

# **Oracle® Advanced Pricing**

Implementation Guide

Release 12.2

**Part No. E48846-12**

November 2023

Oracle Advanced Pricing Implementation Guide, Release 12.2

Part No. E48846-12

Copyright © 2004, 2023, Oracle and/or its affiliates.

Primary Author: Swathi Mathur

Contributing Author: Gowri Arur, Jagan Putta, Vijayarani Bommareddy, Smitha Balaraman

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

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

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

U.S. GOVERNMENT END USERS: Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

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

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

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

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

---

# Contents

**Send Us Your Comments**

**Preface**

## **1 Introduction**

Overview of Oracle Advanced Pricing.....	1-1
Oracle Advanced Pricing Versus Basic Pricing.....	1-2
Terminology.....	1-33
Oracle Advanced Pricing Features.....	1-34
Process Flow for Implementation.....	1-41

## **2 Implementation Overview**

Setup Flow to Implement Oracle Advanced Pricing .....	2-1
---	-----

## **3 Implementation Methodology**

Overview of Implementation Methodology.....	3-1
Using Oracle Advanced Pricing in the HTML Interface.....	3-4
Methodology Steps.....	3-5

## **4 Oracle Advanced Pricing Command Center Implementation**

Advanced Pricing Command Center Configuration.....	4-2
Setting Up Advanced Pricing Command Center.....	4-2
Setup and Configuration Steps for Advanced Pricing Command Center.....	4-2
Reviewing Security Setup for Advanced Pricing Command Center.....	4-2
Profile Options for Advanced Pricing Command Center.....	4-3
Configuring Descriptive Flexfields for Search.....	4-4

Loading Pricing Data.....	4-4
---------------------------	-----

## 5 Profile Options and Pricing Parameters

Overview of Profile Options.....	5-1
Profile Options Setup Summary.....	5-3
Overview of Pricing Parameters.....	5-43
Viewing Pricing Parameter Definitions.....	5-44
Viewing and Updating Pricing Parameter Values.....	5-45
Seeded Pricing Parameters.....	5-46

## 6 Pricing Security

Overview of Oracle Pricing Security.....	6-1
Pricing Security and Operating Units.....	6-4
Setup Steps for Implementing Pricing Security.....	6-5
Changes to Pricing User Interfaces (UI) after Upgrading and Turning On Security .....	6-7
Assigning Pricing Entity Usage.....	6-11
Implementation Suggestions for Privileges.....	6-13
Creating Privileges.....	6-14
Creating Entity Sets .....	6-17
Setting Security Profile Options .....	6-19
Turning On Pricing Security.....	6-26

## 7 Pricing Data Bulk Loader

Overview of Pricing Data Bulk Loader.....	7-1
Populating the Interface Table.....	7-2
QP: Bulk Import of Price List Program.....	7-12

## 8 Price Lists

Overview of Price Lists .....	8-1
Bulk Importing of Price Lists.....	8-4
Usage Price Break Proration.....	8-4

## 9 Price Book

Overview of Price Book.....	9-1
Implementation Steps for Price Book.....	9-4
Setting Up Price Book Profile Options.....	9-4
Setting Up the E-mail Server .....	9-5
Setting Up the Default Printer.....	9-5
Setting Up Oracle XML Publisher.....	9-5

Setting Up the XML Gateway Message Maps.....	9-6
Setting Up the Price Book User Interface (UI).....	9-8
Confirm Pricing Parameter Setup.....	9-8
<b>10 Modifiers</b>	
Overview of Modifiers.....	10-1
Implementation Planning .....	10-4
Modifier Levels and Application Methods.....	10-6
Other Modifier Considerations.....	10-10
Pricing Controls.....	10-13
Modifier Type Setup.....	10-14
Setting up Accrual Discounts.....	10-21
Using Accumulated Range Breaks.....	10-23
Setting Up Runtime Sourcing for Accumulated Range Breaks.....	10-23
<b>11 Archiving and Purging Pricing Entities</b>	
Overview of Archiving and Purging Pricing Entities.....	11-1
Using API to Archive Pricing Entities.....	11-2
Using API to Purge Pricing Entities.....	11-4
<b>12 Auditing</b>	
Auditing .....	12-1
Setting up Auditing.....	12-1
Audit Setup.....	12-2
<b>13 Multicurrency Price Lists and Agreements</b>	
Overview of Multicurrency Price Lists and Agreements.....	13-1
Example of Multicurrency Price List Usage.....	13-3
Fresh Install Using Multiple Currency Price List.....	13-7
Upgrading from Single Currency Price List to Multiple Currency Price List.....	13-8
Implementation Decisions for Creating Multicurrency Conversion Lists.....	13-9
Using Multiple Currency Price List with Other Oracle Products.....	13-13
<b>14 Unit of Measure</b>	
Overview of Unit of Measure.....	14-1
Defining UOM.....	14-1
Pricing Actions: Primary UOM and Pricing UOM.....	14-2
Pricing Controls: Profile Options.....	14-3

## 15 Multiple Organizations

Overview of Multiple Organization Model.....	15-1
Operating Units.....	15-1
Item Validation Organizations.....	15-3

## 16 Precedence and Best Price

Overview of Precedence and Best Price.....	16-1
Default Precedence Numbers.....	16-2
Matched Qualifiers for Modifiers/Price Lists.....	16-2
Price List Incompatibility Resolution.....	16-3
Modifier Incompatibility Resolution.....	16-3
Setting Up Incompatibility Groups.....	16-7
Incompatibility Resolution Examples.....	16-8

## 17 Attribute Management

Overview of Attribute Management.....	17-1
Creating Contexts and Attributes for Pricing Setup Windows.....	17-3
Deleting Contexts and Attributes.....	17-11
Linking Attributes to a Pricing Transaction Entity.....	17-11
Mapping Attributes of Type ATTRIBUTE MAPPING.....	17-18
Running the Build Attribute Mapping Rules Program (Attribute Mapping Only).....	17-21
Creating Source Systems.....	17-25
Creating a New Pricing Transaction Entity.....	17-25
Using Custom Sourced Attributes.....	17-30
Restoring Seeded Data Using the Restore Defaults Button.....	17-31
Troubleshooting While Setting Up Attributes .....	17-33
Troubleshooting in Pricing Setup windows Related to Attribute Management .....	17-33
Troubleshooting During Pricing Setup .....	17-34
Troubleshooting During Integration or Runtime .....	17-34
Upgrading Considerations .....	17-36
Upgrading Context and Attributes.....	17-38
Mapping of Seeded Request Types and Source Systems.....	17-38
Creating PTE and Attribute Links.....	17-39
Upgrade Attribute Mapping Rules.....	17-40
Assigning PTE to Existing Modifiers.....	17-41

## 18 Get Custom Price

Overview of Get_Custom_Price Implementation.....	18-1
--	------

Implementing Get_Custom_Price.....	18-4
Get_Custom_Price_Customized.....	18-7
<b>19 Events and Phases</b>	
Overview of Events and Phases.....	19-1
What Are Pricing Events?.....	19-1
What Are Pricing Phases?.....	19-3
Assigning Pricing Phases.....	19-4
<b>20 Pricing Engine Request Viewer Window</b>	
Overview of Pricing Engine Request Viewer.....	20-1
Setting Up the User Profiles.....	20-2
Regions in the Pricing Engine Request Viewer.....	20-3
Pricing Engine Requests Region.....	20-4
Pricing Engine Request Lines Region.....	20-8
Pricing Engine Request Line Details Region.....	20-13
Attributes Window.....	20-20
Related Lines Window.....	20-27
Formula Step Values Window.....	20-28
Debug Log Window.....	20-30
Analyzing Error Messages.....	20-31
<b>21 Integrating with Oracle Advanced Pricing</b>	
Overview of Integrating with Oracle Advanced Pricing.....	21-1
Integration Steps Required for Pricing.....	21-1
Pricing Engine Interaction Details.....	21-7
Changed Lines API.....	21-28
Oracle Service Contracts (OKS) Integration: Proration and Price List Locking.....	21-30
Integration Flow for Price List Locking.....	21-32
Integration Flow for Proration.....	21-36
Duration and Partial Period Pricing of Service Items.....	21-38
Pricing Features to Support Telecommunications Industry Flows [Oracle Telecommunications Service Ordering (TSO)].....	21-42
<b>22 High Volume Order Processing</b>	
Overview of High Volume Order Processing.....	22-1
<b>23 Technical Considerations</b>	
Basic versus Oracle Advanced Pricing.....	23-1

Oracle Advanced Pricing Engine Processing.....	23-3
Search Engine.....	23-3
Calculation Engine .....	23-21
Extendibility Features.....	23-23

## 24 Diagnostics and Troubleshooting

Overview of Diagnostics and Troubleshooting.....	24-1
Summary of Pricing Engine Messages and Diagnosis.....	24-4
Common Troubleshooting Problems in Pricing Windows.....	24-13
Other Technical Considerations.....	24-26
Attribute Management Troubleshooting.....	24-27

## A Windows and Navigation Paths

Windows and Navigation Paths.....	A-1
-----------------------------------	-----

## B Attribute Seed Data

Overview of Attribute Seed Data.....	B-1
Complex Maintenance Repair and Overhaul PTE Attributes.....	B-2
Demand Planning PTE Attributes.....	B-3
Intercompany Transaction PTE Attributes .....	B-4
Logistics PTE Attributes .....	B-7
Order Fulfillment PTE Attributes .....	B-14
Procurement PTE Attributes.....	B-30

## C Seeded Formulas

Overview of Seeded Formulas.....	C-1
Seeded Cost-to-Charge Conversion Formulas .....	C-2
Seeded Markup Formulas.....	C-5

## D Optimal Performance

Overview of Optimal Performance.....	D-1
Oracle Advanced Pricing Setup Considerations.....	D-4
Qualifier Selectivity.....	D-4
Qualifier Selectivity Examples.....	D-5
Pattern Search.....	D-10
Ignore Pricing.....	D-10
Additional Tips for Better Performance.....	D-11
Analyzing your Data Distributions Using a Script.....	D-11
Technical Improvements.....	D-11

**E Case Study: Pricing Scenarios in the High-Tech Industry**

Overview of Pricing Scenario..... E-1  
Applying Oracle Advanced Pricing..... E-4  
Results of Pricing Scenario..... E-5

**F Case Study: Using Oracle Advanced Pricing Formulas for Healthy Fast Food**

Introduction..... F-1  
Problem Definition..... F-1  
Pricing the Burger ..... F-2

**G Lookups**

Overview of Lookups..... G-1

**Index**



---

# Send Us Your Comments

## Oracle Advanced Pricing Implementation Guide, Release 12.2

### Part No. E48846-12

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

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

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

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

Send your comments to us using the electronic mail address: [appsdoc\\_us@oracle.com](mailto:appsdoc_us@oracle.com)

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

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

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



---

# Preface

## Intended Audience

Welcome to Release 12.2 of the *Oracle Advanced Pricing Implementation Guide*.

This guide assumes you have a working knowledge of the principles and customary practices of the Oracle Advanced Pricing business area. If you have never used Oracle Advanced Pricing, Oracle suggests you attend one or more of the Oracle Applications training classes available through Oracle University. To learn more about Oracle Self Service Web Applications, see the Oracle Self-Service Web Applications Implementation Manual. To learn more about the Oracle Applications graphical user interface, see the Oracle E-Business Suite User's Guide.

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

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

## Documentation Accessibility

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

## Access to Oracle Support

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

## Structure

- 1 Introduction
- 2 Implementation Overview
- 3 Implementation Methodology
- 4 Oracle Advanced Pricing Command Center Implementation
- 5 Profile Options and Pricing Parameters
- 6 Pricing Security
- 7 Pricing Data Bulk Loader
- 8 Price Lists
- 9 Price Book
- 10 Modifiers
- 11 Archiving and Purging Pricing Entities
- 12 Auditing
- 13 Multicurrency Price Lists and Agreements
- 14 Unit of Measure
- 15 Multiple Organizations
- 16 Precedence and Best Price
- 17 Attribute Management
- 18 Get Custom Price
- 19 Events and Phases
- 20 Pricing Engine Request Viewer Window
- 21 Integrating with Oracle Advanced Pricing
- 22 High Volume Order Processing
- 23 Technical Considerations
- 24 Diagnostics and Troubleshooting
- A Windows and Navigation Paths
- B Attribute Seed Data
- C Seeded Formulas
- D Optimal Performance
- E Case Study: Pricing Scenarios in the High-Tech Industry
- F Case Study: Using Oracle Advanced Pricing Formulas for Healthy Fast Food
- G Lookups

## Related Information Sources

### Integration Repository

The Oracle Integration Repository is a compilation of information about the service endpoints exposed by the Oracle E-Business Suite of applications. It provides a complete catalog of Oracle E-Business Suite's business service interfaces. The tool lets users easily discover and deploy the appropriate business service interface for integration with any system, application, or business partner.

The Oracle Integration Repository is shipped as part of the Oracle E-Business Suite. As your instance is patched, the repository is automatically updated with content appropriate for the precise revisions of interfaces in your environment.

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

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

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

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

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



---

# Introduction

This chapter covers the following topics:

- Overview of Oracle Advanced Pricing
- Oracle Advanced Pricing Versus Basic Pricing
- Terminology
- Oracle Advanced Pricing Features
- Process Flow for Implementation

## Overview of Oracle Advanced Pricing

Oracle Advanced Pricing is a rules-based application with an engine component to service the pricing requirements of Oracle applications to price customer transactions. Oracle Advanced Pricing enables you to set up *pricing actions* such as price lists, agreements, formulas, and modifiers that the pricing engine applies to transactions.

Oracle Advanced Pricing also enables you to define a set of *pricing rules* (and pricing controls that can be used in conjunction with the rules) to precisely govern how and when pricing actions are applied to transactions.

The pricing engine is a software component of Oracle Advanced Pricing that is called by an application such as Oracle Order Management or iStore to apply pricing actions to transactions. Applications make calls to the pricing engine when transactions need pricing services or need to be priced.

A calling application must provide the pricing engine with information about the product to be priced and the customer. This information is contained in an internal format called a *pricing request structure*. The pricing engine extracts from its tables the applicable pricing rules, pricing actions, and control information that have been set up. The engine then selects the proper actions and computes their effect. This information is returned by the engine to the calling application.

Oracle Advanced Pricing provides these capabilities to the pricing engine using open API calls. These APIs, along with others, are documented in the *Oracle Order*

## Oracle Advanced Pricing Versus Basic Pricing

The term basic pricing refers to a component of Oracle Order Management that provides pricing functionality when Oracle Advanced Pricing is not installed. Oracle Advanced Pricing and basic pricing have common software components; however, Oracle Advanced Pricing extends and expands the capabilities of basic pricing.

The pricing system software components examine the installation type (either full or shared) to determine the appropriate mode in which to run. Users of basic pricing are installed as "shared" and are not licensed to use Oracle Advanced Pricing capabilities. When in basic mode, the pricing system software components restrict exposure of advanced features in the setup windows. Because the information necessary to drive Oracle Advanced Pricing functionality cannot be set up in a pricing implementation running in basic mode, use of Oracle Advanced Pricing features is also inhibited.

Users who have licensed Oracle Advanced Pricing are installed as "Full." The pricing setup windows enable setup for all information needed to drive features provided by Oracle Advanced Pricing.

The following table describes the primary differences between Oracle Advanced Pricing and the basic pricing capabilities included in Oracle Order Management:

<b>Pricing Features</b>	<b>Basic Pricing in Oracle Order Management</b>	<b>Oracle Advanced Pricing</b>
Archive and Purge Pricing Entities	Same as Advanced Pricing.	Supports removal of price list and modifier list lines data no longer required for current operations
Pricing Security (Privileges)	Same as Advanced Pricing.	The Pricing Security Administrator responsibility can grant access privileges to the following pricing entity types: <ul style="list-style-type: none"><li>• Standard Price List</li><li>• Modifier</li><li>• Agreement Price List</li></ul>

Pricing Features	Basic Pricing in Oracle Order Management	Oracle Advanced Pricing
Pricing Security (Entity Sets)	Entity sets are not available in basic pricing.	Using the Entity Set Page, the Pricing Administrator can create "sets" of pricing objects such as a set of price lists that can be granted access roles.
Pricing Security (Access Levels)	<p>The following access levels are supported by site level profile settings:</p> <ul style="list-style-type: none"> <li>• View Only</li> <li>• Maintain</li> </ul>	Access levels and privileges are the same as in basic pricing.
Pricing Security (Privileges)	<p>You can grant access privileges to the following Grantee types:</p> <ul style="list-style-type: none"> <li>• Global</li> <li>• Operating Unit</li> </ul> <p>Pricing Security supports authorization roles for creating maintaining, and viewing pricing data. Security Privileges are set up and managed in the HTML user interface.</p>	In addition to those in basic pricing, you can grant access privileges to the following Grantee Types: Responsibility and User.

Pricing Features	Basic Pricing in Oracle Order Management	Oracle Advanced Pricing
Price list	<p data-bbox="630 338 862 365">Enables you to set up:</p> <ul data-bbox="630 394 1032 1283" style="list-style-type: none"> <li data-bbox="630 394 927 422">• Price List Header/Lines</li> <li data-bbox="630 464 902 491">• One currency per list</li> <li data-bbox="630 533 886 560">• Effective date at list</li> <li data-bbox="630 602 862 630">• Rounding factors</li> <li data-bbox="630 672 837 699">• Payment terms</li> <li data-bbox="630 741 821 768">• Freight terms</li> <li data-bbox="630 810 837 837">• Freight carriers</li> <li data-bbox="630 879 1032 907">• Percentage price for service items</li> <li data-bbox="630 949 813 976">• Service price</li> <li data-bbox="630 1018 797 1045">• Active Flag</li> <li data-bbox="630 1087 870 1115">• Mobile Download</li> <li data-bbox="630 1157 1008 1209">• Global Flag to support Security feature</li> <li data-bbox="630 1251 1016 1283">• Security enabled rules checking</li> </ul>	<p data-bbox="1089 338 1373 716">Advanced Pricing includes all basic price list features and adds the capability to define point and range price breaks on a price list. In the current release, all price breaks have been made continuous, which means that the high value of one break would be the low value of the next break.</p>
Pricing attributes on Price List	<p data-bbox="630 1346 1032 1440">In basic pricing, you can use one data source, called a context, per order line. Each context can have 100 attributes.</p>	<p data-bbox="1089 1346 1373 1598">Advanced Pricing adds the capability to support multiple contexts. You can use seeded values or you may define your own contexts. You are limited to 100 attributes per context.</p>

Pricing Features	Basic Pricing in Oracle Order Management	Oracle Advanced Pricing
Secondary price lists	One secondary price list is supported.	<p data-bbox="1187 338 1446 527">Advanced Pricing adds capability to define and use unlimited secondary lists. The chaining of secondary lists is <i>not</i> supported.</p> <p data-bbox="1187 558 1446 642">A profile option enables a qualifier check for secondary price lists.</p>

Pricing Features	Basic Pricing in Oracle Order Management	Oracle Advanced Pricing
Qualifiers: Price Lists	Qualifiers are not supported for price lists in basic pricing. In Oracle Order Management, you can create defaulting rules that define when (and thus qualify) a price list is defaulted to the sales order header or line. As an example, you can qualify a price list based on order type, customer, and other attributes from Order Management.	<p>Qualifiers are fully supported for price lists in Advanced Pricing:</p> <ul style="list-style-type: none"> <li>• Ability to attach qualifier groups or enter qualifiers for the price list.</li> <li>• All seeded context values for Advanced Pricing as well as user-defined qualifiers (requires Attribute Mapping) are supported.</li> <li>• Precedence is derived from the Precedence Number field on the Contexts Setup window and can be configured by the user in Advanced Pricing.</li> <li>• Multiple with and/or relations between qualifiers is possible.</li> <li>• Qualifiers enable you to define unlimited price lists for an item.</li> <li>• Note: You cannot define Order Amount as a qualifier.</li> </ul>
Product hierarchy price list notes	Basic pricing only supports flattened categories.	Advanced Pricing supports hierarchical and flattened categories.

Pricing Features	Basic Pricing in Oracle Order Management	Oracle Advanced Pricing
Price breaks on price lists	Not supported in basic pricing.	<p data-bbox="1187 338 1463 457">Advanced Pricing enables price breaks on price lists using the following measures of quantity:</p> <ul data-bbox="1187 489 1446 785" style="list-style-type: none"> <li data-bbox="1187 489 1328 516">• Quantity</li> <li data-bbox="1187 558 1442 611">• Amount (excluding Item Amount)</li> <li data-bbox="1187 653 1312 680">• Weight</li> <li data-bbox="1187 722 1430 785">• Other user defined attributes</li> </ul> <p data-bbox="1230 816 1458 968">The following break types are supported for Application Method of Unit Price and Percent Price:</p> <ul data-bbox="1187 1010 1365 1108" style="list-style-type: none"> <li data-bbox="1187 1010 1354 1037">• Point break</li> <li data-bbox="1187 1079 1365 1108">• Range break</li> </ul>
Price include breaks on price lists: Block Pricing	Not supported in basic pricing.	<p data-bbox="1187 1171 1463 1325">Use Block Pricing (Application Method of Block Price) to define a price for the entire set of a block:</p> <ul data-bbox="1187 1356 1446 1478" style="list-style-type: none"> <li data-bbox="1187 1356 1446 1478">• Block pricing can be used with existing setups (Point Breaks and Range Breaks).</li> </ul>

Pricing Features	Basic Pricing in Oracle Order Management	Oracle Advanced Pricing
Multi-Currency Conversion Lists	Basic pricing does not include multi-currency conversion lists.	<p>Maintain prices in a single list with one base currency and to define multiple currency conversion rates and currency-specific markup/markdown equations.</p> <p>A Multi-Currency Conversion window enables users to define Currency To conversion criteria.</p> <p>Seeded conversion types include: Fixed, Formula, User-defined, Spot, EMU fixed, transaction, and corporate.</p> <ul style="list-style-type: none"> <li>• Some of these seeded conversion types require Oracle General Ledger to be installed.</li> <li>• You can define markup criteria per currency definition.</li> <li>• A concurrent program Update Price Lists with Multi-Currency Conversion Criteria is provided.</li> </ul>

Pricing Features	Basic Pricing in Oracle Order Management	Oracle Advanced Pricing
Formulas	<p data-bbox="724 338 1154 527">Basic pricing enables you to define Static Formulas. Static Formulas require a concurrent manager program to be run that populates the list price column in the price list window so that price is available for engine calls.</p> <p data-bbox="724 554 1154 743">With basic pricing, you can attach formulas to price lists. These formulas can include mathematical operators, numeric operands, and PL/SQL functions, such as min/max, that may be imbedded in a formula.</p> <p data-bbox="724 770 1154 831">Sixteen seeded formulas are included (these can be updated):</p> <ul data-bbox="724 858 1154 982" style="list-style-type: none"> <li data-bbox="724 858 1154 888">• Eight Cost-to-Charge formulas</li> <li data-bbox="724 915 1154 982">• Eight Cost-to-Charge with Markup formulas</li> </ul>	<p data-bbox="1187 338 1463 399">Advanced Pricing adds the following features:</p> <ul data-bbox="1187 426 1463 1035" style="list-style-type: none"> <li data-bbox="1187 426 1463 581">• You can attach dynamic and static formulas to modifiers and price list lines.</li> <li data-bbox="1187 609 1463 1035">• Dynamic Formulas: Dynamic formulas enable the pricing engine to dynamically calculate the formula price based on variable values available at run time. Dynamic formulas can be attached to either price lists or modifiers.</li> </ul> <p data-bbox="1187 1062 1463 1283">Note: Price lists that have a dynamic formula with a Modifier Value as a component cannot be attached to a price list line.</p>

Pricing Features	Basic Pricing in Oracle Order Management	Oracle Advanced Pricing
Formula Components	<p>Component types include:</p> <ul style="list-style-type: none"> <li>• Numeric values</li> <li>• Factor list</li> <li>• One pricing context throughout formula with up to 100 attributes. Context must match context on the price list line to which the formula is attached.</li> </ul> <p>Adjustment factors (factor list): Basic pricing supports multiple factors. However, it has a limitation in basic pricing of one pricing attribute context throughout the formula with up to 100 attributes. Seeded context must be used.</p>	<p>Advanced Pricing includes all formula component types available in basic pricing, and adds the following:</p> <ul style="list-style-type: none"> <li>• List price of an item in a specific price list (product and service) pricing attribute.</li> <li>• Modifier Value is entered in the Modifier Value field on the modifier setup window for a modifier line as a component to a formula.</li> <li>• A FUNCTION type that calls Advanced Pricing API <code>Get_Custom_Price</code>.</li> <li>• Adjustment Factors: Advanced Pricing supports multiple adjustment factors. Both context - seeded and user-defined contexts and attributes can be used as adjustment factors.</li> <li>• Multiple pricing contexts for pricing attributes.</li> </ul>

Pricing Features	Basic Pricing in Oracle Order Management	Oracle Advanced Pricing
Price List Maintenance	Not available in basic pricing.	<p>Advanced Pricing provides search and maintenance capability for a single price list or across multiple price lists.</p> <p>Use the Bulk Change feature for mass updates or update individual price list lines.</p> <p>Updates can be made to price list lines for:</p> <ul style="list-style-type: none"> <li>• Value</li> <li>• Static and Dynamic Formula</li> <li>• Price List Line effective dates</li> </ul> <p>Price List Maintenance is available through the HTML user interface. See the <i>Oracle Advanced Pricing User's Guide</i>, Using the Price List Maintenance feature for more information.</p>
Bulk Import of Price List	<p>Basic pricing:</p> <ul style="list-style-type: none"> <li>• Imports large volume of price lists via the interface tables.</li> <li>• Provides increased performance compared to using the Business Object API.</li> </ul>	Same as basic pricing.
Copy price list	Basic pricing provides this capability.	Same as basic pricing.

<b>Pricing Features</b>	<b>Basic Pricing in Oracle Order Management</b>	<b>Oracle Advanced Pricing</b>
Adjust price list	Basic pricing provides this capability.	Same as basic pricing.
Add items to price list	Supported. (User must have Maintain access privilege to add items.)	Same as basic pricing. (User must have Maintain access privilege to add items.)
GSA pricing	Supported	Same as basic pricing.
Modifier list types	Basic pricing supports the following Modifier List Types: <ul style="list-style-type: none"> <li>• Discount</li> <li>• Surcharge</li> <li>• Freight and Special Charges</li> </ul>	Advanced Pricing provides all Modifier List Types provided in basic pricing, and adds the following: <ul style="list-style-type: none"> <li>• Promotion</li> <li>• Deal</li> </ul>
Modifier application methods	Basic pricing supports the following modifier application methods: <ul style="list-style-type: none"> <li>• Manual</li> <li>• Automatic</li> <li>• Overrideable</li> </ul>	Same as basic pricing and adds Ask For Promotions and Deals.
Active box on Modifier header	Not enabled in basic pricing.	Enabled in Advanced Pricing.
Modifier levels and level code	Order, order line level.	Basic level codes and adds Line Group.

Pricing Features	Basic Pricing in Oracle Order Management	Oracle Advanced Pricing
Modifier header qualifiers and attributes	<p>Basic pricing provides seeded qualifier contexts including:</p> <ul style="list-style-type: none"> <li>• Customer</li> <li>• Price List</li> <li>• Unlimited price lists (new OM feature)</li> <li>• Seeded attributes within contexts. Customer context includes these attributes: Customer class (defined in RA customers); Site; and Customer Name.</li> </ul> <p>Note: Modifier List Type of Freight and Special Charges supports additional modifier header qualifiers and attributes.</p> <p>Security-enabled rules checking.</p>	<p>Includes basic pricing features, plus Advanced Pricing adds these capabilities:</p> <ul style="list-style-type: none"> <li>• User assignment and override of precedence is possible.</li> <li>• Qualifiers can be used with both seeded and user defined contexts possible.</li> <li>• Users can define attributes with user-defined contexts. Up to 100 attributes can be defined for each context, and users can define any number of contexts.</li> <li>• Seeded qualifier attributes can be redirected to other data sources using the attribute mapping feature (see attribute mapping topic).</li> <li>• Users can control qualifier precedence.</li> <li>• Entered Context: Seeded and user-defined is possible.</li> <li>• User control of attribute precedence possible.</li> </ul>

Pricing Features	Basic Pricing in Oracle Order Management	Oracle Advanced Pricing
Modifiers: line types	<p>Types available include:</p> <ul style="list-style-type: none"> <li>• Discount: Fixed amount, Percent, New Price</li> <li>• Surcharge: Fixed amount, Percent, New Price</li> <li>• Freight Charges: Fixed amount, Percent, Lumpsum</li> <li>• Price Break</li> </ul>	<ul style="list-style-type: none"> <li>• Attaching qualifier groups to modifier headers possible.</li> </ul> <p>Advanced Pricing includes the basic pricing types and adds:</p> <ul style="list-style-type: none"> <li>• Coupon Issue</li> <li>• Item Upgrade</li> <li>• Other Item Discount</li> <li>• Terms Substitution</li> <li>• Promotional Goods</li> </ul>

Pricing Features	Basic Pricing in Oracle Order Management	Oracle Advanced Pricing
Modifiers: line qualifiers and attributes	<p>Basic pricing provides the following fixed qualifiers including:</p> <ul style="list-style-type: none"> <li>• Agreement Name</li> <li>• Agreement Type</li> <li>• Order type</li> <li>• Customer PO</li> </ul> <p><b>Usable Product Attributes include:</b></p> <ul style="list-style-type: none"> <li>• Item</li> <li>• Item categories</li> <li>• All items</li> </ul> <p><b>Usable pricing attributes:</b></p> <p>Limited to one context when product attribute is ALL Items</p> <p><b>Note:</b> Modifier Line Type of Freight and Special Charges supports additional modifier line qualifiers and attributes.</p>	<p>Advanced Pricing includes the line level qualifiers provided in basic pricing and adds:</p> <ul style="list-style-type: none"> <li>• Define unlimited number of qualifiers.</li> <li>• Can attach qualifier groups as qualifiers.</li> <li>• Can use seeded contexts and define additional contexts.</li> <li>• User-defined contexts are active.</li> <li>• Multiple qualifiers with AND/OR conditions can be created.</li> </ul> <p>Product attributes in Advanced Pricing include those defined in basic pricing and adds user-defined product attributes.</p> <p>An unlimited number of pricing attributes can be created in Advanced Pricing. The user can change the precedence.</p>

Pricing Features	Basic Pricing in Oracle Order Management	Oracle Advanced Pricing
Modifiers: Buckets for Manual Modifiers	Basic pricing does not provide this feature.	Buckets for manual line/group of line level modifiers are supported for the following modifier types: <ul style="list-style-type: none"> <li>• Discount</li> <li>• Surcharge</li> <li>• Price Break</li> <li>• Freight and Special Charges</li> </ul>
Modifier: Optional Currency	Enables modifier to be used across all transaction currencies: <ul style="list-style-type: none"> <li>• Optional Currency selected from LOV.</li> <li>• Percent, Amount, New Price, Lumpsum.</li> <li>• No currency conversion applied.</li> <li>• Modifier setup numeric value will always be applied in currency units of the transaction currencies.</li> <li>• No restrictions to use; therefore, user discretion advised for appropriate use.</li> </ul>	Advanced Pricing provides the same capabilities available in basic pricing.

Pricing Features	Basic Pricing in Oracle Order Management	Oracle Advanced Pricing
Modifiers: Price Breaks	<p>Line level price breaks in basic include:</p> <ul style="list-style-type: none"> <li>• Percent</li> <li>• Amount</li> <li>• Fixed Price</li> <li>• Equal operator</li> <li>• Break Type Code - Limited to point type only</li> <li>• Volume Type (Item Amount, Item Quantity)</li> </ul>	<p>Line level price breaks in Advanced Pricing includes all price break functionality in basic pricing, and adds the following:</p> <ul style="list-style-type: none"> <li>• Equal, between arithmetic operator</li> <li>• Point and Range</li> <li>• Context - Seeded and user defined</li> <li>• Recurring</li> </ul> <p>Define automatic continuous price breaks based on Net Amount for line and group of line level.</p> <p>Define accumulated range price breaks based upon an accumulated value that is used as starting point of the break calculation. Regular unused pricing attributes with volume context can be assigned as the accumulation attribute for a modifier range break. Accumulation attribute can be sourced by the engine via two methods in attribute management</p> <ul style="list-style-type: none"> <li>• Attribute Mapping</li> </ul>

Pricing Features	Basic Pricing in Oracle Order Management	Oracle Advanced Pricing
Modifiers: Freight and Special Charges	<p data-bbox="630 426 938 457">Supported Header Qualifiers</p> <ul data-bbox="630 478 837 856" style="list-style-type: none"> <li data-bbox="630 478 837 510">• Freight Terms</li> <li data-bbox="630 548 837 579">• Order Amount</li> <li data-bbox="630 617 837 648">• Line Weight</li> <li data-bbox="630 686 837 718">• Order Weight</li> <li data-bbox="630 756 837 787">• Line Volume</li> <li data-bbox="630 825 837 856">• Order Volume</li> </ul> <p data-bbox="630 894 906 926">Supported Line Qualifiers</p> <ul data-bbox="630 947 932 1535" style="list-style-type: none"> <li data-bbox="630 947 837 978">• Order Volume</li> <li data-bbox="630 1016 805 1047">• Order Type</li> <li data-bbox="630 1085 846 1117">• Order Category</li> <li data-bbox="630 1155 786 1186">• Line Type</li> <li data-bbox="630 1224 829 1255">• Line Category</li> <li data-bbox="630 1293 932 1325">• Shipment Priority Code</li> <li data-bbox="630 1362 821 1394">• Shipped Flag</li> <li data-bbox="630 1432 837 1463">• Shippable Flag</li> <li data-bbox="630 1501 870 1535">• Freight Cost Type</li> </ul>	<ul data-bbox="1089 331 1325 363" style="list-style-type: none"> <li data-bbox="1089 331 1325 363">• Runtime Sourced</li> </ul> <p data-bbox="1089 426 1341 552">Advanced Pricing provides the same capabilities available in basic pricing.</p>

Pricing Features	Basic Pricing in Oracle Order Management	Oracle Advanced Pricing
Effectivity date controls	Effectivity Date: order date only	Order date, plus Advanced Pricing adds: <ul style="list-style-type: none"> <li>• Ship Date</li> <li>• Order and Ship Date</li> </ul> Advanced Pricing also adds fields for Promotion Version, parent promotion, and parent version.

Pricing Features	Basic Pricing in Oracle Order Management	Oracle Advanced Pricing
Promotional limits	Basic pricing does not provide this feature.	<p data-bbox="1089 338 1365 457">Advanced Pricing includes this feature. You can define promotional limits by:</p> <ul data-bbox="1089 489 1365 825" style="list-style-type: none"> <li data-bbox="1089 489 1300 516">• Gross Revenue</li> <li data-bbox="1089 558 1203 585">• Usage</li> <li data-bbox="1089 627 1365 680">• Cumulative Discount Amount</li> <li data-bbox="1089 722 1284 749">• Item Quantity</li> <li data-bbox="1089 791 1284 819">• Accrual Units</li> </ul> <p data-bbox="1089 850 1292 877"><b>Limits can be set:</b></p> <ul data-bbox="1089 898 1300 1056" style="list-style-type: none"> <li data-bbox="1089 898 1300 951">• Within current transaction</li> <li data-bbox="1089 993 1268 1056">• Across all transactions</li> </ul> <p data-bbox="1089 1087 1321 1115"><b>Can apply limits for:</b></p> <ul data-bbox="1089 1136 1349 1230" style="list-style-type: none"> <li data-bbox="1089 1136 1349 1163">• Customer attributes</li> <li data-bbox="1089 1205 1333 1232">• Product Hierarchy</li> </ul> <p data-bbox="1089 1264 1268 1291"><b>Types of limits:</b></p> <ul data-bbox="1089 1312 1365 1533" style="list-style-type: none"> <li data-bbox="1089 1312 1349 1396">• Soft Limit (If limit is exceeded, give the full benefit)</li> <li data-bbox="1089 1438 1365 1533">• Hard Limit (If limit is exceeded, adjust or deny the benefit)</li> </ul> <p data-bbox="1089 1564 1365 1591"><b>Hard Limit enforcement:</b></p> <ul data-bbox="1089 1612 1235 1707" style="list-style-type: none"> <li data-bbox="1089 1612 1187 1640">• Hold</li> <li data-bbox="1089 1682 1235 1709">• No Hold</li> </ul> <p data-bbox="1133 1730 1317 1787">Security enabled rules checking</p>

Pricing Features	Basic Pricing in Oracle Order Management	Oracle Advanced Pricing
Duplicate Modifier Lines	Basic pricing supports duplicate modifier lines if the profile option QP: Allow Duplicate Modifiers is set to Yes.	Advanced Pricing allows duplicate modifier lines within a modifier list.
Copy Modifier	Basic pricing does not provide this feature.	You can create a new modifier by copying an existing one.
Pricing Organizer: Modifiers	Basic pricing does not provide this feature.	<p>Advanced Pricing includes this feature, which allows users to query on the setup of modifiers:</p> <ul style="list-style-type: none"> <li>• Modifier List</li> <li>• Modifier Lines</li> <li>• Products (including Excluded Products)</li> <li>• Pricing Attributes</li> <li>• Qualifiers</li> </ul> <p>Query criteria can be saved in Personal or Public folders. Can open Modifiers from the Organizer Summary. Security-enabled rules checking.</p>

Pricing Features	Basic Pricing in Oracle Order Management	Oracle Advanced Pricing
Pricing Engine Request Viewer	<p data-bbox="630 338 1036 394">Basic pricing features include viewing pricing requests information such as:</p> <ul style="list-style-type: none"> <li data-bbox="630 422 935 449">• Pricing Engine Requests</li> <li data-bbox="630 491 1068 800">• Pricing Engine Request Lines <ul style="list-style-type: none"> <li data-bbox="683 548 797 575">- List Price</li> <li data-bbox="683 600 829 627">- Selling Price</li> <li data-bbox="683 653 1008 680">- Service and Serviceable items</li> <li data-bbox="683 705 1068 800">- Price List lines and Modifier lines evaluated and deleted by the pricing engine</li> </ul> </li> <li data-bbox="630 842 1068 1157">• Pricing Engine Request Line Details <ul style="list-style-type: none"> <li data-bbox="683 894 1068 989">- Price List lines and Modifier lines evaluated and deleted by the pricing engine</li> <li data-bbox="683 1014 1019 1073">- Attributes sent to the pricing engine by the calling application</li> <li data-bbox="683 1098 1040 1157">- Attributes used in pricing by the pricing engine</li> </ul> </li> <li data-bbox="630 1199 797 1226">• Debug Log <p data-bbox="691 1257 1036 1318"><b>Note:</b> Accessible from the Tools menu of the Sales Order Pad.</p> </li> </ul>	<p data-bbox="1089 338 1341 527">Same as basic pricing, plus can view the relationship between lines for modifier types promotional goods and other item discounts.</p> <p data-bbox="1089 552 1349 642">Also accessible from the Pricing Manager Responsibility</p>
	Security enabled rules checking	

Pricing Features	Basic Pricing in Oracle Order Management	Oracle Advanced Pricing
Agreements	<p data-bbox="724 338 1146 394">Basic pricing agreement features enable you to:</p> <ul data-bbox="724 422 1146 1062" style="list-style-type: none"> <li data-bbox="724 422 1146 485">• Set payment terms: Invoicing rule, Accounting rule</li> <li data-bbox="724 527 1114 554">• Set freight terms: Freight carrier</li> <li data-bbox="724 596 1114 659">• Create Standard and Agreement Price List</li> <li data-bbox="724 701 1065 764">• Define using customer part numbers.</li> <li data-bbox="724 806 1000 833">• Revise original terms.</li> <li data-bbox="724 875 1146 938">• Enter Revision numbers, date, and reasons at the line level only.</li> <li data-bbox="724 980 967 1008">• Set Effective dates.</li> <li data-bbox="724 1050 1016 1077">• Create Volume breaks.</li> </ul>	Same as basic pricing.

