway system administrator regarding system or process errors, or the trading partner contact for data errors.

The XML Gateway system administrator has the option to “retry” failed outbound processes, or “reprocess” failed inbound processes.

Workflow Support

The raising of this event can be done, as part of the any existing workflow available for Order Header (OEOH) or/and Order Line (OEOL) flow. The following example shows one of the positions where the raise event process can be inserted for sending the 3A6 in OEOH flow (after Booking), the process should be copied (or created same way) from the new workflow (OESO).

Figure 8–25 OEOH OEOL Workflow



Note: This is currently only supported for the OEOH flow (Pack J onwards).

The Raise Show Sales Order Event Sub Process will check if the Sold To customer on the order is set up as a Trading Partner. If the customer is enabled for 3A6 Show SO, the sub-process will raise the event `oracle.apps.ont.oi.show_so.create` to generate and send the 3A6 Show SO XML document for XML orders.

In addition, the event `oracle.apps.ont.oi.xml_int.status` will be also be raised (for both XML and non-XML orders).

Table 8–7 Workflow for The Raise Event

Raise Event Position	-
Standard OM Workflow (OEOH/OEOL)	The workflow will continue to the point where the raise event is entered. When it hit raise event, the Show SO (3A6) with status will be send.

Figure 8–26 Workflow for the Raise Show Sales Order Event Sub Process



Validate Trading Partner/Hub

Verify that the trading partner is defined and required document are enabled.

Additional User Setup to Enable

ON Demand

Note: ON Demand generation of the Show_SalesOrder, can be achieved by running the concurrent program on an as needed basis (this means run the concurrent program to send the Show_SalesOrder).

Attribute Change

No User setup is required for the Show_SalesOrder in event of following attributes change:

- Unit selling Price
- Ordered Qty
- Scheduled arrival date
- Shipped Qty

- Scheduled Ship Date

Any time any of these attributes change for an order (after the order has been booked) Order Management will generate the Show_Salesorder message.

Adding a new line to a booked order will also generate a Show_Salesorder message.

Splitting a line on a booked order will also generate a Show_Salesorder message.

Status Change

No User setup is required for the Show_SalesOrder in event of following status change:

- Booking the order
- Shipping
- Scheduling of lines

When an order or line status changes, the show SO is generated.

Partner Segment Information for Partner type 'Supplier' Not Mapped

The Partner segment Supplier will be derived from the setup in XML Gateway. Trading partner setup allows the setup of source location code, which is a source trading partner location code for the particular trading partner and particular transaction (in many cases this could be the location code of the supplying warehouse). This data will be picked up by Order Management to populate the Supplier location code on the outbound Show_SalesOrder message.

Make sure that while performing the trading partner setup the source location for the outbound transaction (for Order Management) is set up. This source location needs to be a valid EDI location code that will be validated by the XML Gateway and the Oracle Application Server 10g (in case of Rosettanet implementations).

See the [User Procedures](#) section for additional information about this setup.

User Procedures

Defining Trading Partners

The Trading Partner Setup window is used to:

- Enable messages for the trading partner by identifying the internal and external transaction type and transaction subtype codes, and the XML standard associated with the message.

- Access the Trading Partner Code Conversion form.
- Select a message map for the trading partner.
- Identify the communications protocol and address for a message. Optionally, the user can be selected from a hub.

To define a trading partner:

1. Navigate to the Define Trading Partner Setup window from the XML Gateway Responsibility. Setup > Define Trading Partners.
2. Enter the Trading Partner Type, Customer.
3. Select the Trading Partner and Trading Partner Site.
4. Select the transaction type as 'ONT' for this functionality.
5. Select 'ONT_3A6_OAG72_OUT_SSO' as the map.
6. Save your work.

Define Code Conversion Values

The Oracle XML Gateway code conversion function provides a method to cross-reference the codes defined in Oracle Applications to codes used by trading partners, the XML standard, or other standard codes in the transactions.

The Code Conversion Standard code determines which transaction Standard these values are applied to. A Standard Code of 'UNIVERSAL' applies these values across all Standards. In this case run the Code Conversion form from the Navigator Setup menu.

Define the Subscribing System for the event subscription for event oracle.apps.ont.oi.show_so.create'

Subscriber: System - This will usually be the local system, the system on which the subscription code is to run.

Define the Source Trading Partner Location Code

Setup the Source trading partner EDI Location Code. This EDI location Code will be used as the Suppliers EDI Location code which is sent as the PARTNRIDX for Partner 'Ship From.'

This is as part of the Trading Partner Setup window. For each trading partner that you have, specify the supplier location code to send on the Outbound XML transaction.

Setting Up Constraints

This details the setups required for constraint-based generation of an XML integration event and the Show SO XML document.

You can setup a user action called 'Raise Integration Event' in the Constraints window. This can be set for any attribute that is available and for any operation.

This action will raise the 'oracle.apps.ont.oi.xml_int.status' workflow business event. Interested parties can subscribe to the event in a workflow to perform any desired actions. Also, if the Customer on the order is defined as a trading partner in XML Gateway and is enabled to receive the Show SO transaction, a Show SO XML document will be generated and sent to that Customer.

You can also use the validation templates to perform any necessary validations before raising the event and sending the XML document. For example, if the event should be raised only for a specific trading partner then a validation template can be utilized to check for the customer corresponding to that trading partner.

Constraint conditions can be defined to control exactly when your constraint will be triggered. Those conditions can be set up such that multiple conditions need to be satisfied for triggering (i.e. logical AND) or such that satisfaction of any particular condition will trigger the constraint (i.e. logical OR).

Through use of constraint-based generation, you have more flexibility to dynamically control the raising of integration events as well as the generation of the Show SO message. A detailed setup example can be found in the next section.

Please refer to *Oracle Order Management Implementation Manual* for more detailed information on how to setup constraints.

The following is an example of how to set up constraints to raise an Integration event if the salesperson on an order changes:

1. Navigate to the Processing Constraints window using Order Management Super User responsibility (Setup => Rules => Security => Processing Constraints).
2. Create the constraint. Query the constraints for the Application 'Oracle Order Management' and the Entity 'Order Header.'

Figure 8–27 Processing Constraints Window

Application: **Oracle Order Management**
Entity: **Order Header**

Constraints

Operation	Attribute	User Action	Applies To	System Changes	User Changes	Enabled	System
						<input checked="" type="checkbox"/>	<input type="checkbox"/>
						<input type="checkbox"/>	<input type="checkbox"/>
						<input type="checkbox"/>	<input type="checkbox"/>

Applicable To

Group #	Scope	Validation Entity	Record Set	Not	Validation Template	Enabled	System
				<input checked="" type="checkbox"/>		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
				<input type="checkbox"/>		<input type="checkbox"/>	<input type="checkbox"/>
				<input type="checkbox"/>		<input type="checkbox"/>	<input type="checkbox"/>

User Message:

3. Navigate to the Constraints section of the window and enter a new constraint with the following specifications:

Operation => **'UPDATE'** (this field specifies which operation will fire your constraint...i.e. INSERT, UPDATE, or DELETE)

Attribute => **'Salesperon'** (this field restricts your constraint to fire when the operation occurs on a particular attribute)

User Action => **'Raise Integration Event'** (this field specifies what will happen if your constraint is fired)

Enabled => *checked*

Figure 8–28 Processing Constraints Window - Constraints Region

The screenshot shows the 'Processing Constraints' window. At the top, the 'Application' is set to 'Oracle Order Management' and the 'Entity' is 'Order Header'. Below this is the 'Constraints' section, which contains a table with columns: Operation, Attribute, User Action, Applies To, System Changes, User Changes, Enabled, and System. There are three rows of constraints: 1) Operation: Delete, Attribute: (empty), User Action: Not Allowed, Applies To: (empty), System Changes: (empty), User Changes: (empty), Enabled: checked, System: checked. 2) Operation: Delete, Attribute: (empty), User Action: Not Allowed, Applies To: Negotiation, System Changes: (empty), User Changes: (empty), Enabled: checked, System: checked. 3) Operation: Update, Attribute: Salesperson, User Action: Raise Integ..., Applies To: (empty), System Changes: (empty), User Changes: (empty), Enabled: checked, System: unchecked. Below the table are tabs for 'Conditions' and 'Applicable To'. The 'Conditions' tab is active, showing a table with columns: Group #, Scope, Validation Entity, Record Set, Not, Validation Template, Enabled, and System. There is one row with Group # 1, Scope Any, Validation Entity (empty), Record Set (empty), Not unchecked, Validation Template (empty), Enabled checked, and System unchecked. At the bottom is a 'User Message' field.

Operation	Attribute	User Action	Applies To	System Changes	User Changes	Enabled	System
Delete		Not Allowed				<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Delete		Not Allowed	Negotiation			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Update	Salesperson	Raise Integ...				<input checked="" type="checkbox"/>	<input type="checkbox"/>

Group #	Scope	Validation Entity	Record Set	Not	Validation Template	Enabled	System
1	Any			<input type="checkbox"/>		<input checked="" type="checkbox"/>	<input type="checkbox"/>
				<input type="checkbox"/>		<input type="checkbox"/>	<input type="checkbox"/>
				<input type="checkbox"/>		<input type="checkbox"/>	<input type="checkbox"/>

User Message:

Once this constraint has been saved, any updates to the Header-level Salesperson on an order will raise an integration event.

The constraints framework also provides the flexibility to restrict this behavior to orders that satisfy particular conditions. For instance, you could set up your constraint so you only raise integration events for Salesperson changes for the customer “Computer Service and Rentals.” To do this you would need to define a condition on your constraint and, consequently, an appropriate validation template.

To create the necessary validation template:

- 1. Navigate to the Validation Templates window (Setup => Rules => Security => Validation Templates). Then create a new template with the following specifications:
Entity => ‘Order Header’
Template Name => ‘Customer Restriction Template’
Short Name => ‘CUSTRES’
Description => ‘Customer Restriction Template’

Validation Type => 'TBL'

Figure 8-29 Validation Templates Window - Validation Template Region

Entity	Template Name	Seeded?	Short Name	Description	WF	API	TBL
Order Header	Customer Restriction	<input type="checkbox"/>	CUSTRES	Customer Restriction Tem	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
		<input type="checkbox"/>			<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
		<input type="checkbox"/>			<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
		<input type="checkbox"/>			<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
		<input type="checkbox"/>			<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Column	Validation Operation	Value String
Customer	= (Equal To)	Computer Service and Rentals

2. Designate the Validation Type to be 'TBL', and enter the following information in the Validation Semantics section of the window:

Column => 'Customer'

Validation Operation => '= (Equal To)'

Value String => 'Computer Service and Rentals'

Figure 8–30 Validation Templates Window - Validation Semantics Region

Validation Templates

ApplicationOracle Order Management

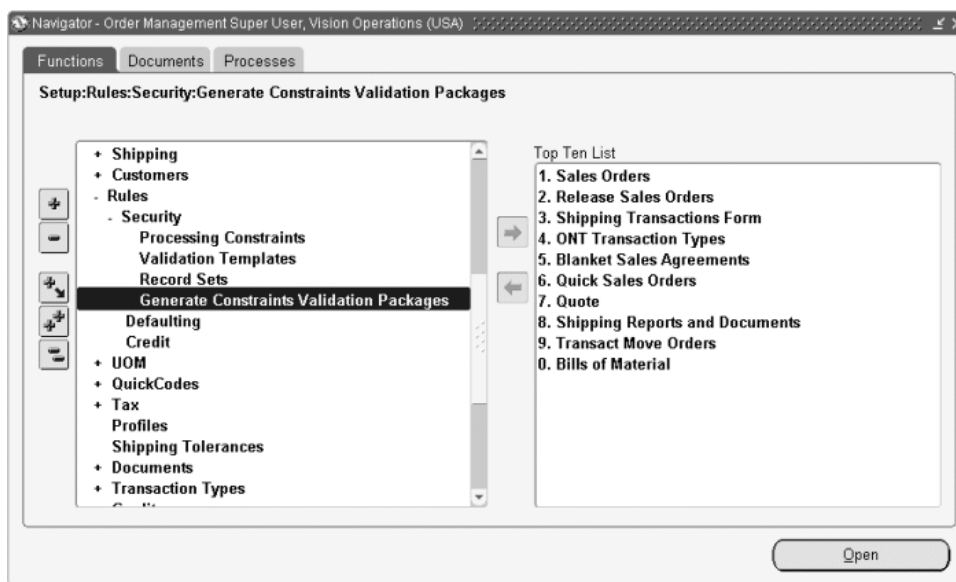
Validation Templates

Entity	Template Name	Seeded?	Short Name	Description	Validation Type		
		WF			API	TBL	
Order Header	Customer Restriction	<input type="checkbox"/>	CUSTRES	Customer Restriction Tem	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
		<input type="checkbox"/>			<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
		<input type="checkbox"/>			<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
		<input type="checkbox"/>			<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
		<input type="checkbox"/>			<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Validation Semantics

Column	Validation Operation	Value String
Customer	= (Equal To)	Computer Service and Rentals

3.
- After saving the validation template, you must run the Generate Constraints Validation Packages concurrent program to make your validation template available in the Constraints window (Setup => Rules => Security => Generate Constraints Validation Packages).

Figure 8–31 Navigator - Functions Tab

After the concurrent process has completed successfully, navigate back to the Constraints window and query your Salesperson constraint. To add a Condition that utilizes your newly created validation template by entering the following information:

Group => '101'

Scope => 'Any'

Validation Entity => 'Order Header'

Validation Template => 'Customer Restriction Template'

Enabled => *checked*

User Message => 'An integration event is being raised as a result of an UPDATE to Salesperson'

Figure 8–32 Processing Constraints Window - Conditions Tab

Processing Constraints

Application

Oracle Order Management

Entity

Order Header

Constraints

Operation	Attribute	User Action	Applies To	System Changes	User Changes	Enabled	System
Update	Salesperson	Raise Integ...				<input checked="" type="checkbox"/>	<input type="checkbox"/>
						<input type="checkbox"/>	<input type="checkbox"/>
						<input type="checkbox"/>	<input type="checkbox"/>

Conditions

Applicable To

Group #	Scope	Validation Entity	Record Set	Not	Validation Template	Enabled	System
101	Any	Order Header	Order	<input type="checkbox"/>	Customer Restriction Templa	<input checked="" type="checkbox"/>	<input type="checkbox"/>
				<input type="checkbox"/>		<input type="checkbox"/>	<input type="checkbox"/>
				<input type="checkbox"/>		<input type="checkbox"/>	<input type="checkbox"/>

User Message

An integration event is being raised as a result of an UPDATE to Salesperson

After this condition is saved, only updates to the Salesperson on orders for Computer Service and Rentals will raise an integration event. If you wanted to add another restriction, for instance that the raise only occur after booking, you would simply add another condition. A seeded validation template called 'Booked' already exists and can be used for this purpose. Note that the Group # of the additional condition should be the same as the Group # of the previously added template. This will cause the conditions to be evaluated together as an AND condition. Differing Group #s would cause the conditions to be treated separately as an OR condition.