Pricing Features	Basic Pricing in Oracle Order Management	Oracle Advanced Pricing
HTML Pricing Setups - Home Page	Basic pricing does not provide this feature.	<p data-bbox="1089 338 1370 590">With the Oracle Pricing User responsibility, Advanced Pricing users can perform the following tasks in Advanced Pricing HTML Pricing Setup from the Home Page:</p> <ul data-bbox="1089 617 1354 1003" style="list-style-type: none"> <li data-bbox="1089 617 1354 674">• Search for price lists and modifier lists</li> <li data-bbox="1089 716 1333 873">• Shortcut links to create price lists, modifier lists, and price list maintenance</li> <li data-bbox="1089 915 1338 1003">• View Recently Created Price Lists and Modifier Lists</li> </ul>

Pricing Features	Basic Pricing in Oracle Order Management	Oracle Advanced Pricing
HTML Pricing Setups - Price List	Basic pricing does not provide this feature.	<p data-bbox="1187 338 1463 558">With the Oracle Pricing User responsibility, Advanced Pricing users can perform the following tasks in Advanced Pricing HTML Pricing Setup:</p> <ul data-bbox="1187 583 1463 1283" style="list-style-type: none"> <li data-bbox="1187 583 1430 611">• Create a Price List</li> <li data-bbox="1187 653 1430 747">• Create a Price List line with Pricing Attributes</li> <li data-bbox="1187 789 1446 884">• Create a Price Break line with Pricing Attributes</li> <li data-bbox="1187 926 1446 978">• Update Price List and Price List Lines</li> <li data-bbox="1187 1020 1463 1052">• Delete a price list line</li> <li data-bbox="1187 1094 1446 1146">• Access Price List Maintenance feature</li> <li data-bbox="1187 1188 1463 1283">• BSA related fields are not available in the HTML user interface</li> </ul>

Pricing Features	Basic Pricing in Oracle Order Management	Oracle Advanced Pricing
HTML Pricing Setups - Modifiers	Basic pricing does not provide this feature.	<p data-bbox="1089 338 1370 558">With the Oracle Pricing User responsibility, Advanced Pricing users can perform the following tasks in Advanced Pricing HTML Pricing Setup for Modifiers:</p> <ul data-bbox="1089 583 1370 1205" style="list-style-type: none"> <li data-bbox="1089 583 1370 709">• Create a modifier list (Discount, Surcharge, Deal, or Promotion List)</li> <li data-bbox="1089 747 1370 1003">• Create a modifier line for discount, surcharge, price break, or promotional goods (additional buy products not supported)</li> <li data-bbox="1089 1041 1370 1136">• Update modifier list and modifier lines for types supported</li> <li data-bbox="1089 1173 1370 1205">• Excluded Products</li> </ul>

Pricing Features	Basic Pricing in Oracle Order Management	Oracle Advanced Pricing
Price Book	<ul style="list-style-type: none"> <li>• Create full price books</li> <li>• Supported publishing options:               <ul style="list-style-type: none"> <li>- Send to Printer</li> <li>- E-mail</li> <li>- View Document (PDF, Excel, RTF)</li> </ul> </li> <li>• Delete Price Book</li> <li>• Attributes for Price Calculation Criteria limited to basic pricing attributes</li> <li>• Accessible from Oracle Pricing User responsibility</li> </ul>	<ul style="list-style-type: none"> <li>• Adds capability to create and publish price books:               <ul style="list-style-type: none"> <li>- Create Delta Price Book</li> <li>- Get Catalog XML message</li> </ul> </li> <li>• Additional publishing option:               <ul style="list-style-type: none"> <li>- Send XML Message</li> </ul> </li> <li>• More extensible attributes used for Price Calculation Criteria</li> <li>• Accessible from Oracle Pricing User responsibility</li> </ul>
Attribute Mapping (Extensibility Feature)	Basic pricing does not provide this capability.	Use attribute mapping to extend pricing to a variety of non-standard sources to drive your pricing. These data sources can be within or from outside Oracle Applications.
Get_Custom_Price API (Extensibility Feature)	Basic pricing does not provide this capability.	Advanced Pricing adds this feature. The Get_Custom_Price API allows you to execute your own code as a part of the Advanced Pricing Engine's execution cycle.
Qualifiers and groups	Not available in Basic Pricing. Defaulting is available for price lists and modifiers.	Included in addition to defaulting for basic pricing.

Pricing Features	Basic Pricing in Oracle Order Management	Oracle Advanced Pricing
Modifiers other differences	Defaulted to single available bucket (bucket 1). User control of incompatibility feature inactive in basic pricing. User control of phase/event mapping inactive in Basic.	Oracle Advanced Pricing adds: multiple buckets, seeded or user defined buckets, user control of phase event mapping, user control of incompatibility/exclusivity modifier control feature, user control of incompatibility resolve method by setting choice of best price or precedence, accrual features, formula in a modifier feature, and active exclude item (product attribute).

Pricing Features	Basic Pricing in Oracle Order Management	Oracle Advanced Pricing
Oracle Order Management integration with Pricing	<p>Order Management integration with basic pricing supports the View Adjustments feature and the following features:</p> <ul style="list-style-type: none"> <li>• Manual price override</li> <li>• Manual discount override</li> <li>• Reason Code</li> <li>• Modifier Dates</li> <li>• Display Qualifier Attributes and display Pricing Attributes buttons on UI</li> <li>• One pricing attribute context</li> <li>• Up to 100 attributes can be used in the single context</li> <li>• Action &gt;Price Order is available.</li> <li>• Action &gt;Price Line is available.</li> <li>• Calculate Price Freeze Flag can be set.</li> <li>• Sales Agreement Order Support: Create a simple price list from the Sales Agreement window.</li> </ul> <p>Security enabled rules checking. Global Flag and security rules enforced. If Global box selected on price list, modifier, or agreement price list, the pricing window can be used regardless of which operating unit created it.</p>	<p>Order Management, when integrated with Advanced Pricing, provides all features supported with basic pricing and adds:</p> <ul style="list-style-type: none"> <li>• View Adjustment: Relationship button, Item Upgrade, Term Substitution</li> <li>• Coupon entry</li> <li>• Ask for promotions</li> <li>• User entered attributes: Multiple attribute contexts can be used</li> <li>• Up to 100 attributes per context</li> <li>• Promotional Limits Hold: <ul style="list-style-type: none"> <li>- Place holds where violated.</li> <li>- No holds are placed.</li> <li>- Place order on hold when any violation occurs.</li> </ul> </li> </ul>

Pricing Features	Basic Pricing in Oracle Order Management	Oracle Advanced Pricing
Oracle Order Management integration with Pricing: Sales Agreement	<p>Create simple in-line price list and modifier list from within the Sales Agreement window. Other tab displays additional information:</p> <ul style="list-style-type: none"> <li>• List Source Document Number (Sales Agreement number) and List Source Code automatically populated on Price List and Modifier List.</li> <li>• Automatic creation of sales agreement as qualifier for Price List and Modifier header and lines</li> <li>• Price List provides fields for: Customer Item, Address, Address Category</li> <li>• Customer Name and Number populated</li> </ul>	<p>Create price break range type modifiers based on accumulated sales agreement fulfillments (across releases). The price break here will be continuous.</p> <p>Volume accumulation attributes have been seeded for Sales Agreement use only. The following accumulation attributes can be selected when setting up the modifier:</p> <ul style="list-style-type: none"> <li>• Sales Agreement Amount</li> <li>• Sales Agreement Line Quantity</li> <li>• Sales Agreement Line Amount</li> </ul>
Pricing Engine	<p>In basic pricing, the pricing engine does not return Oracle Advanced Pricing modifiers or features to the calling application.</p> <p>Security rules checking is enabled. Global box and security rules are used.</p>	<p>In Oracle Advanced Pricing, the pricing engine is enabled to return all advanced features.</p>

Pricing Features	Basic Pricing in Oracle Order Management	Oracle Advanced Pricing
Reports	<p>Reports for basic pricing include:</p> <ul style="list-style-type: none"> <li>• Order Discount Detail Report</li> <li>• Order Discount Summary Report</li> <li>• Diagnostics List Line Details</li> <li>• Diagnostics Performance Analysis</li> </ul> <p>Security-enabled rules checking across all reports.</p>	<p>Advanced Pricing adds the following reports:</p> <ul style="list-style-type: none"> <li>• Accruals Details Report</li> <li>• Attribute Mapping Rules Error Report</li> <li>• Cross Order Volume Report</li> <li>• Modifier Detail Report</li> <li>• Price List Detail Report (including Multi-currency fields when Multi-currency is installed)</li> <li>• Pricing Formulas Report</li> <li>• Qualifier Grouping Report</li> </ul>

Pricing Features	Basic Pricing in Oracle Order Management	Oracle Advanced Pricing
Concurrent Programs	Not available in basic pricing.	<p data-bbox="1089 338 1357 428">Advanced Pricing adds the following concurrent programs:</p> <ul data-bbox="1089 453 1365 1289" style="list-style-type: none"> <li data-bbox="1089 453 1300 516">• Build Attribute Mapping Rules</li> <li data-bbox="1089 558 1292 621">• Build Formula Package</li> <li data-bbox="1089 663 1357 726">• Cross Order Volume Load</li> <li data-bbox="1089 768 1365 831">• Purge Pricing Engine Requests</li> <li data-bbox="1089 873 1341 936">• QP: Bulk Import of Price List</li> <li data-bbox="1089 978 1365 1062">• QP: Maintains the denormalized data in QP Qualifiers</li> <li data-bbox="1089 1104 1357 1188">• Update Price Lists with Multi-Currency Conversion Criteria</li> <li data-bbox="1089 1230 1357 1293">• Update Promotional Limit Balances</li> </ul>

Pricing Features	Basic Pricing in Oracle Order Management	Oracle Advanced Pricing
Support TCA Party Hierarchy	This functionality is only available with Advanced Pricing.	<p>This functionality is only available with Advanced Pricing.</p> <ul style="list-style-type: none"> <li>• Applies To Party Hierarchy check box is not visible on the Price List and Modifier Qualifiers in basic pricing</li> <li>• Party Hierarchy Enabled check box is not visible on Attributes UI in basic pricing.</li> </ul>
Public APIs	Not available in Basic Pricing.	Advanced Pricing provides several public APIs. For more information, see the Oracle Integration Repository (iRepository).

## Terminology

### Calling Application

An application that calls the Oracle Advanced Pricing engine to obtain pricing calls. For example, both Oracle Order Management and iStore call the Oracle Advanced Pricing engine to apply pricing to customer transactions.

### Customer Attributes

Oracle Advanced Pricing provides seeded qualifier attributes with context of Customer (also called Customer Attributes) You can use the preceding attributes by referencing them as qualifiers for either modifier or price list objects. For additional information, see Key Implementation Decision: Are the seeded context values on Oracle customer tables sufficient to contain customer attributes?, page 3-10.

### Pricing Engine

The pricing engine is a program module of Oracle Advanced Pricing. A calling application (such as Oracle Order Management) calls the pricing engine when transactions that need a price are processed.

### **Pricing Request**

A pricing request is the specific information provided to the Pricing engine when the engine is called by the calling application. In general, this includes who the customer is, what the product is, what attributes may be associated with the customer or product that may be used by the pricing engine, the pricing date, and other pricing data attributes that may be required by the pricing engine.

### **Product Hierarchy**

The Oracle Advanced Pricing product hierarchy is pre-seeded with Item context (based on the Oracle Applications Item Master, MTL\_SYSTEM\_ITEMS table). This context delivers a two level product hierarchy consisting of:

- Item number
- Item category

Oracle Advanced Pricing provides two additional capabilities that extend this hierarchy:

- You can define a product hierarchy level more specific than Item by using pricing attributes on a price list. This provides a product hierarchy level below item.
- Oracle Advanced Pricing also recognizes a super category of items called item ALL. Item ALL consists of all the items in a price list.

You can use item, item and pricing attribute, or item categories as defaults to control the operation of price lists and modifiers.

Additional levels of product hierarchy capabilities can be defined in Oracle Advanced Pricing. For example, you may want to define flexfields on the customer tables to house additional customer groupings.

The above hierarchy is built using the Oracle Inventory Item Master. When Oracle Advanced Pricing is installed without Oracle Order Management or the standard item master tables being present, the seeded item attributes listed above will not be available. In that case, you must define an alternative table structure location where the product hierarchy exists (contexts) and attributes it contains. It is not necessary for the table structure supporting your alternative product hierarchy structure to exist within Oracle Applications.

Mapping an alternative product hierarchy can be accomplished using attribute mapping. For more information on attribute mapping, see Overview of Attribute Management, page 17-1.

## **Oracle Advanced Pricing Features**

The following list summarizes the Oracle Advanced Pricing features that are supported by Oracle Applications:

## Qualifiers

Qualifiers determine who is eligible for a price or benefit. Qualifiers and qualifier groups can be linked to price lists and modifiers to define rules for who can receive a particular price, discount, promotion, or benefit. They can assign discounts and promotions to:

- Specific customers
- Customer groups
- Order types
- An order amount

Some example qualifiers are:

- Customer class = VIP
- Order type = Special

You can construct complex qualifier sets consisting of multiple qualifiers that are evaluated together in Boolean AND and OR relationships.

## Qualifier Groups

Qualifier groups enable you to define multiple qualifiers relationships in preparation for association with either price lists or modifiers. You can save these qualifier groups and copy them to one or more price lists and modifiers.

## Price Lists

Price lists relate a list price to a product. (The *list* price is the starting price before any related discounts and adjustments are applied.) Price lists can contain one or more price list lines, price breaks, pricing attributes, qualifiers, and secondary price lists. The price list information includes the price list name, effective dates, currency, pricing controls, rounding factor, and shipping defaults such as freight terms and freight carrier. See the *Oracle Advanced Pricing User's Guide*, Price Lists chapter for information about setting up and using price lists.

You can use the pricing engine to select price lists based on a qualifier rule. However, as an alternative to using price list qualifiers, you may provide a default price list on an order based on any one of the following:

- The sold-to customer
- The ship-to customer
- The bill-to customer
- Order type
- Agreement

Defaulting provides equivalent functionality to the price list defaulting found in Oracle Order Entry/Shipping Release 10.7 and 11.0. You can provide by default some of the same elements you can reference in qualifier rules. Defaulting is available only for price lists. You can use only one default with a price list, and you cannot combine defaults using AND/OR relationships.

**Additional Information:** You can define multiple price lists.

Alternatively, you may enter a specific price list on the order header or at the order line level. For each price list, you can also designate secondary price lists that the engine searches when it cannot find an item on the primary list. Multiple secondary price lists may be searched for each primary list. You may define several price lists as secondary to a primary price list. You can not define multiple levels of secondary price lists.

Price lists may be specified in different currencies. During order entry, if you enter a currency on the order, the pricing engine selects price lists with currency that matches the currency you entered on the order.

### **Multiple Currency Price Lists**

Multiple currency price lists enables businesses that have pricing strategies based on a single price for an item in a base currency to use exchange rates or formulas to convert that price into the ordering currency. At engine run time, the pricing engine will take the currency from the order and search for a price list or price lists with base or conversion currencies matching this currency. The pricing engine converts the price from the base currency and calculates the ordering currency based upon the established conversion rules.

### **Price List Maintenance**

Oracle Advanced Pricing provides a Price List Maintenance feature that enables you to do searches for price lists and price list information, make bulk changes or update individual price list lines across multiple price lists.

The Price List Maintenance is available through the HTML user interface. See the *Oracle Advanced Pricing User's Guide*, Using the Price List Maintenance feature for more information.

### **Pricing Attributes**

Pricing attributes are data elements used in addition to item identifiers. These attributes control what is being priced on a price list or a modifier. Oracle Advanced Pricing is delivered with seeded pricing attributes. The seeded attributes for Oracle Advanced Pricing are the same as those for basic pricing. For more information, see the *Attribute Seed Data appendix* in this guide.

In some situations, an item identifier does not identify the product to a level where a price can be assigned. Pricing attributes enable you to define separate conditions for a product such that the product must be priced under different conditions. A pricing attribute can be any condition you use to further qualify an item.

The following examples show how pricing attributes are used to specify conditions of a

product or service that affects the price. While these examples are appropriate to price lists, the same concept is true for modifiers. Pricing attributes can also be used with formulas.

- Product ID = HMO Plan 15; Pricing Attribute = State = Connecticut; List Price = \$200.00 per month.
- Product = HMO Plan 15; Pricing Attribute = State = New York; List Price = \$250.00 per month.
- Product ID = Motor Oil; Pricing Attribute = Grade = Regular; List Price = \$3.50 per Quart
- Product ID = Motor Oil; Pricing Attribute = Grade = Premium; List Price = \$4.25 per quart
- Product ID = Usage Charge; Pricing Attribute = Time of Day = Evening (7:00pm to 11:00pm); list price =.08 per minute
- Product ID = Usage Charge; Pricing Attribute = Time of Day = Late night (6:00am to 5:30am) list price =.05 per minute

Pricing attributes can be used in combination with each other and are passed to the pricing engine at run-time. The following image shows an example of how to define two attributes, Time of Day and Market Area. Both of these attributes must be obtained by the calling application and then passed to the pricing engine at run time. The pricing engine uses the attributes to match to the combination of product ID and conditions defined by the pricing attributes.



### Attribute Mapping

Attribute mapping is the process used to pass in data (from other applications or systems) that is not seeded in the delivered product into Oracle Advanced Pricing for use in price lists, modifiers, agreements, qualifiers, and formulas.

### Maintaining Price Lists

You can maintain price lists using any one of the following functions:

- Copy price list
- Adjust price list
- Add items to price list

These features are available from the Oracle Pricing Manager responsibility which then submits concurrent manager jobs for each step.

The Price List Maintenance feature (available from the HTML user interface) enables you to make changes to price lists and price list lines for a single price list or across many price lists. For more information, see *Oracle Advanced Pricing User's Guide*, Using the Price List Maintenance feature.

### **Agreements**

Agreements enable you to define prices, payment terms, and freight terms that you negotiated with specific customers. You can:

- Define your agreements using customer part numbers and inventory item numbers.
- Revise the original terms and maintain these changes and their reasons under separate revision numbers.
- Attach an already existing price list to the agreement or define new prices.
- Assign optional price breaks by quantity.
- Set effectivity dates for agreement terms.
- Set payment terms including invoice rule and accounting rule.
- Set freight terms including the freight carrier.
- Apply agreement terms to sales orders by reference agreements.

### **GSA Pricing**

GSA Pricing enables you to define a GSA price list for your GSA customers. The GSA Price List actually uses the modifiers window and uses the new price. You create a discount that adjusts the base price of the item to the GSA price.

### **Formulas**

Formulas, which can be used in both price lists and modifiers, enable a list price to be adjusted according to an algebraic relationship with other variables. Oracle Advanced Pricing enables pricing attributes to be passed as variables to the formula processing at run time. Formulas enable you to define a mathematical expression that the pricing engine uses to determine the list prices of items. A full complement of mathematical operators and numeric operands can be used. An example of a formula is:

- List price of service call = (price from price list \* distance) + adjustment factor

Distance is a numeric value passed to the pricing engine as a pricing attribute at run

time. Adjustment factor is the result of a value in a factor table, based on class of service, which is a pricing attribute variable passed to the engine at run time.

For price lists: When processing formulas, the pricing engine begins by locating a price list line that is linked to a formula. It then applies the mathematical expression to generate a final list price. In Oracle Advanced Pricing, formulas may be static; that is, the variables in the formula must be pre-populated with data by running a concurrent manager job before the formula can be used. Oracle Advanced Pricing provides a dynamic mode of formula operation. In dynamic formulas, the required data to be substituted into formula variables is collected by the pricing engine at run time.

### **Modifiers**

Modifiers are pricing actions that, when applied to a transaction, adjust the selling price up or down. The specific action that a modifier takes is defined by its type. In Oracle Advanced Pricing, a modifier has two levels of functionality that define its action: a list type and a line type. A modifier list type enables you to define behavior characteristics that are common to all lines, which enables you to define a modifier with several different lines, each line representing a specific pricing action.

You can create the following modifier list types in Oracle Advanced Pricing:

- Deal
- Discount
- Freight/Special Charges
- Promotion
- Surcharge

For each list type that you define, you can associate certain line types. The available line types are:

- Coupon issue: Issues a coupon on one order for the customer to redeem for a price adjustment or benefit on a later order.
- Discount: Creates a negative price adjustment.
- Freight charge: Creates a freight charge.
- Item upgrade: Replaces a specific item ordered with another item for the same price.
- Other item discount: Gives a price adjustment or benefit to a specified item on an order when the customer orders one or more other items on the same order.
- Price Break: Applies a variable discount or surcharge price adjustment to a pricing request based meeting the condition of a break type. You can use both point and range type breaks.

- Promotional goods: Adds a new item or service subscription item with a price adjustment or benefit when the customer orders one or more other items on the same order.
- Surcharge: Creates a positive price adjustment.
- Terms Substitution: Replaces freight charges, shipping charges, and payment terms with more favorable charges.

Not all line types can be used with all list types. See the *Oracle Advanced Pricing User's Guide* for a listing of valid modifier list and line type combinations.

You can define a modifier that the pricing engine automatically applies, or you can manually enter a modifier. With proper setup, modifiers can be defined as manual or overrideable.

Modifiers can be used to compute price breaks. You can define breaks at the line level to be computed as percent, amount, or fixed price. Both point and range breaks are supported.

### **Freight and Special Charges**

The freight and special charges capability of Oracle Order Management enables you to capture, store, update, and view costs associated with a shipment, order, container, or delivery. You can either itemize or summarize such charges on your orders. This capability includes functionality to pass customer charge information to Oracle Receivables for invoicing.

Freight and special charges enables you to:

- Apply charges as part of the order.
- Limit the application of charges to orders that meet certain criteria.
- Apply charges until the point of ship confirmation/invoicing.
- Review charges at anytime.
- Create many charge types (duty, handling, freight).
- Support charges at the order or line level.

When using freight and special charges, you set up freight and special charges as pricing modifiers. The pricing engine applies the qualified freight and special charges to order lines. You can view the application of freight and special charges to orders. Oracle Order Management captures costs at shipping and converts them to charges. Freight and special charges appear on invoices.

With Oracle Advanced Pricing, the full functionality of qualifier rules can be used to determine which orders should have freight and logistics charges applied to them, and to exclude certain orders from having charges. Additionally, formulas can be used to mark freight charges up or down before they are placed on the order.

### **Pricing Security**

Oracle Advanced Pricing provides an additional level of security called "pricing security" to enhance the existing functional security. Pricing security enables you to restrict pricing activities such as updating and viewing pricing entities to users granted specific access privileges.

### **Get\_Custom\_Price API**

Oracle Advanced Pricing provides an API that enables the pricing engine to execute user-supplied code to obtain a price, which as an alternative to storing the price in a price list or in a formula. This enables you to obtain a starting or beginning list price from sources that are outside Oracle Advanced Pricing's tables and yet accessible to code that you write.

The Get\_Custom\_Price is called by the pricing engine while evaluating a formula that contains a formula line (step) of type Function.

The technical information necessary to use this API is documented in *Oracle Order Management Open Interfaces, API, & Electronic Messaging Guide*.

## **Process Flow for Implementation**

The process flow for implementing Oracle Advanced Pricing assumes that Oracle Applications, including Oracle Order Management, have been successfully installed, that Oracle Advanced Pricing has been installed as Shared, and that all necessary patches have been applied.

### **Implementing from a Fresh Install**

The following table lists the recommended steps to implement a fresh install of Oracle Advanced Pricing (no previous implementation of Oracle Order Entry/Shipping exists).

#	Process Step	Step Description
1	Analyze and understand business pricing scenarios.	A thorough understanding of pricing business requirements should be established before beginning an implementation of Oracle Advanced Pricing. For more information, see Chapter 3, "Implementation Methodology".
2	Determine data sources and columns needed for product and customer hierarchy definition.	Oracle Advanced Pricing ships with seeded values. You may need to define additional levels or alternative data sources. For more information, see Chapter 3, "Implementation Methodology".

#	Process Step	Step Description
3	Develop logical pricing model solutions.	Plan how you will use Oracle Advanced Pricing for each pricing scenario. For more information, see Chapter 3, "Implementation Methodology".
4	Set up and test prototype pricing solutions.	Before implementing a production system, you should set up Oracle Advanced Pricing prototype solutions for all the pricing scenarios you have identified, and enter test orders against them to determine that they are handled correctly. (The Vision Sample database shipped with the software can be used to do this).
5	Make defaulting decisions.	These decisions must be made before product setup. For more information, see Chapter 3, "Implementation Methodology".
6	Perform Oracle Advanced Pricing product setup tasks.	This step involves taking results of steps 1 through 4 and creating entries in the pricing system tables that enables Oracle Advanced Pricing to act on your plans.
7	Create backup of system as setup.	This step involves creating a backup of the system as set up.
8	Conduct pre-production system functionality and load tests.	Before initial production, you should conduct a system test. This test should exercise all product setups by including examples of all pricing scenarios defined during pre-implementation planning. Verify that testing includes a sufficient volume to stress the system to levels equal to those that are experienced during production operations.

---

## Implementation Overview

This chapter covers the following topics:

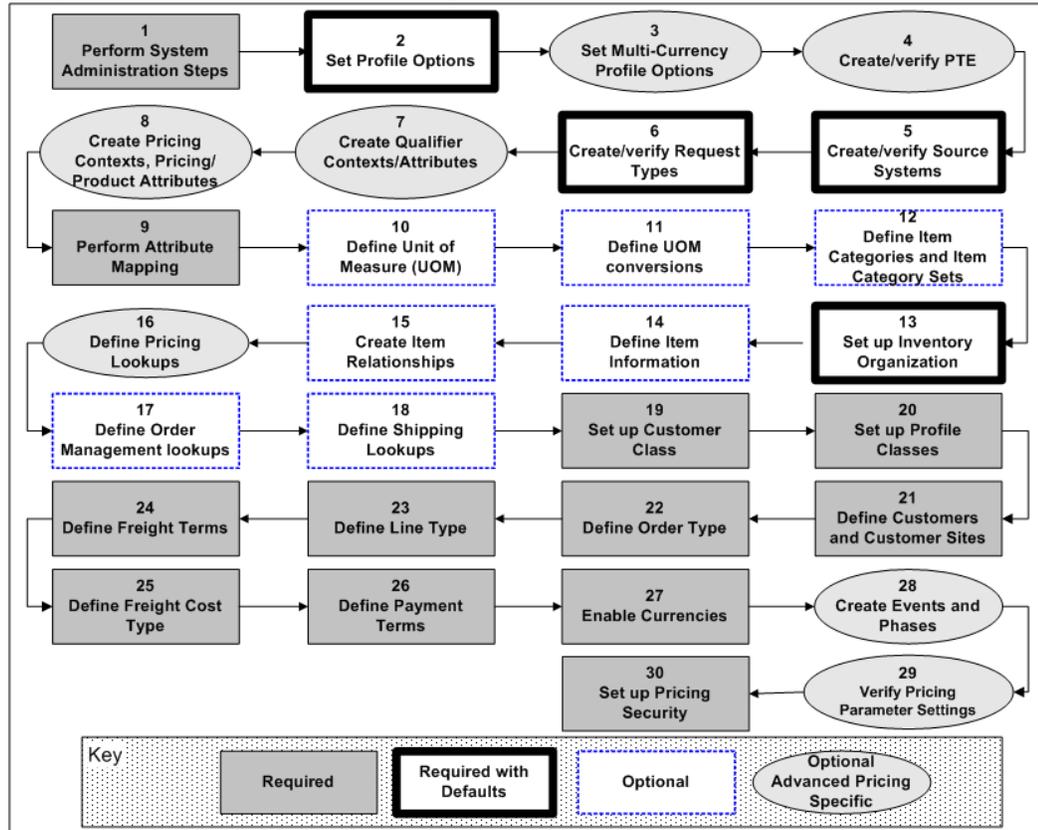
- Setup Flow to Implement Oracle Advanced Pricing

### Setup Flow to Implement Oracle Advanced Pricing

The following image depicts the setup flow for Oracle Advanced Pricing. Some of the steps outlined are required and some are optional. If you have already completed a common-application setup (setting up multiple Oracle Applications products) some of the following steps may be unnecessary.

A Required Step With Defaults is required only if you want to change the seeded default values. Review those defaults and decide whether to change them to better suit your business needs. Do optional steps only if you plan to use the related feature or complete certain business functions.

### Setup Flow for Oracle Advanced Pricing



### To Set Up Oracle Advanced Pricing:

1. Perform system administration steps.

Default: None

Assign users who set up Oracle Advanced Pricing to the Oracle Pricing Manager responsibility.

For more information on completing system administration steps, see *Oracle System Administrator User's Guide*, Responsibilities.

Do this step for each user who accesses the Oracle Pricing Manager responsibility.

2. Set profile options.

During implementation, you set a value for each user profile option to specify how Oracle Advanced Pricing controls access to and processes data.

The system administrator sets and updates profile values.

For more information on setting profile options, see *Oracle Applications System*

*Administrator's Guide*, setting User Profile Options, and *Oracle Advanced Pricing Implementation Guide*, Profile Options.

3. (Optional) Set multi-currency profile options.

Default: None

The profile option QP: Multi-Currency Installed enables the multi-currency price list feature. Using multi-currency price lists enables you to maintain a single price list for multiple currencies. Once you set the profile option to Y (Yes), the price list and agreement forms are converted to multi-currency. Changing the profile option QP: Multi-Currency Installed back to N (No) may cause undesired results if conversion criteria were used. Oracle does not support changing the profile option QP: Multi-Currency Installed back to N (No).

4. Create or verify a Pricing Transaction Entity (PTE).

Default: Order Fulfillment

A PTE is an ordering structure that has associated request types and source systems. Request types and source systems in the same PTE share pricing setup data. You can add additional source system and request types to existing PTEs.

In rare instances, it may be necessary to create a new PTE. You must create a new PTE only if the new request type uses a different ordering structure and a different set of source systems that is not already predefined.

5. Create or verify source systems.

Default: Predefined Oracle Advanced Pricing record

**Warning:** Changing the system source code can severely affect pricing engine behavior.

This step is required if:

- Your pricing data application source is anything other than Oracle Advanced Pricing.
- You are integrating Oracle Advanced Pricing with an application other than Oracle Order Management.

The pricing engine uses request types to determine the source of pricing data to be used when pricing a particular transaction. Request types identify the types of transactions being priced. Whenever the pricing engine prices a request, the request must be stamped with the request type so that the pricing engine can identify the type of transaction. The source system is recorded on all price and modifier lists and identifies which application created this pricing data.

Use the Pricing Transaction Entities Associations window to control which pricing

data is used to price transactions.

Complete this step for each source system that you want to map to a request type.

### **Functional Areas**

Each PTE must have at least one enabled functional area and related category set. Advanced Pricing provides seeded functional areas for each source system in a PTE. You can also add new functional areas. You can use the category set and its related hierarchy of categories to define price list lines or modifiers. Verify that the source systems have the correct associated functional areas for each PTE. You can define category pricing for those categories in the category set associated to those enabled functional areas. Flattened and hierarchical category sets (catalogs) are supported in Advanced Pricing.

#### **6. Create or verify request types.**

Request types are those order applications that request pricing data (for example, *Order Management Order* or *Oracle Contracts for Service*).

You can create or verify request types set up for pricing parameters. Unlike profile options, which can be set at the site, application, responsibility, and user levels, parameter values are set at PTE or Request Type level. Parameters defined at the PTE level may affect behavior for the pricing engine and the user interface level, while parameters defined at the Request Type level mainly affect pricing engine behavior.

#### **7. (Optional) Create qualifier contexts and qualifier attributes.**

Default: Predefined Oracle Advanced Pricing qualifier contexts

If you skip this step, users can choose only predefined Oracle qualifier contexts and qualifier attributes for price and modifier eligibility. If a qualifier attribute references an Oracle Trading Community Architecture (TCA) party, you can select the Party Hierarchy Enabled flag check box. This option enables you to create qualifiers so that associated price lists and modifiers are available to the party hierarchy.

Qualifiers provide a highly configurable and flexible method of defining the rules that your business uses to manage pricing. The pricing engine uses qualifiers to determine eligibility for price lists and modifiers.

#### **8. (Optional) Create pricing contexts, pricing attributes, and product attributes.**

Default: Predefined Oracle Advanced pricing and product attribute contexts.

Pricing and product attributes are a feature of Oracle Advanced Pricing. You can define necessary item attributes to price or apply a modifier and attributes used in formulas.

If you do not complete this step, users can select only predefined Oracle Advanced Pricing contexts and associated attribute values for benefit options.

9. (Optional) Perform attribute mapping.

Default: Predefined Oracle Advanced Pricing attribute mapping rules

Complete this step if you have defined any additional qualifiers or pricing attributes in steps 6 and 7, or if you want to change the defaulting mapping that has been seeded for a seeded qualifier or pricing/product attribute. Oracle provides predefined rules for attribute mapping, for both qualifiers and pricing contexts, that enable a list of values when selecting eligibility criteria.

Qualifier and pricing attribute mapping is required to supply a value for a qualifier or non-user entered pricing attribute before pricing a transaction. A mapping rule is set up to derive the value for the qualifier or pricing attribute from the transaction itself or from another attribute of the transaction. Attribute mapping builds additional information about a transaction that you can use to qualify for or derive a price, benefit, or charge for the transaction.

Attribute mapping includes rules that enable you to configure mapping to source qualifiers and pricing attributes according to your business needs.

10. (Optional) Define units of measure.

Default: None

Units of measure (UOM) are used in Oracle Advanced Pricing to determine the unit value for what the pricing engine is pricing, modifying, returning a benefit, or creating an accrual.

For more information on defining units of measure, see: *Oracle Advanced Pricing Implementation Guide*, Unit of Measure.

Complete this step if you have not installed and set up Oracle Inventory or completed this common-applications setup for another Oracle product.

11. (Optional) Define unit of measure conversions.

Default: None

You must define conversion rates between the base unit of measure and other units of measure within a UOM class if you want to price and discount an item in a UOM other than its primary UOM. Oracle Advanced Pricing uses these conversions to automatically convert transaction quantities to the primary pricing unit of measure defined on the price list when pricing cannot find a price in the transaction unit of measure. In addition, you must define all price adjustments, benefits, and charges in the same unit of measure as the unit of measure used on the price list.

For more information on defining unit of measure conversions, see *Oracle Inventory User's Guide*, Defining Unit of Measure Classes.

Complete this step if you have not installed and set up Oracle Inventory or completed this common-applications setup for another Oracle product.

12. (Optional) Define item categories and item category sets.

Default: Seeded structure name of item categories, and associated default seeded category code combinations. For more information on setting up categories, see *Oracle Product Lifecycle Management User's Guide*.

Item categories have been seeded as an item attribute in product attributes; you can use these item categories when defining price list lines and modifier lines. On the Price List and Modifier windows, the item categories are selected in the product attribute field. The value set for the item category attributes uses the item categories defined in Oracle Inventory. You can define hierarchical categories in a hierarchical catalog using Oracle Advanced Product Catalog. You can set up multiple categories and multiple category sets. For more information, see *Oracle Product Lifecycle Management User's Guide* and *Oracle Advanced Product Catalog User's Guide*.

You can create other item categories in the product attributes item context by creating a new attribute in the item context and attaching a value set. For more information on defining item categories and item category sets, see *Oracle Product Lifecycle Management User's Guide*. For information on flexfields, see *Oracle Application Flexfields User's Guide*, *Key Flexfields in Oracle Applications*, *Item Categories Flexfield*.

Complete this step if you have not installed and set up Oracle Inventory or completed a common-applications setup for another Oracle product. If you do not plan on using Oracle category functionality for associated price or benefits, you can skip this step.