Figure 8–33

Application: **Oracle Order Management**
Entity: **Order Header**

Constraints

Operation	Attribute	User Action	Applies To	System Changes	User Changes	Enabled	System
Update	Salesperson	Raise Integ...				<input checked="" type="checkbox"/>	<input type="checkbox"/>
						<input type="checkbox"/>	<input type="checkbox"/>
						<input type="checkbox"/>	<input type="checkbox"/>

Conditions **Applicable To**

Group #	Scope	Validation Entity	Record Set	Not	Validation Template	Enabled	System
101	Any	Order Header	Order	<input type="checkbox"/>	Customer Restriction Template	<input checked="" type="checkbox"/>	<input type="checkbox"/>
101	Any	Order Header	Order	<input type="checkbox"/>	Booked	<input checked="" type="checkbox"/>	<input type="checkbox"/>
				<input type="checkbox"/>		<input type="checkbox"/>	<input type="checkbox"/>

User Message: **This is a booked order**

Once the new condition is saved, integration events will only be raised for updates to the salesperson on booked orders for Computer Service and Rentals.

Change_SalesOrder

Setup

Customer Setup

Table 8–8 Setup Steps

Step	Description
1	Define Transaction
2	Message Setup
3	Define Trading Partner/Hub

Table 8–8 Setup Steps

Step	Description
4	Define Code Conversion Values
5	Set the Profile Options
6	Loading of Message Map and DTD
7	Enabling and Consuming the Business Event

Define Transaction

Use Oracle XML Gateway to define a cross reference between the Oracle transaction name and the external transaction name. The external transaction name will base on what is meaningful per the XML standard used by the recipient. The external transaction name will appear on the message envelope to support message transport.

Message Setup

To implement a message with a Trading Partner, use the XML Gateway message set up to define the Trading Partner or hub, code conversion values, and internal to external transaction name cross references. In addition, you may identify the XML Gateway system administrator to notify for system or process errors. Assign the Notify PO Update transaction to the trading partner.

Define Trading Partner/Hub

E-Business may be conducted directly with a business partner commonly known as a Trading Partner or via a hub such as Oracle Exchange where many buyers and sellers converge to conduct electronic commerce.

With Oracle XML Gateway, you can define the hub or the actual business partner as a Trading Partner. If you define the hub as the Trading Partner, you can identify all the buyers and sellers who are conducting business on the hub as Trading Partners to the hub.

Included in this definition are the following:

- Trading Partner/Hub name
- Message enabled
- Message confirmation requested
- Message map to use for message creation or consumption

- E-mail address of Trading Partner contact to notify for data errors
- Trading Partner specific code conversion values
- Transport protocol - SMTP, HTTP, HTTPS with credential and username/password as necessary

Please refer to the Oracle XML Gateway documentation for further details.

To define a trading partner:

1. Navigate to the Define Trading Partner Setup window from the XML Gateway Responsibility. Setup > Define Trading Partners.
2. Enter the Trading Partner Type, Customer.
3. Select the Trading Partner and Trading Partner Site.
4. Select the transaction type as 'ONT' for this functionality.
5. Select 'ONT_3A7R_OAG72_OUT_SO' as the map.
6. Save your work.

Define Code Conversion Values

With Oracle XML Gateway, you can create a cross reference between an Oracle code to a meaningful concept to the recipient or vice versa. Common examples are units of measure and currency code.

Code conversion values can be defined to be universal for all Trading Partners and all messages. Additionally, code conversion values may be defined for a specific XML standard or specific to a Trading Partner.

Note: If consuming 3A8 Change PO Response XML documents sent by Oracle Purchasing, you must convert the value Response to the Oracle value of Y using the seeded Code Category DOC_PURPOSE_CODE. This ensures that the column Response_Flag in OE_HEADERS_INTERFACE is populated correctly.

Set the Profile Options

All these profile options are site level.

Set the 'OM: Change SO Response Required' the default value for this profile is No.'

Set the value of the profile `ONT_3A7_RESPONSE_REQUIRED` to either Yes or No. Based on this profile the system will decide whether to put the order line on hold after the changes are triggered and then wait for either a 3A8 response or a 3A9.

Set the 'OM: Order Accept State for XML' profile.

Based on this profile the system decides what status is sent on the outbound Acknowledgement. The default value of this profile is Entered.

This is also governed by the value in the tag `<Booked Flag>` that is mapped in the inbound transactions. The value in this tag determines whether the order is Booked or not when it is created. The tag will have a value of Y or N.

Set the 'OM: Send Acknowledgment for Change PO Response' profile.

Set the value of the profile `ONT_3A8_RESPONSE_ACK` to either 'Yes' or 'No' (default value is 'No').

Based on this profile the system will determine whether or not an Acknowledge PO message is sent when a 3A8 Change PO response message is received. This profile does not govern the behavior Acknowledge PO for normal 3A8 Change PO (i.e. non-response) messages.

Table 8–9 Profile Options

Profile Option Name	User Profile Name	Description
ONT_3A7_RESPONSE_REQUIRED	OM: Change SO Response Required	To determine if the Seller should expect a response to an outbound Change SO XML message
ONT_3A8_RESPONSE_ACK	OM: Send Acknowledgment for Change PO Response	Determines whether an Acknowledgment will be sent for the Change PO Response
ONT_XML_ACCEPT_STATE	OM: Order Accept State for XML	Determines the order Accept status for outbound XML messages

The following table determines the status send on the Outbound Acknowledgment.

Table 8–10 Status Send on the Outbound Acknowledgement

Profile	Booked Flag	Order Status	Acknowledgement Status
BOOKED	Y	Book the Order. Order in Booked state Order in Entered state	Accept Pending
BOOKED	N	Don't book the Order Order in Entered state	Pending
ENTERED	Y	Book the Order Order in Booked state Order in Entered state	Accept 'Accept'
ENTERED	N	Don't book the Order Order in Entered state	Accept

Loading of Message Map and DTD

3A7 transaction message maps are delivered and installed as part of Oracle Order Management file structure and schema. They are automatically loaded into the XML Gateway repository using the LoadMap program. The reports are available to verify the success of the loading process.

Enabling and Consuming the Business Event

The necessary business events and subscriptions are also delivered and installed as part of the schema. They are loaded by the driver.

Data Archive and Purge Procedures

Purge Data From the Acknowledgment Tables

Acknowledgment data is deleted from the Acknowledgment tables after the extraction. Data is left in the tables in case the outbound transaction fails.

Change_PO

Setup

Customer Setup

Table 8–11 Setup Steps

Step	Description
1	Define Transaction Type 'ONT' and Subtype 'CHO'
2	Define Trading Partners
3	Define the Subscribing System for the event subscription for event 'oracle.apps.ont.oi.po_inbound.create'
4	Message Set Up
5	Define Subscription for the Event 'oracle.apps.ont.oi.po_inbound.create'
6	Define System Profile Values

Define Transaction Type 'ONT' and Subtype 'CHO'

Use Oracle XML Gateway to define a cross reference between the Oracle transaction name and the external transaction name. The external transaction name will be based on what is meaningful per the XML standard used by the recipient. The external transaction name will appear on the message envelope to support message transport.

The transaction type ONT and subtype CHO is seeded for this functionality.

Navigate to the Define Transactions window from the XML Gateway Responsibility by selecting Setup /rt.jar:\$Workflow_JAR_file_directory/wfjava.jar: \$Workflow_JAR_file_directory/jdbc/lib/classes111.zip:" oracle.apps.fnd.wf.WFXLoad -d user connect_string lang objectJava

Define Trading Partners

The Trading Partner Setup window is used to:

Enable messages for the trading partner by identifying the internal and external transaction type and transaction subtype codes, and the XML standard associated with the message.

Access the Trading Partner Code Conversion form.

Select a message map for the trading partner.

Identify the communications protocol and address for a message. Optionally, the user can be selected from a hub.

Navigate to the Define Trading Partner Setup window from the XML Gateway Responsibility by selecting Setup DB password:Port mymap.xgm DB password:Port mapname Define XML Standards.

Included in the trading partner/hub definition are the following:

- Trading Partner/Hub name
- Message confirmation requested
- Message map to use for message creation or consumption
- E-mail address of trading partner contact to notify for data errors
- Trading Partner specific code conversion values
- Transport protocol - SMTP, HTTP, or HTTPS with credential and username/password as necessary
- Enable messages for the trading partner by identifying the internal and external transaction type and transaction subtype codes, and the XML standard associated with the message.

To define a trading partner:

1. Navigate to the Define Trading Partner Setup window from the XML Gateway Responsibility. Setup > Define Trading Partners.
2. Enter the Trading Partner Type, Customer.
3. Select the Trading Partner and Trading Partner Site.
4. Select the transaction type as 'ONT' for this functionality.
5. Select 'ONT_3A8R_OAG72_IN' as the map.
6. Save your work.

Define the Subscribing System for the event subscription for event 'oracle.apps.ont.oi.po_inbound.create'

Subscriber: System - This will usually be the local system, the system on which the subscription code is to run.

(This data will be available as the initial setup, it will be an XML file and is loaded to the customer system. The user of the system can change it to work for their system settings).

Define Subscription for the Event ‘oracle.apps.ont.oi.po_inbound.create’

Create a subscription for the event ‘oracle.apps.ont.oi.po_inbound.create’. The event is raised in the Post Process by the XML Gateway engine after the Inbound XML Message is consumed and the data is loaded in the OM Interface tables. This subscription will then consume this event and in turn run a pre-defined workflow.

Use the Oracle XML Workflow Loader utility to download and upload XML definitions for Business Event System objects. When you download Business Event System object definitions from a database, Oracle Workflow saves the definitions as an XML file. This XML file will be delivered to the Customer as part of the Seed Data.

On UNIX, use the following commands to run the Workflow XML Loader:

```
jre -classpath "$JREPATH>/rt.jar:$Workflow_JAR_file_directory>:  
$Workflow_JAR_file_directory>/wfjava.jar: $Workflow_JAR_file_  
directory>/wfapi.jar:  
$ORACLE_HOME>/jdbc/lib/classes111.zip:" oracle.apps.fnd.wf.WFXLoad -d  
user>password> connect_string> protocol> lang> output_file> object> key>Java
```

Define System Profile Values

The following table lists the System Profile options, which need to be set.

Table 8–12 System Profile Values

Profile System Option	Description	Required	Default Value
OM: Run Order Import for XML	Determines whether Order Import will run Asynchronously or Synchronously	No	Synchronous

The directories listed in the above profiles should also be mentioned in the utl_file_dir parameter in init.ora

Please refer to the Oracle XML Gateway User’s Guide for the setup information.

Message Set Up

To implement a message with a trading partner, use XML Gateway message set up to define the trading partner or hub, code conversion values, and internal to external transaction name cross references. In addition, you may identify the XML Gateway system administrator to notify for system or process errors.

Pre-Defined Workflow Event

A pre-defined Workflow event is seeded to run Order Import immediately if the Profile Option OM: Run Order Import for XML is set to 'Synchronous.' This Workflow subscribes to the event raised by the Post Process of the XML Gateway. As part of the Workflow Process it will run Order Import for the XML Message, which is currently being consumed.

Implementation Considerations

Loading of Message Map and DTD

Oracle message maps are delivered and installed as a part of Oracle Order Management file structure and schema. They are automatically loaded into the XML Gateway repository using the LoadMap program. The reports are available to verify the success of the loading process.

Enabling and Consuming the Business Event

The necessary business events and subscriptions are also delivered and installed as part of the schema. They are loaded by the driver.

To install this functionality, the required prereq patch for the XML gateway should be installed first.

Message Map

Note: Most data values in the message map are seeded.

Message Map 'ONT_3A8 R_OAG72_IN.xgm'

The message map 'ONT_3A8R_OAG72_IN' is created using the Oracle XML Gateway Message Designer tool.

The source DTD used is 058_cancel_po_006.dtd, revision 7.2.1 of the Open Application Group. The other associated external reference DTD files are:

oagis_domains.dtd
oagis_resources.dtd
oagis_fields.dtd
oagis_segments.dtd
oagis_extensions.dtd
oagis_entity_extensions.dtd

All the DTD's will be checked in ONT source area under \$ont/xml/oag72.

The target for the Inbound XML Message are the Order Management Open Interface tables OE_HEADERS_INTERFACE & OE_LINES_INTERFACE.

The INBOUND CHANGE_PO DTD is a three level hierarchy with Order Lines split into one or more Shipment Lines. The Order Management architecture is however a two level one with Order Header and one or more Lines. The message map will collapse the three level XML Message into a two level structure when the data is inserted in the Order Management Open Interface tables.

Both the message map created using the Message Designer and its associated DTD's are stored in the database. The following Java programs are available to Load/Delete Maps or Load/Delete DTD's into/from the XML Gateway repository. Please refer to the *Oracle XML Gateway Manual* for more information.

Load/Delete Maps, Load/Delete DTD's

Note: The following process is used only for customizations.