**13. Set up inventory organization.**

Default: None

You must define at least one item validation organization in Oracle Inventory. This is the organization that items are validated and viewed against when entering items on the Price List and Modifier Setup windows.

For more information on setting up inventory organizations, see *Oracle Inventory User's Guide*, *Setting Up Oracle Inventory*.

Complete this step if you have not installed and set up Oracle Inventory or completed a common-applications setup.

**14. (Optional) Define item information.**

Default: None

Define the items that you want to price and discount and assign them to the validation organizations defined in step 10. If you want to define qualifier rules that include the seeded qualifiers Line Volume or Line Weight, you must set the volume or weight attributes of each item (as these attributes are used by attribute mapping to derive the transaction line, weight, or volume).

For more information on defining item information, see *Oracle Inventory User's Guide*, *Items*.

Complete this step if you have not installed and set up Oracle Inventory or completed this common-applications setup for another Oracle product.

15. (Optional) Create item relationships.

Default: None

This step is required if you want to give item upgrade benefits. You must define a Promotional Upgrade item relationship from the ordered item to the item that you want to give as an upgrade. Define your item relationships for the item validation organization.

Set up promotional upgrade items as follows:

- The ordered item and the promotional item must have the same base unit of measure and unit of measure conversions.

If those entities are not the same, the substitution can fail.

See *Oracle Inventory User's Guide*, Item Relationships.

- The modifier unit of measure and the pricing unit of measure on the order line must be the same.

If those entities are not the same, the substitution can fail.

See *Oracle Inventory User's Guide*, Item Relationships.

- Ensure that you do not select the Reciprocal check box.

**Note:** If you want the relationship to be reciprocal meaning that it can go both ways (item A upgrade to item B or item B upgrade to item A), then you must clear the Reciprocal check box, then define a relationship between A and B, and add another record to define the relationship between B and A.

16. (Optional) Define pricing lookups.

Default: Lookup type dependent

Lookup codes supply many of the lists of values in Oracle Advanced Pricing.

Lookup code values are the valid entries that appear in the list of values. They simplify information selection, and ensure that users enter only valid data into Oracle Advanced Pricing. You can add new lookup values at any time. You can set the Enable flag to No so that the lookup no longer appears in the list of values, or you can use Start and End dates to control when a value appears in a list. For a list of lookup types, see *Oracle Advanced Pricing Implementation Guide*, Lookups, page G-1.

17. (Optional) Define Oracle Order Management lookups.

The following table lists lookup types and their descriptions:

Lookup Type	Lookup Description
Define sales channel	Required if you price, provide benefits or charge by sales channel.
Define order categories	Required if you price, provide benefits or charge by order category.
Define line categories	Required if you price, provide benefits or charge by order line category.
Define order sources	Required if you price, provide benefits or charge by order line category.
Define shipment priorities	Required if you price, provide benefits or charge by shipment priority
Define ship methods	Required if you price or provide benefits, including upgrading shipping method or charge by shipment method.

For a list of valid default values for these lookups please refer to *Oracle Order Management User's Guide*, Lookups Appendix. Complete this step if you have not installed and set up Oracle Order Management or completed a common-applications setup.

**18.** (Optional) Define shipping lookups.

Default: Lookup type dependent

For a list of valid default values for these lookups, see *Oracle Shipping Execution User's Guide*.

Complete this step if you have not installed and set up Oracle Order Management or completed a common-applications setup.

**19.** Set up customer class.

Default: None

Required if you price, provide benefits, or charge by customer class.

For more information on setting up customer class, see *Oracle Receivables*, Defining Lookups.

Complete this step if you have not installed and set up Oracle Receivables or completed a common-applications setup.

**20. Set up profile classes.**

Default: None

Required if you price, provide benefits, or charge by customer account type.

For more information on setting up profile classes, see: *Oracle Receivables, Profile Classes*.

Complete this step if you have not installed and set up Oracle Receivables or completed a common-applications setup.

**21. Define customers and customer sites.**

Default: None

Required if you price, provide benefits, or charge by customer.

For more information on defining customers, see *Oracle Receivables, Defining Customers*.

Complete this step if you have not installed and set up Oracle Receivables or completed a common-applications setup.

Customer Sites

Default: None

Required if you price, provide benefits, or charge by customer site.

For more information on defining customer sites, see *Oracle Receivables, Defining Customer Sites*.

Complete this step if you have not installed and set up Oracle Receivables or completed a common-applications setup.

**22. Define order type.**

Default: None

Required if you price, provide benefits, or charge by order type.

For more information on defining order types, see *Oracle Order Management, Defining Order Types*.

Complete this step if you have not installed and set up Oracle Order Management or completed a common-applications setup.

**23. Define line type.**

Default: None

Required if you price, provide benefits, or charge by order line type.

For more information on defining line types, see *Oracle Order Management, Defining Order Line Types*.

Complete this step if you have not installed and set up Oracle Order Management

or completed a common-applications setup.

**24. Define freight terms.**

Default: None

Required if you price, provide benefits (including upgrading freight terms), or charge by freight terms.

For more information on defining freight terms, see *Oracle Order Management, Defining Freight Terms*.

Complete this step if you have not installed and set up Oracle Order Management or completed a common-applications setup.

**25. Define freight cost type.**

Default: None

Required if you price, provide benefits, or calculate charges using freight cost types.

For more information on defining freight cost types, see *Oracle Order Management, Freight Cost Types*.

Complete this step if you have not installed and set up Oracle Receivables or completed a common-applications setup.

**26. Define payment terms.**

Default: None

This step is required if you price, provide benefits, or charge by payment terms.

For more information on defining payment terms, see *Oracle Receivables, Defining Payment Terms*.

Complete this step if you have not installed and set up Oracle Receivables or completed a common-applications setup.

**27. (Optional) Enable currencies.**

Default: All major currencies predefined with Oracle Applications.

The system administrator completes this step. The codes are ISO standard codes for currencies. You must enable the specific currencies that you want to use on your price and modifier lists.

For more information on enabling currencies, see *Oracle General Ledger, Currencies*.

Complete this step if you have not installed and set up Oracle General Ledger or completed a common-applications setup.

**28. Create events and phases.**

Default: Seeded Oracle Advanced Pricing phases

This step is required if you must create additional pricing phases or change the seeded pricing phases. Complete this step if your pricing requires certain types of benefits at a particular point or event in the transaction process flow.

For more information on phases and events, see *Oracle Advanced Pricing Implementation Guide*, What are Pricing Events.

**29. Verify pricing parameter settings.**

You can set values for pricing parameters to control the behavior of the pricing application. Unlike profile options (which can be set at the site, application, responsibility, and user levels), you can set up pricing parameters at the PTE and Request Type levels. Request types are those calling applications that request pricing data (for example, request types such as Order Management Order or Oracle Contracts for Service). Pricing parameters provide flexibility in situations where the parameter value must be different for different request types.

**30. (Optional) Set up pricing security.**

Oracle Advanced Pricing provides pricing security in addition to the existing functional security. Pricing security enables you to grant privileges that control users' access to pricing entities such as price lists, pricing agreements, and modifiers.

During implementation, the Oracle Pricing Administrator can set up pricing security for pricing entities as follows:

- Assign or reassign pricing entities to an operating unit. If Multi-Org Access Control (MOAC) is enabled, you can select from a list of operating units that you can access based on the profile option MO: Security Profile.
- Set Global Usage to share or not share a pricing entity across operating units.
- Assign privileges to pricing entities to control who (the grantee) can view or maintain the specified entity.
- Set up default security profile options that set the access privileges for new pricing entities. See *Overview of Oracle Pricing Security* for more information.



---

# Implementation Methodology

This chapter covers the following topics:

- Overview of Implementation Methodology
- Using Oracle Advanced Pricing in the HTML Interface
- Methodology Steps

## Overview of Implementation Methodology

The methods outlined in this chapter can be used by licensed customers of Oracle Advanced Pricing, Oracle implementation consultants, and Oracle pre-sales consultants.

## Oracle Advanced Pricing Concepts

Before developing a pricing solution model for customer requirements, you should understand how the following pricing concepts are used to describe your pricing rules, pricing actions, controls, and the links to any extensions you employed.

- Pricing rules
- Pricing actions
- Pricing controls
- Pricing extensibility

### Pricing Rules

Pricing rules describe the rules-based logic that the pricing engine uses to apply a pricing action to a transaction.

Pricing rules are built around the customer and product hierarchies, and are often linked to the customer. Because customers belong to single or multi-level groups, Oracle Advanced Pricing enables you to define rules that associate pricing actions with

a group or level of a group that a customer belongs to.

With Oracle Advanced Pricing you can define rules dependent on data other than the pricing hierarchy. For example, discounts are often progressive-based on volume.

You can set up pricing rules in the Qualifier window to describe customer and non-product rules. These qualifier objects are then attached to pricing actions. The product hierarchy definitions are contained in the action objects. For example, if a price list calls for all items in item category A to be priced at \$1.00, the product definition is setup as part of the Price List window rather than using the Qualifier window. To restrict the usage of the same price list to only a VIP customer class, enter the customer class in the Qualifier window.

### **Pricing Actions**

Pricing actions are specific pricing activities that the engine applies to transactions. For example, the pricing engine can establish a list price by applying a price list action to an order line. Similarly, a discount that lowers the list price down to a net selling price is a modifier action.

Oracle Advanced Pricing provides pricing actions that can be used to handle pricing problems. Pricing Actions can be directly related to specific pricing windows, or objects. These objects are:

- Price lists
- Agreements
- Formulas
- Modifiers

Modifiers are a class of pricing actions that modifies the net price either up or down. You can select from several types of modifiers to apply a specific pricing action with specific behavior characteristics.

**Note:** Modifiers are sometimes called adjustments.

### **Pricing Controls**

Pricing controls are used to control how pricing applies actions. For example, effectivity dates can be used to control when rules are in effect and when they are not. They can also be used to define when an action that a rule points to can be applied.

Pricing also supports several other types of controls. For example, you can set up phases, which are groups of pricing actions. You can then tie these phases to physical events in order management. Incompatibility groups are another example of pricing controls. You can assign modifiers to incompatibility groups such that only one modifier within an incompatibility group can be selected by the pricing engine.

There are many different controls in pricing. All are important, but some require more careful planning prior to implementation.

## Pricing Extensibility

Pricing is often driven by customer factors, by trade customs, or by sales channel requirements. Almost no two companies implement pricing in the same way, even if they compete in the same industry.

To address this variability, Oracle Advanced Pricing provides the following extensibility features that help meet your business needs:

- **Flexible Attribute Mapping**

Oracle Advanced Pricing, when licensed with Oracle Order Management, Oracle iStore, or other Oracle products, provides you with defaults for the customer and product hierarchy and other commonly referenced pricing rule drivers. These can be used without extending the product.

For customers whose product or customer hierarchy is more complex, Oracle Advanced Pricing provides flexible attribute mapping. This enables you to add or change levels of the customer and product hierarchy in addition to the seeded values delivered with the product. You can use attribute mapping to substitute an alternative source and structure for either the product or customer hierarchy, rather than using those supplied as defaults when the product is installed.

You can use attribute mapping to link Oracle Advanced Pricing to data maintained outside Oracle Advanced Pricing and outside Oracle Applications. In addition to the customer and product hierarchy related data mentioned, you can use attribute mapping to make pricing read a currency rate from an external source used in a pricing formula computation.

- **Application Programming Interfaces (APIs)**

Oracle Advanced Pricing delivers a set of public APIs that you can leverage using your own code. This enables you to significantly extend the use of Oracle Advanced Pricing.

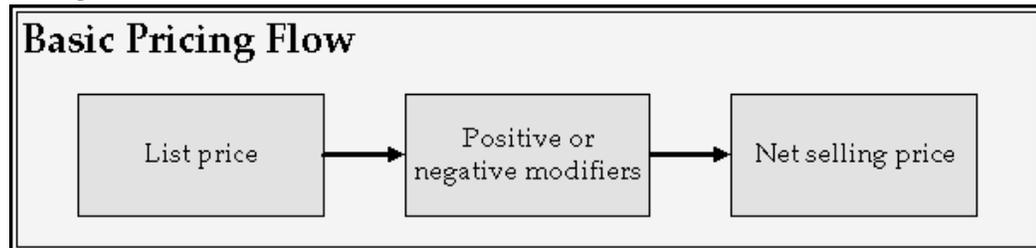
For example, the `Get_Custom_Price` API can be linked to a price list using a formula with a term of type `Function`. You can use this API to make Oracle Advanced Pricing obtain a cost from a purchase order that is used in a calculation that yields a selling price based on actual cost.

Oracle Advanced Pricing APIs are documented in the *Oracle Order Management Open Interfaces, API, & Electronic Messaging Guide*. For users who want to write code for these APIs, a downloadable ARU is available on My Oracle Support that contains coding examples.

## Basic Flow of Pricing

The following image depicts the basic flow for Oracle Advanced Pricing:

### Pricing Flow



Oracle Advanced Pricing prices any transaction by first identifying a list price defined in a price list. Pricing determines whether the price found on the price list must be adjusted. Modifiers control the price of these adjustments, and can modify price either up (positive) or down (negative). Once the pricing engine has selected one or more modifiers that adjust the price or applies a benefit, it computes the final price or net selling price.

#### Pricing windows

Pricing is comprised of business pricing entities, each of which has defined functionality. You can use these entities and their associated functionality to construct pricing solutions. You interact with these Oracle Advanced Pricing business entities using windows (also known as Forms) and HTML pages that you set up to obtain functionality.

Pricing windows are specific tools within the Oracle Advanced Pricing product used to provide pricing solutions. Each object has certain functions that it performs as part of the solution. For further discussion of Oracle Advanced Pricing windows, see *Oracle Advanced Pricing User's Guide*.

## Using Oracle Advanced Pricing in the HTML Interface

Oracle Advanced Pricing provides an HTML-based user interface (UI) that enables you to complete many pricing tasks previously available only in the Oracle Forms-based interface. The HTML UI features guided steps and user-friendly pages that you can use to set up and maintain modifiers, price lists, and qualifiers.

**Note:** The HTML user interface is available only in Oracle Advanced Pricing.

#### Cross-Format Compatibility

Pricing entities such as price lists, modifiers, or qualifiers created in one format can be maintained in the other. For example, if you created a price list in the Forms-based UI, you could update and maintain the price list in the HTML UI. You can still use the Forms-based user interface to set up and maintain pricing functions and features as in previous releases.

**Note:** Some features, such as qualifier groups, can only be set up in the Forms-based UI.

### **Accessing the HTML User Interface**

To access the HTML user interface, select the Oracle Pricing User responsibility and click the Home link to display the **Oracle Advanced Pricing Home** page.

See the *Oracle Advanced Pricing User's Guide* for more information on the features and functions available using the HTML UI.

## **Methodology Steps**

The recommended implementation methodology for Oracle Advanced Pricing consists of three steps.

1. Analyze pricing needs.
2. Develop pricing solutions.
3. Test pricing solutions.

### **1. Analyze Pricing Needs**

The first step in implementing pricing is to understand your customer's pricing requirements. To develop a pricing solution with Oracle Advanced Pricing, you must break down the pricing requirements into their component parts. Once you analyze customer requirements, underlying pricing requirement elements, relationships between the elements, and calculations, you can associate each of these elements and their relationships with the correct business object. From this analysis, you can construct a pricing solution.

#### **Note:** Pricing Terminology

In analyzing pricing requirement prior to implementing Oracle Advanced Pricing, it is important to consider that pricing terminology varies greatly across companies. Terminology such as tiered pricing, block pricing, off-invoice, price ceilings, price floors, and many others vary in their definition from one company to the next.

The best way to avoid misleading terminology is to define each term by its underlying pricing action and the rules that define the precise conditions under which that action is taken.

#### **Step 1. Define Pricing Requirements**

Identify the pricing scenario and its requirements. For example, customers in the Northeast Region may order any two digital widgets and receive 10% off of either analog widget item ABC or DEF ordered at the same time. This is a promotional

discount given for orders received between October 1 and December 31.

### **Step 2. Analyze Pricing Requirements**

Define the underlying component pricing requirements by gathering enough information to precisely define the pricing action itself, and the conditions under which this action is to be taken. Break down the preliminary problem statement further:

- Distinguish between the digital widgets and other widgets. Digital widgets are discounted separately from analog widgets.
- The pricing solution must allow for either 1 or 2 of each digital widget to qualify for the free item.
- The pricing solution model must allow for adding multiple lines to the order to allow line 1 and 2 to be combined to equal the mix and match criteria.
- The 10% discount is taken off the selling price.

### **Step 3. Create Rule and Action Statements**

The next step is to develop rule and action statements that define the pricing actions to be taken and the conditions for the pricing action to be taken. First define the action:

Pricing action: Give a 10% discount taken off normal selling price. This action statement is independent of customer, product, or dates.

- Pricing rule: Discount action taken only for all customers in the Northeast Region.
- Pricing rule: Discount action applied only to analog widgets with item numbers ABC or DEF.
- Pricing rule: Discount action taken only when analog widgets are ordered at the same time as digital widgets.
- Pricing rule: Discount action taken only for orders received between October 1 and December 31.

## **Pricing Structures Example**

In the example it is depicted how a pricing requirement expressed by an end user can be broken down into a pricing action statement and pricing rules that describe the conditions under which the action should be taken. Pricing policies may be defined in broader terms. In this case, before each action is defined along with its associated rules, it is necessary to understand the overall pricing structure of the company.

### **Step 1: Define the Pricing Requirement**

Rick's Souvenir Company sells two different consumer product lines. These products are sold through different sales channels. Rick's Pet Store sells

- Edible products

- Collectible products

Rick's edible products are sold through the following:

- Grocery store retail outlets
- Mass merchandiser retail outlets

Rick's collectible products are completely unrelated to the edible products, and are produced in a separate manufacturing facility. They are sold through the following:

- Hobby and toy stores
- Mass merchandiser retail outlets

Given the overall configuration of the business described, the pricing policies of the company are most likely closely tied to the structure. The following information is discovered after further investigation of Rick's Souvenir Company:

#### **Edible Products**

The edible products sold to retail outlets and mass merchandisers receive price breaks depending on the quantity ordered. For example, product 12345 San Francisco Rice Cakes is sold as follows:

<b>Volume</b>	<b>Price Per Case</b>
0-49	55
49 - 199	50
199 - 499	45
499 and over	40

All edible products are priced similarly for retail and mass merchandisers. All retail and most mass merchandisers also are subject to receive promotional discounts that are offered to induce higher volume and or build market awareness at certain times of year. As an example, promotional discounts are offered on San Francisco Rice Cakes for orders placed between August 1 and October 15 as an inducement to retailers to build up inventory of San Francisco Rice Cakes for Halloween.

One very large mass merchandiser customer, HugeCo Inc., has elected to eschew promotional pricing in favor of what they term every day low price. They negotiated a net price for each item. In the case of San Francisco Rice Cakes, this builds in an additional 10% off the price per case sold to HugeCo to reflect the theoretical average deal rate that other customers receiving promotional pricing receive for all promotions given throughout the year. This gives HugeCo a different pricing schedule for San

Francisco Rice Cakes which is as follows

Volume	Price per case
0-49	50
49 - 199	45
199 - 499	40
499 and over	35

### **Collectible Products**

The collectible products consist of decks of cards where each card has the picture of a popular sports figure. The decks of cards are oriented to particular sports, such as baseball, soccer, and football. Collectible sports cards are limited production runs, and are targeted towards consumers who collect and trade the cards. The pricing of one of the collectible card products, the New Millennium Set, varies depending on sales channel. No promotional pricing is used, and quantity discounts are not given. However, for various business reasons, collectible sports card products including item number 65432 New Millennium Set is priced higher to the hobby dealers than to mass merchandisers. A case containing 24 complete New Millennium Sets is priced at \$375 to hobby dealers, and the price for mass merchandise outlets is \$325.

### **Step 2: Analyze the Pricing Requirement**

Rick's Edible Products have three separate pricing policies or structures: one for grocery, another for mass merchandisers other than HugeCo, and one especially for HugeCo. Within the grocery and mass merchandisers (non-HugeCo) channels, there are two levels or pricing: list price and promotional pricing.

HugeCo also has two levels. In HugeCo's case, these levels are list price and everyday low price. Because HugeCo receives the everyday low price, it should not also receive promotional pricing.

Collectible sports cards have different selling channels each with a different pricing policy. These are hobby list price and mass merchandiser list price. Because there is no promotional price, HugeCo's everyday low price is the same as other mass merchandisers' price.

### **Step 3: Create Rule, Action, and Control Statements**

The following examples are limited to the two products already defined: edibles and collectibles; however, this same approach is used for defining other product pricing.

#### **Edible Product**

Pricing action: Give volume sensitive list price (see table for San Francisco Rice Cakes for prices at specific order volume levels).

- Pricing rule: For product San Francisco Rice Cakes.
- Pricing rule: For all customers belonging to retail customer class and to all customers in mass merchandiser customer class.

Pricing action: Give promotional discount of 10%.

- Pricing rule: For all customers belonging to retail customer class and to all customers in mass merchandiser customer class.
- Pricing rule: For product San Francisco Rice Cakes.
- Pricing rule: Exclude HugeCo from previous rule.
- Pricing rule: Apply to all orders received between August 1 and October 15.

Pricing action: Give HugeCo everyday low price schedule (see table).

- Pricing rule: For product San Francisco Rice Cakes.
- Pricing rule: For customer HugeCo.
- Pricing control: Apply to all orders for HugeCo received from January 1 to December 31.

#### **Collectible Product**

Pricing action: Give list price of \$375 per case.

- Pricing rule: For product New Millennium Set.
- Pricing rule: For all orders for customers belonging to hobby dealers.

Pricing action: Give list price of \$325 per case.

- Pricing rule: For product New Millennium Set.
- Pricing action: For orders for customers belonging to mass merchandiser class.

## **2. Develop Pricing Solutions**

Once you have defined your pricing actions and rule statements, you can define a pricing solution within Oracle Advanced Pricing. You define the implementation solution by designing an overall structure for pricing. The following sections describe decisions you must make when creating this structure.

#### **Implementation Decisions: Single versus Multiple Currency Price Lists**

Oracle Advanced Pricing supports both single currency price lists and multiple currency price lists. For single currency price lists, there is one currency defined per price list. For multi-currency price lists, a Multiple Currency Conversion list is attached to the price list defining multiple currencies and conversion factors and rules for

converting prices.

It is very important to determine which price list strategy your organization supports. Advanced Pricing provides a profile option and concurrent program to convert existing price lists from a Single Currency Price List to Multiple Currency price lists. However, once this profile is enabled, and price lists are converted, users should not return to NON Multi-Currency price lists. Changing the profile back to No may cause undesired results if conversion criteria was used. Oracle does not support changing the setting back to No.

For further discussion on using Multiple Currency Price Lists, see: *Oracle Advanced Pricing User's Guide*, Multiple Currency Price Lists.

### **Single Currency Price Lists**

Single Currency Price Lists are the default setting for Advanced Pricing. These are used when your business requires that you maintain different prices for different currencies, and there is no relationship between prices defined.

### **Multiple Currency Price Lists**

Multiple Currency Price Lists enables businesses that have pricing strategies based on a single price for an item in a base currency and use exchange rates or formulas to convert that price into the ordering currency. At engine run time, the pricing engine will take the currency from the order and search for a price list(s) with base or conversion currencies matching this currency. The pricing engine converts the price from the base currency and calculates the ordering currency based upon the established conversion rules.

### **Implementation Decisions Related to Customer and Product Attributes**

Customer attributes are used to reference single- or multi-level groups with which a customer may be associated. Pricing rules are structured around these groupings which then affect the pricing actions a customer receives.

### **Key Implementation Decision: Are the seeded attributes on Oracle customer tables sufficient to contain customer groupings required for your pricing needs?**

The following qualifier attributes with context of Customer (also called Customer Attributes) are seeded in Oracle Advanced Pricing:

- Customer name
- Customer class
- Site
- Ship to
- Bill to
- Agreement name
- Agreement type

- GSA
- Sales channel
- Account type
- Party ID
- Ship To Party Site
- Invoice To Party Site

**Additional Information:** These attributes can be used as qualifiers for price lists and modifiers. Information about Party ID, Ship To Party Site, and Invoice To Party Site are automatically derived from the sourcing rules.

You can use the preceding attributes by referencing them as qualifiers for either modifiers or price lists.

If the seeded qualifier attributes from the Oracle customer tables are adequate, then further mapping of customer attributes is not required. If the seeded qualifier attributes from the Oracle customer tables are not sufficient, you can create additional customer attributes in Oracle Advanced Pricing using attribute management. For example, you may want to define attributes for additional customer groupings.

For both options, you can use attribute mapping to define the location of the customer data to reference in pricing qualifiers. You must create a default sourcing rule for each defined qualifier attribute because the pricing engine uses mapping rules to locate the attribute values. You have the following options to expand your attributes:

- Use attribute mapping rule to source from Oracle customer tables.
- Use tables outside Oracle to store the customer attributes.

You can map an alternative set of customer attributes using attribute mapping. For more information on attribute mapping, see *Overview of Attribute Management*, page 17-1.

A *customer class* is a grouping of customers. You can define as many classifications as you want and set up pricing data depending on how you want to make prices/benefits available to different classifications. For attributes that reference an Oracle Trading Community Architecture (TCA) Party, you can select the Party Hierarchy Enabled check box. This enables you to create qualifiers so that any associated price lists and modifiers are available to the party hierarchy defined in TCA.

### **Product Hierarchy**

Product hierarchies are single or multi-level groups to which a product may be associated. Pricing rules are structured around these groupings which then affect the

pricing actions a customer receives.

Each level in your product hierarchy where your business sets its pricing and discounting should be defined as a product attribute in the seeded product context ITEM.

The product hierarchy in Oracle Advanced Pricing is seeded with ITEM context based on the Oracle Applications Item Master, MTL\_SYSTEM\_ITEMS table. For this context, the flexibility of Oracle Advanced Pricing enables you to define your product hierarchy as follows:

- Item number
- Item category (supports hierarchical categories)
- Item All
- Pricing attributes to specify a level more detailed than just item

You can define attributes using the attribute management feature in Oracle Advanced Pricing. When defining product attributes, use the precedence value to define the hierarchy. If the pricing engine must choose between an item price and an item category price, set the item attribute as more specific than the product group attribute.

#### **Item categories and category sets**

You can define item categories and category sets for pricing and discounting. Item categories in pricing setups are always based on the item category of the item at the Master Level and not at the Org level.

#### **Pricing Attributes**

Attributes define exactly what is being priced or modified. Attributes can be factors that affect the price of the item, additional definition that does not require creating an item, or discounting at a level higher than item in the product hierarchy.

Creating pricing attributes in different contexts enables attributes to be grouped according to their business requirements. The item context is reserved for defining the product pricing hierarchy. Every level in the product hierarchy at which pricing and discounting is set should be defined as a segment in this context.

#### **Key implementation decision: Must I implement pricing attributes, multiple price lists, or both? What pricing and qualifier attributes do I need, and how do I decide which is which?**

Complex pricing scenarios can be best be solved with a combination of pricing attributes and multiple price lists. Not only can a combination of the two be easier to understand and maintain, but it can improve engine performance.

Pricing attributes further control what is being priced or modified on a price list or modifier list. Each level in your product hierarchy should be defined as a pricing attribute in the seeded context, item. You must create a default sourcing rule for each pricing attribute that you define since the pricing engine uses sourcing rules to locate pricing attribute values.

For multiple price lists, qualifiers offer and/or capability as well as =, between, and not= operators.

For example:

- If you charge different list prices for the same item, depending on conditions in effect at time of order, you can use pricing attributes.
- If conditions are product oriented (example Item 123 Grade A is priced differently from same item Grade B) then product attributes are indicated
- If conditions are not product oriented, but depend on customer's membership in hierarchy or combination of customer and other factors like order type, time of day, then multiple price lists are indicated

## Example: Structuring Your Pricing Rules and Actions

Key implementation decisions:

- What qualifiers and qualifier groups do I need?
- What customer groupings are required in order to develop qualifiers that implement my pricing rules?
- Which pricing actions are needed to process my scenarios?
- What product groupings are needed for my pricing actions?

The structure of your customer and product hierarchies is crucial in setting up and implementing Oracle Advanced Pricing because these hierarchies are essential for defining pricing rules.

- Qualifiers

Qualifiers enable you to define eligibility rules that determine which pricing actions are applied to a transaction. Although a qualifier can be used for any pricing rule, the qualifier window is generally used to define pricing rules related to attributes defined for the customer hierarchy or for other elements not related to the product hierarchy.

When you implement Oracle Advanced Pricing, think in terms of structure for your pricing actions and qualifiers used to select those actions. Use the example of Rick's Souvenirs you used previously in the chapter.

Rick's Souvenirs has two broad product lines each sold through two sales channels. One of the channels overlaps. Pricing followed the channel lines, with the exception of one large customer. Draw some inferences as to how to structure pricing actions and rules based on this information.

- Customer Groupings

Two levels of customer hierarchy must be tied to pricing rules:

- Customer class: grocery, mass merchandiser, hobby
- Customer name: HugeCo

Because customer class and customer name are both seeded elements of the customer hierarchy, no extension to the customer hierarchy is necessary.

- Pricing Actions

The action/rule statements derived earlier show that the need for price list and promotional discounting actions that vary by product and are conditioned by channel.

- Edible Product
  - Price list with breaks for grocery and merchandiser
  - Promotional pricing discounts for grocery and mass merchandiser
  - Alternative price list with breaks for HugeCo
- Collectible Product
  - Price list for hobby
  - Price List for mass merchandisers

Product groupings: Lists are set for the individual products and in this case you have specific item numbers. However, the promotional discount of 10% applies to all edible products. Because item number and item category are both seeded elements in the product hierarchy you need not consider extending the product hierarchy.

### Price Lists:

Set up the following price lists for edible and collectible products

#### Edible Product

Begin by structuring a single price list that handles list pricing. The following table depicts the price list and its related price break lines:

Context	Attribute	Unit Price	Value	Pricing Attribute	Value To	Value From
Product	Item	55	12345	Volume	0	49
Product	Item	50	12345	Volume	49	199
Product	Item	45	12345	Volume	199	499

Context	Attribute	Unit Price	Value	Pricing Attribute	Value To	Value From
Product	Item	40	12345	Volume	499	9999999

In the previous example, you set up a price list action in Oracle Advanced Pricing that gives channel specific pricing for edible products.

### Collectible Product

Now set up a second price list to handle the collectible product 65432 New Millennium Set.

Context	Attribute	Unit Price	Value	Pricing Attribute	Value
Product	Item	\$375	65432	Customer class	Hobby
Product	Item	\$325	65432	Customer class	Mass Merch

The price list is qualified for both customer classes of both Mass Merch and Hobby. Because the price is different for the two classes, within the price list, you used customer class as a pricing attribute. This enables you to define the collectible item on the list twice, once for each channel.

### Combined Edible and Collectible Price Lists

You can combine the two separate price lists into one price list with both price break lines and price lines:

Context	Attribute	Unit Price	Value	Pricing Attribute	Value To	Value From
Product	Item	55	12345	Volume	0	49
Product	Item	50	12345	Volume	49	199
Product	Item	45	12345	Volume	199	499
Product	Item	40	12345	Volume	499	9999999
Product	Item	\$375	65432	Customer class	Hobby	Not applicable

Context	Attribute	Unit Price	Value	Pricing Attribute	Value To	Value From
Product	Item	\$325	65432	Customer class	Mass Merch	Not applicable

### Promotional Discount Modifier

Two adjustments are required for a total solution. The first adjustment: promotional pricing for edible products given to both grocery and mass merchandise class customers. The second adjustment: special everyday low pricing instead of the standard pricing given to HugeCo. A new, additional requirement: the everyday low pricing received by HugeCo must be accounted for as a discount.

First you need a modifier for promotional pricing. The promotional discount is the same percentage for the entire customer class. In the following table, the customer class is Mass Merch and Hobby for the edible products promotional discount.

Context	Attribute	Value	Type	Method	Amount	Incompat. Group
Product	Category	Edible	Discount	Percent	10	GRP01

An incompatibility group assignment is added because the modifier should not be used for HugeCo. Build the modifier for HugeCo:

Context	Attribute	Type	Method	Unit Price	Value	Pricing Att.	Value From-To	Incomp. Group
Product	Item	PBH	Amount	50	12345	Volume	0-49	GRP01
Product	Item	PBH	Amount	45	12345	Volume	49-199	GRP01
Product	Item	PBH	Amount	40	12345	Volume	199-499	GRP01
Product	Item	PBH	Amount	35	12345	Volume	499-999999	GRP01

You defined a modifier for HugeCo that gives HugeCo a new price based on volume. By using a modifier action rather than a price list action, you met the finance requirement to account for the difference in list price and the everyday low price given to HugeCo as a discount. By using Application Method of amount you are substituting the unit prices on the modifier for the unit prices found on the price list shown in either example 1 or example 3 at each volume level. Because modifiers also support price

breaks, you structured the HugeCo modifier as a price break.

Having accomplished this, you are ready to turn to your next implementation decision area: pricing controls.

## Establishing Pricing Controls

Pricing controls describe a group of features in Oracle Advanced Pricing that enable you to specify how the pricing engine interprets your pricing rules and actions. From an implementation perspective, one is a decision that you should make in your implementation process.

### **Key Implementation Decisions: Must I implement incompatibility groups?**

In the example you used earlier in this chapter, by assigning the two modifiers for HugeCo to an incompatibility group, you can force the pricing engine to only choose one.

### **Must I use Oracle Advanced Pricing's exclusivity feature?**

Exclusivity enables you to specify a modifier that, if selected by the pricing engine, is the only modifier applied to a transaction.

### **Do I instruct Oracle Advanced Pricing to use precedence or best price to decide between incompatible modifiers?**

Oracle Advanced Pricing offers two methods to choose between incompatible modifiers: precedence and best price. Precedence is the usually the best choice.

### **Must I implement GSA Pricing?**

GSA pricing enables you to establish a floor level price. It is used to comply with General Services Administration Pricing rules. GSA pricing integrates with Oracle Order Management.

### **Must I use multiple pricing buckets? If so how many levels do I need?**

Pricing Buckets give you the capability to handle discounts with multiple subtotals.

### **Do the seeded pricing phase/Oracle Order Management event relationships work or must I reconfigure them?**

Pricing modifiers and price lists must be associated with a specific pricing phase. When an application calls the pricing engine during a pricing request, the phase is identified in the call; this enables the pricing engine to select only the qualified pricing actions for that phase. In Oracle Order Management, the integration with prices links certain physical events in the order processing cycle to specific phases. By linking events to a specific phase, Order Management can specify particular pricing phases to the engine. The mapping of pricing phases to order management events can be configured by the user.

More detailed information on the pricing phases and how to use them, see the *Events and Phases chapter* in this guide.

### **What effectivity dates must I establish?**

Oracle Advanced Pricing provides very extensive effectivity dating capabilities. When the pricing engine is invoked by a calling application such as Oracle Order Management, it is passed a pricing date. The engine uses this date as a reference point

to compare to effectivity dates found on various pricing forms.

The pricing date can be defaulted in Oracle Order Management. For specifics, see *Oracle Order Management User's Guide*.

### **What unit of measure considerations are necessary for Oracle Advanced Pricing setup?**

If the Pricing engine attempts to determine a base price for a transaction, then the engine attempts to search qualified price lists for a price in the unit of measure that is in transaction line to be priced. If the engine finds such a UOM, the transaction UOM becomes the pricing unit of measure.

If that search is not successful, the engine searches for a conversion factor to convert the UOM on the incoming transaction to the primary UOM on the price list line.

The pricing engine returns only those modifiers defined in the same unit of measure as the pricing unit of measure, or where there is no product UOM specified. The latter may be the case if the modifier is not product specific.

Verify that all modifiers and price lists used together have common units of measure. Verify that if the unit of measure on the price list and modifier is different from the ordering UOM, that a conversion factor has been set up.

### **Do I want to control spending of Promotions?**

Promotional Limits functionality enables you to set maximum values for benefits that a customer can receive for a promotion, deal or other modifier. By limiting the amount of a benefit that can be received, you can keep promotional spending within budget and prevent promotion budget overruns.

For more information on Promotional Limits, see: *Oracle Advanced Pricing User's Guide*.

## **3. Testing Your Pricing Solutions**

The final step in the Oracle Advanced Pricing implementation methodology is to test your pricing solutions.

**Note:** If you upgraded from a release prior to 11*i* and are running Oracle Advanced Pricing in Basic Mode with upgraded data, do your testing in a separate test instance of your system.

1. Verify that you tested each scenario. Develop a documented script for each scenario that describes the setup and expected result, and then gives a step by step outline.
2. Begin with simple scenarios, and then work your way up to more complex ones.
3. Verify that you receive your expected numerical results.
4. Once your setups work in your test environment, you can begin replicating these setups into your production environment. Verify that the price list and modifier

flags are set to inactive. Select beginning effectivity dates with caution to prevent the pricing engine from prematurely using new setups for pricing production transactions.

For information on diagnostics and troubleshooting, see the *Diagnostics and Troubleshooting chapter* in this guide.



---

# Oracle Advanced Pricing Command Center Implementation

This chapter covers the following topics:

- Setting Up Advanced Pricing Command Center
- Setup and Configuration Steps for Advanced Pricing Command Center
- Reviewing Security Setup for Advanced Pricing Command Center
- Profile Options for Advanced Pricing Command Center
- Configuring Descriptive Flexfields for Search
- Loading Pricing Data

# Advanced Pricing Command Center Configuration

## Setting Up Advanced Pricing Command Center

See *Advanced Pricing Command Center Overview, Oracle Advanced Pricing User's Guide*

The Advanced Pricing Command Center configuration setup must be completed after the installation and common configurations are completed as described in My Oracle Support Knowledge Document 2495053.1, *Installing Oracle Enterprise Command Center Framework, Release 12.2*.

See *Setup and Configuration Steps for Advanced Pricing Command Center, page 4-2*.

## Setup and Configuration Steps for Advanced Pricing Command Center

See *Advanced Pricing Command Center Overview, Oracle Advanced Pricing User's Guide*

To set up the Advanced Pricing Command Center:

1. Review the security setup for the Advanced Pricing Command Center, page 4-2.
2. Set profile options, page 4-3.
3. Configure descriptive flexfields for search, page 4-4.
4. Load the pricing setup data, page 4-4.

## Reviewing Security Setup for Advanced Pricing Command Center

See *Advanced Pricing Command Center Overview, Oracle Advanced Pricing User's Guide* and *Setup and Configuration Steps for Advanced Pricing Command Center, page 4-2*.

The Advanced Pricing Command Center uses the same security mechanisms as Oracle Advanced Pricing. You can control who can use the command center's dashboard and which records those pricing users can access by defining pricing security and Multi-Org Access Control (MOAC) and setting up security profile options for pricing entities.

Security rules affect the calculation of metrics and the plotting of charts, which use only the data to which the user has access. When implementing the command center, you must consider the security structure and ensure that the concerned pricing team members have appropriate access to the pricing data. If security is not set up correctly, then the pricing data and any calculation based on that data may be incorrectly displayed for the given user.

**Important:** Security privileges through grants enable you to determine who can access each pricing entity and the permitted level of access: View Only or Maintain. Setting security privileges has no effect on the data load because the application implements these permissions only after the data load. If you use security privileges, then the application displays and enables access to edit data on the dashboard, based on the privileges.

## Profile Options for Advanced Pricing Command Center

See *Advanced Pricing Command Center Overview, Oracle Advanced Pricing User's Guide* and *Setup and Configuration Steps for Advanced Pricing Command Center*, page 4-2.

You must set the following profile options before you run the data load concurrent program.

Profile Option Name	Description
<b>QP: Source System Code</b>	<p>This profile identifies the application through which the pricing information is entered. This source system code is held on all price and modifier lists to identify the origin of the data. By default, the value of the profile option is QP (Oracle Pricing) at the Site level. However, to differentiate between applications and prevent update of modifiers among several applications, the value of this profile can be set up at the Application level.</p> <p>For more information about how to use this profile option, see <i>Profile Options Setup Summary</i>, page 5-3.</p>
<b>QP: Pricing Transaction Entity</b>	<p>This profile option indicates the current Pricing Transaction Entity (PTE) that is in use. Only those contexts and attributes assigned to the current Pricing Transaction Entity are available in the list of values on the setup forms.</p> <p>For more information about how to use this profile option, see <i>Profile Options Setup Summary</i>, page 5-3.</p>

## Configuring Descriptive Flexfields for Search

See *Advanced Pricing Command Center Overview, Oracle Advanced Pricing User's Guide* and *Setup and Configuration Steps for Advanced Pricing Command Center*, page 4-2.

Enterprise command centers support searching on descriptive flexfield (DFF) attributes. After you configure DFFs, you must run the data load process to make the DFF attributes available in the command center.

For additional information about configuring and customizing flexfields, see *Oracle E-Business Suite Flexfields Guide* and My Oracle Support Knowledge Document 2495053.1, *Installing Oracle Enterprise Command Center Framework, Release 12.2*.

The following table describes the DFFs that are available in the Pricing User Dashboard:

Data Set	DFF Title	DFF Name	DFF Attribute Group Name	Displayed in
qp-headers	Additional Info for List Headers	QP_LIST_HEADERS	QPH_INFO	Available Refinements, Headers results table
qp-lines	Additional Info for List Lines	QP_LIST_LINES	QPL_INFO	Available Refinements, Lines results table
qp-qualifiers	Additional information for Qualifiers	QP_QUALIFIER_S	QPQ_INFO	Available Refinements, Qualifiers results table

## Loading Pricing Data

See *Advanced Pricing Command Center Overview, Oracle Advanced Pricing User's Guide* and *Setup and Configuration Steps for Advanced Pricing Command Center*, page 4-2 for more information.

To process and load data from Oracle E-Business Suite to the Advanced Pricing Command Center, run the concurrent program *QP: Enterprise Command Center Data Load*, which is located under the standard **Oracle Pricing Manager** responsibility >**Reports**. You can also run the data load program for individual data sets using the **Data Load Submission** page of the ECC Developer responsibility.

Run this program from the **Submit Request** window.

### QP: Enterprise Command Center Data Load Request

Submit Request

Run this Request... Copy...

Name QP: Enterprise Command Center Data Load

Operating Unit

Parameters

Language American English

Parameters

System Name EBS

Load Type

Languages

SQL Trace

Log Level ERROR Error

OK Cancel Clear Help

Help (C) Submit Cancel

#### To load pricing data:

1. In the **Name** field, enter **QP: Enterprise Command Center Data Load**.
2. In the **System Name** field, accept the default value **EBS**.
3. Select the appropriate **Load Type**.
  - **Full Load:** Loads pricing data and is required to be run for the first data load. The full data load process also includes metadata load. If you run full load for subsequent requests, then this program clears all pricing data from ECC and loads fresh data.
  - **Incremental Load:** Loads only the data that has been modified since the previous load. Schedule incremental loads to run as often as required to keep the ECC dashboard current.
  - **Metadata Load:** Loads Descriptive Flexfield (DFF) metadata. If there are any changes to the DFF definition, then you must run the program first with the Metadata Load option and then the Full Load option so that the DFF changes are displayed in the command center.
4. In the **Languages** field, enter one or more language codes for the output. For multiple language codes, use the format AA,BB,NN. If this field is blank, then the

data will be loaded for the base language only (usually US).

5. Select the log level that you want the program to report. The default value is ERROR. Select **DEBUG** to debug the program. This option creates log files for all data loads that you can use to debug issues with data loading. The QP\_DEBUG\_TEXT table stores debug information.
6. Select **TRUE** to enable SQL Trace. Otherwise, select FALSE.
7. Submit the concurrent request.
8. Review your request using the **tRequests** page.
9. Monitor data loading using the **Data Load Tracking** page of the ECC Developer responsibility.

---

# Profile Options and Pricing Parameters

This chapter covers the following topics:

- Overview of Profile Options
- Profile Options Setup Summary
- Overview of Pricing Parameters
- Viewing Pricing Parameter Definitions
- Viewing and Updating Pricing Parameter Values
- Seeded Pricing Parameters

## Overview of Profile Options

During implementation, profile options can be set to specify how Oracle Advanced Pricing controls access to and processes data. Typically, the System Administrator is responsible for setting up and updating profile option values. See: *Oracle E-Business Suite Setup Guide*, User Profiles and Profile Options in Oracle Application Object Library for more information.

### Find System Profile Values

**Find System Profile Values**

**Display**

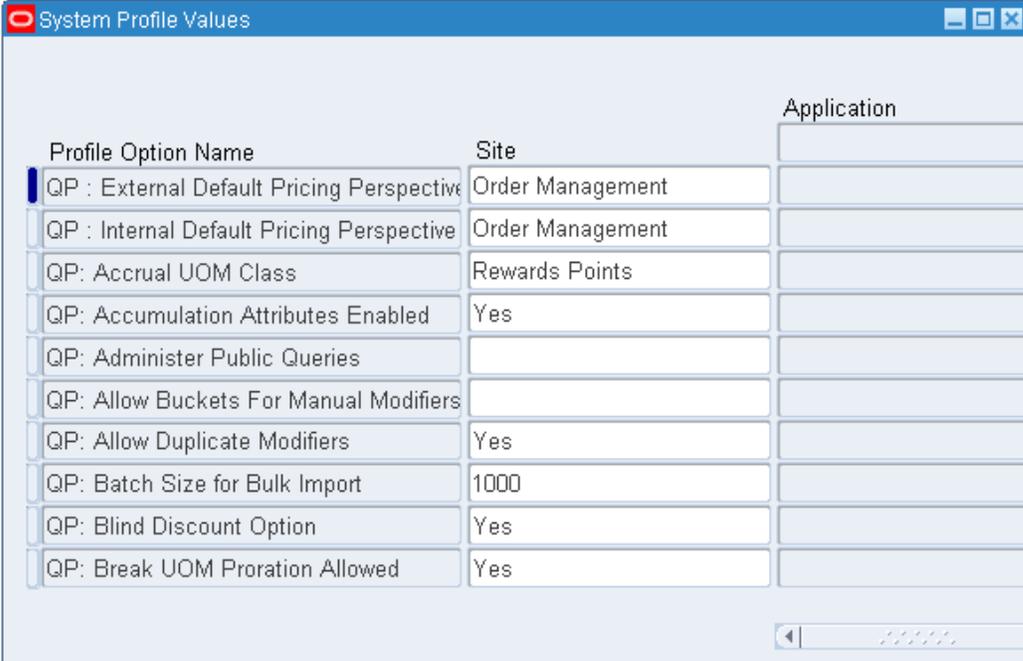
- Site
- Application
- Responsibility
- Server (B)
- Organization
- User
- Profiles with No Values

Profile

Find Clear

In the Profile field, enter QP% and click Find to display all the site level profile options for pricing in the System Profile Value window:

### System Profile Values window



The screenshot shows a window titled "System Profile Values" with a table containing the following data:

Profile Option Name	Site	Application
QP : External Default Pricing Perspective	Order Management	
QP : Internal Default Pricing Perspective	Order Management	
QP: Accrual UOM Class	Rewards Points	
QP: Accumulation Attributes Enabled	Yes	
QP: Administer Public Queries		
QP: Allow Buckets For Manual Modifiers		
QP: Allow Duplicate Modifiers	Yes	
QP: Batch Size for Bulk Import	1000	
QP: Blind Discount Option	Yes	
QP: Break UOM Proration Allowed	Yes	

For each profile option, select a Site value that supports your implementation requirements for Oracle Advanced Pricing.

## Profile Options Setup Summary

This section provides a summary of Advanced Pricing profile options and the System Administrator level at which the profile options can be viewed and updated: Site, Application, Responsibility, or User. The short names and definitions used in the tables are described as follows:

Value	Definition
SA Site	System Administrator: Site level
SA App.	System Administrator: Application level
SA Resp.	System Administrator: Responsibility level
SA User	System Administrator: User level
User	User level

Value	Definition
Yes	You can update the profile option.
No	You cannot change the profile option value.
Blank	You must specify a default value or you may encounter a system error.
Req? (Required)	<ul style="list-style-type: none"> <li>Required (Req.): Requires you to provide a value for the profile option.</li> <li>Optional: (Opt.) Already provides a default value, so you must change it only if you do not want to accept the default.</li> </ul>

For the following table, if the value No is displayed in the Default Value column, the default for the profile option is No.

If implemented with Oracle Order Management, the Order Management (OM) profiles indicated must also be considered in order to work with related pricing (QP) profile options.

Profile Option	SA Site	SA App.	SA Resp.	SA User	Default Value	User	Req?
OM: Discounting Privilege	No	Yes	Yes	Yes	Full	No	Opt
OM: GSA Discount Violation Action	View Only	No	No	No	Warning	No	Opt
OM: Promotion Limit Violation Action	Yes	No	No	No	Null	View Only	Opt
OM: Negative Pricing	View Only	View Only	View Only	No	Blank	View Only	Opt

Profile Option	SA Site	SA App.	SA Resp.	SA User	Default Value	User	Req?
OM: Display Qualified Ask for Modifiers (See: Setting up Non-Seeded Profiles)	Yes	Yes	Yes	Yes	No	View Only	No
<p><b>Note:</b> Since it is a non-seeded profile, it does not already have a set value. You can set the profile and its values to suite your requirement. The mentioned values serve as examples.</p>							
QP: Accrual UOM Class	Yes	Yes	No	No	Blank	View Only	Opt
QP: Accumulation Attributes Enabled	Yes	Yes	No	No	No	View Only	Opt
QP: Archive Header Qualifiers	Yes	Yes	Yes	Yes	No	Yes	Opt
QP: Administer Public Queries	Yes	Yes	Yes	No	Blank	No	Opt
QP: Allow Buckets for Manual Modifiers	Yes	No	No	No	Blank	Yes	Opt
QP: Allow Duplicate Modifiers (Used by Basic Pricing Only)	Yes	No	No	No	Yes	No	Opt

<b>Profile Option</b>	<b>SA Site</b>	<b>SA App.</b>	<b>SA Resp.</b>	<b>SA User</b>	<b>Default Value</b>	<b>User</b>	<b>Req?</b>
QP: Batch Size for Bulk Import	Yes	Yes	Yes	Yes	1000	Yes	Opt
QP: Blind Discount Option	Yes	Yes	No	No	Yes	View Only	Opt
QP: Break UOM Proration Allowed	Yes	Yes	No	No	Blank	Yes	Req
QP: Build Attributes Mapping Options	Yes	No	No	No	No	View Only	Opt
QP: Continuous Price Breaks	Need inputs	Need inputs	Need inputs	Need inputs	Yes	Need inputs	Need inputs
QP: Cross Order Volume Period1	Yes	Yes	No	No	Blank	View Only	Req
QP: Cross Order Volume Period2	Yes	Yes	No	No	Blank	View Only	Req
QP: Cross Order Volume Period3	Yes	Yes	No	No	Blank	View Only	Req
QP: Custom Ignore Pricing	Yes	No	No	No	Blank	View Only	Opt
QP: Custom Sourced	Yes	No	No	No	No	View Only	Opt
QP: Debug	No	No	No	Yes	Request Viewer Off	Yes	Opt
QP: E-mail - From Address	Yes	No	No	Yes	--	Yes	Req
QP: Enable Auditing	Yes	No	No	No	No	Yes	Opt

<b>Profile Option</b>	<b>SA Site</b>	<b>SA App.</b>	<b>SA Resp.</b>	<b>SA User</b>	<b>Default Value</b>	<b>User</b>	<b>Req?</b>
QP: Enable Best Price Formula Evaluation	Yes	No	No	Yes	--	Yes	Opt
QP: External Default Pricing Perspective	Yes	No	No	Yes	Order Management	View Only	Req
QP: Exclude Qualifiers	Yes	No	No	No	Blank	Yes	Req
QP: Get Custom Price Customized	Yes	No	Yes	No	No	No	Opt
QP: Ignore Price List Date Validation	Need inputs	Need inputs	Need inputs	Need inputs	Need inputs	Need inputs	Need inputs
QP: Inbound XML Messaging Responsibility	No	Yes	No	No	--	Yes	--
QP: Insert Formula Step Values into Temp Table	Yes	No	No	No	No	No	Opt
QP: Internal Default Pricing Perspective	Yes	No	No	No	Order Management	View Only	Req
QP: Inventory Decimal Precision	Yes	Yes	Yes	Yes	10	Yes	Opt
QP: Item Validation Organization	Yes	No	Yes	No	Blank	No	Opt
QP: Limit Exceed Action	Yes	No	Yes	Yes	Hard - Adjust Benefit Amount	Yes	Opt
QP: Line Volume UOM Code	Yes	Yes	No	No	Blank	No	Req

<b>Profile Option</b>	<b>SA Site</b>	<b>SA App.</b>	<b>SA Resp.</b>	<b>SA User</b>	<b>Default Value</b>	<b>User</b>	<b>Req?</b>
QP: Line Weight UOM Code	Yes	Yes	No	No	Blank	No	Opt
QP: Licensed for Product	No	Yes	No	No	Blank	--	Req
QP: Modifier Find Window - Show records	Yes	Yes	Yes	Yes	No	Yes	
QP: Multi-Currency Installed	Yes	No	No	No	No	No	Opt
QP: Multi Currency Usage	No	Yes	Yes	No	Blank	No	Req
QP: Negative Pricing	Yes	Yes	No	No	No	No	Opt
QP: Pass Qualifiers to Get_Custom_Price API	Yes	No	No	No	No	No	Req
QP: Pattern Search	Yes	No	No	No	Blank	No	Opt
QP: Pattern Search Path	Yes	No	No	No	Blank	No	Opt
QP: Performance Control	Yes	No	No	No	None	No	Opt
QP: Price Rounding	Yes	No	No	No	Blank	No	Req
QP: Pricing Party Hierarchy Type	Yes	No	No	No	Blank	No	Opt

Profile Option	SA Site	SA App.	SA Resp.	SA User	Default Value	User	Req?
QP: Pricing Perspective Request Type	Yes	Yes	No	No	Order Management Order	No	Req
<p><b>Note:</b> For each application, there is a separate default value.</p>							
QP: Pricing Transaction Entity	Yes	Yes	No	Yes	Order Fulfillment	No	Opt
QP: Promotional Limits Installed	Yes	No	No	No	No	No	Opt
QP: Qualify Secondary Price Lists	Yes	No	No	No	No	Yes	Opt
QP: Return Manual Discounts	Yes	Yes	Yes	Yes	Yes	Yes	Opt
QP: Satisfied Qualifiers Option	Yes	Yes	Yes	Yes	Yes	Yes	Opt
QP: Security Control	Yes	No	No	No	Off	No	Opt
QP: Security Default Maintain Privilege	Yes	No	No	No	Global	No	Opt
QP: Security Default ViewOnly Privilege	Yes	No	No	No	Global	No	Opt
QP: Selling Price Rounding Options	Yes	No	Yes	Yes	Individual: = round(listprice) + round(adj)	No	Opt

Profile Option	SA Site	SA App.	SA Resp.	SA User	Default Value	User	Req?
QP: Set Request Name	Yes	Yes	Yes	Yes	Blank	Yes	Req
QP: Source System Code	Yes	Yes	No	Yes	Oracle Pricing	Yes	Opt
QP: Unit Price Precision Type	Yes	Yes	No	No	Standard	No	Opt
QP: Valueset Lookup Filter	Yes	Yes	Yes	Yes	Yes	Yes	Opt
QP: Verify GSA Violations	Yes	No	No	No	No	No	Opt

### Setting Up Non-Seeded Profiles

Apart from setting up values for the seeded profiles, you must:

- Create the non-seeded ONT\_DISPLAY\_QUALIFIED\_ASK\_FOR\_MODIFIERS profile
- Set the profile value as Yes, if you want to see only the eligible ask-for modifier in Oracle Order Management.

To create the non- seeded profile:

1. Navigate to System Administrator > Define Profile Options > open the Profile window.
2. Enter the following information:
  - Name: ONT\_DISPLAY\_QUALIFIED\_ASK\_FOR\_MODIFIERS
  - Application: Order Management
  - User Profile Name: OM: Display Qualified Ask For Modifiers
  - Description: Decides whether only eligible ask for modifiers should be displayed or all (in Promotions LOV in Promotion/Pricing Attributes block of Sales Order window).
  - SQL Validation:

```
SQL="SELECT LOOKUP_CODE, MEANING
INTO :PROFILE_OPTION_VALUE
, :VISIBLE_OPTION_VALUE
FROM OE_LOOKUPS
WHERE LOOKUP_TYPE = 'YES_NO' "
COLUMN="MEANING( * )"
```

### 3. Save

## Profile Options

You must set a value for profile options followed by the word "required," no default is supplied. Ordinary users can see profile options followed by the word "exposed," only system administrators can see the rest. Further details follow the list, click an item to find them.

- OM: Discounting Privilege
- OM: Display Qualified Ask for Modifiers
- OM: GSA Discount Violation Action
- OM: Negative Pricing
- OM: Promotion Limit Violation Action
- QP: Accrual UOM Class
- QP: Accumulation Attributes Enabled
- QP: Administer Public Queries
- QP: Allow Buckets for Manual Modifiers
- QP: Allow Duplicate Modifiers
- QP: Archive Header Qualifiers
- QP: Batch Size for Bulk Import
- QP: Blind Discount Option
- QP: Break UOM Proration Allowed
- QP: Build Attributes Mapping Options
- QP: Continuous Price Breaks
- QP: Cross Order Volume Period1
- QP: Cross Order Volume Period2
- QP: Cross Order Volume Period3
- QP: Custom Ignore Pricing
- QP: Custom Sourced
- QP: Debug

QP: E-mail - From Address  
QP: Enable Auditing  
QP: Enable Best Price Formula Evaluation  
QP: Exclude Qualifiers  
QP: External Default Pricing Perspective  
QP: Get Custom Price Customized  
QP: Ignore Price List Date Validation  
QP: Inbound XML Messaging Responsibility  
QP: Insert Formula Step Values into Temp Table  
QP: Internal Default Pricing Perspective  
QP: Inventory Decimal Precision  
QP: Item Validation Organization  
QP: Licensed for Product  
QP: Limit Exceed Action  
QP: Line Volume UOM Code  
QP: Line Weight UOM Code  
QP: Modifier Find Window - Show records  
QP: Multi-Currency Installed  
QP: Multi Currency Usage  
QP: Negative Pricing  
QP: Pass Qualifiers to Get\_Custom\_Price API  
QP: Pattern Search  
QP: Pattern Search Path  
QP: Performance Control  
QP: Price Rounding  
QP: Pricing Party Hierarchy Type  
QP: Pricing Perspective Request Type  
QP: Pricing Transaction Entity  
QP: Promotional Limits Installed  
QP: Qualify Secondary Price Lists  
QP: Return Manual Discounts  
QP: Satisfied Qualifiers Option

QP: Security Control  
QP: Security Default Maintain Privilege  
QP: Security Default ViewOnly Privilege  
QP: Selling Price Rounding Options  
QP: Set Request Name  
QP: Source System Code  
QP: Unit Price Precision Type  
QP: Valueset Lookup Filter  
QP: Verify GSA Violations

### **OM: Discounting Privilege**

This profile option determines control of a user's ability to apply discounts to an order or order line.

#### **Values**

- None: Users cannot apply any manual discounts.
- Full: Ability to apply any valid discount against an order or order line, as long as the order type of the order does not enforce list prices.
- Non-overridable only: Ability to apply only non-overridable discounts against an order or order line.
- Unlimited: Ability to apply any valid discount against any order or order line, regardless of whether the order type of the order enforces list prices.

The default value is Full.

### **OM: Display Qualified Ask for Modifiers**

Create and set Yes as the value for this profile to avail the functionality of getting only eligible ask-for modifiers in Oracle Order Management.

This profile enables you to obtain only eligible ask-for modifiers qualified for the sourced attributes in Oracle Order Management. When you set this profile to Yes, Order Management displays the eligible ask-for modifiers; else the application retains the original behavior of listing all the modifiers. When this profile is set Yes, the application makes a Pricing engine call to retrieve all the eligible ask-for modifiers based on the sourced attributes. This enables you to view all the eligible modifiers when you click on the Promotion list of values in the Promotions/Pricing Attributes window (Actions from Sales Order window in Oracle Order Management). See: Setting up Non-Seeded Profiles

### **OM: GSA Discount Violation Action**

This profile option determines the user is notified when you define a discount that results in GSA violation.

This profile must be set as Yes if you want order entry personnel to receive a GSA violation warning message, and if the QP: Verify GSA is set to Yes.

### **OM: Negative Pricing**

This profile option determines whether or not a negative list price or selling price can be entered on an order. Select either Yes or No. For example: a trade-in item may be entered on an order with the trade-in value as a negative value.

### **OM: Promotion Limit Violation Action**

Determines the hold action Order Management will take when encountering a initial promotional hold returned by the pricing engine. If the pricing engine returns a possible promotional hold for an order or order line, Order Management will use the value of this profile option to determine the course of action for the order or line. Messages are generated and can be viewed within the Process Messages Window.

Select from:

- Place holds where violated (either Line or Order): If the pricing engine returns an initial promotional hold, place a hold for the corresponding order or order line.
- No holds applied: If the pricing engine returns an initial promotional hold, do not apply a hold for either an order or order line. Allow the order or order line to continue processing within its associated workflow.
- Place order on hold when any violation occurs (both Line and Order): If the pricing engine returns an initial promotional hold, irrespective of the hold level, place the order on hold, in addition to any order lines that may be marked for promotional hold.

This profile option is optional.

The default value is Null.

Ordinary users can't see this profile option.

System administrators can see this profile option at the Site level.

### **QP: Accrual UOM Class**

This is required if your business gives non-monetary accruals as benefits.

Specifies the unit of measure class to be used for defining accrual units of measure. The Modifier Setup window displays all units of measure in this class when entering the Benefit UOM for an accrual.

**Values**

All UOM classes defined to Oracle Applications.

This profile option is visible and can be updated at site and application levels.

The default value is Null.

**QP: Accumulation Attributes Enabled**

This profile option enables the use of accumulation attributes for modifiers. The value of the accumulation attribute is sourced from the pricing request when evaluating accumulated range price breaks for modifiers. To use the accumulation attribute feature, the profile QP: Accumulation Attributes Enabled must be set to Yes.

**Note:** Ensure that you set this profile to Yes at the application level only for applications that support the Accumulation attributes feature. Else, you must set this profile to No.

**Values**

- Yes: Enables the use of accumulation attributes for modifiers. The profile option also enables the Accumulation Attribute field in the Price Breaks tab on the Define Modifier window.
- No: Cannot use the accumulation attribute feature. The Accumulation Attribute field in the Price Breaks tab does not display in the Define Modifier window.

This profile option can be set at the Site and/or Application level.

The default value is No.

**QP: Administer Public Queries**

Enables a query saved in the Public folders to be deleted or renamed (when value of the profile QP: Administer Public Queries is set to Yes).

**Values**

- Yes: Enables queries saved in Public folders to be deleted or renamed.
- No: Queries saved in Public folders cannot be deleted or renamed.

The default value is Null.

**QP: Allow Buckets for Manual Modifiers**

This profile option enables you to define buckets for manual line or group of line level modifiers

**Values**

- Yes: Enables you to define buckets for manual line or group of line level modifiers.

- No: You cannot define buckets for manual line or group of line level modifiers.

If the profile value is changed from Yes to No and there are manual modifiers defined with buckets, the manual modifiers defined with buckets need to be end-dated or deactivated. The pricing engine will return the manual bucketed adjustment even if the profile is set to No.

This profile option is set at the site level.

The default value is No.

### **QP: Allow Duplicate Modifiers**

Used by Basic Pricing Only. The profile option QP: Allow Duplicate Modifiers, which is typically set by the System Administrator, determines if duplicate modifiers are permitted. If set to Yes (the default), an existing modifier can be duplicated. If set to No, you must change the new modifier line before you can save it. A modifier line is considered a duplicate if any of the following attributes of the original and duplicated line match within the same modifier list:

- List Line Start Date Active
- List Line End Date Active (Lines with overlapping dates are considered duplicates)
- Modifier Level Code (Order/Line)
- Automatic Flag (Selected/Cleared)
- Product UOM code
- Product Attribute
- Product Attribute Value
- Pricing Attributes
- Set of Qualifiers

#### **Values**

- Yes: Duplicate modifiers can be copied within the same modifier list for Basic Pricing.
- No: Duplicate modifiers cannot be copied within the same modifier list for Basic Pricing.

This profile option is visible and can be updated at the site level.

The default value is Yes.

### QP: Archive Header Qualifiers

Use this profile option to archive header-level qualifiers of the price lists. When you archive inactive pricing entities, the application deletes the header-level qualifiers based on the value of the profile option.

#### Values

Yes: Select Yes to archive header-level qualifiers of the pricing entities.

No: Select No if you do not want to archive the header level qualifiers of the pricing entities.

The default value is No..

### QP: Batch Size for Bulk Import

This profile option value determines the number of records loaded into the memory for bulk import processing. Setting an appropriate value for this profile based on hardware configuration improves performance; however, the system may "hang" if the profile value is set too high.

The default value is 1000.

### QP: Blind Discount Option

The default value for this profile option should only be changed if you never define blind discounts. If you never define blind discounts, set this profile option to No; this bypasses part of the search engine processing. A blind discount is defined as a modifier that has all of the following:

- No list qualifiers on the modifier list header
- No line qualifiers on the modifier
- No products or pricing attributes

**Note:** If your business must define a modifier as previously described, verify that this profile option is set to Yes. If this is not done, the modifiers will not be selected by the search engine.

#### Values

- Yes: Blind discounts are enabled.
- No: Blind discounts are disabled; bypass blind discount processing in search engine.

This profile option is visible and can be updated at the site and application levels.

The default value is Yes.

### QP: Break UOM Proration Allowed

This profile option controls whether break ranges for price lists need to be prorated or not. This profile is useful for service usage pricing engine calls.

#### Values

- Yes: The pricing engine prorates the break ranges before evaluating which break is satisfied.
- No: The pricing engine does not prorate the break ranges before evaluating which break is satisfied.

This profile option can be updated at the site and application levels.

The default value is Blank.

### QP: Build Attributes Mapping Options

This profile option enables you to set attribute mapping rules for attributes in both the active and inactive setups.

#### Values

- Map attributes used in active pricing setup: The Build Attribute Mapping Rules program will map the attributes that are used only in the active pricing setup.
- Map all attributes: This program will source the attributes that are used in both the active and inactive setups.

The default value is No.

### QP: Continuous Price Breaks

This profile option enables you to disable the continuous price break functionality. If you do not set the profile value, then the application uses the continuous price break functionality.

#### Example

##### Values

- Yes: Oracle Advanced Pricing allows continuous price breaks for the newly created price lists or modifiers and automatically upgrades the non-continuous price break to continuous price break during copy price list functionality.
- No: Oracle Advanced Pricing allows non-continuous price breaks for the newly created price lists or modifiers and copy price list functionality retains the price lists as it is. All the newly created price lists or modifiers allow non-continuous price breaks.

**Warning:** Ensure that changing the default value of the profile is based

on correct business needs as switching the profile back and forth may lead to unexpected results.

### **QP: Cross Order Volume Period1**

This is required if you will be running the cross order volume load program. This defines the number of days of order lines that the load program will accumulate and total. This value must not be the same as the value in QP: Cross Order Volume Period 2 or QP: Cross Order Volume Period 3.

#### **Values**

Always expressed in days.

This profile option is visible and can be updated at the site and application level.

The default value is Blank.

### **QP: Cross Order Volume Period2**

This is required if you will be running the cross order volume load program. This defines the number of days of order lines that the load program will accumulate and total. This value must not be the same as the value in QP: Cross Order Volume Period 1 or QP: Cross Order Volume Period 3.

#### **Values**

Always expressed in days.

This profile option is visible and can be updated at the site level.

The default value is Blank.

### **QP: Cross Order Volume Period3**

This is required if you run the cross order volume load program. This defines the number of days of order lines that the load program will accumulate and total. This value must not be the same as the value in QP: Cross Order Volume Period 1 or QP: Cross Order Volume Period 2.

#### **Values**

This value is always expressed in days.

This profile option is visible and can be updated at the site level.

The default value is Blank.

### **QP: Custom Ignore Pricing**

This profile option provides users an opportunity to inform the pricing engine to ignore lines that have no significance from pricing perspective. These lines can be marketing or manufacturing option items whose lines are sourced to pricing for processing. After considerable unnecessary processing, the pricing engine fetches a value of zero for these

lines, which can be avoided. Due to this unnecessary processing, these lines are an extra burden on pricing and cause performance issues.

Using this profile option, you can save significant amount of time especially in case of bigger orders by implementing the QP\_CUSTOM\_IGNORE.IGNORE\_ITEMLINE\_FOR\_PRICING custom hook API. If you set the profile to Yes, then the application calls the custom API before processing the line in price engine. You must write logic that can identify that the incoming line is a zero priced line.

## QP: Custom Sourced

### Values

- **Yes:** When this profile is set to Yes, the Build Contexts program builds the contexts from the dynamic package generated by the Build Attribute Mapping Rules program and from the custom package created by the customers.
- **No:** When this profile is set to No, the Build Contexts program builds the contexts only from the dynamic package generated by the Build Attribute Mapping Rules program.

The default value is No.

## QP: Debug

This profile option enables you to set the Request Viewer and how it captures request details into the pricing debug tables and debug log information into the debug log table.

### Values

- **Request Viewer Off, Generate List Line Details for Debug:** If the QP: Debug profile value is set to "Request Viewer Off, Generate List Line Details for Debug," then a menu option "Debug: Generate List Line Details" displays on the Tools menu in the pricing setup windows (price list, modifier, and pricing agreements windows). To collect list line details, click Debug: Generate List Line Details from the list line record for which the line details needs to be generated. A message will confirm that the debug information for the list line is being generated through a concurrent request.
- **Request Viewer Off:** When off, nothing is written into pricing debug tables and debug log table. The debug log text file will not be created.
- **Request Viewer Off, show Diagnostic details in Trace:** Some pricing timings sql are controlled by this value. When set to "Request Viewer Off, show Diagnostic details in Trace," the pricing timing sqls are executed. If not set, the timings sqls are not executed.
- **Request Viewer On, but Debug Log is not visible in Viewer:** When this is set, the Request Viewer captures pricing request details into the pricing debug tables, but debug log information is not written into the debug log table. The debug log text

file will be created.

- **Request Viewer On:** When on, the Request Viewer captures pricing request details into the pricing debug tables and debug log information into the debug log table. The debug log text file is also created.

This profile option can be updated at the user level and is active for the transactions of the user who set this profile option—other users' transactions are not affected.

**Note:** The profile option, QP: Set Request Name can be used in conjunction with the QP: Debug profile option. When the QP: Set Request Name is set to Yes, the Request Name field will be prefixed with the Order ID.

The default value is Request Viewer Off.

### QP: E-mail - From Address

This profile option defines the sender's e-mail address when a price book is e-mailed from the price book feature.

You need to set up the profile option QP: E-mail - From Address to specify the *From Email Address* for any e-mails that are sent from the price book feature. When the price book is e-mailed, the recipient can identify the sender and respond to the sender's return e-mail address. The default return e-mail address is used only when e-mailing from the price book feature not from a different e-mail application such as Outlook.

**Note:** To send the price book via e-mail, select the Send Email option from the Create Price Book page.

To set the return e-mail address, enter the return e-mail address such as joe.smith@vision.com in the profile option QP: E-mail - From Address. This serves as the return address when a price book is e-mailed from the price book feature.

**Important:** Publishing a price book to a printer or as an e-mail attachment is dependent on the availability of the Delivery Manager Component of the Oracle XML Publisher (XDO) product.

### Values

You can enter any valid e-mail address. This profile option is visible and can be updated at the site and user level.

The default value is Blank.

### QP: Enable Auditing

Use this profile option to audit the pricing tables in Oracle Advanced Pricing. Set the

profile option to **Yes** to enable auditing at the site level. Set the profile option to **No** to disable auditing.

After you set the profile option, run the *QP: Audit Pricing Tables* process to enable the triggers for the tables that need to be audited. See *QP: Audit Pricing Tables* in *Oracle Advanced Pricing User's Guide*.

The default value is No.

### **QP: Enable Best Price Formula Evaluation**

Use this profile option to evaluate the Best Price functionality for modifiers with attached formula. Set the value to Yes to enable the pricing engine to evaluate the price for modifiers having formulas attached when Precedence is set to Best Price for the phase in which the modifier is defined. For setting up Best Price as Incompatibility Resolve code, refer to the Precedence and Best Price chapter in the *Oracle Advanced Pricing Implementation Guide*.

### **QP: Exclude Qualifiers**

The profile option determines the evaluation of the exclusive qualifiers for multi-value attributes. When you set this value to Yes, the Exclude button is enabled on the Qualifier window.

#### **Example**

- Yes - Set the option to Yes to enable the exclusive qualifiers setup to evaluate the multi-value attributes.
- No - Set the option to No if you do not want to enable the exclusive qualifiers setup to evaluate the multi-value attributes.

The default value is Blank.

### **QP: External Default Pricing Perspective**

This profile option is used for the price book feature and sets the default pricing perspective for all external users. This value defaults to the Pricing Perspective field in the Create Price Book page. However, an internal user can override the defaulted value by selecting a different pricing perspective.

**Note:** External users cannot view or change the Pricing Perspective field. The value Purchasing is valid for internal users but not external users.

The default value is Order Management.

### **QP: Get Custom Price Customized**

This profile option indicates, when processing formulas, that the pricing engine

evaluates the line type function. If your organization wants to use this formula line type, you must:

- Customize the GET\_CUSTOM\_PRICE function.
- Set this profile option to Yes.

#### **Values**

- Yes: When processing formulas, the pricing engine evaluates the line type function. This profile option must be set as Yes if the package body for QP\_CUSTOM is coded.
- No: When processing formulas, the pricing engine does not evaluate the line type function.

This profile option can only be updated at the site level. This profile is required if your business needs more pricing formulas than are provided in basic pricing.

A pricing formula consists of a formula expression (mathematical expression) consisting of step-numbers and formula lines that correspond to each step-number in the formula expression. Each formula line associates with a formula line type. There are six formula line types in Oracle Advanced Pricing (three in basic pricing). One type is function.

Oracle Advanced Pricing provides a function called Get\_Custom\_Price with a standard set of input parameters. The user is provided the flexibility of writing any custom code in the function body and use the input parameters supplied by the pricing engine. The value returned by the Get\_Custom\_Price function can be used in a formula expression.

To use the customized Get\_Custom\_Price function's return value in a formula, the user must set up a formula with a line type of function (the formula may or may not have other formula lines).

The Get\_Custom\_Price function can be used in any number of formulas at the same time because the formula ID is an input parameter to the function and assists the user differentiate code logic.

The default value is No.

#### **QP: Ignore Price List Date Validation**

This profile option enables the application to ignore the price list date validation and enables you to update the lines and header details of the price lists created using the Copy Price List feature. To update the price list lines and header details for price lists that are created using the "Copy Price List" feature, you must set Y as the value for this profile option.

The default value is No..

#### **QP: Inbound XML Messaging Responsibility**

This profile option defines the responsibility (at the Application or User level) for

inbound XML messages. It can be set only at the System Administrator (SA) Application level and by the user. This profile option is used by the price book feature to set the responsibility context for an inbound XML message request for a price book; for example, a Get Catalog XML message.

**Note:** The Application corresponds to the pricing perspective of the price book request.

The default value is Blank.

### QP: Insert Formula Step Values into Temp Table

This profile option can be set only at the site level using the System Administrator responsibility.

#### Values

- Yes: If set to Yes, the step values of each formula attached to a price list line is evaluated by the Pricing Engine and inserted into the temporary table QP\_FORMULA\_STEP\_VALUES\_TMP. This can be referenced by customized code.
- No: If set to No, the step values are not inserted into the temporary table described above.