1. java LoadMap DB username> DB password> Hostname>:Port>:SID>
mymap.xgm>

Example: java oracle.apps.ecx.loader.LoadMap apps apps
ap505dbs:1521:dev115 ONT_3A8R_OAG72_IN.xgm
2. java DeleteMap DB username> DB password> Hostname>:Port>:SID>
mapname>

Example: java oracle.apps.ecx.loader.DeleteMap apps apps
ap505dbs:1521:dev115 ONT_3A8R_OAG72_IN.xgm

The Message Map is a .xgm file which will be stored in the Order Management Source area under \$ont/patch/115/map/ ONT_3A8R_OAG72_IN.xgm

Note: Maps and DTD's must be kept in sync between the Message Designer and the XML Gateway repository. When in doubt, always reload the map and DTD as a pair.

key: H => OE_HEADERS_INTERFACE, L => OE_LINES_INTERFACE

Sample Business Flow for 3A4 Process PO/3A4 Acknowledge PO

Table 8–13 Sample business flow for 3A7 Change SO/3A8 Change PO response

Step	Buyer System E.G. Oracle Purchasing	Oracle Order Management
0		Figure 8–34, "Trading Partner Setup completed for Oracle Order Management via XML Gateway forms" Figure 8–35, "Profile Option OM: Change SO Response Required set to 'Yes' at site-level"
1	Figure 8–36, "Buyer approves a Purchase Order"	
2		Figure 8–37, "Order Management receives the Purchase Order electronically as a Process PO XML document and a corresponding sales order is createdHeader" Figure 8–38, "Order Management receives the purchase order electronically as a Process PO XML document and a corresponding sales order is created Line"

Table 8–13 Sample business flow for 3A7 Change SO/3A8 Change PO response

Step	Buyer System E.G. Oracle Purchasing	Oracle Order Management
3		<p>Figure 8–39, "After booking the order, the seller modifies the ordered quantity on one of the lines"</p> <p>Figure 8–40, "The order line goes on hold"</p> <p>Figure 8–41, "3A7 Change SO XML document is generated and sent to the buyer"</p> <p>Figure 8–42, "Results Summary Outbound Message Details"</p>
4	<p>Figure 8–43, "Buyer receives seller-initiated changes"</p> <p>Figure 8–44, "Buyer sends a 3A8 Change PO response acknowledging the changes"</p>	
5		<p>Figure 8–45, "Seller receives 3A8 Change PO response"</p> <p>Figure 8–46, "Results Summary Inbound Message Details"</p> <p>Figure 8–47, "3A8 Change PO response is processed successfully and the hold on the sales order line removed"</p>

Figure 8–34 Trading Partner Setup completed for Oracle Order Management via XML Gateway forms

Trading Partner Setup

Trading Partner Type: **Customer**

Trading Partner Name: **XMLCUST02**

Trading Partner Site: **Cyber XMLCUST02A Chattanooga TN 37401**

Company Admin Email: **rparamat@oracle.com**

Code Conversion

—Trading Partner Details—

Transaction Type	Transaction SubType	Standard Code	External Transaction Type	External Transaction SubType	Direction	Map	Connection/Hub	Protocol Type
ECX	CBOD0	OAG	BOD	CONFIRM	OUT	ECX_CBOD0	DIRECT	SMTP
ONT	POA	OAG	PO	ACKNOWLED	OUT	ONT_3A4A_O	DIRECT	SMTP
ONT	CPO	OAG	PO	CANCEL	IN	ONT_3A9R_O		
ONT	CHO	OAG	PO	CHANGE	IN	ONT_3A8R_O		
ONT	POI	OAG	PO	PROCESS	IN	ONT_3A4R_O		
ONT	CSO	OAG	SALESORDE	CHANGE	OUT	ONT_3A7_OA	DIRECT	SMTP
ONT	SSO	OAG	SALESORDE	SHOW	OUT	ONT_3A6_OA	DIRECT	SMTP

Figure 8–35 Profile Option OM: Change SO Response Required set to 'Yes' at site-level

Personal Profile Values

Profile Name	Default Value	User Value
OM: Change SO Response Req	Yes	

Figure 8–36 Buyer approves a Purchase Order

Purchase Orders (Vision Operations) - [New]

PO, Rev	12122 0	Type	Standard Purchase	Created	18-JUN-2004 18:19
Supplier	ABC Order Manag	Site	SUPPLIER SITE	Contact	
Ship-To	V1- New York City	Bill-To	V1- New York City	Currency	USD
Buyer	Stock, Ms. Pat	Status	Incomplete	Total	27,696.25
Description					
P-Card	Transaction Code				

Lines Price Reference Reference Documents More Agreement Temporary Labor

Num	Rev	Job	Category	Description	UOM	Quantity
1			PRODUCTN.FIN	Dimension 4550	Each	10
2			PRODUCTN.FIN	Dimension 4550	Each	15

Item AS54888 Dimension 4550

Catalog... Currency... Terms Shipments Approve...

Figure 8–39 After booking the order, the seller modifies the ordered quantity on one of the lines

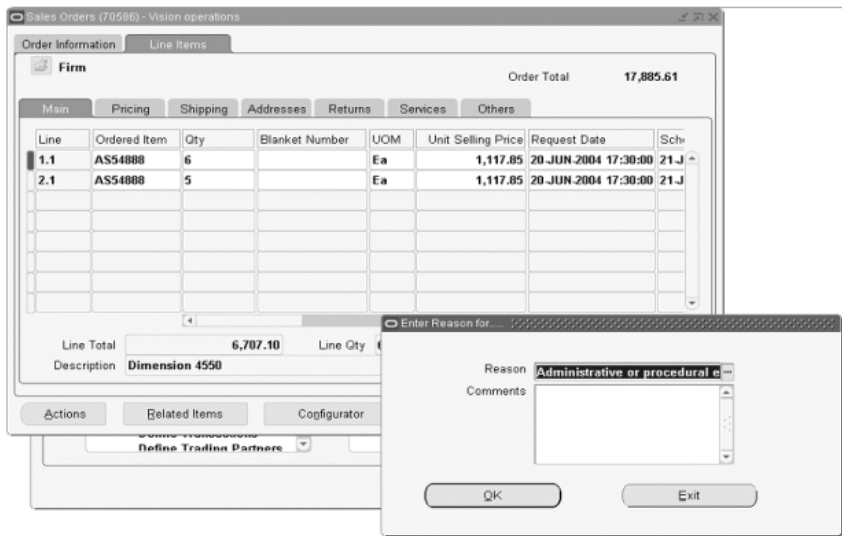


Figure 8–40 The order line goes on hold

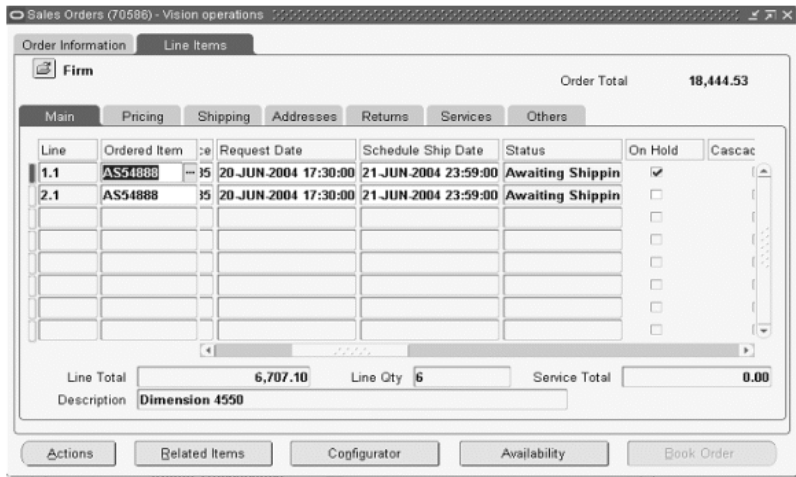


Figure 8-43 Buyer receives seller-initiated changes

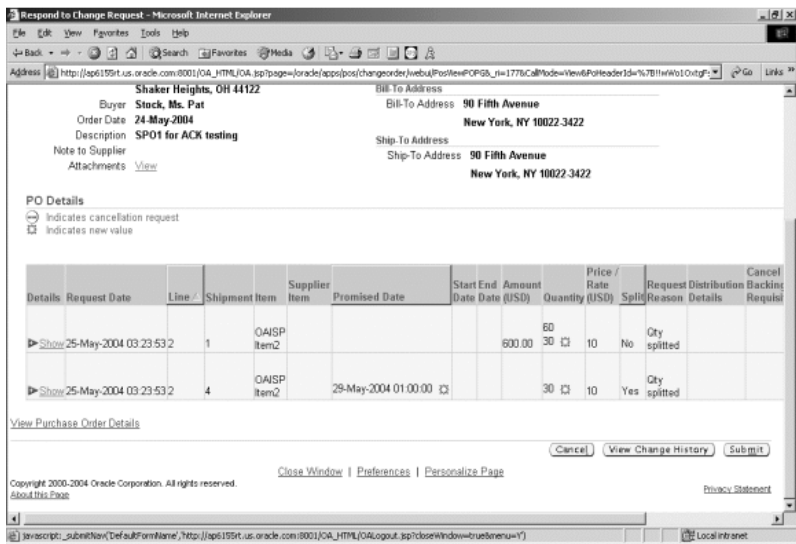


Figure 8-44 Buyer sends a 3A8 Change PO response acknowledging the changes

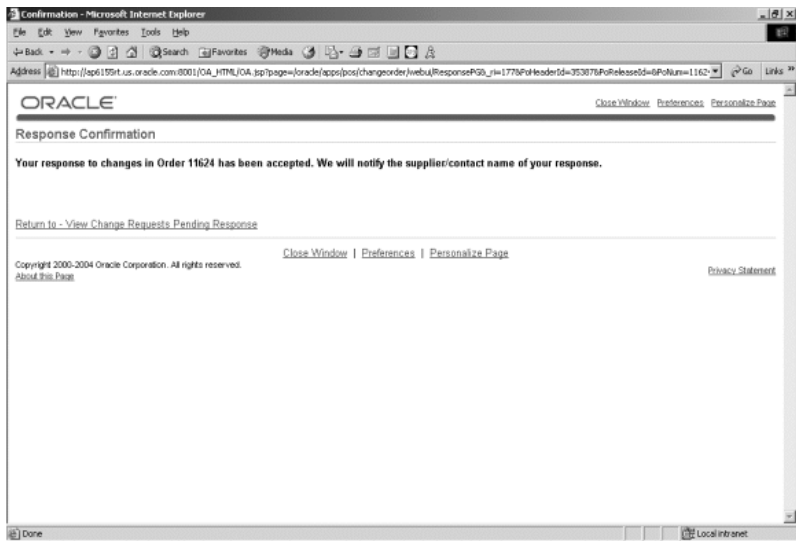


Figure 8–47 3A8 Change PO response is processed successfully and the hold on the sales order line removed

Sales Orders (70586) - Vision operations

Order Information Line Items

Firm Order Total **18,444.53**

Main Pricing Shipping Addresses Returns Services Others

Line	Ordered Item	Qty	Blanket Number	UOM	Unit Selling Price	Request Date	Sch
1.1	AS54888	6		Ea	1,117.85	20 JUN 2004 17:30:00	21 J
2.1	AS54888	5		Ea	1,117.85	20 JUN 2004 17:30:00	21 J

Line Total **6,707.10** Line Qty **6** Service Total **0.00**

Description **Dimension 4550**

Actions Related Items Configurator Availability Book Order

Cancel_PO

Setup

Customer Setup

Using the Oracle Order Management Super User responsibility, Customer Standard window (Customers-> Standard), query a customer to create XML orders for. On one of the customer's addresses, create a primary **Sold To** site usage and specify an EDI Location (e.g. Q7Q-1A1). Save your work. When provided on the inbound XML document, this information will be used to denote a particular customer site and also to determine the destination of the acknowledgment.

Figure 8–48 Customer Addresses Window

Customers - Standard

Customer Addresses - Computer Service and Rentals, 1006

CountryUnited States

Address301 Summit Hill Drive

Alternate Name

Postal Code37401

EDI LocationQ7Q-1A1

Addressee

Site Number1034

CityChattanooga

StateTN

Province

CountyHamilton

☐ Identifying Address

☒ Active

Business Purposes

Characteristics

Communication

Contacts

Contacts : Roles

Bank Accounts

Usage	Location	Bill To Location	Primary	Active
<input checked="" type="checkbox"/> Bill To	Chattanooga (OPS)		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
<input checked="" type="checkbox"/> Ship To	Chattanooga (OPS)	Chattanooga (OPS)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
<input checked="" type="checkbox"/> Sold To			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
<input type="checkbox"/>			<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>			<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>			<input type="checkbox"/>	<input type="checkbox"/>

New

Open

Table 8–14 Setup Steps

Step	Description
1	Define Transactions
2	Define Trading Partners
3	Define Code Conversions
4	Setup Oracle Workflow Business Event System

Define Transactions

Use Oracle XML Gateway to define a cross reference between the Oracle transaction name and the external transaction name. The external transaction name will be

based on what is meaningful per the XML standard used by the recipient. The external transaction name will appear on the message envelope to support message transport.

Define Trading Partners

e-Business may be conducted directly with a business partner commonly known as a trading partner or via a hub such as Oracle Exchange where many buyers and sellers converge to conduct electronic commerce.

With Oracle XML Gateway, you can define the hub or the actual business partner as a trading partner. If you define the hub as the trading partner, you can identify all the buyers and sellers who are conducting business on the hub as trading partners to the hub.

Included in the trading partner/hub definition are the following:

- Trading Partner/Hub name
- Message confirmation requested
- Message map to use for message creation or consumption
- E-mail address of trading partner contact to notify for data errors
- Trading Partner specific code conversion values
- Transport protocol - SMTP, HTTP, or HTTPS with credential and username/password as necessary
- Enable messages for the trading partner by identifying the internal and external transaction type and transaction subtype codes, and the XML standard associated with the message.

To define a trading partner:

1. Navigate to the Define Trading Partner Setup window from the XML Gateway Responsibility. Setup > Define Trading Partners.
2. Enter the Trading Partner Type, Customer.
3. Select the Trading Partner and Trading Partner Site.
4. Select the transaction type as 'ONT' for this functionality.
5. Select 'ONT_3A9R_OAG72_IN' as the map.
6. Save your work.

Define Code Conversions

The Oracle XML Gateway code conversion function provides a method to cross-reference the codes defined in Oracle Applications to codes used by trading partners, the XML standard, or other standard codes in the transactions.

The Code Conversion Standard code determines which transaction Standard these values are applied to. A Standard Code of 'UNIVERSAL' applies these values across all Standards. Here, run the Code Conversion form from the Navigator Setup menu.

Setup Oracle Workflow Business Event System

Oracle XML Gateway leverages the Oracle Workflow Business Event System to publish and subscribe to application business events of interest to automatically trigger message creation or consumption.

Seeded business events and event subscriptions to send outbound or consume inbound messages are delivered for all Oracle pre-built messages. The seeded event subscriptions may be configured during implementation to perform activities to address specific business requirements.

Define System Profile Values

The following table lists the System Profile options, that need to be set.

Table 8–15 Profile Options

Profile System Option	Description	Required	Default Value
OM: Run Order Import for XML	Determines whether Order Import will run Asynchronously or Synchronously.	No	Synchronous

The directories listed in the above profiles should also be mentioned in the utl_file_dir parameter in init.ora

Please refer to the *Oracle XML Gateway User's Guide* for the setup information.

Setup the Subscribing System

Query the Inbound event and change the Subscribing system accordingly.

Define the Subscribing System for the event subscription for event 'oracle.apps.ont.oi.po_inbound.create.'

Subscriber: System - This will usually be the local system, the system on which the subscription code is to run.

(This data will be available as the initial setup, come as an XML file and will be loaded to the customer system. User of the system will be able to change it to work for their system settings).

Message Set Up

To implement a message with a trading partner, use XML Gateway message set up to define the trading partner or hub, code conversion values, and internal to external transaction name cross references. In addition, you may identify the XML Gateway system administrator to notify for system or process errors.

Cancelled Reason Behavior

Order Import populates the OE_ORDER_LINES_HISTORY column with the reason code for the cancellation. When change reasons are specified in the incoming Cancel_PO document at both the header and line level, the line level reason populates the above mentioned table (OE_ORDER_HEADERS_HISTORY does exist but is not populated). The only exception to this behavior occurs when the incoming document does not contain any POLINE tags (i.e. no lines specified but entire order being cancelled). In this case, the header level cancel reason is entered in the history table for each line being cancelled.

Note: This behavior is consistent with the Sales Orders window.

If the incoming document does not contain this tag, then the message map defaults the cancelled code to 'Not provided,' that corresponds to the meaning: 'No reason provided.'

Pre-Defined Workflow Event

A pre-defined Workflow event is seeded to run Order Import immediately if the Profile Option OM: Run Order Import for XML is set to 'Synchronous.' This Workflow subscribes to the event raised by the Post Process of the XML Gateway. As part of the Workflow Process it will run Order Import for the XML Message, which is currently being consumed.

Note: Cancel_PO utilizes the same workflows as Process_PO.

Implementation Considerations

Loading of Message Map and DTD

Oracle message maps are delivered and installed as a part of Oracle Order Management file structure and schema. They are automatically loaded into the XML Gateway repository using the LoadMap program. The reports are available to verify the success of the loading process.

Enabling and Consuming the Business Event

The necessary business events and subscriptions are also delivered and installed as part of the schema. They are loaded by the driver.

Open Interface Tracking

Setup

Customer Setup

To setup for Open Interface Tracking:

1. Set up the appropriate Electronic Messages.
2. Set the profile option OM: Electronic Message Integration Event Sources. This profile indicates the level at which electronic messages are tracked and has the values XML, EDI & XML and All (if the value is NULL, this is treated as XML). If the profile option is set to XML, this feature will track only orders with an order source of XML. Similarly, if the profile option is set to EDI & XML, only orders with order source of EDI or XML will be tracked.

Note: Order Management enables users to extract EDI 855 and 865 Acknowledgment data for orders with order source Online. To enable these orders for tracking, the value of the profile option must be set to All; setting the profile to EDI & XML will not track these orders.

3. Set the user access to the form. To allow the form to link to the OA Framework pages for Oracle Workflow and XML Gateway Transaction monitor, ensure that

the responsibility with the Open Interface Tracking form has access to the form functions WF_G_DIAGRAM and TXMONITORRESULTSPG – these functions are part of the responsibility Workflow Administrator Web Applications, but you may need to add them to a menu within the responsibility containing the form.

4. Schedule the WF_DEFERRED Agent Listener.
5. Schedule the Workflow Background Process for Item Type OM: Open Interface Tracking.

Integration Event – oracle.apps.ont.oi.xml_int.status

Oracle Order Management has a Business Event System-based mechanism to provide subscribers with a way to perform processing at desired integration points.

Currently, there are three ways these events are triggered:

1. During processing of Electronic Messages (i.e. XML, EDI transactions) and Order Import. In this scenario, the event is raised automatically via the seeded workflows.
2. Because of Processing Constraint-based generation. In this scenario, you can determine the point where the event is raised by defining a Processing Constraint.
3. By inserting the workflow sub-process 'Raise Show Sales Order Event Sub Process' from the item type 'OM Show Sales Order' (OESO) in an Order Header (OEOH) workflow process. This method can be used, for example, to raise an event when an order is booked.

To leverage this functionality, you can define custom synchronous or asynchronous subscriptions to this event and perform any desired processing via the event subscription. For further information about defining event subscriptions, please refer to the Oracle Workflow documentation.

These events are controlled by the profile option, 'OM: Electronic Message Integration Event Sources'. The possible values are 'XML' (default value), 'XML & EDI' and 'All.' Users can use this profile option value to determine the Order Sources for which this event is raised.

1. Integration Event Parameter Details

The following tables summarize the event parameters that are supported by the integration event. Note that not every parameter will be populated in all cases, as described in subsequent sections.

The event **oracle.apps.ont.oi.xml_int.status** will contain the following parameters:

Table 8–16 Parameters and Descriptions

Parameter Name	Parameter Description	Comments
XMLG_INTERNAL_CONTROL_NUMBER	Unique document number generated by XML Gateway for inbound messages	Overloaded to provide a unique id for EDI inbounds and Order Import as well
XMLG_MESSAGE_ID	Unique message id generated by XML gateway for outbound messages	
XMLG_INTERNAL_TXN_TYPE	Typically 'ONT' except for Confirm BOD which is owned by 'ECX'	
XMLG_INTERNAL_TXN_SUBTYPE	Identifies XML transaction type	Overloaded to also contain EDI and Order Import transaction type, namely '850', '860', '855', '865', 'GENERIC'
DOCUMENT_DIRECTION	'IN' or 'OUT'	
XMLG_DOCUMENT_ID	Unique ID provided by OM for outbound messages	Overloaded to provide a unique id for outbound EDI transactions
TRADING_PARTNER_TYPE	'C'	
TRADING_PARTNER_ID	TP Party ID	
TRADING_PARTNER_SITE	TP Party Site ID	
DOCUMENT_NO	Sales Order Number	
ORG_ID	Organization ID	
PARTNER_DOCUMENT_NO	Purchase Order Number	
DOCUMENT_REVISION_NO	Change Sequence	
ONT_DOC_STATUS	'ACTIVE', 'SUCCESS', 'ERROR'	
MESSAGE_TEXT	FND MESSAGE detailing what has occurred.	

Table 8–16 Parameters and Descriptions

Parameter Name	Parameter Description	Comments
WF_ITEM_TYPE	Workflow Info for XML/EDI processing	
WF_ITEM_KEY	Workflow Info for XML/EDI processing	
ORDER_SOURCE_ID	Order Source	
SOLD_TO_ORG_ID	Customer ID	
ORDER_TYPE_ID	Order Type ID	
CONC_REQUEST_ID	Processing Concurrent Request ID	
PROCESSING_STAGE	Code to identify why this event was raised. Sample values include 'INBOUND_IFACE', 'OUTBOUND_SENT'.	
SUBSCRIBER_LIST	Comma delimited list of product short names of the intended subscribers eg 'ONT,CLN'	
DOCUMENT_STATUS	'SUCCESS', 'ERROR'	
RESPONSE_FLAG	'Y' for 3A8 Change PO Response documents	
HEADER_ID	Order Header identifier	
LINE_IDS	Only applicable for events raised via Processing Constraints	

2. Integration Event Parameter Population

Electronic Messaging/Order Import

The following table contains a Y if a particular parameter is populated for XML/EDI transactions as well as Order Import.

Table 8–17 Inbound/Outbound Messages Parameter Names

Parameter Name	INBOUND MESSAGES Parameter Populated (Electronic Message Types)	OUTBOUND MESSAGES Parameter Populated (Electronic Message Types)
XMLG_INTERNAL_ CONTROL_NUMBER	Y (XML, EDI, Order Import)	
XMLG_MESSAGE_ID		Y (XML, if document sent successfully)
XMLG_INTERNAL_TXN_TYPE	Y (XML, EDI, Order Import)	Y (XML, EDI)
XMLG_INTERNAL_TXN_ SUBTYPE	Y (XML, EDI, Order Import)	Y (XML, EDI)
DOCUMENT_DIRECTION		Y (XML, EDI)
XMLG_DOCUMENT_ID		Y (XML, EDI)
TRADING_PARTNER_TYPE		Y (XML, except CBOD)
TRADING_PARTNER_ID		Y (XML, except CBOD)
TRADING_PARTNER_SITE		Y (XML, except CBOD)
DOCUMENT_NO	Y (XML, EDI, Order Import) (if import succeeds) (information is also derived for XML updates via 3A8/3A9 and processing stage IMPORT_ FAILURE)	Y (XML if import succeeds or sales order already exists, EDI)
ORG_ID	Y (XML, EDI, Order Import)	Y (XML, EDI)
PARTNER_DOCUMENT_NO	Y (XML, EDI, Order Import)	Y (XML, EDI)
DOCUMENT_REVISION_NO	Y (XML, EDI, Order Import)	Y (XML, EDI)
DOCUMENT_STATUS	Y (XML, EDI, Order Import)	Y (XML, EDI)
ONT_DOC_STATUS	Y (XML, EDI, Order Import)	Y (XML, EDI)

Table 8–17 Inbound/Outbound Messages Parameter Names

Parameter Name	INBOUND MESSAGES Parameter Populated (Electronic Message Types)	OUTBOUND MESSAGES Parameter Populated (Electronic Message Types)
MESSAGE_TEXT	Y (XML, EDI, Order Import)	Y (XML, EDI)
WF_ITEM_TYPE	Y (XML except when PROCESSING_STAGE is IMPORT_SUCCESS or IMPORT_FAILURE)	Y (XML, EDI *)
WF_ITEM_KEY	Y (XML except when PROCESSING_STAGE is IMPORT_SUCCESS or IMPORT_FAILURE)	Y (XML, EDI *)
ORDER_SOURCE_ID	Y (XML, EDI, Order Import)	Y (XML, EDI)
SOLD_TO_ORG_ID	Y (XML, EDI, Order Import)	Y (XML, EDI)
ORDER_TYPE_ID	Y (XML, EDI, Order Import) (if import succeeds) (information is also derived for XML updates via 3A8/3A9 and processing stage IMPORT_FAILURE)	Y (XML if import succeeds or sales order already exists, EDI)
HEADER_ID	Y (XML, EDI, Order Import) (if import succeeds) (information is also derived for XML updates via 3A8/3A9 and processing stage IMPORT_FAILURE)	Y (XML, EDI)
CONC_REQUEST_ID	Y (XML, EDI, Order Import)	Y (XML for 3A6 Show SO generated via concurrent program)

Table 8–17 Inbound/Outbound Messages Parameter Names

Parameter Name	INBOUND MESSAGES Parameter Populated (Electronic Message Types)	OUTBOUND MESSAGES Parameter Populated (Electronic Message Types)
PROCESSING_STAGE	Y (XML, EDI, Order Import)	Y (XML, EDI)
SUBSCRIBER_LIST	Y (XML, EDI, Order Import)	Y (XML, EDI)
RESPONSE_FLAG	Y (XML only for Change PO response documents)	

* The EDI support is only active when the 11.5.10/Pack J EDI workflow is being used

The following table maps the value of the event parameter PROCESSING_STAGE to the message text for that stage:

Table 8–18 PROCESSING_STAGE to the Message Text For That Stage

XML	EDI	Order Import	Processing Stage Description (CODE)	Sample Message Text
Y (3A8 Change PO Response only)			Data processed by inbound gateway (INBOUND_GATEWAY)	"Change PO - Buyer response contained a rejected header and was not processed by Order Management" "Change PO - Buyer response contained one or more rejected lines which were not processed by Order Management"
Y	Y		Data received in Open Interface Tables (INBOUND_IFACE)	"Process PO – Data successfully entered into Open Interface tables" "850 POI – Data successfully entered into Open Interface tables"
Y			Import mode (PRE_IMPORT)	"Process PO – Order Import will be run in Asynchronous mode" "Process PO – Order Import will be run in Synchronous mode"

Table 8–18 PROCESSING_STAGE to the Message Text For That Stage

XML	EDI	Order Import	Processing Stage Description (CODE)	Sample Message Text
Y	Y	Y	Order Import result (IMPORT_SUCCESS, IMPORT_FAILURE)	“Process PO - Order Import successful” “Process PO – Order Import failed”
Y (except CBOD)			Outbound message triggered (OUTBOUND_TRIGGERED)	“OM is ready to send the Acknowledge PO message” “OM is ready to send the Show SO message”
Y (except CBOD)	Y *		Outbound setup result (OUTBOUND_SETUP)	“Acknowledge PO – Setup validation successful” “Acknowledge PO – Setup validation failed”
Y	Y		Outbound sent (OUTBOUND_SENT)	“Acknowledge PO message has been sent” “855 POAO message has been sent”

Note that for Outbound_Setup, the EDI support is only active when the 11.5.10/Pack J EDI workflow is being used

Constraint-based Event Raise

The following table contains a Y if a particular parameter is populated for Processing Constraint-based Event Raise:

Table 8–19 Populated Parameters for Processing Constraint-based Event

Parameter Name	Parameter Populated (Electronic Message Types)
DOCUMENT_NO	Y
ORG_ID	Y
PARTNER_DOCUMENT_NO	Y
DOCUMENT_REVISION_NO	Y
ORDER_SOURCE_ID	Y
SOLD_TO_ORG_ID	Y

Table 8–19 Populated Parameters for Processing Constraint-based Event

Parameter Name	Parameter Populated (Electronic Message Types)
HEADER_ID	Y
SUBSCRIBER_LIST	Y ('DEFAULT')
LINE_IDS	Y 'ALL' for header-level triggering Colon-delimited string of Line Ids for line-level triggering