The default value is No.

### QP: Internal Default Pricing Perspective

Sets the default value for the pricing perspective for all internal users using the price book feature. These values can be applications such as Oracle Order Management and Oracle Purchasing that request information from the pricing application. This value defaults to the Pricing Perspective field in the Create Price Book page; however, an internal user can override this value by selecting a different pricing perspective. An external user cannot view the Pricing Perspective field or change its value.

**Note:** Oracle Advanced Pricing is not one of the valid values for this profile option.

The default value is Order Management.

### QP: Inventory Decimal Precision

Used to set maximum decimal precision for uom conversion when calculating pricing quantity. If not set, it is defaulted to 10 digits decimal precision.

#### Example 1

Consider the following set up:

- Primary UOM = YR (year)

- Order UOM = MTH (month)
- Order Quantity = 12

The pricing engine rounds the pricing quantity based on the decimal precision setting. If the default precision is 10 digits, then the resulting pricing quantity will be  $12 * (1/12) = 0.9999999999\dots$  the number will be rounded to 1YR.

### Example 2

Consider the following set up:

- Primary UOM = DZ
- Order UOM = EA
- Order Quantity = 16

The pricing engine rounds the pricing quantity based on the decimal precision setting. If a user sets the Profile QP: Inventory Decimal Precision to 6 digits, then the resulting pricing quantity is calculated as follows:  $16 * (1/12) = 1.333333333333\dots$  which is rounded to 1.333333 DZ.

The default value is 10.

## QP: Item Validation Organization

Set this profile, by site or responsibility, to an organization at the level in your organization hierarchy at which you set prices for items.

### Values

This profile option is visible and can be updated at the site and responsibility levels.

Before setting this profile option, you need to set up values for:

- HR: Security profile
- HR: Business Group profile options

Valid inventory master organizations will be available based on values of HRMS profile settings. For more information on these profiles, see: *Configuring, Reporting and System Administration in Oracle HRMS, Security* chapter.

**Note:** For more information on multiple organizations, see the *Multiple Organizations* chapter in this guide.

The default value is None.

## QP: Licensed for Product

This profile option identifies which Oracle software application can use Oracle Pricing. The value can be set after you buy the license to use Oracle Pricing along with other

Oracle Applications. For example, to use Oracle Pricing for Oracle Procurement applications, you must be licensed to use Oracle Pricing and set the value of this profile to PO (Oracle Procurement).

#### **Values**

The value of this profile need to be set to the Oracle Application which is planning to use Oracle Pricing.

This profile option can be updated at the application and user levels.

The default value is Blank.

### **QP: Limit Exceed Action**

This profile option defines the default action codes for promotion and modifier limits. It specifies the action for the pricing engine if a pricing request exceeds a promotional limit.

This profile option is based on the lookup type Limit Exceed Action.

#### **Values**

- **Soft--Full Benefit Amount:** Applies the full benefit to the order even if the transaction exceeds the defined limit.
- **Hard--Adjust Benefit Amount:** Adjusts the order benefit amount so that the order meets but does not exceed the promotional limit. A status message is sent to the calling application such as Order Management to place a promotional hold on the order.

This profile option is visible and can be updated site level using the System Administrator responsibility.

The default value is Hard--Adjust Benefit Amount.

### **QP: Line Volume UOM Code**

Required if your business must define qualifier rules which include the seeded qualifier line volume.

This profile option specifies the unit of measure of the line volume qualifier. The attribute sourcing API converts the item on the request line to its primary UOM, then uses the volume attributes of the item to derive the line volume of the item in the UOM specified in the profile option.

Order Volume:

The Order Volume qualifier calculates the total volume of all the order lines for the order. In order to use this qualifier the following must be set up:

1. The profile, QP: Line Volume UOM Code, must be set to the correct volume uom code. This uom code is the uom for the total order volume.
2. There must be a uom conversion set up to convert from the ordered\_uom for each

order line to the volume uom specified in the profile. For example if line 1 of your order is in Ea and the profile is set to CBM, there must be a conversion set up from Ea to CBM for the item.

The calculation of Order Volume:

Each order line will be converted into the line volume (order\_quantity \* uom\_conversion\_rate) and the line volumes will be totaled to get the Order Volume.

**Values**

All units of measure currently defined to Oracle.

This profile option is visible and can be updated at the site and application levels.

The default value is Blank.

**QP: Line Weight UOM Code**

This profile option specifies the unit of measure for the line weight qualifier. This is required if your business needs to define qualifier rules which includes the seeded qualifier line weight.

The attribute sourcing API converts the item on the request line to its primary UOM, and then uses the weight attributes of the item to derive the line weight of the item in the UOM specified in this profile option.

The Order Weight qualifier is calculated as the total weight of all the order lines for the order. In order to use this qualifier the following must be set up:

1. The profile QP: Line Weight UOM Code must be set to the correct weight uom code. This uom code is the uom for the total order weight.
2. A uom conversion must be set up to convert from the ordered\_uom for each order line to the weight uom specified in the profile. For example, if line 1 of your order is in KGM and the profile is set to LBS, there must be a conversion set up from KGM to LBS for the item.

How order weight is calculated:

Each order line will be converted into the line weight (order\_quantity \* uom\_conversion\_rate) and the line weights will be totaled to get the Order Weight.

This is similar to the Line Weight qualifier setup.

**Values**

All units of measure currently defined to Oracle.

This profile option is visible and can be updated at the site and application levels.

The default value is Blank.

**QP: Modifier Find Window - Show records**

This profile option enables you to use a Find window in the Modifiers window to query

modifier records. However, typically, you would use the Pricing Organizer feature to find modifiers and modifier information.

**Values**

- Yes: Enables you to use modifier Find window.
- No: When set to No, the modifier Find Window does not display. Instead, the Pricing Organizer window displays.

The default value is No.

**QP: Multi-Currency Installed**

If you have global customers or do pricing in different currencies, the multicurrency feature enables you to maintain a single price list for multiple currencies.

Once the profile option is set to Yes, the concurrent program Update Price Lists with Multi-Currency Conversion Criteria must be run to enable the price list windows for multicurrency usage.

**Note:** Once the concurrent program has been run successfully, all existing price list and agreement windows are converted to multicurrency price lists. Users should not return to NON multicurrency price lists. Changing the profile option back to No may cause undesired results if conversion criteria have been used. Oracle does not support changing the setting back to No.

The program must be run initially only once to activate the multicurrency feature; otherwise, data corruption may result.

**Values**

- Yes: If the profile QP: Multi-Currency-Installed is Yes, the incoming pricing request will send order currency, pricing date, product attributes, pricing attributes and qualifier attributes. The pricing engine matches the order currency with the price list's Base Currency or Conversion Criteria To-Currency
- No: When the profile QP: Multi-Currency-Installed is No, the order currency is matched with the price list base currency, which is the current behavior.

This profile option is visible and can be updated at the site and application level.

The default value is No.

**QP: Multi Currency Usage**

This profile option determines if the calling application can use multicurrency price lists.

**Values**

- Yes: Enables an application to use multicurrency price lists.

- No: Do not use multi currency price lists.

This profile option can be updated at the site, responsibility and application levels.

The default value is Blank.

### **QP: Negative Pricing**

The default value should only be changed if your business needs to define a negative price on a price list line. Controls whether a negative price can be entered in the Price List setup window.

#### **Values**

- Yes: Allows a negative price to be entered.
- No: Does not allow a negative price to be entered.

This profile option is visible and can be updated at the site and application levels.

The default value is No.

### **QP: Pass Qualifiers to Get\_Custom\_Price API**

#### **Values**

- Yes: If selected, then only the qualifiers are passed to Get\_Custom\_Price.
- No: If selected, qualifiers will not be passed to Get\_Custom\_Price.

The default value is Blank.

### **QP: Pattern Search**

You must set this profile option at the site level to turn on the pattern search engine.

You can set this profile to any of the following values:

- Pattern Search Off - To turn off the pattern search engine.
- Modifier - To turn on the pattern search engine for modifiers search only.
- Price List - To turn on the pattern search engine for price lists search only.
- Both Modifier and Price List - To turn on the pattern search engine for both price lists and modifiers.

### **QP: Pattern Search Path**

This profile directs the search path during the search. Setting a value for this profile is not mandatory. The default value is Search Lines First. You need to run few common pricing flows and decide which is the best search path for your pricing setup.

You can set this profile to any of the following values:

- Search Lines First – Search engine evaluates the lines first and then the headers.
- Search Header First - Search engine evaluates the headers first and then the lines.

### QP: Performance Control

This profile resolves performance issues while updating formulas having huge factor lists in which the expensive query is updating the pricing attributes. If you set the value to Yes, then the application defers updating pricing attributes. This makes the update process of the formula fast, resulting in better performance.

### QP: Price Rounding

This profile option controls how the value for the rounding factor is derived and used in price lists and related windows. The Advanced Pricing - Price Lists window rounds, stores, and displays the list price based on the profile option setting.

The value entered in the Round To field from the price list is used to store the rounded value, while currency precision determines the displayed list price.

For example, if the Round To value is -2 and the currency precision is -5, the following list prices display:

115.24000

9.23000

100.00000

If the QP: Price Rounding profile option is set to Enforce Currency Precision, then the value in the Round To field in the Advanced Pricing - Price Lists window cannot be updated. Also, the values permitted for the rounding factor will be limited to the price list currency precision.

**Warning:** If the QP: Price Rounding profile option is set to Enforce Currency Precision, and you enter a list price greater than the precision amount allows, you will be unable to save the price list. An error message only displays when you try to save the price list, not when you first enter the list price.

The following table shows how different settings for QP: Price Rounding affect the list price. Assume that the price list price = 6.15 and the markup change = 1.52% resulting in 6.24348.

If QP: Price Rounding value is: (see right)	Blank (Default)	Enforce Price List Rounding Factor	Enforce Currency Precision
Value in Round To field on Price List (see right for values)	-2	-2	-4
List Price	6.15	6.15	6.15
Mark up	1.52 %	1.52 %	1.52 %
List Price (new)	6.24348	6.24	6.2435

### Values

- Blank (Default): If this option is selected, the following occurs:
  - No limit is imposed on the number of places that can be entered on the price list after the decimal point.
  - The value for a price list line is not rounded.
  - The list price that displays will not be rounded by either Currency Precision or the Round To value.
  - Static formula calculation results will not be rounded.
  - The Round To value specified for the price list rounds the pricing engine results.
- Enforce Price List Rounding Factor: If this option is selected, the value entered in the Round To field of the Advanced Pricing - Price Lists window is used for:
  - Rounding the value on the price list line.
  - Rounding the pricing engine calculation results.
  - Calculating the results for static formula calculations.

**Note:** The currency precision setting determines the display of the list price.
- Enforce Currency Precision: If selected, the Rounding Factor field on the price list cannot be updated by the user. Instead the Rounding Factor value defaults from the

profile QP: Unit Price Precision Type (either Standard/Extended precision) for the price list currency. The decimal places that display for the list price is determined by the Currency Precision.

**Note:** Formula Prices: For dynamic formulas, the calling application passes the rounding factor and the resulting rounding factor displays regardless of the profile setting.

### **Rounding Behavior**

The profile option settings for QP: Price Rounding affect the rounding behavior as described below:

#### **Formula Prices**

For dynamic formulas, the calling application passes the rounding factor and the resulting rounding factor displays regardless of the profile setting.

#### **Price List**

- The Round To value is used to round and store the list price if the profile option QP: Price Rounding is set to Enforce Rounding Factor or Enforce Currency Precision.
- Currency Precision is used to display the list price if profile option QP: Price Rounding is set to Enforce Rounding Factor or Enforce Currency Precision.
- The Round To can not be modified if profile option QP: Price Rounding is set to Enforce Currency Precision.

#### **Adjust Price List**

The list price, after adjustment by amount or percent, will be rounded and stored using Rounding Factor if the profile option QP: Price Rounding is set to Enforce Rounding Factor or Enforce Currency Precision.

#### **Add Items to Price List**

The list price will be rounded and stored using Rounding Factor if the profile option QP: Price Rounding is set to Enforce Rounding Factor or Enforce Currency Precision and Set List Price Equal to Cost From is checked on the window.

#### **Update Formula Prices**

The list price will be rounded and stored using Round To if the profile option QP: Price Rounding is set to Enforce Rounding Factor or Enforce Currency Precision.

#### **Update Formula Prices**

The list price will be rounded and stored using Round To if the profile option QP: Price Rounding is set to Enforce Rounding Factor or Enforce Currency Precision.

#### **Agreement**

The Round To is used to round and store the list price if the profile option QP: Price Rounding is set to Enforce Rounding Factor or Enforce Currency Precision.

- Currency Precision displays the list price if the profile option QP: Price Rounding is set to Enforce Rounding Factor or Enforce Currency Precision.
- The Round To cannot be modified if the profile option QP: Price Rounding is set to Enforce Currency Precision.

The default value is Blank.

### **QP: Pricing Party Hierarchy Type**

This profile option specifies the hierarchical party relationship type from Trading Community Architecture (TCA) to be used in pricing. A value must be assigned to this profile before you can select the Party Hierarchy Enabled check box for TCA Party qualifier attributes. The following is a list of seeded TCA Party attributes:

- Customer Party (qualifier context = Party Information (ASOPARTYINFO))
- Party Id (qualifier context = Customer (CUSTOMER))
- Buying Groups (qualifier context = Customer Group (CUSTOMER GROUP))
- Supplier (qualifier context = Party Information (PARTY))
- Buyer (qualifier context = Party Information (PARTY))

It can be set at the Site level.

The default value is None.

### **QP: Pricing Perspective Request Type**

This profile option is used for the price book feature and defines the mapping between the Pricing Perspective and Request Type. It can be set at SA Site and SA Application levels. Default at the site level is Order Management Order. Each relevant (valid pricing perspective) application level has a different default.

#### **Values**

The following lists the values for the request type to pricing perspective mappings:

- Complex Maintenance Repair and Overhaul: Complex CMRO
- Order Capture: Order Capture
- Transportation Execution: Oracle Transportation Shipment
- Inventory: Inter Company Invoicing
- Demand Planning: Demand Planning

- Contracts Core: Oracle Contracts
- Service Contracts: Oracle Contracts for Service
- Order Management: Order Management Order
- Purchasing: Purchase Order

The default value is Order Management Order.

## QP: Pricing Transaction Entity

This profile option indicates the current Pricing Transaction Entity (PTE) in use. Only those contexts and attributes assigned to the current Pricing Transaction Entity will be available in the list of values on the setup forms. Likewise, querying up setup data for price list lines, modifiers, and qualifiers, etc. will cause the description to be shown only for those contexts and attributes that are assigned to the current Pricing Transaction Entity. Therefore it is important to set this profile option to the correct value before creating or querying any setup data in the Pricing Application.

It is not a good idea to change this profile often as you will see the other context-attributes combinations for a different PTE when querying on the pricing setup forms. If this happens, you will see the internal ID code.

This profile can be set at the site, application, and user levels.

### Values

The valid values for this profile are all the Pricing Transaction Entities that belong to QP Lookup type QP\_PTE\_TYPE.

The seeded value of this profile for various applications is shown in the table below.

Application name	Default Value
Site Level Profile Value	Order Fulfillment (ORDFUL)
Oracle Transportation Execution (FTE)	Logistics (LOGSTX)
Oracle Inventory (INV)	Intercompany Transaction (INTCOM)
Oracle Demand Planning (MSD)	Demand Planning (DEMAND)
All other Applications	Defaulted from Site

The default value is Order Fulfillment.

### QP: Promotional Limits Installed

This profile option activates the promotional limits feature in Oracle Advanced Pricing to enable users to manage promotional limits and related functions. The initial default value is *N* for No which means the limit feature is not active. The System Administrator must change this value to *Y* for Yes to be able to use limits. Only the System Administrator can change the value of this profile at site and application levels only. Other users can view the profile only.

#### Values

- Yes: Enables the promotional limits features to be used.
- No: Disables the promotional limits features.

**Note:** Once the QP: Promotional Limits Installed profile option is enabled, leave the promotional limits active with the value of *Y*. Do not disable the QP: Promotional Limits Installed profile option once it has been enabled!

The default value is No.

### QP: Qualify Secondary Price Lists

This profile option enables the pricing engine to use the qualifier rules defined for the secondary price list. For example, suppose you have distributors who can buy only certain product families. If you do not use qualifiers on secondary price lists, then you must create a price list for each possible combination of product families and distributors. If you use the QP: Qualify Secondary Price Lists option, then you can have a single parent price list with multiple secondary price lists, each for a product family with a qualifier that identifies the distributor by name or type.

**Note:** This profile option is set typically by the System Administrator.

#### Values

- Yes: If you select Yes, the pricing engine evaluates qualifiers for the secondary price list if the item's price is not found on the primary price list. Select Yes only if you plan to use qualifiers on your secondary price lists.
- No: If you select No, the pricing engine *does not* evaluate qualifiers for the secondary price list if the item's price is not found on the primary price list. If you select this value, do not set up qualifiers on secondary price lists in your pricing implementation.

The default value is No.

## QP: Return Manual Discounts

A modifier can be applied to the order automatically at the time of pricing or can be returned to the calling application (Oracle Order Management) and stored in price adjustments table, so that users can choose adjustments to be applied on the order line.

The following modifiers types can be manual:

- Discounts
- Surcharges
- Price break lines
- Freight Charges

Manual modifiers can be set at all levels (line, group of lines, and order); manual line/group of line level modifiers can be used with modifier buckets.

### Manual Discount Flag in the Pricing Engine Request Viewer window

In the Pricing Engine Request Viewer window, the Manual Discount Flag check box reflects the setting of the profile QP: Return Manual Discounts. If the Manual Discount Flag box is selected, then the QP: Return Manual Discounts is set to Yes. The profile option settings determine which modifier is evaluated by the pricing engine. For example, assume the following setup:

Modifier Name	Incompatibility Level	Precedence
Manual_1	Incompatibility 1	100
Manual_2	Incompatibility 1	200

When QP: Return Manual Discounts = Yes, then the list of values (LOV) will show both Manual\_1 and Manual 2.

When QP: Return Manual Discounts = No, then the LOV will show Manual\_1 since it has the highest precedence (determined by the lower precedence number).

### Values

- Yes: In basic pricing, Y is the default and should not be changed. This means that one automatic discount is applied, and all manual discounts are returned for user selection.

All the automatic discounts that are deleted as part of incompatibility processing return as manual discounts available for user selection All unapplied manual discounts are returned and all automatic discounts not considered are returned as manual discount.

- No: All automatic and manual discounts run through incompatibility processing and one from each incompatibility group is returned. An automatic discount may be deleted and a manual discount may be selected. Discounts (automatic or manual) deleted as part of incompatibility processing are not returned as manual discounts.

The Pricing Engine does not consider applied manual modifiers during incompatibility processing. Discounts (automatic or manual) deleted as part of incompatibility processing will not be returned as manual discounts.

The default value is No.

### QP: Satisfied Qualifiers Option

The profile option QP: Satisfied Qualifiers Option impacts performance when entering and booking an order. It controls whether satisfied qualifiers are returned to the calling application or not.

#### Values

- Yes: The pricing engine returns all the satisfied qualifiers to the calling application. This increases pricing engine processing time.
- No: Processing time is reduced because the pricing engine does not return the satisfied qualifiers to the calling applications.

The default value is No.

### QP: Security Control

This profile option is view-only and displays the current status of pricing security for your entire installation--either On or Off. You can use only the Security Control concurrent program to change the security setting. Before using this concurrent program, ensure that you have completed all the set up and implementation steps. For more information, see: Pricing Security, *Oracle Advanced Pricing Implementation Guide*.

#### Values

- Off: Indicates that pricing security is turned on for the entire installation. After upgrading to the pricing security feature, but prior to turning security on, the pre-upgrade functionality is maintained. This means that any user with functional access to the Pricing Manager responsibility has full access to maintain and update all pricing entities regardless of operating unit.
- On: After pricing security is turned on, then the profile option displays as On.

**Warning:** Pricing security should not be turned on until all mapping has been completed. All functional users require access privileges to view or maintain their pricing entities.

The default value is Off.

### QP: Security Default Maintain Privilege

This profile option controls the default maintain privileges for NEWLY CREATED price lists and modifiers after security is turned on. For example, if the profile option is set to Operating Unit, then the maintain privileges for that price list or modifier are restricted to the default operating unit of the responsibility that created the price list or modifier. This controls which users (if any) have access to maintain specific price lists and modifiers. View and maintain responsibilities are controlled separately by different profile options.

#### Values

Global, Operating Unit, Responsibility, User, or None.

The default value is Global.

### QP: Security Default ViewOnly Privilege

This profile option determines the default view-only privileges for NEWLY CREATED price lists and modifiers after security is turned on. View and maintain responsibilities are controlled separately by different profile options. This controls which users (if any) have access to view specific price lists and modifiers.

#### Values

Global, Operating Unit, Responsibility, User, or None.

The default value is Global.

### QP: Selling Price Rounding Options

This rounding option rounds the selling price after adding unrounded list price and adjustments:  $\text{selling price} = \text{round}(\text{list price} + \text{adjustments})$

**Note:** The profile OM: Round Unit Selling Price has been migrated to QP: Selling Price Rounding Options.

#### Values

- NO: = unrounded listprice + unrounded adjustments: No rounding is done.
- Individual: = round (listprice) + round (adj): The adjustments are calculated using rounded list price.
- Additive: = round (listprice + adj); unrounded Freight: The adjustments are calculated using unrounded list price.

This profile option can be viewed and updated at the site and responsibility level.

#### Freight Charge Rounding

Freight Charge Rounding: If the QP: Selling Price Rounding Options profile is set to NO

or ADDITIVE then freight charges will not be rounded. If the profile is set to INDIVIDUAL then freight charges will be rounded. The rounding flag in the control record passed by calling application may have one of the following values:

- Y (Yes): Rounds selling price and adjustments.
- N (No): No rounding.
- Q: Behavior depends on the profile setting for QP: Selling Price Rounding (NO, INDIVIDUAL, ADDITIVE). If rounding flag is passed as Q, but QP: Selling Price Rounding Options is NULL, the default behavior is no rounding.
- Null: Rounds selling price and adjustments.

**Case 1)**

Rounding Flag = Q

Profile QP: Selling Price Rounding Options = NO

List Price = 12.60, Rounding Factor = 0, Discount 25%

Adjustment amount = -3.15

Selling price =  $12.60 - 3.15 = 9.45$

**Case 2)**

Rounding Flag = Q

Profile QP: Selling Price Rounding Options = INDIVIDUAL

List Price = 12.60, Rounding Factor = 0, Discount 25%.

Adjustment amount =  $-\text{round}(\text{round}(12.60) * 0.25) = -3$

Selling price =  $\text{round}(12.60) - 3 = 10$

**Case 3)**

Rounding Flag = Q

Profile QP: Selling Price Rounding Options = ADDITIVE

List Price = 12.60, Rounding Factor = 0, Discount 25%.

Adjustment amount = -3.15

Selling price =  $-\text{round}(12.60 - 3.15) = 9$

**Case 4)**

Rounding Flag = N

List Price = 12.60, Rounding Factor = 0, Discount 25%.

Adjustment amount = -3.15

Selling price =  $12.60 - 3.15 = 9.45$

**Case 5)**

Rounding Flag = Y

List Price = 12.60, Rounding Factor = 0, Discount 25%.

Adjustment amount =  $-\text{round}(\text{round}(12.60) * 0.25) = -3$

Selling price =  $\text{round}(12.60) - 3 = 10$

**Case 6)**

Rounding Flag = NULL

List Price = 12.60, Rounding Factor = 0, Discount 25%

Adjustment Amount =  $-\text{round}(\text{round}(12.60) * 0.25) = -3$

Selling price =  $\text{round}(12.60) - 3 = 10$

The default value is Individual:  $= \text{round}(\text{listprice}) + \text{round}(\text{adj})$ .

**QP: Set Request Name**

The profile option QP: Set Request Name can be used in conjunction with the QP: Debug profile option so that the Request Name field will be prefixed with the Order ID.

**Values**

Any valid values such as the Name or User ID of the user submitting the price request.

The QP: Set Request Name profile option is visible and can be updated at the site, application, responsibility, and user levels.

The default value is Blank.

**QP: Source System Code**

This profile option is used in all pricing setup windows to identify the application through which the pricing information is being entered. This source system code is held on all price and modifier lists to identify the origin of the data. At the time of pricing, the pricing engine may restrict its search to pricing information which originated from a particular application depending on the request type to source system setup.

By default, the value of the profile option QP\_SOURCE\_SYSTEM\_CODE is QP (Oracle Pricing) at the Site level. However, to differentiate between applications and prevent update of modifiers among several applications, the value of this profile can be set up at the Application level.

If modifiers are created in different applications such as Oracle Advanced Pricing, Order Management, or Trade Management, then changes to the modifier can only be made in the application where the modifier was originally created. These changes include changes to the modifier header (as deleting is already prevented) or changes such as inserting, updating, or deleting modifier lines, pricing attributes, and qualifiers.

If you do not want to differentiate between applications or prevent update of modifiers created by different applications, then no specific setup is required. In this case, all

modifiers are created with same value for source system code (assuming the value of this profile is set only at site level by default or same value if set for all applications which create modifiers).

**Example**

The following customer requirements need to be established when setting up the profile options:

- Modifiers created by the Trade Management application can only be changed by Trade Management.
- Modifiers created by Oracle Pricing and Order Management applications can be interchangeably updated but not by other applications.
- All other applications can update the modifiers of other applications except Trade Management, Oracle Pricing and Order Management.

To accommodate these requirements, the value of the profile QP\_SOURCE\_SYSTEM\_CODE is set up as follows:

- Site level: QP (default)
- Trade Management: XXX
- Oracle Pricing: YYY
- Order Management: YYY (same as Oracle Pricing)

The default value should only be changed if the source of the pricing data is any application other than Oracle Advanced Pricing. Set this to the source system lookup code of the application which is interfacing the pricing data.

After Attribute Mapping Manger is installed, this profile can be set at Site, Application and User level. The default value for this profile at the time of installation at the Site level is Oracle Pricing. The seeded value of this profile for various applications is shown in the table below:

<b>Application name</b>	<b>Default Value</b>
Site Level Profile Value	Oracle Pricing (QP)
Oracle Transportation Execution (FTE)	Oracle Transportation Execution (FTE)
Oracle Inventory (INV)	Oracle Inventory (INV)
Oracle Marketing (AMS)	Oracle Marketing (AMS)

Application name	Default Value
All other Applications	Defaulted from Site

**Note:** Set the QP: Source System Code Profile to the source system lookup code of the application from which the QP Setup or other application setup windows are called.

### Values

QP: Oracle Pricing (and other source system lookup codes of related applications).

This profile option can be updated at the site and responsibility levels.

The default value is Oracle Pricing.

### QP: Unit Price Precision Type

This profile option determines the Round To value that is defaulted on the price list. The rounding factor is limited by the number of positions allowed in the standard or extended precision format of the price list currency.

**Note:** If you update either the extended or standard precision value for a currency, then the updated value is applied only to new price lists but not existing price lists.

### Values

- Extended: Rounding factor defaults to the currency's extended precision.
- Standard: Rounding factor defaults to the currency's standard precision.

This profile option can be updated at the site and application levels.

**Important:** At the beginning of a price request, the item amount is sourced and then calculated using quantity \* unit price. This value is then converted to a canonical format. However, if the value is larger than the canonical format, the conversion fails. Currently, for number type attributes, the pricing engine handles numbers of up to 21 digits to the left of the decimal, and 40 digits to the right of the decimal. Since the item amount and item quantity are also attributes (sourced internally), the restriction applies to these attributes as well.

The default value is Standard.

### **QP: Valueset Lookup Filter**

Use this profile option to enable or disable a search criteria window for qualifier value lookups in qualifiers, price lists, and modifiers. Some qualifiers use large valuesets, for example, those based on all customers, and searches may take a long time. If you want to reduce the number of items that display in the list of values, you can enter search criteria. If you do not enter search criteria and click the list of values indicator for the fields Value From or Value To, you see a window which advises that you have not entered search criteria and that the search may take a long time.

#### **Values**

- Yes: The message displays.
- No: The message does not display. Use this value if you do not expect to have large qualifier valuesets and do not need to enter search criteria to reduce the display.

This profile option is visible and can be updated at the site, application, responsibility, and user levels.

The default value is Yes.

### **QP: Verify GSA Violations**

This profile option indicates whether the pricing calculation engine should test for GSA violations. You can change the value to Yes if you require GSA pricing functionality.

The evaluation is performed if: 1) a request is for a non-GSA customer, and 2) GSA rules are violated if the selling price of an item is calculated to be less than the price of the item on any GSA price list.

#### **Values**

- Yes: Pricing engine tests for GSA violations, and any violating request lines are returned to the calling application with GSA violation status.
- No: Does not test for GSA violations.

This profile option can be updated at the site, application, responsibility, and user levels.

The default value is No.

## **Overview of Pricing Parameters**

Pricing parameters provide controls for pricing applications that can be set by a Pricing Transaction Type (Request Type) or by a group of transactions that share pricing setups (PTE). Unlike profile options, which can be set at the site, application, responsibility, and user levels, parameter values are set at Pricing Transaction Entity (PTE) or Request Type level.

**Note:** Request types are those order applications that request pricing data (for example, Request Types such as *Order Management Order* or *Oracle Contracts for Service*).

Pricing parameters provide flexibility in situations where the parameter value needs to be different for different Request Types.

**Note:** Parameters are defined at either the Pricing Transaction Entity (PTE) level or the Request Type level: parameters defined at the PTE level may affect behavior for the pricing engine and the user interface level, while parameters defined at the Request Type level mainly affect pricing engine behavior.

For example, if you wanted price break ranges returned for Oracle Transportation but not Oracle Order Management, you could not set an application level profile option simultaneously to *No* for Oracle Order Management and *Yes* for Oracle Transportation. However, the pricing parameter *Return only satisfied Price List break lines* gives you the flexibility to define different parameter values for each Request Type: *No* for Oracle Order Management or *Yes* for Oracle Transportation. This enables price breaks to be returned for Oracle Transportation but not for Oracle Order Management.

## Related Topics

Overview of Pricing Parameters, page 5-43

Viewing Pricing Parameter Definitions, page 5-44

Setting Pricing Parameter Values, page 5-45

Seeded Pricing Parameters, page 5-46

## Viewing Pricing Parameter Definitions

You can search for parameters by parameter level and/or name to view details about seeded parameters. Pricing parameters enable you to control the functionality of the pricing engine at the PTE (pricing transaction entity) and Request Type levels.

### Notes:

In the Parameter Definition page, you can review parameter details and update parameter values (if applicable):

- **Level:** Displays the level at which the parameter is set such as Pricing Transaction Entity or Request Type.
- **Update:** Click Update to update the seeded value. If this button is grayed out then the value cannot be updated.

- **Details:** Click Details to view additional details about the selected parameter such as the Value Set and Default Value.
  - **Default Value:** Displays the default value for the parameter.
  - **Advanced Pricing Only:** If selected, then the parameter is only available for Advanced Pricing customers.

**Note:** If the Advanced Pricing Only check box for the parameter is selected and the installation is only for Basic Pricing, then the parameter will not display or the User Assigned Value cannot be updated.

## Related Topics

Overview of Pricing Parameters, page 5-43

Setting Parameter Values, page 5-45

Seeded Pricing Parameters, page 5-46

## Viewing and Updating Pricing Parameter Values

You can do searches to find available pricing parameters, view detailed information about a parameter, and if desired, change the parameter value. Pricing parameters enable you to control the functionality of the pricing engine at the PTE (pricing transaction entity) and Request Type levels.

Use the Parameters Values page to search for and update existing values for a parameter. A parameter value (either seeded or overridden with a user-assigned value) controls the default behavior for a parameter. For example, the seeded values (*Yes* or *No*) for the parameter Return Only Satisfied Price List Break Lines control which price break lines are returned.

You can update a parameter value by entering a new User Assigned Value to take precedence over the seeded value.

**Note:** You must be assigned the Oracle Pricing Administrator or Pricing Manager responsibility to view or update pricing parameters.

### To view and update pricing parameter values:

In the Parameter Values page, do a search to display the parameter name(s) and related information:

- **Level Meaning:** Defines the level (such as Request Type or Pricing Transaction

Entity) at which the parameter is defined.

- **Level Name:** Displays the default level for the parameter such as Order Management Order.
- **Seeded value:** Displays the default seeded value such as Yes or No assigned for each Parameter Name/Level Name combination. Since the same Parameter Name can be paired with different Level names, the seeded value may be different for each Parameter Name/Level Name combination. Optionally, to override a seeded value, enter a User Assigned Value.
- **User Assigned Value: (Optional)** To override the default seeded value, enter a User Assigned Value (this takes precedence over the default seeded value).

## Related Topics

Overview of Pricing Parameters, page 5-43

Viewing Parameter Definitions, page 5-44

Seeded Pricing Parameters, page 5-46

## Seeded Pricing Parameters

This section describes the seeded pricing parameters. These parameters can be set only at *Request Type* level.

## Price Book Pricing Events

This parameter identifies which pricing events should be used in the generation for a price book submitted for that request type. You can enter the name of a different valid pricing event in the User Assigned Value field or enter multiple pricing events by separating each event name with a comma; for example, PRICE, LINE (two events) or PRICE, LINE, BATCH (three events). Enter the event name(s) in capital letters; the system does not validate your entry or warn you if you make an incorrect entry such as entering an invalid event name. For more information on seeded events, see *Oracle Advanced Pricing Implementation Guide*, What are Pricing Events.

**Note:** A pricing event is a point in the transaction life cycle when you want to price the transaction (or certain transaction lines), or when you want to apply price adjustments, benefits, or charges to the whole transaction or specific transaction lines.

### Example

The default value for the Order Management request type is BATCH event. If the customer has defined modifiers that apply at order booking time (for example, for

BOOK event) and if these modifiers are used to calculate net prices for the price book, set the parameter value for Order Management request type to BATCH, BOOK.

## **Return Only Satisfied Price List Price Break Lines**

This parameter value indicates if the pricing engine should return only satisfied price break lines from the price list or all price break lines. Set parameter value to Y (Yes) for improved performance.

### **Example of break processing**

Suppose a price list line is set up as a Point Break with the following break ranges by quantity:

- 0-10
- 10-100
- > 100

If the quantity value for the current request is 15, then the break processing differs depending on the parameter value selected:

- If parameter value is Y (Yes); If the quantity value for the current request is 15, then the pricing call only returns 10-100 because the quantity is eligible for this break line.
- If the parameter value is N (No): All the break lines will be returned.



---

## Pricing Security

This chapter covers the following topics:

- Overview of Oracle Pricing Security
- Pricing Security and Operating Units
- Setup Steps for Implementing Pricing Security
- Changes to Pricing User Interfaces (UI) after Upgrading and Turning On Security
- Assigning Pricing Entity Usage
- Implementation Suggestions for Privileges
- Creating Privileges
- Creating Entity Sets
- Setting Security Profile Options
- Turning On Pricing Security

### Overview of Oracle Pricing Security

In Oracle Applications, a basic level of security called *functional security* is used to manage users' access to each application and control their access to windows, functions, and reports within an application.

Typically, the system administrator administers function security and assigns operating unit, responsibility, and system access to users. See the *Oracle E-Business Suite Security Guide* for more information about function security.

In addition to the existing function security, Oracle Advanced Pricing provides an additional level of security called *pricing security*. Pricing security enables you to restrict pricing activities such as updating and viewing pricing entities to users who are granted specific access privileges. Pricing entities include price lists, pricing agreements, and modifiers.

Pricing security can be set up and maintained in the HTML user interface by a user who

is assigned the Oracle Pricing Administrator responsibility. The Oracle Pricing Administrator has the authorization to access and update all pricing entities for all functional users. With pricing security, you can implement a higher level of control by:

- Assigning pricing entities to operating units: A pricing entity can be assigned ownership to a specific operating unit. You can restrict usage to one operating unit or allow usage by all operating units.
- Assigning privileges that control which grantee (Global, Operating Unit, Responsibility, or User level) can view or maintain the specified entity: You can use security privileges to control users' access to pricing entities in the following ways:
  - Grant view-only or maintain access privileges to functional users at the Global, Operating Unit, Responsibility, or User level.
  - Grant temporary access - for example, to auditors or temporary employees - for a specified date range.
  - Assign or reassign Operating Unit ownership to price lists and modifiers and control which operating units can use them for pricing transactions.
  - Create entity sets (a set consists of grouped pricing entities) and assign access privileges to the entire set. The Entity Set function is available only with license to Advanced Pricing.
- Setting default rules for security access for new pricing entities.

**Warning:** Before turning on pricing security, you must create privileges for existing pricing entities.

## Advanced Pricing Responsibilities

A *responsibility* defines a level of authority in an application. Each responsibility lets you access a specific set of Oracle Applications windows, menus, reports, and data to fulfill your role in an organization. Several users can share the same responsibility, and a single user can have multiple responsibilities.

You can assign users the following seeded responsibilities to enable access to pricing windows, menus, reports, and data:

- Oracle Pricing Administrator

The user who is assigned the Oracle Pricing Administrator responsibility has complete access to all pricing entities without restriction and is responsible for the global setup and administration of pricing security. Pricing security features include privileges, entity sets, and entity usage. The Oracle Pricing Administrator has unrestricted access and can select any operating unit and can access pricing

entities across all operating units. For more information about access privileges by operating unit, see Pricing Security and Operating Units, page 6-4.

- Oracle Pricing Manager

The user who is assigned the Oracle Pricing Manager responsibility can access all features for setting up, using, and maintaining pricing features and functions (except pricing security) such as price lists, pricing formulas, modifiers, attribute management, and item and category management.

- Oracle Pricing User

This responsibility provides access to the HTML user interface where you can create price lists and modifiers (Deal, Discount, Promotion, Surcharge), access the price list maintenance feature, and attach qualifiers and qualifier groups to modifiers and price lists.