Workflow Subprocess-based Event Raise

The following table contains a Y if a particular parameter is populated for Workflow Subprocess-based Event Raise (using the 'Raise Show Sales Order Event Sub Process'):

Table 8–20 Parameters Populated for Workflow Subprocess-based Event

Parameter Name	Parameter Populated (Electronic Message Types)
DOCUMENT_NO	Y
ORG_ID	Y
PARTNER_DOCUMENT_NO	Y
DOCUMENT_REVISION_NO	Y
ORDER_SOURCE_ID	Y
SOLD_TO_ORG_ID	Y
ORDER_TYPE_ID	Y
HEADER_ID	Y
SUBSCRIBER_LIST	Y ('DEFAULT')
LINE_IDS	Y ('ALL')

User Procedures

To query electronic messages:

1. Navigate to the Access the Open Interface Tracking Find window.

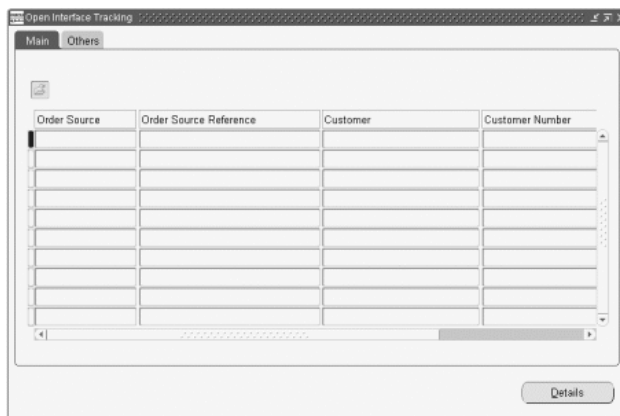
Figure 8–49 Find Window

The screenshot shows a web-based application window titled "Open Interface Tracking". Inside, there is a "Find" dialog box. The dialog box has the following fields and controls:

- Customer:
- Customer Number:
- Order Source:
- Order Source Reference:
- Includes Transaction:
- Start Date: -
- Last Updated: -
- Buttons:

The background window has tabs for "Main" and "Others". The "Main" tab is active, showing a table with columns "Order Source" and "Number". A "Details" button is located at the bottom right of the background window.

2. Enter in the desired search criteria and execute the query.
3. To view all the electronic messages for a particular order, select that message and click Details.

Figure 8–50 Open Interface Tracking Window**To view workflow, XML, and delete records:**

1. Navigate to the Access the Open Interface Tracking Find window and query your records.
2. To view the workflow or XML for a particular electronic message, click Details.
3. To view all the electronic messages for a particular order, select that message and click Details.

Note: A concurrent program, Purge Open Interface Data, exists to delete multiple records from the table. This purge procedure can also be used to purge the Interface or Acknowledgment tables.

To purge records:

1. Navigate to the Concurrent Program Manager.
2. Choose the Purge Open Interface Data concurrent program. This concurrent program can also be used to purge the Interface and Acknowledgment tables.

To purge workflows:

Each Open Interface Tracking event is consumed with corresponding information being stored in the OE_EM_INFORMATION_ALL table using the Write History

Entry process of the Item Type OM: Open Interface Tracking. Once the workflow for an event completes, the Workflow can be safely purged.

Since OM_PF J is a part of SCM_PF J (which includes CLN), the dependency below will not be an issue.

NOTE: If the customer has Order Management Pack J, but is not on Oracle Applications 11.5.10, then Order Management's seeded subscription to the event **oracle.apps.ont.oi.xml_int.status (required by this feature)** may not process correctly due an issue with another seeded subscription to the same event. This issue can also affect any other product or custom subscriptions to the above mentioned event. To avoid this problem, customers can either:

- Apply 11i.CLN.B Patch #2734844

OR

- Manually disable the seeded event subscription to the event oracle.apps.ont.oi.xml_int.status owned by Supply Chain Trading Connector (CLN), if the customer is not using the CLN product. The issue with this event subscription has been resolved in CLN patchset B.

Order Management Suite Products

- [Release Management](#) on page 9-2
- [Shipping](#) on page 9-3
- [Transportation Execution](#) on page 9-7
- [DSNO](#) on page 9-12

Release Management

Planning Schedule

The purpose of the SYNC PLANSCHD BOD is to communicate requirement information (part number, quantity, etc.) between a customer and their supplier on a regular basis, for example daily, weekly, etc., or a user-defined time bucket type definition that is sent as part of this PLANSCHD.

SYNC PLANSCHD enables adding new requirements and the modification of previously established requirements through various SYNC Indicator values. Existing SYNC Indicator values from OAG include A for adding, C or change, and D for delete. In addition, a new SYNC indicator value R for replacement is defined for this PLANSCHD (as well as in the related SHIPSCHD BOD).

Shipping Schedule

The purpose of the SYNC SHIPSCHD Business Object Document is to enable the exchange of shipment schedule information, authorizing a shipment quantity and date for specific trading partners and addresses.

The ship schedule is generated by a material planning application and transmitted to an order or material planning application.

Receive message

The XML Gateway Execution Engine does the following during inbound processing:

- Dequeue Message from Inbound Queue
- Validate Message via XML Parser

Uses the XML Parser to validate the inbound message to determine if it is well-formed and valid (based on DTD stored in DTD directory) before proceeding further.

- Validate Trading Partner or Hub

If the inbound message is both well-formed and valid, the Execution Engine proceeds to validate that the Trading Partner and document are defined. If the Trading Partner is not defined or the document is not defined for the Trading Partner, the XML message will not be processed further.

- Get Message Map from Repository

If the message map associated with the Trading Partner is not available in the XML Gateway repository, the XML message will not be processed further.

- Maps Data

If the Trading Partner is valid and the message map exists in the repository, the Execution Engine maps the data in the XML message to its target data fields in Oracle e-Business Suite tables and columns identified in the message map. These are often the Application Open Interface tables. Mapping process also includes:

- Apply Code Conversion

- Apply Actions

- Detect and Report Processing Errors

Errors may be detected by the Oracle XML Gateway Execution Engine, Oracle Advanced Queuing, Oracle Workflow, or Oracle Transport Agent. Information regarding the error is enqueued onto the Error Queue. An e-mail notification is sent via Oracle Workflow to notify the trading partner regarding data errors, or the XML Gateway system administrator regarding system or process errors. In addition, for system or process errors, a copy of the XML message is placed in the XML message directory for use in troubleshooting the reported error. For trading partner-related data errors, the trading partner can refer to the copy of the XML message.

Process DSP and Other RLM Processes

Similar to the EDI message, when successfully XML processed, the transactions by XML gateway appear in the Release management workbench as “available to process” schedules.

For more information, see the *Oracle XML Gateway User's Guide*.

Shipping

Purpose Codes

EDI Demand Transaction Purpose Codes vs. Oracle Purpose Codes

For a customer demand transaction to be processed correctly in the Demand Processor, the appropriate Oracle Purpose Codes must be associated with it.

EDI Gateway Code Conversion cross-references for RLA_TRX_PURP are pre-defined for the standard Purpose Codes used in EDI transactions for ASC X12 and EDIFACT standards.

You must define additional customer-specific Code Conversion cross-references in the EDI Gateway that will uniquely map to the correct Oracle Purpose Code if:

Your customer uses a different EDI standard whose Purpose Codes are not pre-defined.

You have EDI Trading Partner customers who send demand transactions with a Purpose Code that differs in meaning from the pre-defined cross-reference to the Oracle Purpose Code (external values 2-4 can contain Trading Partner Code, Schedule Type, Schedule Type Qualifier, or other element as needed).

For example, your customer may use an **Original** Purpose Code for the first schedule of a new model year, which is intended to replace all previously issued demand from the prior model year. If customer-specific mapping is defined to link this schedule to an Oracle **Replace** Purpose Code, the demand will be handled properly. However, if the customer-specific mapping is not defined, the Oracle **Original** Purpose Code will be assigned. This will result in overstated demand, because demand from the last schedule of the prior model year will remain on file, and demand from the first schedule of the new model year will be added.

Assuming that existing and new demand came from the same schedule type (e.g. 830, 862, or 866), the following chart describes Oracle Purpose Codes and their functionality:

Table 9–1 Oracle Purpose Code

Code	Demand Processor Interpretation
Original	The requirements on this schedule have not been previously issued. They will all be added, even if existing demand is on file.
Cancellation	The requirements on this schedule will be canceled.
Change	This is a partial revision of a previous schedule for selected items; the requirements on this schedule will replace what was sent on a previous schedule for these items within the schedule horizon.
Replace	The requirements on this schedule will replace existing requirements from a previous schedule, subject to the following exceptions: for items not on the schedule, all existing demand remains intact for schedule items, existing demand dated after the schedule horizon end date remains intact.

Table 9–1 Oracle Purpose Code

Code	Demand Processor Interpretation
Add	The requirements on this schedule will be added to existing requirements from a previous schedule.
Delete	The requirements on this schedule will be deleted from existing requirements from a previous schedule.
Confirmation	The requirements on this schedule will be archived but not processed, assuming that changes were made manually before the schedule was received. They are changes to a previous schedule which confirm verbal emergency requirements.

Planning/Shipping Inbound

Transaction Descriptions

Planning Schedule With Release Capability (830)

A Planning schedule with release capability transaction set provides for customary and established business practice relative to the transfer of forecasting/material release information between organizations.

The planning schedule transaction may be used in various ways or in a combination of ways, such as:

1. A simple forecast.
2. A forecast with the buyer's authorization for the seller to commit resources, such as labor or material.
3. A forecast that is also used as an order release mechanism, containing such elements as resource authorizations, period-to-date cumulative quantities, and specific ship/delivery patterns for requirements that have been represented in buckets, such as weekly, monthly, or quarterly.

The order release forecast may also contain all data related to purchase orders, as required, because the order release capability eliminates the need for discreet generation of purchase orders.

Shipping Schedule (862)

The shipping schedule transaction set can be used by a customer to convey precise shipping schedule requirements to a supplier, and is intended to supplement the planning schedule transaction set (830). The shipping schedule transaction set will

supersede certain shipping and delivery information transmitted in a previous planning schedule transaction, but it does not replace the 830 transaction set. The shipping schedule transaction set shall not be used to authorize labor, materials, or other resources.

The use of this transaction set will facilitate the practice of Just-In-Time (JIT) manufacturing by providing the customer with a mechanism to issue precise shipping schedule requirements on a more frequent basis than with the issuance of a planning schedule transaction, e.g. daily shipping schedules versus weekly planning schedules. The shipping schedule transaction also provides the ability for a customer location to issue shipping requirements independent of other customer locations when planning schedules are issued by a consolidated scheduling organization.

Delivery Schedule (DELFOR)

A message from buyer to supplier giving product requirements regarding details for short term delivery instructions and/or medium to long term product/service forecast for planning purposes according to conditions set out in a contract or order. The message can be used to authorize the commitment of labor and materials resources; however, authorizations are not actually used in Europe.

Delivery Just In Time (DELJIT)

A message provides the ability for a customer to convey precise delivery sequence and Just In Time schedule requirements to a supplier, and is intended to supplement the Delivery Schedule message (DELFOR). It provides a mechanism to issue precise shipping schedule requirements in terms of a 24-hour clock and on a more frequent basis than with the instance of a delivery schedule message, e.g. daily shipping schedules versus weekly planning schedules.

Delivery Instruction (DELINS)

The Delivery Instruction Message is sent by a buyer and is intended to provide information regarding details for both short term delivery instructions and medium to long term requirements for planning purposes according to conditions set out in a contract or order.

Note: The analysis is based upon the Odette message DELINS, version 4. The Odette Delivery Instruction message corresponds in purpose and content to the ASC X12 830 - Planning Schedule with Release Capability and the X12 862 - Shipping Schedule.

KANBAN Signal (KANBAN)

The KANBAN Signal Message is issued by a consignee giving authorization to the consignor to ship material based upon receiving a kanban signal and following the principles of the Just-In-Time philosophy.

Note: The analysis is based upon the Odette KANBAN message, version 2. The Odette message corresponds in purpose and content to the ASC X12 862 - Shipping Schedule when used as a kanban Signal.

For more information, see the *Oracle XML Gateway User's Guide*.

Transportation Execution

ShowShipment Outbound XML is used as the outbound freight bill audit from Oracle Transportation Execution to a third party audit firm. It is the XML equivalent of ASC X12 940 transaction. The message contains all of the details of the shipment that are required to audit the freight bill.

XML based carrier tracking enables you to accept inbound XML ShowShipStatus message from carriers. This is equivalent to the EDI 214 Shipment Status Message. The carrier response includes tracking information such as the current location of the delivery, status of the delivery, delivery exceptions, and the date the message was sent. The carrier can create an XML.

Delivery Tracking

ShowShipStatus message, or an EDI 214 that is converted to the XML ShowShipStatus format, and is transmitted to Oracle Transportation Execution.

Freight bill payment and audit (FPA) refer to the process of verification of freight bills for charges and bill payment once a shipment has been tendered to a transportation carrier and delivery has been completed. Once a carrier completes delivery of a shipment a freight bill is generated and forwarded to the party responsible for payment.

The manual process of freight bill auditing and payment can be very time consuming and expensive. Shippers can realize cost savings by implementing procedures to automate these processes. Some shippers out source this activity to 3rd party freight payment firms that perform auditing and payment services. Shippers use freight payment firms to reduce administrative costs associated with

auditing freight bills. The Freight Payment and Audit module of Oracle Transportation Execution will help shippers to perform in house auditing of freight bills in an automatic manner. For more information, see the *Oracle XML Gateway User's Guide*.

Tracking Message

The tracking message lets shipper communicate with carrier about current status information of a specific delivery. Once the information is processed by Oracle Transportation Execution, the shipper can easily view all the information of the delivery on the form and track the progress of the shipping process. The tracking message is based on OAG standard DTD, so any shipper and carrier can adopt it.

Support for Track and Trace

Tracking can be defined as receiving continuous feedback at regular intervals on the status and whereabouts of a shipment while in route from its original pickup location to its final destination. Tracking occurs while the shipment is in progress. There are three types of tracking: automatic, bulk, and individual request (adhoc) tracking.