## Pricing Security Terms

The following terms are used in Oracle pricing security:

- Pricing Entity Security: The highest level of security administration for Oracle Pricing. This level of security is in addition to functional security and pricing transaction entity (PTE) plus source system code security. Functional security is established for each user by responsibility setup. The Oracle Pricing Administrator responsibility has complete access to all pricing entities without restriction and is used for the global administration of pricing entities. This security is administered in the Oracle HTML user interface.
- Pricing Entity: A pricing entity can be a price list, modifier list, or pricing agreement.
- Entity Set: A set of pricing entities that can be used as an Entity Type to which you can grant privileges with Maintain or View-Only access levels.
- Entity Type: A term used to describe one of the following pricing entities: Standard Price list, Modifier List, Pricing Agreement, and Entity Set.
- Entity Usage: Grants the usage of the entity to one or all operating units so that it can be used during pricing engine calls.
- Global Usage: When Global Usage is set to Yes for a pricing entity, it can be used across all operating units for processing orders. If No is selected, the usage of the entity is restricted to the operating unit that created or owns it.

When security is turned on, a Global box indicating Global Status is dynamically added to the header region of all price lists and modifiers. A user with Maintain access privileges can update the Global box. The Oracle Pricing Administrator can

also update the Global Usage settings in the Entity Usage pages.

- **Grantee:** The specific user or users for a Grantee Type who are given permission to view or maintain a pricing entity. Used in combination with a Grantee Type.
- **Grantee Type:** The level to which privileges are granted:
  - **Global:** Includes all users with access to pricing menus.
  - **Operating Unit:** Includes users who have the named operating unit as the default operating unit (as specified in MO: Default Operating Unit profile).
  - **Responsibility:** Includes users within the named responsibility.
  - **User:** Specifies a named user.
- **Access Level:** Provides Maintain or View-Only access to a pricing entity:
  - **View-Only:** Enables the user to view but not update the pricing entity.
  - **Maintain:** Enables the user to view and update pricing entities. Not all of the entities support delete capabilities.

## Pricing Security and Operating Units

Pricing Security allows you to create pricing data that is specific to an operating unit. The multi-organization access control (MOAC) feature further enables users to access multiple operating units within one responsibility, and to create multiple pricing entities (price lists, modifiers) for different operating units without changing responsibility. This feature is controlled by the profile option MO: Security Profile. When MOAC is enabled, you can enable pricing security to provide centralized control of pricing entities for use by operating unit or across all operating units for pricing orders. Pricing security is still used to assign access roles at Global, Organization, Responsibility, or User levels with roles of View Only or Maintain. However, when MOAC is enabled, you should review the following changes in pricing security action:

**Note:** See the *Oracle Applications Multiple Organizations Implementation Guide* for information about setting the profiles MO: Security Profile and MO: Default Operating Unit.

## Multi-Organization Access Control

**Profile option MO: Default Operating Unit automatically creates pricing security privileges**

When you create a price list or modifier list, the default pricing security privileges are created for the operating unit that is set in the MO: Default Operating Unit regardless of

whether the price list or modifier list is created as Global or for a different operating unit. This occurs when:

- Profile option MO: Default Operating Unit is enabled (MO: Security Profile is set)
- Pricing security is ON (profile option QP: Security Control is ON)
- One of the pricing profile options, QP: Security Default ViewOnly Privilege or QP: Security Default Maintain Privilege, is set to Operating Unit

For example, suppose you are assigned the Pricing Manager responsibility with access to the following operating units - OU1, OU2, and OU3 - and the following conditions exist:

- Pricing security is ON.
- MO: Default Operating Unit profile is set to **OU1**.
- QP: Security Default Maintain Privilege and QP: Security Default View Only Privilege profiles are set to Operating Unit.

If you then create a global price list (PL1) or price list for OU2 (PL2), the view and maintain privileges will be created for **OU1** because the operating unit defaults from the MO: Default Operating Unit profile (the default operating unit that is set for the responsibility) for both PL1 and PL2.

#### **Update to Operating Unit field allowed if users have Maintain access**

In Oracle Advanced Pricing, the operating unit on the price list or modifier list must match the operating unit of the transaction (for example, a sales order) that is being priced. If pricing security is ON, any user who is granted maintain access to the price list or modifier list can update the Operating Unit field of the modifier or price list.

For example, suppose you have a price list PL1 from operating unit OU1 that is assigned a maintain privilege of Global, but you log into a responsibility with access to only OU2, OU3 as assigned by the MO: Security Profile. You could update the price list PL1 due to Global security privilege and update it from OU1 to OU2; however, you cannot change it back to OU1 through the same responsibility because the security profile does not provide access to OU1.

## **Setup Steps for Implementing Pricing Security**

After you upgrade to pricing security, pricing security is not switched on automatically. Pricing users with functional access can still fully view and maintain existing price lists and modifiers as before the upgrade.

Before turning security on, you should review and complete the following setup steps for implementing pricing security, *otherwise, pricing users may be unable to query any price lists or modifiers in the pricing windows*. After you have completed the security setup steps, you can run the concurrent program *QP: Security Control with Views Conversion* to

turn on pricing security.

**Note:** The profile option QP: Security Control (read-only) displays the current setting of the security option for your entire installation (either on or off).

You must be assigned the Oracle Pricing Administrator responsibility to set up and maintain the pricing security features for all functional pricing users.

### **Step 1: Map security access requirements**

Identify and map all price lists, modifiers, and agreement price lists to:

- Operating units that should own and maintain them.
- The users in those operating units who require View-Only or Maintain access (view and update) to pricing entities.
- Operating units that use them when pricing transactions.

### **Step 2: Assign ownership of pricing entities (Entity Usage page)**

The next step is to assign preexisting price lists and modifiers to an operating unit. Use Global Usage settings to restrict the entity to a specific operating unit or make it available across all operating units.

### **Step 3: Create privileges (Privileges page)**

The next step is to create access privileges for all users in all operating units. You can assign view or maintain access to a pricing entity.

Optionally, you may want to create entity sets that enable you to group multiple entities of the same entity type, and then grant access to the entity set. For example, you may want to create a set called Summer Set that contains all active modifiers with Summer Promotion in the modifier name. Then you can assign privileges to the entity set rather than to each entity separately.

**Note:** You must have a license for Advanced Pricing to use entity sets.

### **Step 4: Set security profile options**

Use the following security profile options to set the default security privileges for pricing entities that are newly-created:

- QP: Security Default ViewOnly Privilege: Sets the default Viewing privileges for newly-created pricing entities.
- QP: Security Default Maintain Privilege: Sets the default Maintain privileges for

newly-created pricing entities.

- **QP: Security Control (read-only):** This profile option displays the current setting of the security option for your entire installation (either on or off). This profile option value cannot be directly updated - only the concurrent program *QP: Security Control with Views Conversion* can turn pricing security on and off.

These profile options are delivered in default settings that maintain the existing functional security features of Oracle Pricing. Before you can change these profile settings, the Oracle Pricing Administrator must map the complete security access requirements for each pricing entity. No security profile option should be changed until these steps have been completed.

### **Step 5: Turn on pricing security**

To activate pricing security, set the concurrent program *QP: Security Control with Views Conversion* to ON. This is the "switch" that turns security on or off for your installation. Before setting the program to ON, ensure you have completed all the preceding implementation steps.

## **Related Topics**

Assigning Pricing Entity Usage (Entity Usage page), page 6-11

Creating Pricing Entity Sets, page 6-17

Creating Privileges, page 6-14

Setting Security Profile Options, page 6-19

## **Changes to Pricing User Interfaces (UI) after Upgrading and Turning On Security**

This section summarizes the changes that occur to pricing entities after you upgrade to pricing security and turn on security. Some of the changes, such as the new Global check box on price lists and modifiers, are visible only to users after pricing security is turned on.

### **Entity Usage**

After the upgrade to security, all existing price lists and modifiers are assigned the default entity usage of Global Usage. Global usage enables the pricing entity to be used across all operating units. When security is turned on, a Global box is added to the header of all modifiers and price lists to indicate the global usage status for the entity. If the Global check box is:

- Selected, then global usage is enabled for the entity.
- Cleared, then global usage is not enabled for the pricing entity, and an operating

unit must be assigned to the entity.

The Global check box is not visible to users until the concurrent program Security Control is turned ON. When the check box is visible, a user with Maintain access privileges can select or clear the Global check box. However, users with view-only privileges cannot change the Global check box. If a user creates a new pricing entity (such as a price list) and clears the Global check box, then an operating unit must be assigned to that entity. If MOAC is not enabled, the operating unit defaults to the value of the profile MO: Operating Unit.

With MOAC, this operating unit will default to the value in the MO: Default Operating Unit profile. You can override this default and select from any operating unit that is assigned to the MO: Security Profile option. If the Global check box is left selected (the default value), then the entity can be used across all operating units when transactions are priced.

Alternatively, the Pricing Administrator can also update the Global check box for one entity at a time or in bulk using the Bulk Update Entity Usage page, which is available from the Entity Usage page.

### Changes to Price Lists

After the upgrade, you can review the operating unit and global usage settings for an entity on the Entity Usage page. An example of the information that appears for a selected entity is outlined in the following table:

Entity Name	Type	Global Usage	Owned by Operating Unit
Name of the entity (for example, Summer Pricelist)	Type of entity (for example, Standard Pricelist)	Yes	Blank (not assigned to an operating unit)

The following price list changes occur after the upgrade to pricing security:

- Price lists that are assigned Global usage cannot be assigned to an operating unit as well. Any such global price lists will be updated to clear out the operating unit.
- Once security is turned on, all new price lists have their view and update properties determined by the pricing security profile options.
- You need at least view-only access privileges to display or query price lists in the price list windows. With view-only access, you cannot change header or any associated information such as price list lines, pricing attributes, qualifiers, or secondary price lists.
- Users who have view-only privileges on a price list as per pricing security rules will be in view-only mode on the price list window. To update a price list, the user

requires specific maintain-access privileges.

- For secondary price lists, you can select only the price lists with view-only or maintain privileges for the secondary price list. In addition, secondary price lists are also restricted by the entity usage that is assigned to the primary price list: if the primary price list is global, you can select any secondary price list (global or assigned to any operating unit). If an operating unit is assigned to the primary price list, you can select a global price list for secondary price list or a secondary price list that has the same operating unit as the primary price list.
- The Public API, `QP_PRICE_LIST_PUB.PROCESS_PRICE_LIST`, will update only price lists as per price list security rule.
- You can select Price Lists > Copy Price Lists to copy price lists. A copied price list is assigned the default privilege from the security profile options. During copying, you can override defaults that are derived from Copy From price list and specify your own settings for global flag and operating unit for the Copy To price list. However, if the default security privilege is set to Operating Unit, the copied price list is still assigned the default privilege based on the MO: Default Operating Unit profile.

#### **Changes to Modifier windows**

The following modifier changes occur after the upgrade to pricing security:

- Modifiers that are assigned Global usage cannot be assigned to an operating unit. Any such global modifiers will be updated to clear the Operating Unit field.
- After pricing security is turned on, the default view and maintain properties for all new modifiers are determined by the security profile options.
- You need at least view-only access privileges to display or query modifiers in the Define Modifier window. With view-only access privileges, you can view all list and line limits for a modifier that include attributes and transactions for the limit.
- With view-only access privileges, you cannot modify the header information, lines, list or line qualifiers, pricing attributes, and related modifier information. A message will appear to advise you about the view-only status.
- Modifier lines of the type Promotional Goods can attach to price lists that are viewable, as per pricing security, in the Get Price column list of values (LOV) in the Get region.

In addition, the list of values in the Get Price column on Promotional Goods is also restricted by entity usage that is assigned to the modifier:

- If the modifier is global, you can select only a global price list in the Get Price column.

- If an operating unit is assigned to the modifier, you can select a global price list or a price list that has the same operating unit as the modifier.
- The Public API, `QP_MODIFIERS_PUB.PROCESS_MODIFIERS`, will update only modifiers as per modifiers security rule.
- In the Modifier Incompatibility Setup window, only those modifier lines belonging to a modifier list that can be viewed or maintained will get queried as per pricing security rules. Modifiers that are opened by clicking the Modifiers button can be viewed or maintained depending on the privileges that are defined by the Pricing Security Administrator.
- With view-only access to a modifier, you can copy the modifier by choosing modifiers > Copy Modifiers.
  - Assign the default privileges from the security profile options. If the profile option is set to Operating Unit, the default privileges are always by the MO: Default Operating Unit profile regardless of the operating unit that is assigned to the copied modifier.
  - Global flag and operating unit value for the copied modifier defaults from the Copy From modifier. But you can override these defaults, and select new values for global flag and operating unit for the Copy To modifier.

### Changes to Calling Applications

All calling applications that display a list of values for price lists can use the pricing view `QP_PRICELISTS_LOV_V` to display the valid global and operating unit (OU) price lists that are specific to their transaction. This view displays the either the global price lists and price lists that are specific to the OU on the transaction or all price lists depending on whether the pricing security feature is turned ON or OFF.

### Changes to Other Pricing windows

The following table outlines the impact of pricing security and security privileges on various windows in Advanced Pricing:

<b>For the following:</b>	<b>Security Privileges are enforced:</b>
Copy Price Lists	Yes. You need at least view-only access.
Copy Modifier	Yes. You need at least view-only access.
Modifier Incompatibility setup	Yes, can be updated if you have maintain access.

<b>For the following:</b>	<b>Security Privileges are enforced:</b>
Pricing Organizer	Yes. You can view and access the modifier if it appears.
Pricing Mass Maintenance	Yes. You need maintain access.
Adjust Price List	Yes. You need maintain access.
Add Items to Price Lists	Yes. You need maintain access.
Multi-currency conversion	No security at present.
Formulas	No security at present.
Agreement Header	Agreement inherits security rules of attached price list.
Price List report	Yes. You must have at least view-only access.
Modifier Detail report	Yes. You must have at least view-only access.
Archive and Purge	Yes, you must have maintain access.

## Assigning Pricing Entity Usage

By default, a new price list or modifier is assigned Global usage. If the Global check box is deselected for a pricing entity such as a modifier, the Operating Unit field is enabled. The operating unit defaults from the profile option MO: Default Operating Unit if MOAC is enabled. If MOAC is not enabled, the value defaults from the profile option MO: Operating Unit.

This operating unit field value appears in the Owned by Operating Unit field on Pricing Entity Usage user interfaces. A pricing entity that is assigned to an operating unit can be used only for that operating unit and not across all your operating units. However, pre-existing price lists and modifiers are not assigned a default operating unit, so you can use the entity usage feature to:

- Assign or reassign ownership of pre-existing price lists and modifiers to the appropriate operating unit.
- Grant or revoke Global Usage for pricing entities. Global usage enables the pricing entity to be accessed across all operating units.

When assigning pricing entity usage to a pricing entity such as a price list or modifier, you should consider the following:

- Identify which entities are to be used across all operating units (Global Usage ) in pricing transactions.
- Identify which entities are to be restricted to only one operating unit (Owned by Operating Unit).
- Identify pricing entities that are used by *multiple* operating units but not *all* operating units:
  - Select Yes for Global Usage.
  - Create modifier or price list qualifiers for the specific operating units. Qualifiers need to be created using the price list or modifier user interfaces.

Based on the security policy of your organization, the Oracle Pricing Administrator can then grant access privileges to the pricing entities, once entity usage has been set up.

**Warning:** The Oracle Pricing Administrator should assign ownership to all price lists and modifiers prior to upgrading or implementing Oracle Pricing Security. This can also be done using the Bulk Update Entity Usage feature on the Entity Usage page.

### To assign pricing entity usage:

On the Entity Usage page, search for the entities and assign the following entity usage values:

**Note:** For fresh upgrades or new installations, the Global Usage check box is Yes (selected) and the Owned by Operating Unit field is blank.

- **Global Usage:** To make the entity available across all operating units, select Yes for Global Usage. If this is not selected (cleared), global usage is not enabled for the pricing entity, and the usage of the entity is restricted to the assigned operating unit. The Global Usage status is also displayed to users through the Global box on price list and modifier windows.
- **Owned by Operating Unit:** To restrict the entity's usage to a specific operating unit, select the operating unit name.

To make bulk changes to multiple pricing entities, click Bulk Update Entity Usage.

### **To use bulk update entity usage:**

Use the Bulk Update Entity Usage page to quickly apply settings for global usage and operating unit assignment across multiple pricing entities; for example, to assign the same operating unit across all price lists.

From the Entity Usage page, find the pricing entities to be updated. Alternatively, to select all pricing entities on a page, click Select All. If additional entities are listed on subsequent pages, click the Next link, and then click Select All. Repeat this process until all the entities to be updated are selected. Click Bulk Entity Usage to update the following settings:

#### **Notes**

- Global Usage: Select Yes or No to update the global usage for the entities.
- Owned by Operating Unit box: Select this box (and an Operating Unit) to assign the entity to a specific operating unit.

## **Implementation Suggestions for Privileges**

Complete the following planning and implementation steps before turning on pricing security. This mapping should be completed by someone with complete knowledge about the following: the pricing users and their operating units; all price lists; modifier lists; and any specific business requirements for granting access to any of the many pricing entities.

1. Identify and list all users with functional access to Advanced Pricing menu.

Identify all responsibilities within your installation that have functional access to the Oracle pricing menus. This helps to determine whether a pricing entity can be granted access by users with these responsibilities. When an access privilege is granted by responsibility, then all users with this responsibility will have this privilege.

Add to the listing of all responsibilities with access to pricing menus, all individual users, by name. Some users may not require Maintain privileges to any pricing entities, but may actually require view-only access. These users should be identified and associated with the pricing entities to which they require view access.

This mapping assists in granting an access privilege to a specific user. A user may have access privileges by virtue of their responsibility. If the user, whose responsibility has been granted an access privilege of ViewOnly to a pricing entity, needs to have Maintain access, a privilege may be granted to the user for Maintain that is a higher privilege than that granted to his or her Responsibility.

2. List all users by new access privileges.

A listing of all users and their access privileges should be maintained by the Pricing

Administrator. Once mapping has been completed and access privileges granted, you can query the privileges that are granted in a variety of ways using the Privileges page of the Security pages. A search by entity type such as Standard Price List displays all standard price lists by entity name, grantee type, grantee name, access level (ViewOnly or Maintain), and effective dates. Your listing of new access privileges can be checked against the results.

## Creating Privileges

Security privileges enable you to define who can access each pricing entity and the level of access that is permitted: View Only or Maintain. You can grant the following access privileges:

- Grant access privileges to functional users at the Global, Operating Unit, Responsibility, or User level:
  - Global: Includes all users with access to pricing menus.
  - Operating Unit: Includes users that have the named operating unit as the default operating unit (as specified in MO: Default Operating Unit profile).
  - Responsibility: Includes users within the named responsibility.
  - User: Specifies a named user.
- Grant access level of View Only or Maintain.
- Grant temporary access - for example, to auditors or temporary employees - and give them automatic start and end effective dates.
- Assign privileges to entity sets. You can create entity sets to group similar entities (for example, modifiers for a specific customer) and assign privileges to that entity set rather than assign a privilege separately to each entity. The entities that are contained within that entity set inherit the privileges that are assigned to the entity set. For more information, see *Creating Pricing Entity Sets*, page 6-17.

**Note:** You must be assigned the Oracle Pricing Administrator responsibility to grant privileges.

You can assign privileges using the following pages:

- Privileges page: To search for and update existing privileges.
- Express Create Privilege page: To create an access privilege for one specific pricing entity.

- Bulk Create Privileges page: To select multiple pricing entities and create access privileges for a grantee.

To assign default security privileges for newly-created pricing entities, see Setting Default Security Profile Options, page 6-19.

### **Precedence Levels for Multiple Privileges**

A user belonging to a responsibility classification such as Pricing User will typically have the access privileges that are associated with that responsibility. However, if a user has only View Only access to a pricing entity by virtue of his or her responsibility, but requires Maintain access, you can assign a Maintain access privilege to the user. A Maintain access privilege is a higher privilege than View Only, and therefore, the higher Maintain privilege prevails for the named user.

If a user has a Maintain access privilege to a given entity at any level of his or her user hierarchy (Responsibility, Operating Unit, and Global), the user will have Maintain access regardless of any other privileges. For example, if a user has Maintain access at his operating unit level but a view-only access at his user level, his Maintain access privilege will have precedence.

### **To create privileges (directly in Privileges page):**

In the Search region, search by Entity Type to view and assign privileges directly on the Privileges page:

- To revoke privileges, select the line to delete and click Delete.
- To assign or update an access level, select Maintain or View Only.
- Enter or update the effective start and end date, and click OK to save your changes.

**Note:** If the message *No data exists* appears in the Results: Privilege(s) region then no privileges exist for the entity.

Alternatively, select the entities and click either Express Create Privilege or Bulk Create Privileges.

### **To create a privilege for one specific pricing entity (Express Create Privilege):**

1. To create a privilege for one specific pricing entity, select the entity and click Express Create Privilege.
2. Select the entity type and entity name of the pricing entity to be granted privileges.
3. Select from the following Grantee Types and, if applicable, select a Grantee Name:
  - Responsibility: Grants the privilege to a specific responsibility such as Pricing

User, Guest User (the specific Grantee Names depend on the setup for your specific business).

- User: Grants the privilege to a specific user such as John Smith in the Pricing Department.
  - Global: If Grantee Type is Global, leave Grantee Name blank. This makes the privilege available to all users with functional access to pricing menus.
  - Operating Unit: Grants the privilege to a specific operating unit. For example, select Vision1 to give a privilege to all users who have Vision 1 as the default operating unit.
4. Access Level: Select the access level to be granted to the grantee:
    - Maintain: Enables users to delete, view, and update pricing entities.
    - View Only: Enables users to view but not update the pricing entity.
  5. Start and End Dates: Select the start and end dates. For example, to provide temporary access to a temporary employee, you could enter a start date of 02-Jul-2007 and an end date of 31-Aug-2007. Alternatively, accept the system dates.

### **To create privileges to multiple pricing entities (Bulk Create Privileges):**

Use the Bulk Create Privileges page to quickly create and assign privileges to multiple entities such as price lists or modifiers. For example, you could search for all price lists belonging to the operating unit of *Vision France* and then use the bulk create privileges feature to grant them all Maintain access. Alternatively, as a shortcut, you could create an entity set for the entities to be changed, and use the bulk update to update the entity set. The changes are then applied to all entities within that entity set.

1. Do a search by entity type, then select the pricing entity or entities to be granted privileges.
2. Click Next to display the Bulk Create Privileges: Provide Additional Privileges Information page, and complete your entries:
  - Entity Type and Entity Name: Select the Entity Type and Entity Name of the pricing entity to be granted privileges.
  - Grantee Types/Grantee Name: Select one of the following:
    - Responsibility: Grants the privilege to a specific responsibility such as Pricing User, Guest User (the specific Grantee Names depend on the setup for your specific business).

- User: Grants the privilege to a specific user such as John Smith in the Pricing Department.
- Global: If Grantee Type is Global, leave Grantee Name blank. This makes the privilege available to all users with functional access to pricing menus.
- Operating Unit: Grants the privilege to a specific operating unit. For example, select Vision1 to give a privilege to all users that have Vision1 as the default operating unit.
- Access Level: Select the access level to be granted to the grantee:
  - Maintain: Enables users to delete, view, and update pricing entities.
  - View Only: Enables users to view but not update the pricing entity.
- Start and End Dates: Select the start and end dates. For example, to provide temporary access to a temporary employee, you could enter a start date of 02-Jul-2007 and an end date of 31-Aug-2007. Alternatively, accept the system dates.

## Creating Entity Sets

You can create a set of pricing entities that contain multiple pricing entities of the same entity type; for example, an entity set for price lists and an entity set for modifiers. This facilitates assigning privileges to the entire entity set rather than to each separate entity. Here are some examples of entity set usage:

- You could create an entity set to give Maintain access to a few individual users. You can then use the same set to give view-only access privilege to all other users.
- You could create an entity set consisting of all price lists for a specific customer, then grant maintain access to a specific user who is responsible for maintaining those price lists. Only that user would be authorized to view and maintain the price lists for that entity set.

To use entity sets, you need to:

1. Create an entity set using the Create Entity Set page. On this page, you can select only header level criteria to create the set.
2. Use the entity set as the grant object (with object type as ENTITY SET) and grant access roles to any grantee type and grantee.

You should identify the selected criteria in the description of the entity set. Once an entity set is created, you cannot copy or update it. If changes are required, a new entity set must be created.

**Note:** You can revoke or add privileges as needed. However, the entity set cannot be deleted if any existing privileges are on that entity set. The Entity Set feature is available only to licensed users of Oracle Advanced Pricing.

For entity sets, consider the following guidelines:

- You can create an entity set for a specified set of criteria that does not currently exist in the system.
- You can create access privileges for this entity set even when no records currently exist in the system.
- Any new records that are created that meet the set criteria are automatically assigned to the set and inherit the privileges that are assigned to the set.

If this entity set is used in an access privilege, the newly created entity will be included in the set and will have those privileges.

#### **Example of Entity Set Usage**

You create a new entity set named SET1 for all active modifiers for USD currency containing Wireless in the customer name. Next you query on the set name SET1 on the Entity Sets page. After clicking the Go button, no records are displayed in the Results region. This occurs because there are no records that currently exist meeting these criteria.

Next, you create a privilege for entity set SET1 and assign view-only access for the Vision Operating Unit. Next, a user creates a new modifier - MOD 1 in the USD currency for the customer Totally Wireless - and makes the modifier active.

The MOD 1 modifier will automatically be assigned to the SET1 entity set and will inherit view-only access.

#### **To create an entity set:**

1. In the Create Entity Set page, define your set criteria:
  - **Set Name and Description:** Enter a name that uniquely identifies the entity set that you are creating, and a description that is simple, meaningful, and includes all the criteria that is selected for this entity set. The criteria to define the set should be included in the description for the set.
  - **Pricing Entity Types:** Select a pricing entity type to be included in the entity set. Only one pricing entity type can be included in an entity set.

**Note:** An entity set can contain only one unique pricing entity type. For example, Entity Set1 cannot contain entity types of

both Standard Price List and Modifier.

- Pricing Entity Name: Select an operator - is, is not, contains, starts with, ends with - and then enter specific details about the pricing entity name to be included in the set. For example, if you select Pricing Entity Name is Summer Price List, then the price list that is named Summer Price List will be included in the entity set. (Assuming Standard Price List was selected as the pricing entity type.)
2. Optional Qualifier Criteria: Select criteria from the Add Criteria field to add additional criteria, and click the Add button. Add only the criteria that you need for your new entity set. Remember to add the additional criteria to the Set Description. Your entity set will include only those pricing entities exactly matching your criteria.

#### **To delete an entity set:**

To delete an entity set, you must first revoke all privileges on this set and then delete it.

1. Navigate to the Entity Sets page and do a search for an existing entity set.
2. In the Results: Entity Set(s) region, click the Delete icon to delete a specific entity set.
3. If the Delete icon is grayed out, the entity set still has privileges assigned to it. Before the entity set can be deleted, you must first revoke the privileges, and then delete the entity set.

## **Setting Security Profile Options**

You can set security profile options to define the default security privileges that are assigned to newly-created price lists and modifiers. These profiles should be left in default setting (maintaining current functionality) and not be changed until you have decided which users should have automatic privileges of View Only or Maintain whenever a pricing entity is newly created. These privileges are automatically created as soon as the creating user saves the new entity. Security access for *existing* pricing entities is set by the Oracle Pricing Administrator using pricing security.

The two security profile options, QP: Security Default Maintain Privilege and QP: Security Default ViewOnly Privilege, control the default access privileges that are assigned to newly-created price lists or modifiers only:

- QP: Security Default ViewOnly Privilege  
Controls the default *view-only* privileges for *newly created* price lists and modifiers. View and maintain responsibilities are controlled separately by different profile

options. This profile option enables you to set view-only privileges at one of the following levels: Global (Default), Operating Unit, Responsibility, User, or None. This controls which users (if any) can view newly-created price lists and modifiers.

- QP: Security Default Maintain Privilege

Controls the default *maintain* privileges for NEWLY CREATED price lists and modifiers. This profile option enables you to set maintain privileges at one of the following levels: Global (Default), Operating Unit, Responsibility, User, or None.

**Note:** If either of these default privileges is set to Operating Unit, the privilege is created for the operating unit that is specified in MO: Default Operating Unit profile and *not* the operating unit that is assigned to the pricing entity.

- QP: Security Control

The profile option QP: Security Control (read-only) displays the current setting of the security option for your entire installation (either on or off). This profile option value cannot be directly updated and can only be turned on using the concurrent program Security Control.

Before setting the security profile options and changing the defaulting privilege profiles, complete all security setup requirements. To change the access privileges for *pre-existing* price lists and modifiers, use the Security Privileges window.

## Additional Implementation Considerations

The following discussion will assist you in choosing the combination of profile option settings to meet your security policy.

### Resolving conflicts between multiple access levels

If the user has two different access privileges to the same pricing entity, the access level of Maintain always prevails. For example, if a pricing user has Maintain access at the User level to certain price lists, and view-only access at the Responsibility level, the user has Maintain privileges to those price lists.

In all cases, the highest access level (the Maintain access privilege) prevails over the View-Only privilege. This rule applies regardless of what operating unit ID the user is in.

### Security profile option settings compared

The following section lists possible combinations of security profile option settings that define the default view and maintain access privileges for newly created pricing entities. Review the combinations of profile option settings and select the combination that suits the requirements for your installation. When security is turned on, a price list and modifier that is newly created will be assigned the default view and maintain security privileges from the profile option settings.

**Security Profile ON: Behavior when you are creating a new pricing entity**

The following table shows behavior by combinations of profile settings when you are setting up new price lists and modifiers. Available values are None, User, Responsibility, Operating Unit, and Global.

<b>QP: Default View Only Privilege</b>	<b>QP: Default Maintain Privilege</b>	<b>Behavior while being created</b>	<b>After saving and exiting the Entity's (Price list or Modifier) setup windows</b>
None	None	Entity can be viewed and updated while being created.	1. The new entity cannot be viewed or updated by anyone.
None	User	Entity can be viewed and updated while being created.	2. The new entity can be viewed and updated only by the user who created it only.
None	Responsibility	Entity can be viewed and updated while being created.	3. The new entity can be viewed and updated only by users with the same responsibility as the user who created it only.
None	Operating Unit	Entity can be viewed/updated while being created.	4. The new entity can be viewed and updated by all users with the same default operating unit as the user who created the entity only.
None	Global	Entity can be viewed and updated while being created.	5. The new entity can be viewed and updated by all users.

**Security Profile ON: Behavior when you are creating a new pricing entity for combination: values for User**

The following table shows behavior by combinations of profile settings when you are setting up new price lists and modifiers. Available values are: None, User, Responsibility, Operating Unit, and Global.

<b>QP: Default View Only privilege</b>	<b>QP: Default Maintain Privilege</b>	<b>Behavior while being created</b>	<b>After saving and exiting the Entity's (Price list or Modifier) setup windows</b>
User	None	Entity can be viewed and updated while being created.	The user who created it can view the new entity. Nobody can update it.
User	User	Entity can be viewed and maintained by user who created it.	The new entity can be viewed and updated only by the user who created it only.
User	Responsibility	Entity can be viewed and maintained by user who created it.	Similar to the None/Responsibility settings, except that the user can still view the entity even if he or she is exempted from the responsibility.
User	OU	Entity can be viewed and maintained by user who created it.	Similar to None/Operating Unit settings, except that, the user can still view the entity even if he or she has a different default operating unit.
User	Global	Entity can be viewed and maintained by user who created it.	Same as None/Global settings. The new entity can be viewed and updated by all users.

**Security Profile ON: Behavior when you are creating a new pricing entity for combination: values for Responsibility**

The following table shows behavior by combinations of profile settings when you are setting up new price lists and modifiers. Available values are: None, User, Responsibility, Operating Unit, and Global.

QP: Default View Only Privilege	QP: Default Maintain Privilege	Behavior while being created	After saving and exiting the Entity's (Price list or Modifier) setup windows
Responsibility	None	Entity can be viewed and maintained by user who created it.	All the users can view the new entity with the same responsibility as the user who created it. Nobody can update it.
Responsibility	User	Entity can be viewed and maintained by user who created it.	All the users can view the new entity with the same responsibility as the user who created it. And, only the user who created it can update it.
Responsibility	Responsibility	Entity can be viewed and maintained by user who created it.	Same as None/Responsibility settings. The new entity can be viewed and updated only by users with the same responsibility as the user who created it only.
Responsibility	Operating Unit	Entity can be viewed and maintained by user who created it.	All the users can view the new entity with the same responsibility as the user who created it. And, all the users with the same default operating unit as the user who create it can also update it.

<b>QP: Default View Only Privilege</b>	<b>QP: Default Maintain Privilege</b>	<b>Behavior while being created</b>	<b>After saving and exiting the Entity's (Price list or Modifier) setup windows</b>
Responsibility	Global	Entity can be viewed and maintained by user who created it.	Same as None/Global. The new entity can be viewed and updated by all users.

**Security Profile ON: Behavior when you are creating a new pricing entity for combination: values for Operating Unit**

The following table shows behavior by combinations of profile settings when you are setting up new price lists and modifiers. Available values are: None, User, Responsibility, Operating Unit, and Global.

<b>QP: Default View Only Privilege</b>	<b>QP: Default Maintain Privilege</b>	<b>Behavior while being created</b>	<b>After saving and exiting the Entity's (Price list or Modifier) setup windows</b>
Operating Unit	None	Entity can be viewed and maintained by user who created it.	All the users with the same default operating unit as the user who created it can view the new entity. Nobody can update it.
Operating Unit	User	Entity can be viewed and maintained by user who created it.	All the users with the same default operating unit as the user who created it can view the new entity. Only the user who created it can update it.

<b>QP: Default View Only Privilege</b>	<b>QP: Default Maintain Privilege</b>	<b>Behavior while being created</b>	<b>After saving and exiting the Entity's (Price list or Modifier) setup windows</b>
Operating Unit	Responsibility	Entity can be viewed and maintained by user who created it.	All the users with the same default operating unit as the user who created it can view the new entity. All the users with the same responsibility as the user who created it can update it.
Operating Unit	Operating Unit	Entity can be viewed and maintained by user who created it.	Same as None/OU settings. The new entity can be viewed and updated only by users with the same default operating unit as the user who created the entity.
Operating Unit	Global	Entity can be viewed and maintained by user who created it.	Same as None/Global settings. The new entity can be viewed and updated by all users.

**Security Profile ON: Behavior when you are creating a new pricing entity for combination: values for Global**

The following table shows behavior by combinations of profile settings when you are setting up new price lists and modifiers. Available values are: None, User, Responsibility, Operating Unit, and Global.

QP: Default View Only Privilege	QP: Default Maintain Privilege	Behavior while being created	After saving and exiting the Entity's (Price list or Modifier) setup windows
Global	None	Entity can be viewed and maintained by user who created it.	All the users can view the new entity. But nobody can update it.
Global	User	Entity can be viewed and maintained by user who created it.	All the users can view the new entity. Only the user who created it can update it.
Global	Responsibility	Entity can be viewed and maintained by user who created it.	All the users can view the new entity. All the users with the same responsibility as the user who created it can update it.
Global	Operating Unit	Entity can be viewed and maintained by user who created it.	All the users can view the new entity. All the users with the same default operating unit as the user who created it can update it.
Global	Global	Entity can be viewed and maintained by user who created it.	Same as None/Global. The new entity can be viewed and updated by all users

The Oracle Pricing Administrator can assign or change ownership of a pricing entity using the Entity Usage page (or the Bulk Update Entity Usage feature from the Entity Usage page).

## Turning On Pricing Security

**WARNING:** The concurrent program QP: Security Control with Views Conversion turns pricing security on or off for your entire installation. If you are upgrading or freshly installing the security feature for the first time, ensure that you have completed the following setup and implementation steps before turning pricing security on or setting the default security profile options; otherwise, users will be unable to query any

price lists or modifiers in the pricing windows.

- Assess and map out the behavior that your business requires when a new price list or modifier is created.
- Assign an operating unit owner for existing pricing entities.
- Grant privileges at all levels based on your security policy and needs.

When security control is first turned ON, a Global check box appears in the header region of all price lists and modifiers. If the Global check box is enabled for the entity, then that entity is available across all operating units in your organization. The Global check box is visible to end-users and can be updated (cleared or selected) by users with Maintain access privileges.

You can update the Global check box for each price list and modifier one at a time, or do bulk updates in the Bulk Update Entity Usage page. For more information, see Assigning Pricing Entity Usage, page 6-11.

Prior to your turning security on, pricing entities are not identified by an operating unit. It is very important that the Oracle Pricing Administrator assigns ownership to all price lists and modifiers prior to upgrading to or implementing pricing entity security. You can use the Bulk Update Entity Usage feature in the Entity Usage page to assign or reassign global usage values.

After you turn pricing security on, the default operating unit is used if the Global check box is deselected.

The following table shows the behavior of existing pricing entities when pricing security is turned ON and no pre-security is assigned:

<b>QP: Default View Only Privilege</b>	<b>QP: Default Maintain Privilege</b>	<b>Privileges from Pricing Security Administrator</b>	<b>Behavior</b>
Not applicable	Not applicable	No privileges granted	Entity cannot be viewed or updated by anybody except the Oracle Pricing Administrator through the security management pages that are selected from the Oracle HTML user interface.

QP: Default View Only Privilege	QP: Default Maintain Privilege	Privileges from Pricing Security Administrator	Behavior
Not applicable	Not applicable	Maintain	Entity can be viewed and updated by the user with Maintain access privileges.

---

## Pricing Data Bulk Loader

This chapter covers the following topics:

- Overview of Pricing Data Bulk Loader
- Populating the Interface Table
- QP: Bulk Import of Price List Program

### Overview of Pricing Data Bulk Loader

The Pricing Data Bulk Loader application programming interface (API), which is available in Basic and Advanced Pricing, enables you to complete the following tasks:

- Import new price lists.
- Update and delete price list data.

This Pricing Data Bulk Loader API consists of a set of interface tables and a concurrent program. All validations and defaulting is done before the data is inserted or updated. The Pricing Data Bulk Loader can be used as an alternative to the price list user interface (UI) and Price List Business Object API for importing large volumes of price list data such as from a legacy system.

The following price lists are not imported by the Pricing Data Bulk Loader:

- Price lists that are specific to pricing agreements (List Type Code of AGR)
- Price lists that are created from sales agreements (List Source Code of BSO)
- Price lists that are created from service contracts (List Source Code OKS)

### Profile Options That Are Used in Pricing Data Bulk Loader

The following profile options are used with the Pricing Data Bulk Loader API:

- QP: Pricing Transaction Entity: When the Pricing Data Bulk Loader API is called, the price list data is picked up from the interface table when the Pricing Transaction Entity attribute (column) is either:
  - NULL
  - Same as the profile value
- QP: Source System Code: When the Pricing Data Bulk Loader API is called, the price list data is picked up from the interface table when the Source System Code attribute (column) is either:
  - NULL
  - Same as the profile value
- QP: Batch Size for Bulk Import: This profile value determines the number of records that are loaded into the memory for processing. For improved performance, set an appropriate value for this profile based on your hardware configuration. If the profile value is set too high, then the system may "suspend." The default value is 1000.

**Note:** For a Basic Pricing installation, only the Basic Pricing data is imported into the system.

## Related Topics

Examples of the Interface Table Setup, page 7-11

## Populating the Interface Table

As a first step, the Pricing Data Bulk Loader imports data from the following interface tables:

- QP\_INTERFACE\_LIST\_HEADERS: This table captures the Price List header data.

**Important:** START\_DATE\_ACTIVE, END\_DATE\_ACTIVE: When populating the interface tables ensure that the following format is used for the start and end dates: YYYY/MM/DD (for example, 2006/06/24):

Data type	Format
-----------	--------

Date	YYYY/MM/DD
DateTime	YYYY/MM/DD HH:MM:SS

- QP\_INTERFACE\_LIST\_LINES: Price list line data is captured in this interface table.
- QP\_INTERFACE\_QUALIFIERS: This table contains the header qualifiers that are associated with the price lists to be imported.
- QP\_INTERFACE\_PRICING\_ATTRIBS: The product and pricing attributes data is captured in this table. When updating the product value for a price list line, you can use the `product_attr_value` column and the `Product_attr_val_Dis` column in the `qp_interface_pricing_attribs` table.

## Attributes in the Interface Tables

The following key attributes in the interface tables control the import of the price list data:

- ORIG\_SYS\_HEADER\_REF: This attribute is used in all four interface tables. In the QP\_INTERFACE\_LIST\_HEADERS table, the value in this attribute uniquely identifies a price list header and refers to the primary key of the price list header record in the external/legacy system.

For the other interface tables, this attribute value determines the association of the respective entity with the price list header. This attribute is stored in the price list core tables as a map between the price list data and the data in the interface tables. It is used with Update and Delete actions for interface table records. Therefore, during Insert, the uniqueness of this attribute in the price lists data is checked.

- ORIG\_SYS\_LINE\_REF: This attribute is used in the tables: QP\_INTERFACE\_LIST\_LINES and QP\_INTERFACE\_PRICING\_ATTRIBS. Similar to the ORIG\_SYS\_HEADER\_REF, this field uniquely identifies a price list line. For updating and deleting a price list line, this attribute value identifies the price list line. Also the uniqueness of this field is checked during insert.
- ORIG\_SYS\_QUALIFIER\_REF: This attribute is used in the table: QP\_INTERFACE\_QUALIFIERS. Similar to the ORIG\_SYS\_HEADER\_REF, this field uniquely identifies a price list qualifier. For updating and deleting a qualifier, this attribute value is used to identify the qualifier. Also the uniqueness of this field is checked during insert.
- ORIG\_SYS\_PRICING\_ATTR\_REF: This attribute is used in the table:

QP\_INTERFACE\_PRICING\_ATTRIBS. Similar to the ORIG\_SYS\_HEADER\_REF, this field uniquely identifies a product/pricing attribute.

When you are updating and deleting, this attribute value identifies the product or pricing attribute. Also, the uniqueness of this field is validated during insert.

- **PROCESS\_STATUS\_FLAG:** This attribute is used in all four interface tables, and indicates the status of a record during the import process. The API processes the records only when the value of the attribute is P. The records with errors display a value of NULL in this field. The successfully processed records will have the value I in this field, but all the successful records are deleted at the end of the process, so no records will exist with this attribute value I.
- **INTERFACE\_ACTION\_CODE:** This attribute is used in all four interface tables. The attribute indicates the action to be performed on the record:
  - **INSERT:** Indicates that the entity needs to be inserted into the price list.
  - **UPDATE:** Indicates that the entity needs to be updated.
  - **DELETE:** Indicates that the entity needs to be deleted.

## Mandatory Columns for Bulk Loader

The following table specifies the mandatory columns for the pricing bulk loader interface:

**Mandatory Columns for Bulk Loader Interface Tables**

---

<b>Operation</b>	<b>Table</b>	<b>Mandatory Columns</b>
Create (INTERFACE_ACTION_CODE : INSERT)	QP_INTERFACE_LIST_HEADERS	ORIG_SYS_HEADER_REF  LIST_TYPE_CODE  NAME  CURRENCY_CODE  CURRENCY_HEADER_ID (If Multi Currency is installed)  ROUNDING_FACTOR  SOURCE_LANG  LANGUAGE  orig_org_id (if the QP security is on and global_flag is N then)  INTERFACE_ACTION_CODE  PROCESS_FLAG  PROCESS_STATUS_FLAG

---

Operation	Table	Mandatory Columns
Create (INTERFACE_ACTION_CODE : INSERT)	QP_INTERFACE_LIST_LINE	ORIG_SYS_LINE_REF ORIG_SYS_HEADER_REF LIST_LINE_TYPE_CODE ARITHMETIC_OPERATOR OPERAND OR price_by_formula_id OR generate_using_formula_id (Any one of three) PRICE_BREAK_TYPE_CODE (If list_line_type_code is 'PBH') RLTD_MODIFIER_GRP_TYPE E (If list_line_type_code is 'PBH') PRICE_BREAK_HEADER_REF F (If the line is a PBH child line) INTERFACE_ACTION_CODE E PROCESS_FLAG PROCESS_STATUS_FLAG

Operation	Table	Mandatory Columns
Create (INTERFACE_ACTION_CODE : INSERT)	QP_INTERFACE_PRICING_ATTRIBS	<p>ORIG_SYS_PRICING_ATTR_REF</p> <p>ORIG_SYS_LINE_REF</p> <p>ORIG_SYS_HEADER_REF</p> <p>INTERFACE_ACTION_CODE</p> <p>PROCESS_FLAG</p> <p>PROCESS_STATUS_FLAG</p> <p>If any one is present in the below set then all should be present</p> <p>[</p> <p>PRODUCT_ATTRIBUTE_CONTENT</p> <p>product_attribute OR</p> <p>PRODUCT_ATTR_CODE (Anyone of two)</p> <p>product_attr_value</p> <p>OR</p> <p>PRODUCT_ATTR_VAL_DISP (Anyone of two)</p> <p>]</p> <p>If any one is present in the below set then all should be present</p> <p>[</p> <p>PRICING_ATTRIBUTE_CONTENT</p> <p>PRICING_ATTR_CODE OR</p> <p>pricing_attribute (Anyone of two)</p> <p>PRICING_ATTR_VALUE_FROM_DISP OR</p> <p>pricing_attr_value_from (Anyone of two)</p> <p>PRICING_ATTR_VALUE_TO_DISP OR</p> <p>pricing_attr_value_to (Anyone of two)</p> <p>(if</p> <p>comparison_operator_code is BETWEEN)</p> <p>]</p> <p>PRODUCT_UOM_CODE</p>

Operation	Table	Mandatory Columns
Update (INTERFACE_ACTION_CODE : UPDATE)	QP_INTERFACE_LIST_HEADERS	ORIG_SYS_HEADER_REF  INTERFACE_ACTION_CODE  PROCESS_FLAG  PROCESS_STATUS_FLAG
Update (INTERFACE_ACTION_CODE : UPDATE)	QP_INTERFACE_LIST_LINES	ORIG_SYS_LINE_REF  INTERFACE_ACTION_CODE  PROCESS_FLAG  PROCESS_STATUS_FLAG
Update (INTERFACE_ACTION_CODE : UPDATE)	QP_INTERFACE_PRICING_ATTRIBUTES	ORIG_SYS_PRICING_ATTRIBUTE_REF  INTERFACE_ACTION_CODE  PROCESS_FLAG  PROCESS_STATUS_FLAG

## Updating or Deleting Price Lists Using the Bulk Loader

When you update or delete price lists, all original system reference columns must be populated on the interface tables to identify the record being updated. For example, to delete or update header records, specify `orig_sys_header_ref`, and for line records, specify `orig_sys_header_ref` and `orig_sys_line_ref`.

### Internal Price Lists or Pricing APIs

For internal price lists created from pricing user interfaces or from pricing APIs where original system reference columns were not entered by the user, the system defaults these columns. For you to populate the original system reference columns for data created prior to the R12 release, you must run the concurrent program *QP: Upgrade Pricing data for Bulk Loader*. For data to be updated or deleted on these internal price lists, these defaulted values must be populated in original system reference columns on the interface table:

The `Orig_sys_header_ref` column on all entities (list header, list lines, pricing attributes, qualifiers) defaults to `list_header_id` prefixed by INT (for Internal).

**Important:** The prefix INT is only for the upgraded price list headers from 11.5.10 and earlier releases. For price lists created in 12.0 and later releases, the prefix INT is not added. For example, if the `header_id` is 12345, the `orig_sys_header_ref` for all entities for that price list will default to INT12345.

Other original system reference columns on list line, pricing attribute, and qualifier entities, default to the primary key ID value. As an example, `orig_sys_line_ref` defaults to the `list_line_id`. If there are duplicate `orig_sys_header_ref`, the prefix "INT-D-" is added to the `list_header_id`.

Here are some examples of how the naming conventions are applied:

Price List Header Name	List_header_id	Orig_sys_hdr_ref (After Naming Convention Applied)
PL_11510	12345	INT12345
PL_120	67890	67890
PL_120	67890	INT-D-67890

Note: If the `Orig_sys_hdr_ref` = 67890 already exists for another price list, then the prefix "INT-D-" is added.

**Notes:**

- If you are using the bulk loader operation to update a price break child line and to delete the price break simultaneously (a usage not recommended), then both the price break and the price break child line are deleted. Similarly, you cannot update the price break child line and delete the pricing attribute because a price break child line must have a pricing attribute. The correct usage is to delete (using DELETE action) both the line and the pricing attribute so that no error message appears.
- For the table `qp_interface_list_headers`, do not use the INSERT and UPDATE operations for the same price list at the same time (the bulk loader cannot create and update the price list header information in the same run). For example, the following setup shows the INSERT and UPDATE operations being used at the same time:

INTERFACE_ACTION_CO	HEADER_ID	ORIG_SYS_HEADER_REF
DE		

INSERT	—	123
UPDATE	—	123
UPDATE	—	123

Line 1: Create a new price list with `orig_sys_header_ref` of 123.

Line 2: Update a price list with `orig_sys_header_ref` of 123.

Line 3: Same as line 2.

### To Update Selected Columns:

For updating selected columns, you need to enter only updated column values on the interface tables. For example, to update only the operand column for a price list line, specify the original system reference columns to identify the record and new value for the operand column on the list lines interface table.

To update a column to a null value populate that column in the interface table with one of the following constants:

- `QP_BULK_LOADER_PUB.Get_Null_Date` for column of datatype Date
- `QP_BULK_LOADER_PUB.Get_Null_Char` for column of datatype Varchar2
- `QP_BULK_LOADER_PUB.Get_Null_Number` for column of datatype Number

**Note:** To do validations, the Pricing Data Bulk Loader populates all the empty columns on the interface tables (set to current values in the database) with the current data in the database. In the previous example, all columns other than the operand column will be updated by the bulk loader. If errors occur, you might see all columns populated after the bulk loader is run even though the user only populated the operand column.

If you are creating, updating or deleting a child record, the parent record (and grandparent record if applicable) must be present in the interface table. Even if the parent (and grandparent) record does not need any processing (insert, update, or delete) it should still be present with `process_status_flag = I` and original system columns populated. If the header record does not exist, child entity records will not be processed by the bulk loader.

**Table Names and Upload Action Validations**

Table Name	Select	Insert	Update	Delete
QP_INTERFACE_LIST_HEADER	X	NA	X	X
QP_INTERFACE_LIST_LINES	X	NA	X	X
QP_INTERFACE_QUALIFIERS	X	NA	X	X
QP_INTERFACE_PRICING_ATTRIBS	X	NA	X	X
QP_INTERFACE_ERRORS	NA	X	NA	NA
QP_LIST_HEADERS_B	NA	X	X	X
QP_LIST_HEADERS_TL	NA	X	X	X
QP_LIST_LINES	NA	X	X	X
QP_QUALIFIERS	NA	X	X	X
QP_PRICING_ATTRIBUTES	NA	X	X	X
QP_RLTD_MODIFIERS	NA	X	X	X

**Note:** NA = Not applicable

## Examples of Interface Table Setup

The following sample scripts show how to set up the interface tables for the Pricing Data Bulk Loader feature. The scripts are located in the `$QP_TOP/patch/115/sql` directory:

- QPBLKEX1.sql: Script to insert price list header and price list line.
- QPBLKEX2.sql: Script to insert price list header and price list line and pricing attributes.
- QPBLKEX3.sql: Script to insert price list header and price break line.
- QPBLKEX4.sql: Script to update price list header with qualifiers.

- QPBLKEX5.sql: Script to attach secondary price lists to the primary price list.

## QP: Bulk Import of Price List Program

The Pricing Data Bulk Loader API is implemented as the concurrent program QP: Bulk Import of Price List. Using the concurrent program, you define the pricing data to be imported (bulk loaded), then run the concurrent program to import the price list data from interface tables into the Oracle Advanced Pricing tables. To improve processing efficiency, you can schedule the concurrent program to run at optimal times, such as when no active users are on the system.

The QP: Bulk Import of Price List program enables you to complete the following tasks:

- Do bulk validations and bulk loading.
- Provide all validation error messages in the same run for records that were not uploaded rather than returning errors one after another.
- Provide the ability to multi-thread the interface loader similar to Order Import with the ability to multi-thread within the same price list.

You can set the profile option QP: Batch size for Bulk Upload (QP\_BATCH\_SIZE\_FOR\_BULK\_UPLOAD) to restrict the number of records that are read to the PL/SQL tables. This helps to improve the bulk import performance. The default value for the profile is 1,000, but you can change this value to suit your hardware configuration. For more information, see the *Advanced Pricing Implementation Guide*, Profile Options.

### Scenarios for Using the QP: Bulk Import of Price List Program

Your business practices may require that you upload price list data on a regular basis, such as in the following scenarios:

- When a new price list must be created by uploading a price change file.
- When new products lines are created, new offerings to new channels, acquisitions of product lines or companies, new product rollout, and when new price lists are created.
- When you are implementing a new sales territory, to bulk load existing price lists or to upload pricing from regions that may have their pricing on spreadsheets.

### Processes performed by the QP: Bulk Import of Price List Program

The QP: Bulk Import of Price List program completes the following processes:

- Creates and updates price list headers, lines, pricing attributes, qualifiers, and breaks using the interface tables.
- Deletes price list lines, pricing attributes, qualifiers, and breaks using the interface tables.

- (Optionally) Uses the Process Parent parameter in the concurrent program to load list lines even if the pricing attribute validation fails. If the Option is NO, and a pricing attribute fails, both parent and child lines are not uploaded.
- Stores the interface program ID in the main pricing tables so that you know which batch ID updated the records.

This program updates only the IDs that are derived from Value To ID conversions back to the interface tables. If a record generates errors, the record remains in the interface tables where you can view the IDs.

- Marks a record as being processed so that multiple processes do not pick up the same record.
- Automatically deletes the successfully uploaded rows from the interface tables.
- Processes a specific set rather than the entire eligible data.

**Note:** For more information on populating the interface tables for bulk loader, see *Oracle Advanced Pricing Implementation Guide*.

### **Considerations for Running the QP: Bulk Import of Price List Program**

Review the following considerations before running the QP: Bulk Import of Price List program:

- Source System/Pricing Transaction Entity (PTE): The concurrent program retrieves records only when the source system code and the PTE code match with the same profile values or null.
- Price lists that are attached to the following entities are not retrieved for bulk loader processing:
  - Sales Agreements (list source code of BSO)
  - Service Contracts (list source code of OKS)
  - Pricing Agreements (list type code of AGR)
- Importance of Log and Output file: The request output contains the following information:
  - Number of records processed
  - Number of successful records
  - Error messages (if applicable)

The concurrent request log file provides the processing details. The log file also lists

debug messages if Debug On parameter was set to Yes. If you enter an entity name that is not unique for the eligible records, none are selected for processing.

- Child lines, parents, and grandparent records must be populated to insert, update, and delete records: To insert, update, or delete records, ensure that the corresponding parent and grandparent records (if applicable) are also in the interface table so they are eligible for retrieval by the concurrent program.
- Errors before resubmit: For records with errors in the interface tables, set the request\_id and process status flag columns to NULL so that these records are selected in the next run. When the Entity Name is specified as a Report parameter, you do not need to reset these values.
- Data for error messages in a QP interface table: The table QP\_INTERFACE\_ERRORS is populated with the processing error messages that are then written to the output file of the concurrent request. Occasionally, you must purge this table of this data.

### Report Submission

In the Submit Request window, Name field, select QP: Bulk Import of Price List.

### Report Parameters

#### QP: Bulk Import of Price List Parameters

Entity	PRL	Price List
Entity Name		
Process ID		
Process Type		
Process Parent	Yes	
No of Threads	1	
Turn Debug On	No	
Enable Duplicate Line Check	Yes	

- Entity: Indicates which pricing entity is to be processed. The default value is PRL (price list).
- Entity Name: This is an optional parameter. Enter the name of the price list to be processed. If this field is left blank, the system processes all entities of the selected entity type that are available for processing in the interface tables.

If you enter a price list name, that price list is considered for import even if it failed in a previous import and regardless of the other flags (for example, even if

PROCESS\_STATUS\_FLAG not equal to P, or PROCESS\_FLAG not equal to Y or REQUEST\_ID not equal to NULL in the QP\_INTERFACE\_LIST\_HEADER record).

- Process ID: This value corresponds to the process\_id in the interface tables. The Process ID groups data in the interface table. You can select criteria to specify which records to process.

If this field is left blank, all available records for processing are considered subject to the restrictions entered in the Entity Name field.

- Process Type: This field corresponds to the process type that is listed in the interface tables such as XML. If specified, the bulk loader processes only the records with PROCESS\_TYPE column matching the value entered here.
- Process Parent: Select Yes (the default) or No to determine if the loader should process a price list line when validation for a child pricing attribute fails. Valid values are Yes (the default) and No.
- No of Threads: Enter a number that indicates the total number of child processes or threads to achieve multithreading for list lines and their child records. The default value is 1.

Multithreading occurs when the child concurrent processes are initiated after a user submits the Process Parent request. The parent process first processes the list header records and then the qualifiers records before spawning child processes. The child processes upload price list lines and attributes.

**Note:** Multithreading can impact the length of processing time that is required; therefore, you should not exceed the number of Central Processing Units (CPUs) that are available when running the program. Ideally, you should schedule this process for a slow processing period or when few users are on the system.

- Turn Debug On: Select Yes or No (default):
  - Select No to improve loading performance; however error messages are not sent to the Debug log.
  - Select Yes to have all error messages sent to the Debug Log.

The Debug On parameter controls only the log files. However, error messages, number of records processed, and other related information is written to the output file, regardless of the value set for Debug On.

- Enable Duplicate Line Check: The bulk loader can upload thousands of lines into the pricing (QP) lines table. If Enable Duplicate Line is set to Yes, then the QP: Bulk Import of Price List checks for duplicates lines and displays an error message if

duplicate lines are found. If it is set to No, then the duplicate check is not done. The default value is Yes.

---

## Price Lists

This chapter covers the following topics:

- Overview of Price Lists
- Bulk Importing of Price Lists
- Usage Price Break Proration

### Overview of Price Lists

Price lists are essential to ordering products because each item entered on an order must have a price. Each price list contains basic list information and one or more pricing lines that define prices for all levels of your product hierarchy: at the item level, category level, or any level that you define. Basic price list information includes the price list name, effective dates, currency, pricing controls, and shipping defaults such as freight terms and freight carrier. For a price list, you can define price breaks, pricing attributes, qualifiers, and secondary price lists.

Oracle Advanced Pricing provides an HTML-based user interface (UI) that features guided steps, user-friendly pages, and shortcut links for setting up and maintaining modifiers, price lists, and qualifiers. You can use this HTML format for many tasks that were previously available using only the Oracle forms-based UI.

Information about the following price list topics (for both the HTML UI and the forms-based UI) is available in the *Oracle Advanced Pricing User's Guide*, Price Lists chapter:

- Creating price lists and price list lines
- Creating price breaks for a price list line.
- Updating price lists and price lists lines
- Adding and adjusting items on a price list
- Using the Price List Maintenance feature (HTML interface)

- Using secondary price lists
- Creating a GSA price list
- Copying a price list and price list lines
- Archiving, deleting, and purging price list information
  - You can only view or update price lists for your pricing transaction entity.
    - To view and update the price list, the pricing transaction entity (PTE) for the price list and the profile option QP: Pricing Transaction Entity must match.
    - You can only view or update a price list only in your source system. To update the price list, the source system for the profile option QP: Source System Code and the price list must match. However, if the source system of the price list is different than the value defined in the profile but within the same pricing transaction entity, then the user can only view the price list.
 

Price rounding considerations: the profile option QP: Selling Price Rounding Options affects the rounding of list price and adjustments. For more information on this and other profile options, see: *Oracle Advanced Pricing Implementation Guide*, Profile Options.
  - Copying price breaks created in a release before R12: When you copy a price break (for example, by copying a price list or modifier list with price breaks) from a pre-R12 release, the price breaks are updated to the price break format described in the preceding section. You need to review the changes to ensure that the converted price break setup meets your pricing objectives.

## Pricing for Hierarchical Item Categories

You can set up prices and modifiers based on hierarchical item categories and flattened item categories. You must define category-based pricing for hierarchical categories for price lists or modifiers; however, to do this, you must set a default hierarchical category set (catalog) for the functional area of the source system creating the data.

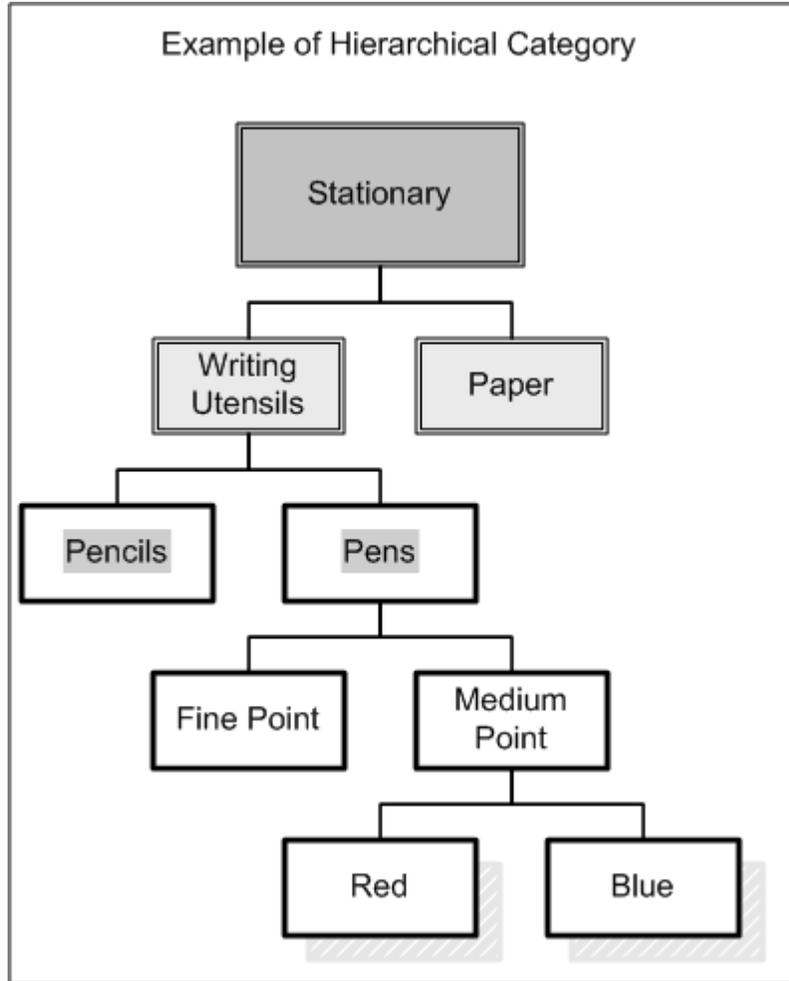
- **Hierarchical category**

In a hierarchical category, you can set up pricing for segments of the hierarchy because each segment is a distinct category.

For example, consider the following hierarchical category:

- Stationery > Writing Utensils > Pens > Fine Point/Medium Point > Red/Blue

**Example of Hierarchical Category**



Your pricing can be at any level of the product hierarchy. For example, you can give a 5 percent discount for Stationery, a 10 percent discount for Writing Utensils, and a 20 percent discount for Pens. If the Item ABC belongs in the Pens category, and you defined the above 3 discounts in the setup, then all 3 discounts would be applied. If Item XYZ belongs in the Pencils category, then only the 5 percent and 10 percent discounts would apply because Pencils falls under the Stationery and Writing Utensils categories.

- **Flattened category**

A flattened category represents a category and its subcategories in a single group. For example, consider a flattened category of Computers.Hardware.Keyboard.

Wireless. If Item A belonged to this category, then you could set up pricing for only the Computers.Hardware.Keyboard.Wireless category not for each item or sub-category. To set pricing for segments of a flattened category, use attribute mapping to source each segment as a product attribute or define hierarchical categories. Flattened categories can be defined from the Item Manager responsibility in the forms-based and HTML user interfaces.

If you are creating category-specific price lists and modifiers, then you can select (from the Product field) the categories associated with those enabled functional areas for a given source system for that pricing transaction entity (PTE). You can select from hierarchical or flattened categories because one or more functional areas can be defined for a source system. The default category set (or catalog) associated with the functional area may be hierarchical.

**Note:** Item categories for price lists are always based on the item category of the item at the Master level and not at the Org (organization) level.

## Related Topics

*Oracle Advanced Pricing User's Guide, Price Lists chapter*

## Bulk Importing of Price Lists

The Pricing Data Bulk Loader API, which is available for Basic and Advanced Pricing, enables you to:

- Import new price lists.
- Update and delete the price list data.

This API consists of a set of interface tables and a concurrent program that enable you to import large volumes of price list data (for example, from a legacy system).

## Related Topics

Pricing Data Bulk Loader API, page 7-1

## Usage Price Break Proration

You can prorate price breaks to support the following business scenarios:

- To simplify the definition of usage pricing, and use one price break definition for different billing periods.

- For contracts that start in the middle of a billing period.
- For contracts that are terminated in the middle of a billing period.

Some bills are based on predefined cycles; for example, a service provider may bill at the end of every calendar quarter. However, a customer may want to start service with that provider on a date other than the quarter start date. Therefore, the contract start date would fall into the middle of a billing period.

One usage line on a contract may have a complex billing schedule. For example, a bill for one period of 15 days, followed by two periods of 1 month, then four periods of 1 quarter. However, the usage line would be associated with only one pricing definition in the price list.

That pricing definition may be based on monthly usage. So when the 15 days or the quarter billing periods must be billed, the usage breaks must be prorated to suit the circumstances.

Prorating price breaks is optional and controlled by the profile QP: Break UOM Proration Allowed. This profile must be set to Yes to use this feature.

**Example 1: Price Break by Quarter**

The following table displays the conversion for a break defined by quarter:

- Requested Break UOM = Month
- Conversion Factor = 1/3

**Value From/To Conversion**

Value From/To Before Conversion)	Value From/To (After Conversion)
	<b>Note: Prorated breaks will not be truncated.</b>
0 - 5	0 to 1.666...6
5 - 10	1.666...6 to 3.333...3
10 - 20	3.333...3 to 6.666...6
20 - 99	6.666...6 to 33

**Example 2: Price Break by Month**

The following table displays the conversion for a break defined by month:

- Requested Break UOM = Quarter

- Conversion Factor = 3

***Value From/To Conversion***

<b>Value From/To (Before Conversion)</b>	<b>Value From/To (After Conversion)</b>
0 - 5	0 - 15
5 - 10	15 - 30
10 - 20	30 - 60
20 - 99	60 - 297

If the incoming value is 15, which belongs to the first two breaks, the first break is selected because the calculation is greater than Value From and less than and equal to Value To.

---

## Price Book

This chapter covers the following topics:

- Overview of Price Book
- Implementation Steps for Price Book
- Setting Up Price Book Profile Options
- Setting Up the E-mail Server
- Setting Up the Default Printer
- Setting Up Oracle XML Publisher
- Setting Up the XML Gateway Message Maps
- Setting Up the Price Book User Interface (UI)
- Confirm Pricing Parameter Setup

### Overview of Price Book

You can generate a price book for a specific customer to publish lists of products and their prices. You can create a *full* price book for all selected products or a *delta* price book that shows only the items for which prices have changed since the corresponding full book was published. To create a price book, you select the items that you want to see on the price book, select a publishing option, then run a concurrent request (either immediately or at a predetermined time) to generate the price book. You can select from various printing and delivery options:

- Print the price book by either:
  - Viewing the document online and printing it.
  - Specifying the printer in the publishing options of the price book. This uses the Delivery Manager of XML Publisher (a component of the Oracle E-Business Suite).

- Send the price book in e-mail as an attachment to the external customer using the Delivery Manager option of the XML Publisher.
- Publish the price book using the available seeded templates in the following formats:
  - Excel spreadsheet
  - PDF (Portable Document Format)
  - RTF (Rich Text Format)

**Note:** You can also define your own template using XML publisher.

- Send as an XML message.

Alternatively, the input criteria for generating the price book can also be accepted by an OAG XML Transaction (GET\_CATALOG\_002) from the trading partner (external customer). The price book is then transmitted to the trading partner as the OAG XML Transaction (SYNC\_CATALOG\_003). The technology stack components that are involved in generating a price book includes Oracle XML Gateway, Oracle Workflow Business Event System, and Oracle Supply Chain Trading Connector.

Typically, both internal and external customers can create price books for a variety of business reasons as described in the following scenarios:

#### **Internal Customers**

- Pricing Analyst or Pricing Super-user: Can generate price book to define pricing reports for audit or planning purposes and for ongoing maintenance activities for pricing rules.
- Product Managers: Product managers who want to define list prices and dealer or distributor prices could generate price books to review existing prices and generate hard copies of prices that are related to a specific product or product category for a customer.
- Sales Managers: Sales managers who call on customers or want to review existing prices can use price books to generate hard copies of prices that are related to a specific customer.

#### **External Customers**

A specific customer of a dealer may want to look up prices on a vendor or dealer web site and view a published list of prices. Using a Price Book API, the customer can retrieve the price book information and make it available on their web site.

These are some typical scenarios in which external customers may use price books:

- Dealers and distributors of a client company want to look up prices they must pay to a vendor.
- Distributors wanting to create price books for general pricing or for specific customers who want to know the current prices on the day the price book is generated.
- Companies want to publish price books for future dates: If a price increase will go into effect October 1st, the company could print and mail out price books in September.

## Workflow Business Events

For price books, Oracle Advanced Pricing provides seeded workflows that are associated with business events . A workflow - a sequence of activities that can be performed either automatically or by end-user intervention - can help automate your business process. When a defined business event occurs, such as booking an order, it triggers a related workflow process.

- `oracle.apps.qp.pricebook.catso`: This event is raised by the pricing concurrent program that generates or initiates the publishing of price books.
- `oracle.apps.qp.pricebook.catgi`: This event is raised by the XML Gateway (ECX) when it receives the GET\_CATALOG\_002 message from a Trading Partner.

Oracle Pricing uses the following workflows that are seeded in the Oracle Supply Chain Trading Connector (CLN) application for the collaboration history component:

- `oracle.apps.cln.ch.collaboration.create`: This event is linked to the pricing concurrent program that generates or initiates the publishing of price books. A CLN workflow for this event creates tracking information for the price book collaboration in the collaboration history tables.

## Price Book Public APIs

Advanced Pricing provides the following public application programming interfaces (APIs) for the price book feature: Create Price Book API and Get Price Book API. Applications that are integrated with Advanced Pricing can call these APIs to generate or retrieve a price book. The following list describes the price book public APIs:

- `Create_Price_Book` API: Use this API to create, overwrite, and publish new or existing price books (either full or delta price books). Based on the input criteria, a concurrent request is submitted by the system (a concurrent request is also submitted when you are creating price books either through the price book feature or Get\_Catalog). The Request ID, a return status, and any errors that are encountered during input parameter validation or when you are submitting a concurrent request will be returned as output parameters. Once you have created

the price book, use the `Get_Price_Book` API to retrieve it.

- `Get_Price_Book` API: Use this API to retrieve an existing price book by entering the following input parameters: price book name, `price_book_type`, and customer.

## Related Topics

*Oracle Order Management Open Interfaces, API, & Electronic Messaging Guide*

## Implementation Steps for Price Book

Before using the price book, complete the following implementation steps:

1. Set up price book profile options, page 9-4
2. Set up the e-mail server, page 9-5
3. Set up a default printer, page 9-5
4. Set up the Oracle XML Publisher, page 9-5
5. Set up the Oracle XML Gateway, page 9-6
6. Set up the price book user interface (UI), page 9-8
7. Confirm pricing parameter setup, page 9-8

## Setting Up Price Book Profile Options

The price book feature uses the following profile options to define the default settings:

- QP: E-mail From Address: Defines the *from e-mail* address for any e-mails that are sent from the price book feature.
- QP: External Default Pricing Perspective: Sets the default pricing perspective for all external users.
- QP: Inbound XML Messaging Responsibility: Defines the responsibility to be used for inbound XML messages.
- QP: Internal Default Pricing Perspective: Sets the default pricing perspective for all internal users
- QP: Pricing Perspective Request Type: Defines the mapping between a pricing perspective (calling application) and the request type it uses.

For information about setting up and using the profile options, see *Oracle Advanced*

## Setting Up the E-mail Server

When sending a price book as e-mail, you enter the e-mail addresses in the Email Address field on the Create Price Book page. After the price book is generated, it will be sent to the specified e-mail addresses using the e-mail server that is specified in the configuration. Price books that are sent by e-mail use the default e-mail server. To specify the e-mail server, you must add it to the Oracle XML Publisher Delivery Manager configuration and specify it as the default entry. For information about how to configure the Delivery Manager, see the *Oracle XML Publisher User's Guide*.

**Note:** When sending the price book as e-mail, you must select a template which defines the layout and appearance of the price book that you e-mail.

## Setting Up the Default Printer

When printing the price book to a specific printer, enter the printer name in the Printer field on the Create Price Book page. If no printer name is specified, then the default printer that is defined in the configuration is used. To define the default printer, you must add the printer for your site to the Oracle XML Publisher Delivery Manager configuration and specify it as the default entry. You can also set up additional printers by adding more entries into the configuration. For more information about configuring the Delivery Manager, see the *Oracle XML Publisher User's Guide*.

**Note:** The printer name that you enter in the Printer field on the Create Price Book page must correspond to a printer server name in the Oracle XML Publisher Delivery Manager configuration. When printing a price book, you also must select a template that defines the layout and appearance of the printed price book.

## Setting Up Oracle XML Publisher

Using Oracle XML Publisher, you can publish your price books in a variety of formats such as PDF, RTF, and Excel.

**Note:** The Delivery Manager component of the Oracle XML Publisher (XDO) product is required to directly publish a price book to a printer or e-mail attachment. The Delivery Manager component with the publishing options to print and send e-mail is available in XML Publisher 5.0.

### **To select an XML Publisher Template:**

Oracle Advanced Pricing provides seeded XML Publisher templates that you can use to generate your price books. You can use Oracle XML Publisher to create and maintain additional templates that define the layout for the published price book. You may want to create multiple templates depending on the desired publishing option (e-mail or print) and the desired output format (Excel, PDF, RTF). For example, you may want a separate template for print versions and another for .PDF output.

When printing or sending the price book by e-mail, you select the appropriate template for the chosen publishing option. If you do not want to show the List Price for products, you can copy one of the seeded price book templates and then remove the List Price column (or any other column) from your new template.

## **Setting Up the XML Gateway Message Maps**

Price book requires the following XML gateway message maps to generate and transmit price book information to and from external customers:

- GET\_CATALOG\_002 message map: To receive input criteria from the trading partner (external customer) and generate the price book by means of an OAG XML transaction.
- SYNC\_CATALOG\_003 message map: To transmit the price book to the trading partner by means of the OAG XML transaction.

XML Gateway Message Maps are created in the XML Gateway Message Designer and stored as .xgm files. These maps are loaded into XML Gateway as seed data. XML Messaging also requires a Trading Partners and a Trading Partner Detail record each for the Get Catalog and Sync Catalog messages.