Automatic Tracking: an automated way to obtain tracking for a particular delivery. You indicate on a delivery to track it, and the messages are sent to the carrier system, and received from the logistics provider, in a automated fashion.

Bulk Tracking: a daily transmission for all shipments that a particular carrier picked up from a particular supplier. On a daily basis, the carrier sends the tracking information for each shipment still in process. The carrier continues to send this information until the shipment is delivered to its final destination.

Adhoc Tracking: a request for tracking information on a particular shipment. The Oracle Transportation Execution user at the supplier site will send a request for tracking an individual shipment to a particular carrier, and the carrier will respond with the shipment's current tracking information.

Tracing locates a shipment by determining where it has been. Tracing a shipment typically happens after shipment is overdue and an inquiry is made by the consignee as to its Estimated Time of Arrival.

Track and Trace would be used for the following reasons:

- **Expediting a critical shipment** A person may trace a shipment to determine if the shipment will arrive prior to stockout (inventory depletion) resulting in a production line shutdown or plant shutdown. If the shipment will meet the deadline, no action is taken. If the shipment will not meet the deadline or the

delivery schedule is too close for comfort, action is taken to expedite movement of the shipment with the carrier, or find an alternative supply.

- **Diverting a shipment** Circumstances may exist that cause the shipper to consider diverting a shipment; also known as a re-consignment. For example, the shipper's warehouse inventory of an item is depleted when a customer in desperate need of product calls in an order. The shipper traces several outbound shipments en route to customers with less critical needs (customers with sufficient inventory) so that they can re-consign a shipment to the customer in critical need of product. The shipper must trace the shipments to verify their location and status before selecting the shipment for re-consignment and issuing a diversion to the carrier.
- **Locating a lost shipment** If a shipment fails to deliver, then tracing the shipment to identify its last known location can help pinpoint where progress of the shipment ceased. This information assists with the investigation of items that have been stolen, damaged, or waylaid (diverted to an unplanned destination).
- **Verifying guaranteed service level** Trace a shipment to verify that the shipment was delivered within a guaranteed time limit. If a shipper has selected a ship method, such as, 2-day air for which the shipper incurs a premium freight rate the carrier may be obligated to deliver within the agreed upon time limit to justify payment of the freight charge. A shipment not delivered on time may result in a request by the shipper for the carrier to refund all or a portion of the freight charge.
- **Assessing carrier performance** The shipper and carrier may have an agreed performance standard for specific service lanes (also known as routes). For example, the agreed transit time standard between the shipper's origin (San Francisco) and a particularly service sensitive customer in Chicago is five days. The shipper will periodically trace completed shipments to assess the carrier's performance over a period of several months and compile performance statistics, e.g. 90% of completed shipments were delivered within 5 days.
- **Processing claims** Identifying where a shipment was at any point in time can greatly assist shippers with supporting evidence when they process claims against carriers and other parties. This information is usually of a historical nature, such as, where a certain shipment was on a particular date and time. The same information may also be used by carriers to refute claims.
- **Locating transportation equipment** Shippers and carriers who are responsible for transportation equipment, including trucks, railcars, vessels, and containers depend on tracing to conduct equipment planning and scheduling activities.

Therefore, in addition to tracing the contents of a shipment, you should have the ability to trace the equipment. You must be able to trace proprietary (owned or leased) equipment regardless of whether the equipment is loaded or empty. The equipment always has an identification number (LPN - License Plate Number) associated with the piece of equipment which can be used for tracing purposes.

Bulk Tracking

Oracle Transportation Execution receives inbound tracking messages for a specific delivery on both delivery level and container level. On delivery level, a carrier can use waybill number, bill of lading number, and booking number to send a tracking message. On the container level, a carrier can track individual container based on container license plate number, seal identifier, and carrier reference number. In order for the shipper to view the complete list of the tracking message, the carrier should repeat the information in the previous message when he is sending a new message with the same tracking id type.

View the Delivery Activity

You can view the delivery activity based on the delivery name. This page shows both delivery level tracking and container level tracking. On delivery level tracking, the page shows the latest header information the system received from the carrier and the system will not distinguish different tracking id type. On container level, the page lists all the containers the carrier tracked based on the container license plate number.

View the Delivery Detail Information

You can view the delivery detail information by clicking the Detail icon on the Delivery Activity page. It shows the complete list of all the delivery details sent to Oracle Transportation Execution ordered by report date.

View the Delivery Exception

You can view the delivery exception information by clicking on the Exception icon on the Delivery History page. It lists all the exceptions the system received from the carrier under this delivery. You can also view the delivery exception information by clicking on the Exception icon on the Delivery Detail page. It will list all the exceptions under this delivery detail.

Bulk Tracking

Oracle Transportation Execution accepts inbound tracking messages asynchronously from a logistics provider to a supplier that represents all shipments the carrier has for that supplier. Oracle Transportation Execution has links to the history tables and tracking tables. You can log in and see the resulting status updates after they arrive.

XML or EDI Messages

Depending on what the logistics provider can receive, the request and response messages can be sent in either an XML format, or an EDI format. This is determined by the trading partner setup for a carrier in the XML or e-Commerce Gateway. If a carrier uses EDI transactions, a translator converts the EDI transaction into the XML format for Oracle Transportation to use the message.

Data Received on the Response Transaction

When the logistics provider responds with new tracking information for a delivery, the following data is sent on the response transaction, and is updated in the Oracle Transportation Execution tables:

- Shipment or Package Status
- Shipment or Package Status Reason Code
- Location of shipment or package

View the Delivery History

You can view the history of all the deliveries. The tracking information comes from the inbound message from the carrier. The carrier name, the delivery status, activity date, activity type, and the sequence of the trip segment display. You can drill down to either shipment exceptions or the delivery leg details.

View the Delivery Detail Information

You can view the delivery leg details information from the inbound message. The summarized header information displays for the delivery as well as the detail information of each delivery leg. The user interface allows the history information to be shown relative to a delivery, showing all the legs, or a specific delivery leg.

View the Delivery Exception

You can view the delivery exceptions from the inbound message, and all the exceptions listed for the chosen delivery. For more information, see the *Oracle XML Gateway User's Guide*.

DSNO

Departure Ship Notice Outbound 856

DSNO is a notice used by the supplier to provide advance notification of a shipment to the buyer or trading partner that the material has been shipped. It is extremely flexible, and permits the seller to transmit shipment information in varying levels of detail. It is also referred to as ASC X12 856 transaction EDI.

Interface Trip Stop (ITS): DSNO is generated as a part of Interface Trip Stop process. DSNO can also be generated after Ship Confirm using the concurrent program Departure Ship Notice Outbound.

The Outbound Execution engine is responsible for extracting data related to a transaction from Oracle Applications. A transaction is referred to as a Business Document or a message i.e. purchase order, invoices, shipment notices, etc. The Extracted data can be staged in a staging table, fed directly to the EDI Flat File process to create an Outbound EDI File, or sent to the XML Engine for producing XML Documents.

A given transaction can have multiple representation for the outside world i.e. a purchase order document can be an XML document for one customer and EDI Flat File for another customer. The data related to different maps for the same transaction is stored in the EDI repository. The triggering process will identify which map to process and then pass on the value to the engine.

The input to the Execution Engine is the Map Identifier for the message and the transaction type. The map identifier stores the mapping information between the Oracle Applications tables and the Flat File or XML or Staging table.

The Execution engine extracts the data from EC messaging views, defined on top of business views for each transaction. In addition to the data obtained from the view, the engine executes stored procedures, functions, and custom code to pull out the remaining business data. For more information, see the *Oracle XML Gateway User's Guide*.

Electronic Messaging Messages

Messages covered in this appendix include:

- [Process_PO and Acknowledge_PO](#) on page A-2
- [Show_SalesOrder](#) on page A-3
- [Cancel_PO](#) on page A-4
- [Change SO](#) on page A-6
- [Inbound Change PO Request](#) on page A-7

Process_PO and Acknowledge_PO

1. The error out message if it is not determined whether document delivery is required for that TP.

OE_OI_TP_NOT_FOUND - Missing Primary SOLD_TO site usage for the customer id &CUST_ID

2. The error out message if the outbound acknowledgment data is not generated correctly.

OE_OI_ACK_DATA_NOT_FOUND - Error in getting data from &TABLE for sending the acknowledgment

3. The error out message if a particular type of EDI location is not found.

OE_OI_EDI_LOC_NOT_FOUND - &EDI_LOCATION not found for the site usage of &SITE_USAGE

4. The error out message if the org cannot be determined from the address.

OE_OI_ORG_NOT_FOUND - &SITE_USAGE site usage is not found for address id &ADDRESS_ID

5. The error out message if the customer site cannot be determined when generating acknowledgment data.

Process_PO and Acknowledge Purchase Order

OE_OI_CUST_SITE_NOT_FOUND - Could not determine account/party site id for customer id &CUST_ID

6. The error out message if the customer cannot be identified before raising the acknowledgment event.

OE_OI_CUST_NOT_FOUND - No Customer Found & OPT_TABLE for document id &DOC_ID

Show_SalesOrder

No seeded messaged for this transaction exist.

Cancel_PO

1. Error message if invalid quantity is sent on the inbound Cancel_PO XML.
OE_OI_INVALID_QTY_3A9 - Attempt to cancel Invalid Quantity
2. Error message if we cannot determine whether document delivery is required for that trading partner.
OE_OI_TP_NOT_FOUND - Missing Primary SOLD_TO site usage for the customer id &CUST_ID
3. Error message if we cannot generate the outbound acknowledgment data correctly.
OE_OI_ACK_DATA_NOT_FOUND - Error in getting data from &TABLE for sending the acknowledgment
4. Error message if a particular type of EDI location is not found.
OE_OI_EDI_LOC_NOT_FOUND - &EDI_LOCATION not found for the site usage of &SITE_USAGE
5. Error message if the org cannot be determined from the address.
OE_OI_ORG_NOT_FOUND - & SITE_USAGE site usage is not found for address id &ADDRESS_ID
6. Error message if the customer site cannot be determined when generating acknowledgment data.
OE_OI_CUST_SITE_NOT_FOUND - Could not determine account/party site id for customer id &CUST_ID
7. Error message if the customer cannot be identified before raising the acknowledgment event.
OE_OI_CUST_NOT_FOUND - No Customer Found &OPT_TABLE for document id &DOC_ID

NOTE: Integration with the Collaboration History Module (CLN)

Order Management provides integration with Collaboration History Module (CLN).

The Collaboration History module records the transaction history when the trading partners exchange messages. To enable this Order Management has incorporated the API calls during the processing of the XML messages.

Refer to the Collaboration History Module documentation for more details.

Change SO

New Message

Following message will be raised when status of 3A8 response is 'ACCEPT' but 3A8 does not match with the changes proposed on the original 3A7.

1. Message_Name: ONT_3A8_RESPONSE_COL_MISMATCH

3A8 Request Purchase Order Change response does not match with changes proposed on the original 3A7.

Message Type: Error

1. One or more rejected lines was received on a Change PO Response
Change PO - Buyer response contained one or more rejected lines which were not processed by Order Management
Change PO Response was rejected at header level.
Change PO - Buyer response contained a rejected header and was not processed by Order Management.
2. OM is ready to send Change SO
OM is ready to send Change SO message.
3. Setup validation done (Successful)
Change SO - Setup validation successful.
4. Setup validation done (Failed)
Change SO - Setup validation failed.
5. Change Sales Order message sent
Change SO message has been sent.

Inbound Change PO Request

1. When the Change PO message is processed and Order Import tables are populated with the data the message recorded using Collaboration History Module API is:

On success – Change PO - Data successfully entered into Open Interface tables

2. Order Import is going to be run immediately:

Change PO - Order Import will be run in Synchronous mode

3. Order Import will be run in Asynchronous mode.

Change PO - Order Import will be run in Asynchronous mode

4. Order Import has completed.

Change PO-Order import has completed (Sucessfully)

5. Order Import has completed.

Change PO-Order import has completed (Failure)

Index

Symbols

\$ont/xml/oag72/oagis_entity_extensions.dtd, 7-29
\$ont/xml/oag72/oagis_extensions.dtd, 7-29
_PTE_VAL_REC Parameters, 5-53

Numerics

091_show_salesorder_006.dtd, 7-60
3A6 Show Sales Order, 7-50

A

Accounting Rule Hierarchy for Sales Order Lines, 2-162
Acknowledge Cancellation of Purchase Order, 7-117
Acknowledge PO Setup Steps, 8-25
Acknowledge Purchase Order, 7-34
Add Customer, 2-7
Adhoc Tracking, 9-8
AGREEMENT_REC_TYPE, 5-4
AGREEMENT_TBL_TYPE Parameters, 5-8
AGREEMENT_VAL_REC_TYPE Parameters, 5-8
AGREEMENT_VAL_TBL_TYPE Parameters, 5-9
Agreements Public Application Program Interface, 5-3
All Customers, 2-6
APIs and Open Interfaces, 1-2
Application Parameter Initialization, 4-39
Apply Automatic Attachments, 2-155
Apply Hold, 2-155

ASC X12 856, 9-12
ASC X12 862, 9-7
ASC X12 940, 9-7
Assessing carrier performance, 9-9
asynchronous, 7-4
Attribute Level Security Check, 2-81
Attribute Mapping Application Program Interface, 5-49
Attribute Validation, 2-81
Auto_Pack API, 4-128
Autocreate_Del_Trip API, 4-113
Autocreate_Deliveries API, 4-110
AutoInvoice
 accounting rules, 2-162
 ATO configurations, 2-165
 credit methods, 2-164
 internal sales orders, 2-164
 invoice sources, 2-160
 invoicing rules, 2-163
 sales tax calculation, 2-161
 transaction sources, 2-160
Automatic Tracking, 9-8

B

B2B server, 7-2
Basic Business Needs, 1-2
Book Order, 2-154
booked order, 7-51
BUILD_CONTEXTS (overloaded), 5-304
Bulk Tracking, 9-8, 9-10
Business Object for Modifier Setup Application Program Interface, 5-60
Business Object for Pricing Formulas Application

Program Interface, 5-133
Business Object for Pricing Limits Application
Program Interface, 5-157

C

Cancel Purchase Order, 7-117
Cancel_PO, 8-63
 Setup Steps, 8-48, 8-64
Cancel_PO Message Map, 7-122
CANCEL_PO OAG, 7-120
Cancelled Reason Behavior, 8-67
CBOD, 7-7, 7-118
Change_PO, 8-48
ChangedAttributesRecType RECORD
 DEFINITION, 4-100
CHECK_FUNCTION, 5-291
Clear Dependent Attributes, 2-81
CLN, A-4
Closed for Days, 7-50
code conversion values, 8-7
Collaboration History Module, A-4
Columns Enabled for Code Conversion, 7-133
columns enabled for code conversion, 7-8
communications protocol, 8-5, 8-35
Concurrent Programs
 Order Import Statistics, 2-17
Confirm BOD, 7-6, 7-118
Connect to data collection device, 1-2
Connect to external systems, 1-2
Connect to other systems, 1-2
consortium, 7-2
Consume the Cancel_PO Message Coming Inbound
 From Purchasing, 7-117
Consume the Process_PO Document, 7-4
Container_Actions API, 4-131
CONTEXTS_RESULT_REC_TYPE, 5-304
CONTEXTS_RESULT_TBL_TYPE, 5-304
Control Columns, 1-11
control processing, 1-2
CONTROL_REC_TYPE, 5-261
COPY_QUALIFIER_RULES Parameters, 5-323
Corrected Data, 2-16
Create_Batch API, 4-148
Create_Containers API, 4-116

CREATE_UPDATE_DELIVERY Mapping, 4-168
Create_Update_Freight_Costs API, 4-134
CURR_DETAILS_REC_TYPE, 5-191
CURR_DETAILS_TBL_TYPE, 5-193
CURR_DETAILS_VAL_REC_TYPE, 5-193
CURR_DETAILS_VAL_TBL_TYPE, 5-193
CURR_LISTS_REC_TYPE, 5-189
CURR_LISTS_TBL_TYPE, 5-190
CURR_LISTS_VAL_REC_TYPE, 5-190
CURR_LISTS_VAL_TBL_TYPE, 5-191
CURRENCY_CODE_TBL, 5-178
CURRENCY_REC, 5-178
CUST_TRX_TYPE_ID, 2-179
Customer Addresses Window, 8-3, 8-24
Customer PO No From, 7-51
Customer PO No To, 7-51

D

Data Columns, 1-11
Data Received on the Response Transaction, 9-11
Database View, 1-12
Defaulted Columns, 7-9, 7-134
Defaulting Rules, 2-11
Define Code Conversion, 8-3
Define Code Conversion Values, 8-64
Define Trading Partner/Hub, 8-3
Define Trading Partners, 8-64
Define Transaction, 8-64
Define Transactions, 8-3
Define Transactions Window, 8-4
Defining System Profile Values, 8-34
Defining Trading Partners, 8-34
Delete_Freight_Costs API, 4-142
Delink Config, 2-157
Deliveries Public Application Program
 Interface, 4-59
Delivery Actions, APIs, and Parameters, 4-30
Delivery Detail Actions, APIs, and
 Parameters, 4-34
Delivery Instruction (DELINS), 9-6
Delivery Just In Time (DELJIT), 9-6
Delivery Schedule (DELFOR), 9-6
Delivery_Action API, 4-70
DELIVERY_ACTION Mapping, 4-167

DELIVERY_PUB_REC_TYPE RECORD

DEFINITION, 4-62

Departure Ship Notice Outbound 856, 9-12

Derived Columns, 1-11, 7-9, 7-134

destination application, 1-11

Detail_To_Delivery API, 4-90

Diverting a shipment, 9-9

DSNO, 9-12

E

e-Business Suite codes, 8-31

EC, 7-2

EDI 214 Shipment Status Message, 9-7

Electronic Message Status-Message
Mapping, 7-143

e-mail address, 8-5

Enabling and Consuming the Business Event, 8-23

error handling, 7-4

failed processing, 7-119

unexpected errors, 7-119

Errors table, 1-12

Event Parameters, 7-144

Event Parameters & Population, 7-144

Event/Event Subscription, 7-144

Exception_Action API, 4-79

Exception_Actions Parameters, 4-80

Exceptions Application Program Interface, 4-79

Expediting a critical shipment, 9-8

export data, 1-2

Extension Tags, 7-28, 7-135

F

Form Related Messages That Are Seeded, 7-142

FORMULA_LINES_REC_TYPE Parameters, 5-138

FORMULA_LINES_VAL_REC_TYPE
Parameters, 5-142

FORMULA_LINES_VAL_TBL_TYPE
Parameters, 5-143

FORMULA_REC_TYPE Parameters, 5-135

FORMULA_TBL_TYPE Parameters, 5-138

FORMULA_VAL_REC_TYPE Parameters, 5-138

Function calls, 1-3

G

Gateway Message Designer, 7-27

Generate Confirm BOD (CBOD), 7-6

Generate_Documents API, 4-76

Generate_Documents Parameters, 4-77

Get Currency Application Program Interface, 5-177

Get Custom Price Application Program
Interface, 5-179

Get Price List Application Program Interface, 5-185

Get Ship Method, 2-158

GET_ATTR_MAPPING Parameters, 5-57

GET_ATTRIBUTES, 5-300

Get_Currency, 5-177

Get_Custom_Price, 5-180

Get_Exceptions API, 4-87

Get_Exceptions Parameters, 4-87

Get_Price_List, 5-185

GET_QUALIFIER_RULES Parameters, 5-322

Get_User_Item_Pricing_Contexts, 5-306

H

Header_Adj_Rec_Type, 2-114

Header_Adj_Val_Rec_Type, 2-118

Header_Rec_Type, 2-110

Header_Val_Rec_Type, 2-111

Holds and Releases, 2-16

hub, 8-8

I

Identifier Columns, 1-11

iIntegrations, 8-9

Import Types, 2-13

Importing from External Systems, 2-11

Inbound EDI Messages, 7-143

Inbound PO, 7-4

Inbound XML Messages, 7-143

Integration With the Collaboration History Module
(CLN), 7-70

Interface tables, 1-3

Interface Tables and Column Descriptions, 2-20

Interface Trip Stop (ITS), 9-12

Interface views, 1-3

internal and external transaction type, 8-5

Internal Sales Orders, 2-11
Internal sales orders
 Receivables Interface, 2-164
Invoicing Rule Hierarchy for Sales Order
 Lines, 2-163
Is_attribute_used, 5-307
Items and Bills, 2-5

J

java LoadMap, 7-28

K

KANBAN Signal (KANBAN), 9-7

L

Limit_Attrs_Rec_Type, 5-164
Limit_Attrs_Tbl_Type, 5-165
Limit_Attrs_Val_Tbl_Type, 5-166
Limit_Balances_Rec_Type, 5-166
Limit_Balances_Tbl_Type, 5-168
Limit_Balances_Val_Rec_Type, 5-168
Limit_Balances_Val_Tbl_Type, 5-168
Limits_Tbl_Type, 5-163
Limits_Val_Rec_Type, 5-163
LINE_ATTR_REC_TYPE, 5-284
LINE_DETAIL_REC_TYPE, 5-271
LINE_REC_TYPE, 5-266
Line_Rec_Type, 2-134
Line_Val_Rec_Type, 2-135
Load Function, 1-12
Load/Delete DTD's, 7-28
Load/Delete Maps, 7-28
Loading of Message Map and DTD, 8-23
LoadMap program, 8-9
Locating a lost shipment, 9-9
Locating transportation equipment, 9-9
LOCK_ATTR_MAPPING Parameters, 5-54
LOCK_ATTRIBUTES, 5-297
LOCK_QUALIFIER_RULES Parameters, 5-320

M

Maintain Function, 1-13
Manual and Automatic Pricing, 2-14
Match and Reserve, 2-157
message confirmation requested, 8-5
message enabled, 8-5
Message Map, 8-9
message map, 8-5
 loading, 8-9
Message Map 'ONT_3A9R_OAG72_IN.xgm, 7-121
Message Map Details, 7-7, 7-35, 7-133
Message Set Up, 7-60
Migration from Open Interfaces to Public
 APIs, 4-163
MODIFIER_LIST_REC_TYPE Parameters, 5-63
MODIFIER_LIST_VAL_REC_TYPE
 Parameters, 5-69
MODIFIER_LIST_VAL_TBL_TYPE
 Parameters, 5-70
MODIFIERS_REC_TYPE Parameters, 5-70
MODIFIERS_TBL_TYPE Parameters, 5-79
MODIFIERS_VAL_REC_TYPE Parameters, 5-80
MODIFIERS_VAL_TBL_TYPE Parameters, 5-83
Multi-Currency Conversion Setup Application
 Program Interface, 5-187

N

Notations, 5-330

O

OAG, 7-2, 7-4
oagis_domains.dtd, 7-27, 7-47, 7-60
oagis_entity_extensions.dtd, 7-27
oagis_extensions.dtd, 7-27
oagis_fields.dtd, 7-27, 7-47, 7-60
oagis_resources.dtd, 7-27, 7-47, 7-60
oagis_segments.dtd, 7-27, 7-47, 7-60
OE_Acknowledgment_Pub.Process_SSO_CONC_
 PGM, 7-62
OE_ACTIONS_IFACE_ALL, 2-68
OE_ACTIONS_IFACE_ALL Derived Values, 2-69
OE_CREDITS_IFACE_ALL, 2-63
OE_CREDITS_IFACE_ALL Conditional

- Settings, 2-64
- OE_CREDITS_IFACE_ALL Derived Values, 2-64
- OE_CUSTOMER_INFO_IFACE_ALL, 2-21
- OE_HEADERS_IFACE_ALL, 2-7
- OE_HEADERS_IFACE_ALL Conditional Settings, 2-38
- OE_HEADERS_IFACE_ALL Derived Values, 2-37
- OE_HEADERS_INTERFACE, 7-27
- OE_LINES_IFACE_ALL, 2-40
- OE_LINES_IFACE_ALL Conditional Settings, 2-55
- OE_LINES_IFACE_ALL Derived Values, 2-54
- OE_LINES_INTERFACE, 7-27
- OE_LOT_SERIAL_NUMBERS, 2-147
- OE_LOTSERIALS_IFACE_ALL, 2-65
- OE_LOTSERIALS_IFACE_ALL derived Values, 2-66
- OE_OI_ACK_DATA_NOT_FOUND, A-2, A-4
- OE_OI_CUST_NOT_FOUND, A-2, A-4
- OE_OI_CUST_SITE_NOT_FOUND, A-2, A-4
- OE_OI_EDI_LOC_NOT_FOUND, A-2, A-4
- OE_OI_INVALID_QTY_3A9, A-4
- OE_OI_ORG_NOT_FOUND, A-2, A-4
- OE_OI_TP_NOT_FOUND, A-2, A-4
- OE_ORDER_HEADERS, 7-66
- OE_ORDER_HEADERS_ALL, 2-104
- OE_ORDER_LINES_ALL, 2-123
- OE_ORDER_PRICE_ATTRIBS, 2-118, 2-142
- OE_ORDER_PUB.GET_ORDER, 2-94
- OE_ORDER_PUB.LOCK_ORDER, 2-98
- OE_ORDER_PUB.PROCESS_ORDER, 2-85
- OE_PRICE_ADJ_ASSOCs, 2-120, 2-145
- OE_PRICE_ADJ_ATTRIBS, 2-119, 2-143
- OE_PRICE_ADJS_IFACE_ALL, 2-58
- OE_PRICE_ADJS_IFACE_ALL Derived Values, 2-60
- OE_PRICE_ADJUSTMENTS, 2-114, 2-139
- OE_PRICE_ATTS_IFACE_ALL, 2-61
- OE_PRICE_ATTS_IFACE_ALL Derived Values, 2-62
- OE_RESERV_TNS_IFACE_ALL, 2-66
- OE_RESERV_TNS_IFACE_ALL Derived Values, 2-67
- OE_SALES_CREDITS, 2-122, 2-146
- OEOH OEOL Workflow, 8-32
- OESO Workflow, 7-61
- OEXSSOCP, 7-62
- OM
 - Item Validation Organization, 2-5
- ON Demand, 8-33
- ONT_3A4A_OAG72_OUT.xgm, 7-47
- ONT_3A4R_OAG72_IN, 8-6
- ONT_3A4R_OAG72_IN.xgm, 7-7
- ONT_3A6_OAG72_OUT_SSO, 8-35
- ONT_3A6_OAG72_OUT_SSO.xgm, 7-60
- ONT_3A7_OAG72_OUT_SO, 7-72
- ONT_3A8R_OAG72_IN.xgm, 7-87, 7-108
- ONT_CUST_PO_NUMBER, 7-66
- ONT_TP_LOCATION_CODE, 7-65
- Open Application Programmatic Interface, 1-9
- Open Applications Group, 7-4
- Open Interface and Public API Comparison, 4-164
- Open Interface diagram, 1-8
- Open Interface Tracking, 7-140, 8-68
- Open Interface Tracking Window, 8-78
- Open Orders Only, 7-50
- Optional Columns, 1-11
- Oracle Exchange, 8-4
- Oracle Order Management APIs/Open Interfaces, 1-4
- Oracle Order Management with Oracle Receivables and Invoicing, 2-159
- Oracle Purpose Code, 9-4
- Oracle Workflow Business Event System, 8-8
- Oracle XML Gateway Details, 8-27
- oracle.apps.ont.oi.po_acknowledge.create, 8-26
- oracle.apps.ont.oi.po_inbound.create, 8-7
- oracle.apps.ont.oi.show_so.create, 8-35
- oracle.apps.ont.oi.showso.create, 8-32
- Order Changes, 7-51
- Order Import, 2-4, 7-4
 - failed processing, 7-5
 - successful processing, 7-5
 - unexpected Errors, 7-5
- Order Import Flow - Generic, 7-105, 7-129, 8-10
- Order Management Parameters, 2-5
- Order Status, 2-13
- OUTBOUND, 7-51
- Outbound EDI Messages, 7-143
- Outbound Execution engine, 9-12
- Outbound XML Messages, 7-143

P

P_ATTR_MAPPING_PUB Application Program Interface, 5-303
P_CON_REC, 5-294, 5-297
P_CON_VAL_REC, 5-294, 5-298
P_PTE_REC Parameters, 5-51
P_PTE_REC_TYPE Parameters, 5-55
P_PTE_VAL_REC Parameters, 5-55
P_SEG_TBL, 5-295
P_SEG_VAL_TBL, 5-295
p_serial_range_tab, 4-99
Parameters to Add, 7-63
partial cancellations, 7-117
Partner Collaboration - RosettaNet, 7-3
Payment Term Comparison, 2-16
periodical, 7-50
Periodical Support, 8-29
PIP, 7-2
PIP 3A4, 7-2
PIP 3A6, 7-2
PIP 3A9, 7-2
Planning Schedule, 9-2
Planning/Shipping Inbound, 9-5
PL/SQL Record Structures, 2-103
Populated Parameters, 7-146
Post Process Action, 8-23
pre-defined Workflow event, 7-105, 7-128, 8-9
Prerequisites and Set-Up, 2-5
Price Comparisons, 2-16
Price List Setup Application Program Interface, 5-199
Price List Setup Group Application Program Interface, 5-240
Price Request Application Program Interface, 5-255
PRICE_LIST_LINE_REC_TYPE, 5-204, 5-246
PRICE_LIST_LINE_REC_TYPE Parameters, 5-13
PRICE_LIST_LINE_TBL_TYPE, 5-207, 5-248
PRICE_LIST_LINE_TBL_TYPE Parameters, 5-16
PRICE_LIST_LINE_VAL_REC_TYPE, 5-249
PRICE_LIST_LINE_VAL_REC_TYPE Parameters, 5-16
PRICE_LIST_LINE_VAL_TBL_TYPE, 5-250
PRICE_LIST_REC, 5-186
PRICE_LIST_REC_TYPE, 5-201, 5-243
PRICE_LIST_REC_TYPE Parameters, 5-9
PRICE_LIST_TBL, 5-186
PRICE_LIST_TBL_TYPE, 5-203, 5-245
PRICE_LIST_TBL_TYPE Parameters, 5-11
PRICE_LIST_VAL_REC_TYPE, 5-204, 5-245
PRICE_LIST_VAL_REC_TYPE Parameters, 5-11
PRICE_LIST_VAL_TBL_TYPE, 5-204, 5-208, 5-245
PRICE_LIST_VAL_TBL_TYPE Parameters, 5-13, 5-18, 5-21
PRICE_REQUEST, 5-260
PRICE_REQUEST(Overloaded), 5-261
Pricing Agreements, 2-14
Pricing Attributes and Qualifier Attributes, 5-288
PRICING_ATTR_REC_TYPE, 5-210, 5-251
PRICING_ATTR_REC_TYPE Parameters, 5-18, 5-88
PRICING_ATTR_TBL_TYPE, 5-253
PRICING_ATTR_TBL_TYPE Parameters, 5-20, 5-92
PRICING_ATTR_VAL_REC_TYPE, 5-212, 5-253
PRICING_ATTR_VAL_REC_TYPE Parameters, 5-20, 5-93
PRICING_ATTR_VAL_TBL_TYPE, 5-213, 5-254
PRICING_ATTR_VAL_TBL_TYPE Parameters, 5-93
Process Function, 1-13
Process Order Application Open Interface, 2-69
Process Order Interface (API), 2-15
Process Order Usage, 2-148
Process Orders Entities and Associated Tables, 2-70
Process R_SHOW_SALES_ORDER, 7-81
Process_PO (3A4 Request PO), 7-4
PROCESS_AGREEMENT Parameters, 5-3
PROCESS_ATTR_MAPPING Parameters, 5-50
PROCESS_ATTRIBUTES, 5-293
PROCESS_CURRENCY, 5-188
PROCESS_MODIFIERS Parameters, 5-61
Process_PO, 7-4
 Setup Steps, 8-3
PROCESS_PO DTD, 7-28
Process_PO inbound message, 7-34
PROCESS_PRICE_LIST, 5-200, 5-241
PROCESS_QUALIFIER_RULES, 5-309
Processing claims, 9-9

Processing Constraints, 2-16
profile option
 OM Run Order Import for XML, 7-4, 8-51, 8-67
Profile Options, 2-5, 8-66
PubBatchInfoRecType RECORD
 DEFINITION, 4-152
PubFreightCostRecType RECORD
 DEFINITION, 4-137, 4-144
Purge Open Interface Data Concurrent
 Program, 2-18
Purge Procedure, 7-141

Q

QP_ATTR_MAPPING_PUB, 5-303
QP_ATTRIBUTES_PUB Application Program
 Interface, 5-293
QP_LIMITS, 5-161
QP_LIMITS_PUB.GET_LIMITS Parameters, 5-160
QP_LIMITS_PUB.LOCK_LIMITS
 Parameters, 5-159
QP_LIMITS_PUB.PROCESS_LIMITS
 Parameters, 5-158
QP_PREQ_GRP.INSERT_LINE_ATTRS2, 5-259
QP_PREQ_GRP.INSERT_LINES2, 5-257
QP_PRICE_FORMULA.PROCESS_PRICE_
 FORMULA Parameters, 5-134
QUAL_REC_TYPE, 5-283
QUALIFIER_RULES_REC_TYPE, 5-310
QUALIFIER_RULES_VAL_REC_TYPE, 5-313
QUALIFIER_RULES_VAL_TBL_TYPE
 Parameters, 5-314
QUALIFIER_VAL_REC_TYPE, 5-319
Qualifiers Application Program Interface, 5-308
QUALIFIERS_REC_TYPE, 5-209, 5-250
QUALIFIERS_REC_TYPE Parameters, 5-83, 5-314
QUALIFIERS_TBL_TYPE, 5-209, 5-250
QUALIFIERS_TBL_TYPE Parameters, 5-87, 5-319
QUALIFIERS_VAL_REC_TYPE, 5-209, 5-250
QUALIFIERS_VAL_REC_TYPE Parameters, 5-87
QUALIFIERS_VAL_TBL_TYPE, 5-210, 5-251
QUALIFIERS_VAL_TBL_TYPE Parameters, 5-88,
 5-320

R

RA_INTERFACE_LINES, 2-166
RA_INTERFACE_SALESCREDITS, 2-181
Raise Send Acknowledgement Sub Process, 7-108,
 7-132, 8-13
Raise Send CBOD Out Sub Process, 7-106, 7-130,
 8-11
Related Customers, 2-6
RELATED_LINES_REC_TYPE, 5-283, 5-285
Release Hold, 2-156
Release_Batch API, 4-160
REQ_LINE_ATTRS_REC, 5-180
REQ_LINE_ATTRS_TBL, 5-180
Required Columns, 1-11
Returns, 2-11
Reverse Limits Application Program
 Interface, 5-328
Reverse_Limits Parameters, 5-329
RLA_TRX_PURP, 9-4
RLM_INTERFACE_HEADERS_ALL, 3-2
RLM_INTERFACE_LINES_ALL, 3-22
RosettaNet, 7-2
Rosettanet and OAG Transactions, 7-2
RosettaNet Partner Interface Processes, 7-2
Round Price Application Program Interface, 5-333
round_price Parameters, 5-334
Run Order Import Sub Process, 7-107, 7-131, 8-12

S

Sales Order date from, 7-51
Sales Order date to, 7-51
Sales Order No From, 7-50
Sales Order No To, 7-50
Scheduling, 2-14
Search For EDI/XML Electronic Messages, 7-140
seed codes for the code conversion, 8-28
seeded event subscriptions, 8-8
Seeded Pricing Events, 5-265
Seeded workflow OEM/ Open Interface
 Tracking, 7-144
SEND DOCUMENT function, 7-61
Send XML PO Acknowledgment, 7-49
Setting Up the Lock_Order Procedure, 2-97

- Setting Up the Process Order Procedure, 2-84
- Setup, 8-63
- Setup Oracle Workflow Business Event
 - System, 8-64
- Setup the Oracle Workflow Business Event
 - System, 8-3
- Shipping API Flow Scenario 1, 4-10
- Shipping API Flow Scenario 2, 4-11
- Shipping API Flow Scenario 3, 4-12
- Shipping API Flow Scenario 4, 4-13
- Shipping API Flow Scenario 5, 4-14
- Shipping Execution APIs
 - Container Public API, 4-116
 - Deliveries Public API, 4-59
 - Delivery Details Public API, 4-90
 - Freight Costs Public API, 4-134
 - Stop Public API, 4-49
 - Trip Public API, 4-40
- Shipping Schedule, 9-2
- Shipping Schedule (862), 9-5
- Shipping Transaction Form/Public API
 - Correlation, 4-3
- Short Name Key, 5-242
- Show Sales Order - Concurrent Program, 7-62
- Show_SalesOrder, 8-29
 - Setup Steps, 8-30
- Show_salesorder, 7-50
- Show_SalesOrder Message Map, 7-52
- Show_SalesOrder Workflow, 7-67
- ShowShipment Outbound, 9-7
- ShowShipStatus, 9-7
- Simplified graphical representation of an ATO
 - configuration, 2-79
- Single Customer, 2-6
- Solution Provider, 7-2
- source application, 1-10
- SP, 7-2
- Split_Line API, 4-94
- Standard Code Conversion Window, 8-7
- Status change due to business events, 7-51
- Status Code, 5-335
- Status Send on the Outbound
 - Acknowledgement, 8-47
- Stop Actions, APIs, and Parameters, 4-27
- Stop Public Application Program Interface, 4-49

- Subscriber
 - System, 8-35
- subscriber, 8-67
 - system, 8-7
- successful processing, 7-119
- Support For Split Lines, 7-85
- Support for Track and Trace, 9-8
- SYNC PLANSCHD, 9-2
- SYNC SHIPSCHD, 9-2
- synchronous, 7-4, 7-105, 7-128, 8-9

T

- Tracking Message, 9-8
- trading partner, 8-8
- Trading partner id, 7-50
- Trading Partner Setup Window, 8-6
- trading partner specific code conversion
 - values, 8-5
- trading partner/hub name, 8-5
- transaction name, 8-4
- Transaction Sources, 2-11
- Transaction Sources Window
 - Required Settings, 2-160
- Transaction Type, 8-6
- transport protocol, 8-5
- trigger points, 7-35
- Trip Actions, APIs, and Parameters, 4-26
- Trip Public Application Program Interface, 4-40
- TRIP_PUB_REC_TYPE Record Definition, 4-43
- TRIP_STOP_PUB_REC_TYPE RECORD
 - DEFINITION, 4-52, 4-122

U

- UNIVERSAL, 8-7
- Update Container API, 4-120
- Update_Shipping_Attributes API, 4-97
- UPDATE_SHIPPING_ATTRIBUTES
 - Mapping, 4-165
- USER_ATTRIBUTE_REC_TYPE, 5-306
- USER_ATTRIBUTE_TBL_TYPE, 5-306
- USERAREA, 8-28
- USERAREA Data, 8-28

V

Validate Function, 1-12
validate imported data, 1-2
Validate_Freight_Cost_Type API, 4-141
Validate_Price_list_Curr_code Application Program Interface, 5-336
Validate_Price_list_Curr_code Parameters, 5-336
Validation of Pricing Object Security API, 5-292
Verifying guaranteed service level, 9-9
View the Delivery Activity, 9-10
View the Delivery Detail Information, 9-10, 9-11
View the Delivery Exception, 9-10, 9-12
View the Delivery History, 9-11
View Workflow for XML Messages, 7-140
View XML Data For a Particular Message, 7-140

W

WF Support, 8-29
window
 Trading Partner Code Conversion, 8-5
 Trading Partner Setup, 8-5, 8-36
Workflow Administrator responsibility, 8-29
Workflow Event Setup, 7-10, 7-36, 7-60, 7-135
Workflow for The Raise Event, 8-33
Workflow for the Raise Show Sales Order Event Sub Process, 8-33
WSH_CONTAINER_PUB.AUTO_PACK
 Parameters, 4-128
WSH_CONTAINER_PUB.CONTAINER_ACTIONS
 Parameters, 4-131
WSH_CONTAINER_PUB.CREATE_CONTAINERS
 Parameters, 4-116
WSH_DELIVERIES_PUB.CREATE_UPDATE_
 DELIVERY Parameters, 4-60
WSH_DELIVERIES_PUB.DELIVERY_ACTION
 Parameters, 4-71, 4-169
WSH_DELIVERY_DETAILS, 4-120
WSH_DELIVERY_DETAILS_PUB.AUTOCREATE_
 DEL_TRIP Parameters, 4-113
WSH_DELIVERY_DETAILS_PUB.AUTOCREATE_
 DELIVERIES Parameters, 4-110
WSH_DELIVERY_DETAILS_PUB.DETAIL_TO_
 DELIVERY Parameter, 4-91

WSH_DELIVERY_DETAILS_PUB.SPLIT_LINE
 Parameters, 4-94, 4-97
WSH_FREIGHT_COSTS_PUB.CREATE_UPDATE_
 FREIGHT_COSTS Parameters, 4-135
WSH_FREIGHT_COSTS_PUB.DELETE_FREIGHT_
 COSTS Parameters, 4-141, 4-142
WSH_PICKING_BATCHES_PUB.CREATE_BATCH
 Parameters, 4-149
WSH_PICKING_BATCHES_PUB.RELEASE_
 BATCH Parameters, 4-161
WSH_TRIP_STOP_PUB.STOP_ACTION
 Parameters, 4-56
WSH_TRIP_STOPS_PUB.CREATE_UPDATE_STOP
 Parameters, 4-50
WSH_TRIPS_PUB.CREATE_UPDATE_TRIP, 4-40
WSH_TRIPS_PUB.CREATE_UPDATE_TRIP
 Parameters, 4-41
WSH_TRIPS_PUB.TRIP_ACTION
 Parameters, 4-46

X

X_CON_REC, 5-295, 5-299, 5-301
X_CON_VAL_REC, 5-296, 5-299, 5-302
x_exceptions_tab Fields, 4-89
X_PTE_REC_TYPE Parameters, 5-52, 5-56, 5-58
X_PTE_VAL_REC Parameters, 5-56
X_SEG_TBL, 5-296, 5-300, 5-302
X_SEG_VAL_TBL, 5-296, 5-300, 5-302
XC_ACTION_REC_TYPE Parameters, 4-81
XML Gateway, 8-8
XML Gateway Execution Engine, 8-8
XML message, 7-105, 7-128, 8-9
XML or EDI Messages, 9-11