### **Map for Get\_Catalog\_002 XML message**

The XML Gateway Message Map, QP\_CATGI\_OAG72\_IN.xgm, is provided to get the GET\_CATALOG\_002 XML document. The following tables are mapped to the incoming XML Message:

- QP\_PB\_INPUT\_HEADERS\_B
- QP\_PB\_INPUT\_HEADERS\_TL
- QP\_PB\_INPUT\_LINES

The GET\_CATALOG\_002 XML document conforms to the following Open Applications Group (OAG) document type definition (DTDs), 129\_get\_catalog\_002.dtd and the oagis\_extensions.dtd. These and other relevant DTDs (oagis\_domains.dtd, oagis\_entity\_extensions.dtd, oagis\_fields.dtd, oagis\_resources.dtd, oagis\_segments.dtd) are loaded to the XML Gateway, and are located at \$qp/xml/oag.

Tables and views that are used by this map are ECX\_OAG\_CONTROLAREA\_TP\_V (provided by XML Gateway for the Control Area Mapping), and the tables listed

previously.

### **Map for SYNC\_CATALOG\_003 XML message**

The XML message SYNC\_CATALOG\_003 is an OAG standard that provides an electronic method of publishing a price book for business collaborations involving Oracle Applications. The SYNC\_CATALOG\_003 enables a seller to publish product catalogs from Oracle Advanced Pricing to the buyer.

The seller uses SYNC\_CATALOG\_003 to publish a new price book. If the price book information changes, the seller can re-publish the price book so the buyer can replace the price book with the more recent price book.

An XML Gateway Message Map, QP\_CATSO\_OAG72\_OUT.xgm, is provided to generate the SYNC\_CATALOG\_003 XML document and has the following inputs:

- Price Book Header: Generated using QP\_PRICE\_BOOK\_HEADERS\_V
- Price Book Lines: Generated using QP\_PRICE\_BOOK\_LINES\_V (price book summary lines)
- Price Book Line Details: Generated using QP\_PRICE\_BOOK\_LINE\_DETAILS\_V (price list lines and modifier lines)
- Pricing Attributes of the Price List Lines: Generated using QP\_PRICE\_BOOK\_ATTRIBUTES\_V (price list lines)
- Price Break Lines of Price Break Header Modifiers and Price List Lines: Generated using QP\_PRICE\_BOOK\_BREAK\_LINES\_V (price list lines and modifier lines of type Price Break Header)

The SYNC\_CATALOG\_003 XML document conforms to the following OAG DTDs: 128\_sync\_catalog\_003.dtd and the oagis\_extensions.dtd. These and other relevant DTDs (oagis\_domains.dtd, oagis\_entity\_extensions.dtd, oagis\_fields.dtd, oagis\_resources.dtd, and oagis\_segments.dtd) are located at \$qp/xml/oag. Tables and views that are used by this map include ECX\_OAG\_CONTROLAREA\_TP\_V (provided by XML Gateway for the Control Area Mapping), and the views listed previously.

### **Setting up the XML Gateway Trading Partners:**

To enable the inbound GET\_CATALOG\_002 and outbound SYNC\_CATALOG\_003 XML messages for an external customer, you must first define the customer as a trading partner in the Oracle XML Gateway Trading Partner Setup. In this setup, add the following two transactions to the trading partner:

```
GET_CATALOG_002
  Transaction Type = QP
  Transaction SubType = CATGI
  Standard Code = OAG
  External Transaction Type = CATALOG
  External Transaction Subtype = GET
  Direction = IN
  Map = QP_CATGI_OAG72_IN

SYNC_CATALOG_003
  Transaction Type = QP
  Transaction SubType = CATSO
  Standard Code = OAG
  External Transaction Type = CATALOG
  External Transaction Subtype = SYNC
  Direction = OUT
  Map = QP_CATSO_OAG72_OUT
```

For more information about setting up a trading partner, see the *Oracle XML Gateway User's Guide*. You can use the responsibility Workflow Administrator Web Applications > Transaction Monitor to monitor the status of the XML messages.

## Setting Up the Price Book User Interface (UI)

If your company uses customer item numbers and customer item descriptions, then you may want to customize the Price Book Details page. Hidden columns are available for the customer item number and the customer item description. Enabling these columns will enable your users to see the customer item number and description when available for an item.

## Confirm Pricing Parameter Setup

Price book uses the pricing parameter Price Book Pricing Events (QP\_PRICE\_BOOK\_PRICING\_EVENTS) to associate pricing events with request type codes for use with price book engine calls. Typically, you do not need to change the seeded value.

This pricing parameter is at the Request Type level. You can associate more than one pricing event with a request type by separating the events with commas. Ensure that the events that are used as values for this pricing parameters are valid, applicable pricing events.

---

## Modifiers

This chapter covers the following topics:

- Overview of Modifiers
- Implementation Planning
- Modifier Levels and Application Methods
- Other Modifier Considerations
- Pricing Controls
- Modifier Type Setup
- Setting up Accrual Discounts
- Using Accumulated Range Breaks

### Overview of Modifiers

Modifiers are pricing actions that drive your pricing processes in addition to price lists, formulas, and agreements. Modifiers can adjust net price either up or down. Modifier actions include discounting, adding surcharges, charging for shipping, changing order terms (payment, freight, or shipping method) or adjusting price based on promotional pricing actions.

**Key Implementation Decision: How do I use modifiers in pricing actions? Which modifiers best demonstrate my pricing processes?**

Oracle Advanced Pricing provides seeded modifier lists and modifier line types. The modifier list types are: discount, surcharge, deal, promotion, and freight and special charges.

The following table displays modifier types that you can create within a modifier list.

Modifier Type (Top row)	Discount List	Surcharge List	Deal (must be associated with a promotion)	Promotion	Freight and Special Charges List
Discount	Yes	Yes	Yes	Yes	No
Surcharge	Yes	Yes	Yes	Yes	No
Other Item Discount	No	No	Yes	Yes	No
Terms Substitution	No	No	Yes	Yes	No
Item Upgrade	No	No	Yes	Yes	No
Price Break	Yes	Yes	Yes	Yes	No
Promotional Good	No	No	Yes	Yes	No
Coupon Issue	No	No	Yes	Yes	No
Freight and Special Charge	No	No	No	No	Yes

**Important:** You can view or update modifier lists only for your pricing transaction entity.

- To view and update the modifier list, the pricing transaction entity (PTE) for the modifier list and the PTE set for the profile option QP: Pricing Transaction Entity must match.
- You can only view or update a modifier list in your source system. To update the modifier list, the source system for the profile option QP: Source System Code and the modifier list must match.

## Pricing for Hierarchical Item Categories

You can set up prices and modifiers based on hierarchical item categories and flattened item categories. You can define category-based pricing for hierarchical categories for

price lists or modifiers; however, for you to do this, a default hierarchical category set (catalog) must be set for the functional area of the source system creating the data. For more information, see Pricing for Hierarchical Item Categories, page 8-2.

## Using the HTML User Interface with Modifiers and Modifier Lines

Oracle Advanced Pricing provides an HTML-based user interface (UI) that features guided steps, user-friendly pages, and shortcut links for setting up and maintaining modifiers, price lists, and qualifiers.

In the HTML user interface, you can create the following modifier list and modifier line types:

### Modifier List Types

- Deal
- Discount List
- Promotion
- Surcharge List

**Note:** You must use the Oracle Forms-based user interface to create a Freight and Special charge List.

### Modifier Line Types

Once you have created a modifier list, you can add modifier lines that define the type of price adjustments and benefits that the pricing engine applies to pricing requests. The following modifier line types are available in the HTML UI:

- Discount: Creates a negative price adjustment.
- Promotional Goods: Adds a new item to the order line with a price adjustment or benefit when the customer orders a particular item on the same order.
- Price Break: Applies a variable discount or surcharge price adjustment to an eligible break type. You can use both point and range type breaks.
- Surcharge: Creates a positive price adjustment. For example, a 10 percent handling charge is applied to a customer order.

## Related Topics

*Oracle Advanced Pricing User's Guide, Modifiers*

## Implementation Planning

As you analyze and understand a client's business pricing scenario, you need to address some key implementation decisions to develop a logical pricing solution. These decisions will be discussed throughout the chapter. Visualizing the price bands of an organization may assist you in developing an effective implementation plan for Oracle Advanced Pricing.

The following example provides a visual guide for mapping the modifiers of an organization to the components of Oracle Advanced Pricing.

The source pricing data is based upon the assumptions that item Super Wine has an item category of *Wine*. An order for a quantity 15 was placed on *15-Jun-2000* for a customer *XYZ Corporation*, who belongs to a customer class *VIP*.

Bucket	Pricing Phases	Incompat. Group	Qualifiers/ Product Attributes (Precedence)	Modifiers/ Price List	Disco unt	Sub Total
Base Price	List Line Base Price	EXCLUSIVE	Customer Class - VIP (310) Item Code - Super Wine (220)	Corporate List - \$1000	Null	Null
Base Price	List Line Base Price	EXCLUSIVE	Item Category - Wine (290)	Preferred Vendors - \$800	Null	Null
Base Price	List Line Base Price	Null	Null	Null	Null	\$1,000
Bucket 1	List Line Adj.	Level 1	Item Quantity > 10 (800) Item Code - Super Wine (220)	4 <sup>th</sup> July Promotion 10%	\$100	Null
Bucket 1	List Line Adj.	Level 1	Customer Class - VIP (310)	Summer Promotion 15%	Null	Null
Bucket 1	List Line Adj.	Level 2	Customer id - XYZ Corp (260)	VIP discount \$40	\$40	Null

Bucket	Pricing Phases	Incompat. Group	Qualifiers/ Product Attributes (Precedence)	Modifiers/ Price List	Discount	Sub Total
Bucket 1	List Line Adj.	Level 2	Customer id - XYZ Corp (260)	Weekday Discount \$20	Null	Null
Bucket 1	All Line Adj.	Level 1	Item Category - Wine (290)	General Discount - \$20	\$10	Null
Bucket 1	All Line Adj.	Exclusive	Order Date < 01-Dec-2000 (510)	Seasonal Discount - \$10	Null	Null
Bucket 1	Header Level Adj.	Level 1	Null	Preferred Customer 10%	100	Null
Bucket 1	Line Charges	Level 1	Null	Handling Charge \$20	\$20	Null
Bucket 1	Header Level Charges	Level 1	Null	Null	Null	Null
Bucket 1	Header Level Charges	Null	Null	Null	Null	\$750
Bucket 2	Adj.	Level 2	Null	High usage surcharge 2%	(\$15)	Null
Bucket 2	All Line Adj.	Level 2	Null	Null	Null	Null
Bucket 2	Header Level Adj.	Level 2	Null	Null	Null	Null
Bucket 2	Line Charges	Level 2	Null	Null	Null	Null

Bucket	Pricing Phases	Incompat. Group	Qualifiers/ Product Attributes (Precedence)	Modifiers/ Price List	Discount	Sub Total
Bucket 2	Header Level Charges	Level 2	Null	Null	Null	Null
Bucket 2	Header Level Charges	Level 2	Null	Null	Null	Null
Bucket 2	Null	Null	Null	Null	Null	\$765

**Note:** Key to Table Short Names

Adj. = Adjustments

## Modifier Levels and Application Methods

The pricing engine uses modifier levels to determine pricing request eligibility for specific modifiers and to determine at which level the modifier should be applied: Order, Line, or Group of Lines.

- Line: Applies only to a specific order line.
- Group of Lines: Applies to groups of lines in the order.
- Order: Applies to an entire order.

For example, a volume-based discount modifier applied at the line level will consider only volume on the qualifying line. A volume-based discount modifier applied at the group of lines level will process the volume across all qualifying lines on the pricing request.

### Key Implementation Decision: How do I assign modifier application methods?

You can select from the available application methods to create different pricing actions:

- Amount: Creates a fixed price adjustment on each unit for the amount that is specified in the value field.
- Percentage: Creates a percentage price adjustment on each unit for the percentage that is specified in the value field.

- New price: Overrides the selling price of the item and makes it the new price.
- Lumpsum: Creates a price adjustment for the entire sum amount of the line.

The application methods that are available depend on the modifier line type and level selected. This can be found on the Discounts/Charges tab. You can use a formula to drive the value of your pricing action.

The following tables depict the application methods allowed based on various modifier levels (the application method is listed in the top row).

**Modifier Application Methods: Line Level**

Modifier Line type (column below)	Per. Value	Per. Form.	Amt. Value	Amt. Form.	New Price Value	New Price Form.	Lmpsm. Value	Lmpsm. Form.
Discount	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Surcharge	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Freight/ Special Charge	Yes	Yes	Yes	Yes	No	No	Yes	Yes
Other Item Discount (Get)	Yes	No	Yes	No	Yes	No	Yes	No
Price Break	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Promotional Good (Get)	Yes	No	Yes	No	Yes	No	Yes	No

**Note:** Key to Table Short Names

- Amt. = Amount
- Form. = Formula
- Lmpsm = Lumpsum
- Per. = Percent

**Modifier Application Methods: Group of lines Level**

<b>Modifier Line type (column below)</b>	<b>Per. Value</b>	<b>Per. Form.</b>	<b>Amt. Value</b>	<b>Amt. Form.</b>	<b>New Price Value</b>	<b>New Price Form.</b>	<b>Lmpsm. Value</b>	<b>Lmpsum. Form.</b>
Discount	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Surcharge	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Other Item Discount (Get)	Yes	No	Yes	No	Yes	No	Yes	No
Price Break	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Promotional Good (Get)	Yes	No	Yes	No	Yes	No	Yes	No

#### **Modifier Application Methods: Order Level**

<b>Modifier Line type (column below)</b>	<b>Per. Value</b>	<b>Per. Form</b>	<b>Amt. Value</b>	<b>Amt. Form.</b>	<b>New Price Value</b>	<b>New Price Form.</b>	<b>Lmpsm. Value</b>	<b>Lmpsm. Form.</b>
Discount	Yes	Yes	No	No	No	No	No	No
Surcharge	Yes	Yes	No	No	No	No	No	No
Freight/ Special Charge	No	No	No	No	No	No	Yes	Yes

## **Considerations for Group of Lines Behavior**

The following section outlines special considerations when applying discounts at the Group of Lines level for the following modifier types: Discounts, Other Item Discount, Price Break Header, and Promotional Goods.

For modifier types at the Group of Lines level, be aware of the following considerations:

- You can select only the Volume Type of Item Quantity and Item Amount.
- If some excluded items or item categories are set up, then order lines that are based on an excluded item or item category that qualify for this modifier, are not

considered for group of lines calculation.

**Example: Group of Lines processing and Excluded Items**

To demonstrate how a modifier evaluates order lines at the "group of lines" level (with an excluded item), two modifiers have been set up:

- Modifier\_Shampoo1 is set up without any excluded items.
- Modifier\_Shampoo2 is set up with an excluded item "Shampoo1."

The setup for both modifiers is summarized in the following table:

Modifier Name	Level	Item Category	Volume	Excluded Item
Modifier_Shampoo1	Group of Lines	Shampoo	> 100 Qty	None
Modifier_Shampoo2	Group of Lines	Shampoo	> 100 Qty	Shampoo1

Suppose that a customer then placed an order for two shampoo items (Shampoo1 and Shampoo2) and conditioner as outlined in the following table:

Order Line ID	Item Category	Item	Quantity
1	Shampoo	Shampoo 1	70
2	Shampoo	Shampoo 2	40
3	Conditioner	Conditioner	30

Both modifiers Modifier\_Shampoo1 and Modifier\_Shampoo2 are then applied against the Shampoo order. However, each modifier evaluates the group of lines differently based on the modifier setup (one modifier considers the exclude item and the other does not) as described in the following outline.

**Modifier\_Shampoo1 applied to Shampoo order**

Before Modifier\_Shampoo1 can be given, the lines from the Shampoo order are evaluated:

- Order lines 1 and 2 qualify for Modifier\_Shampoo1 because the items are from the Item Category of Shampoo.
- The quantity of the order lines 1 and 2 that qualify are summed to determine whether the ordered quantity is greater than 100. The sum is 110 which is greater than 100.

As a result of this check, the modifier Modifier\_Shampoo1 is given because the:

- Item category for the considered items is Shampoo.
- Ordered quantity of the lines for Item Category shampoo is greater than 100.

#### **Modifier\_Shampoo2 applied to Shampoo order**

Before Modifier\_Shampoo2 can be given, the order lines are evaluated in the following way:

- Order line 2 qualifies for Modifier\_Shampoo2.
- Order line 1 is rejected because the item Shampoo1 is an exclude item (see the set up for Modifier\_Shampoo2 where Shampoo1 is listed as the Exclude Item, and is therefore excluded).
- The quantity of order line 2 (the only order line that qualifies so far) is summed to determine whether the ordered quantity is greater than 100. The sum is 40 which is less than 100.

As a result of this check, the modifier Modifier\_Shampoo2 is not given because the ordered quantity is not greater than 100.

## **Related Topics**

Functional Areas for Source System region, page 17-29

## **Other Modifier Considerations**

### **Key Implementation Decision: How will my customer and product hierarchies affect the structure of my modifiers?**

Using qualifiers, you can tie the structure of your modifiers to your customer hierarchy. For more information about this topic, see Implementation Methodology, page 3-1.

You can set up modifiers for item number, item category, or all items. Through pricing attributes, you can further define your product hierarchy.

## **Excluding Items and Item Categories**

By selecting exclude, you can exclude individual items or item categories from being applied to a modifier. If an item is in an excluded item category and in a non-excluded item category, the excluded item category is honored. If a hierarchical category is specified in excluded items, then any items that belong in the child categories will be excluded, as well as any end items that are associated with that category.

For example, a modifier for all items excludes item category "IC 1." You have an item Z that is in IC1, IC2, and IC3. When the engine assesses that Item Z is in IC1, the modifier is

not applied, even though the item is in IC2 and IC3.

## Unit of Measure (UOM)

No UOM conversions are available for modifiers. The pricing engine evaluates only modifier lines that have matching UOMs or null value that is selected for the pricing UOM. For example, item A has a UOM of each with primary UOM checked for the price list line. The ordered UOM for item A is dozen. The engine will therefore consider only modifier lines with each or with null values. UOM is not a mandatory field for modifier types other than price breaks.

**Note:** The UOM is not mandatory for promotional modifiers if the volume type is Item Amount. The UOM is mandatory only in the following cases:

- List line type is Promotional Goods, Other Item Discount, or Item Upgrade and the volume type is not Item Amount, Period1 Item Amount, Period2 Item Amount, or Period3 Item Amount.
- The Modifier line volume type is Item Quantity.

## Pricing Phase

The pricing phase determines when the pricing engine applies a modifier. For example, modifiers in the list line adjustment phase are applied only after leaving the line. Modifiers in the all lines adjustment phase are applied only after saving the line. Consider unnecessary pricing engine calls that can affect performance.

## Precedence

The values for precedence will default based on the setup of the product attribute (such as item number and item category) and any qualifier attributes that are defined in the Context and Attributes window. For more information, see Implementation Methodology, page 3-1 and Precedence and Best Price, page 16-1.

## Incompatibility Level

You can make modifiers incompatible with each other by placing them in the same incompatibility group level. When this is done, the pricing engine applies only one modifier per phase to the order line or order. For more information on incompatibility levels, see Precedence and Best Price, page 16-1.

## Buckets

Pricing buckets control how discounts and other benefits are calculated across phases.

Grouping discounts and benefits into buckets helps determine the net selling price across all pricing phases.

### Null Bucket

Modifiers that are assigned to a Null bucket are applied last and always adjust from the list price. Order level modifiers must be in the Null bucket. The pricing engine uses the following steps when calculating the selling price for Null buckets:

1. Calculates percent discounts using the list price.
2. Sums all bucket modifier values to create a bucket subtotal.
3. Applies the subtotal after the last numbered bucket.

Suppose that the following buckets are set up, each with a different adjustment:

- Null bucket: a 50% discount is assigned to the Null bucket.
- Bucket 1: a 10% discount is assigned.
- Bucket 2: a 10% surcharge is assigned.

The buckets are used to calculate the price for an SP ATO Model with a list price of \$55. The following table shows how the final price is calculated:

Bucket	Modifier Adjustment	Calculated Price	Final Price
Bucket 1	10% discount	$55 - (10\% * 55)$	\$49.50
Bucket 2	10% surcharge	$49.5 + (10\% * 49.5)$	\$54.45
Null bucket	50% discount. Note: The modifiers are applied last and taken out of the list price.	$54.45 - (50\% * 55)$	\$26.95

## Key Implementation Decision: How many bucket levels do I need?

Determine the number of buckets that you need by counting subtotals on your client's invoice. Control the modifier calculation sequence by adding the numbers between beginning list price and final net price. Plan the assignment of discounts to buckets based on these subtotals.

The null bucket calculates the modifier off of the list price and then applies the adjustment to the last bucket's subtotal. Order level modifiers are always in the null bucket.

Oracle Advanced Pricing does not restrict the number of buckets that you can define.

## Manual Modifiers

If a manual modifier is applied for a given bucket, all modifiers in subsequent buckets will get re-evaluated.

## Pricing Controls

The following pricing controls can be set up to define how the modifier and modifier lines are applied:

- Incompatibility occurs when the pricing engine finds more than one modifier to return but only one can be applied. The engine resolves this incompatibility using either precedence or best price.
- Automatically Apply: Select Automatically Apply to have the modifier applied automatically by the pricing engine. Certain modifiers must be applied automatically and in these cases, you cannot change the value.
- Customer Must Ask For List (or Ask For): Modifiers with the Customer Must Ask For List box selected are given only if requested by the user through calling applications. This field can be used when the list type is Promotion or Deal.
- Override: Select Override to enable the modifier adjusted price to be manually overridden. If this box is not selected, then the modifier adjusted price cannot be overridden. The Override box appears only for those types of modifier that supports override.
- Effectivity dates at the modifier list and modifier line: Effectivity dates on the modifier line must fall in between the effectivity dates for the modifier list.
- Additional date types for order date and requested ship date: These are additional parameters that can be set to control the effectivity of the modifier list.
- The Comparison Value field: This field holds the approximate value for the benefit item(s) for item upgrade, term substitution, coupon issue, other item discount, and promotional good. This value is used during best price processing (the field does not impact Oracle General Ledger).
- Calculate Price flag: This is passed by the calling application to enable the calling application to fully or partially freeze a price request. Depending on the value of the flag, the price can be completely frozen, or additional modifiers can be applied in certain phases.

**Warning:** Do not change the Calculate Price field to Calculate Price on a free good item.

# Modifier Type Setup

## Manual Modifiers

You can define manual adjustments for discounts, surcharges, freight charges, and point price breaks. To override the selling price directly on the order line, you must define a manual discount or a manual surcharge. Remember the following guidelines when defining manual adjustments:

- The Automatic box must be cleared.
- The Override box must be selected for overrideable manual adjustments.
- Buckets are available for manual modifiers.
- Manual adjustments are applied based on pricing phase.
- The engine returns manual discounts based on the value that is defined in the profile option QP: Return Manual Discounts.
  - If the profile option is set to Y, then all manual discounts are returned and all automatic discounts that are not considered are returned as manual discounts. This is the default.
  - If the profile option is set to N, then all manual and automatic discounts undergo incompatibility processing and one per incompatibility group is returned.

Discounts (automatic or manual) that are deleted as part of incompatibility processing are returned as manual discounts.

- For manual discounts that use formula calculation, all attributes for the formula must be sent to the pricing engine in the pricing request. The engine returns the manual discount, which can be applied to the order line.
- On the Manual Adjustments field for the unit selling price on the sales order line, *only* line and line group manual adjustments appear.
- After applying manual adjustments to the line the calculate\_price\_flag remains Y. If you do not want other adjustments applied to the line, set the calculate\_price\_flag to P or N. If you want to apply order level manual adjustments, do so through the View Adjustments screen. If the calculate\_price\_flag on any of the lines is P or N, order level adjustments cannot be applied to the order. If no applicable manual adjustments are available for a line, a note appears that indicates that no applicable discounts are available.

To apply a manual adjustment in Oracle Order Management, see the *Oracle Order*

*Management Suite Implementation Manual, or the Oracle Order Management Suite User's Guide.*

## Other Item Discount

For an other item discount to apply, all items must be correctly sequenced on the order. The pricing phase must be tied to an event that considers all lines in the order. Other item discounts may not be manual or overrideable. Other item discounts ignore incompatibility for the get line.

### Get Region and Benefit Items

Benefit items in the Get region can be defined only at the item level. Users should not enter values for the get price and get UOM fields. The price of the benefit item should be on the order.

Because the pricing engine does not support recurring other item discounts, get quantity should be 1. The engine will apply the discount to all quantities of the item. For example, for every 5 pastries and 2 cookies you order, you get 2 cookies at 50% off. Using this modifier, if you order 10 pastries and 4 cookies, you receive 4 cookies at 50% off. Similarly, if you order 10 pastries and 3 cookies, all 3 cookies are 50% off.

If you discount using the percent application method, percent is based on the bucket value that is defined in the modifier summary tab. For example, if the other item discount is in bucket 2, then a 5% discount is given after all adjustments for bucket 1 are applied to the get item.

## Promotional Goods

A promotional good such as *Buy 1 PC and get a free mouse, or buy 1 PC and get free speakers* can be set up in the following ways using modifiers:

### Method 1

1. Set up two modifiers:
  - Modifier A: Buy 1 PC and get 1 free mouse.
  - Modifier B: Buy 1 PC and get free speakers.
2. Make the two modifiers incompatible with each other and set the precedence so that the modifier with the higher precedence will be automatically applied.

### Method 2

1. Set up two Other Item discount modifiers:
  - Modifier A: buy 1 PC and buy 1 mouse, then get the mouse free.
  - Modifier B: buy 1 PC and buy speakers, then get the speakers free.
2. Make these two modifiers incompatible with each other and then set the

precedence. When the PC is ordered and the mouse or speakers are added to the order, one of these items is free.

**Note:** In the Get region of the Define Modifier Details window, only price list lines that are standalone price list lines are displayed in the list of values. All service items and all price break lines (headers and children) are not displayed and cannot be selected from the list of values.

Benefit items that are set up in the Get region can only be defined only at the Item level.

The pricing engine returns an additional order line for the promotional good and sets the Calculate Price Flag to No. Additional modifiers are not applied to the line unless the flag value is changed to Yes or P. This prevents negative selling prices on free good items. If order lines exist where the Calculate Price flag is set to No, then the application does not apply the additional order level modifiers.

**Note:** Promotional goods are not supported at the Order level.

**Warning:** Do not change the Calculate Price Flag to Calculate Price on a free good item.

### Promotional Service Items

After you define a line level promotional modifier for an item (for example, a PC) in the Define Modifier window, you can add a service item (for example, warranty for a specific period) as a 'promotional get item' for the parent item. Use the Get region in the Define Modifier Details window of the parent item to specify the following for the promotional get item:

- The service item quantity.
- The UOM, which the application uses as the primary UOM for conversion.
- The duration of the service program such as 1, 2, or more.
- The period of the service program such as day, month, or year.
- The percent price for the service item.

**Note:** The calling application such as Oracle Order Management decides the start date and end date of the service program. If you define the service duration and period for a different UOM other than the one defined on the get item line, then the pricing engine returns the appropriate conversion to the calling application. This

set up adds the service line item automatically when users of calling applications such as Order Management create a sales order for a parent line item that qualifies for a promotional service get item.

### **Promotional Subscription Service Items**

After you define a line level promotional modifier for an item (for example, a PC) in the Define Modifier window, you can add a subscription service item (for example, a year of support) as a "promotional get item" for the parent item. Use the Get region in the Define Modifier Details window of the parent item to specify the following for the promotional get item:

- The subscription service item.
- The Subscription Service check box must be selected.
- The duration of the subscription service such as 1, 2, or more.
- The period of the subscription service such as day, month, or year.
- The subscription template such as, day, week, month, year.

**Note:** The calling application such as Oracle Order Management decides the start date and end date of the service program. This set up adds the service line item automatically when users of calling applications such as Order Management create a sales order for a parent line item that qualifies for a promotional subscription service get item.

### **Term Substitution**

Payment, freight, and shipping terms can be substituted using the terms substitution modifier.

- The modifier level is always line or order level.
- Define product attribute by item number or item category for a term substitution modifier at the line level.

### **Item Upgrade**

For item upgrade modifiers, the relationships between items and upgraded items must be defined in Oracle Inventory.

- Promotional upgrade is the relationship type.
- The UOM for the upgraded item will be the same as the UOM of the original item.
- The pricing engine recognizes only the original item for additional modifiers. It never recognizes the upgraded item.

## Price Breaks

Price breaks can be created for both price list lines and modifier lines with the Pricing Context of Volume.

Price breaks have the following characteristics:

Price Break Characteristic	Example
Consist of price break lines without gaps between subsequent break ranges.	0 to 100 (greater than 0 but less than or equal to 100).  100 to 200  200 to 300  <b>Note:</b> For example, a quantity of 100.0 would be eligible for the first price break range (0-100); however, a quantity of 100.1 would be eligible for the second price break range (100-200).
The Value From of the first break range starts with 0.	0 to 100
The Value From of the current price break matches the Value To from the previous break (applies to breaks other than the first break range).	0-100 100-200 200-300

**Important:** Copying price breaks created in a release before Release 12: When you copy a price break (for example, by copying a price list or modifier list with price breaks) from a pre-Release 12 release, the price breaks are updated to the price break format that is described in the preceding section. You need to review the changes to ensure that the converted price break setup meets your pricing objectives.

If the ordered quantity does not fall in the price range of the price list, the price is still

calculated and defaulted in the sales order line. Therefore, ensure that the price ranges are set up correctly to get the pricing results you desire. For example, suppose you create the following range price breaks for price list Item AS54888 (Application Method is Unit Price):

- Value From/To: 0 to 5 Price: 1.00 each
- Value From/To: 5 to 7 Price: .75 each

If Price Type is Point: The price is not returned if the ordered quantity does not fall in the price range of the price list.

If Price Type is Range: A price of 0 is retrieved for the ordered quantity that does not fall in the price range.

For example, if a Sales Order for 10 items of AS54888 were placed, the price would be calculated as follows (Price Type is Range):

$$(5*1) + (2*0.75) + (3*0)/10 = 0.65 \text{ Unit Price}$$

Notice that the last 3 items fall out of the price range and therefore a unit price of 0 is returned. Therefore it is important that the price breaks are set up correctly to get the desired price results.

## Types of Price Breaks

When setting up price breaks, you can choose from either point or range breaks to determine price break processing:

### Point Price Break

Consider the following example discount rules:

- Item quantity ordered 0-10: 1% discount
- Item quantity ordered 10-50: 2% discount
- Item quantity ordered 50-999: 5% discount

If this is a line level price break and the ordered quantity is 55, then a 5 percent discount is applied to the order line. If this is a group of lines modifier, the total quantity over all the group of lines on the order receive the discount.

### Range Price Break

Using the point price break example, if the ordered quantity is 55, then the first 10 items ordered receive a 1% discount, the next 40 receive a 2% discount, and the remaining 5 receive a 5% discount. The discount is then averaged and applied to the order line.

### Accumulated Range Breaks

Accumulated range breaks extend the regular range break feature for modifiers. You can use accumulated range breaks to enable calling applications such as Oracle Order Management to pass accumulation values to the pricing engine using accumulation attributes. For more information about setting up and using accumulated range breaks, see the *Modifiers chapter* in this guide.

**Important:** For Oracle Order Management, when you split a line, the selling price changes if the volume falls into a different price bracket. At that point, the outcome depends on the value of the Calculate Price Flag prior to the split. To prevent the price from automatically changing, set the Calculate Price Flag field to freeze price prior to splitting the line.

## Coupon Issue

Two modifier lines must be set up when you are defining a coupon issue in the modifiers window. The price adjustment or benefit is linked to the coupon issue. When you are setting up the coupon issue line, a modifier number is mandatory. The pricing engine uses the number to generate a unique number series for the coupon which it passes to the calling application. The customer quotes the number when redeeming the coupon in the calling application.

## Ask For Promotions

You can create "Ask For" promotions by selecting one of the following when setting up a modifier:

- Customer Must Ask For List box (HTML UI)
- Ask For box: (Forms-based UI)

**Note:** This field can be used when the list type is Promotion or Deal.

When the Ask For box is selected, the customer must "ask for" the promotion by supplying a promotion name or number.

The pricing engine validates order eligibility for the asked for promotion after the order is sent to the pricing engine.

You can also set up both automatic and manual asked for promotions. The methods in which the calling application applies asked for promotions depends on the calling application.

## Recurring Modifiers

You can mark modifiers as recurring in the break type field on the modifier summary tab. The following modifier types enable recurring:

- Discount
- Surcharge

- Promotional good (additional buy items do not recur)
- Coupon issue

The following example illustrates a recurring coupon issue: for every 5 items ordered, get a coupon for 10% off a future order. If you order 15 items, then you receive 3 coupons.

## Setting up Accrual Discounts

Both monetary and non-monetary accruals can be created through the modifier setup window. Accruals are given as a percentage, amount, or lumpsum and do not affect order line selling price. Accruals can have expiration dates attached to them and do not appear as chargeable items on invoices. Accrual accumulation is stored in the OE\_Price\_Adjustments table. You can make reference to accruals may be made by selecting the Accrual flag.

Remember the following when setting up accrual discounts:

- Select the Accrual box on the Discounts/Charges tab.
- Benefit quantity and benefit UOM are of the benefit that you want to accrue.

**Note:** For Coupon Issue modifiers, the Benefit Quantity and Benefit UOM fields are not supported.

- Expiration date specifies when the accrued transactions expire (optional).
- For the expiration period and expiration type, the expiration period begins when the item begins to accrue. The pricing engine will calculate the expiration date.

**Note:** The expiration period and expiration type fields cannot be entered if expiration date has been entered.

- Accrual conversion rate is used to specify the conversion of the benefit UOM to the primary currency. The following example illustrates a conversion rate: if one air mile is 0.50 currency units, the accrual conversion rate is 0.50. The UI window exists for marking accruals as redeemed.

### Fields reserved for future use:

- Rebate Transaction
- Percent Estimated Accrual Rate

### Buckets with Monetary Accruals

Because accruals do not affect selling price on the order line, the pricing engine does not

include accruals in bucket calculations. The engine uses bucket numbers to determine the price with which to calculate accrual value. The following table illustrates this concept.

Bucket	Modifier	Price Adjustment	Accrual Amount	Bucket Subtotal	Selling Price
List Price	n/a	n/a	n/a	n/a	\$100.00
1	7% discount	\$7.00	n/a	\$7.00	\$93.00
1	10% accrual	n/a	\$10.00	n/a	\$93.00
2	10% accrual	n/a	\$9.30	n/a	\$93.00
2	\$5 discount	\$5.00	n/a	\$5.00	88.00

### Accounting Limitations

- Accrual discounts are accounted for in the same manner as regular discounts in Oracle Advanced Pricing, Oracle Order Management, and Oracle Accounts Receivable. Oracle General Ledger account information is currently not used in Oracle Advanced Pricing modifier and accrual redemption forms.
- Oracle General Ledger account information from Oracle Order Management cannot be passed to Oracle Accounts Receivable without a workaround (for example, Oracle Trade Management) using standard memo lines for auto invoicing. Discounts are expensed as a sales expense and accruals are deferred expense. Each implementation will require establishment of Oracle General Ledger accounts and mapping information flow of accounting for discount expenses.
- No interface exists for Oracle Accounts Receivable and Oracle Accounts Payable to research available accruals, balance, and decrementing for payments.
- Currently only net revenue is posted to the Oracle General Ledger. This posting occurs between Oracle Order Management and Oracle Accounts Receivable products.

### Redeeming Accruals

Accruals can be reviewed and marked as redeemed in the accrual redemption screen. This screen is an online view that reflects all accrual records from transactions. It can be used to view all redeemed and unredeemed records, or you can view using a more specific query, such as modifier number, customer, or redeemed only.

Querying any customer name or customer number returns all redeemed and unredeemed records for the customer regardless of modifier number. Querying by modifier returns specific modifiers with records for all customers who qualify for it.

Transaction type and source system are populated by any automated redemption processes. This reflects both manual redemption and those created by other transaction sources such as Oracle Accounts Receivable, Oracle Accounts Payable, or other source systems. You can manually update accrual redemption by keying in:

- Transaction reference: credit memo, check number, or write off codes
- Payment system: Oracle Accounts Receivable, Account Payable, or other system
- Redeemed date (defaults as current date, but can be changed by user)

## Using Accumulated Range Breaks

Accumulated range breaks extend the regular range break feature of modifiers by using an "accumulation value" to start a break point instead of zero.

The pricing engine evaluates regular breaks and accumulated range breaks differently:

- Accumulated range breaks: The accumulation value is used as the starting point when the pricing engine is evaluating a break.
- Non-accumulated (regular) price breaks: Regular breaks are evaluated starting from zero.

Calling applications, such as Oracle Order Management, pass accumulation values to the pricing engine using accumulation attributes. The accumulation attributes are used in modifier setups, and are attached to a modifier price break header.

For example, an order for a quantity of five of Item XYZ with an accumulation value of 8 would be priced at the appropriate range break(s) for the 9th through 13th items.

With accumulated range breaks, orders can be priced starting at higher-ranged breaks while with regular breaks, all orders start at the lowest break. See the *Oracle Advanced Pricing User's Guide* for examples and additional information about accumulated range break features.

The following methods are used to pass the accumulation value to the pricing engine:

- Attribute Mapping: This is the conventional method of passing attributes in a pricing request.
- Runtime Sourcing: This is set up in the Attribute Management feature to source an accumulation attribute at runtime. Runtime occurs when the modifier is selected and breaks are evaluated. However, unlike attribute mapping, runtime sourcing is used only for accumulated range breaks.

## Setting Up Runtime Sourcing for Accumulated Range Breaks

Unlike attribute mapping, runtime sourcing can be used only for accumulated range

breaks. During accumulated range break calculations, the pricing engine calls the Runtime Sourcing application programming interface (API) to acquire an accumulation value for the accumulation attribute (defined as RUNTIME SOURCE in the Attribute Mapping setup).

The RUNTIME SOURCED API is a PL/SQL function that returns the accumulation value. It is similar to Get\_Custom\_Price API in that the user must write the function, but is used only for accumulation. It is different from a standard attribute mapping rule because sourcing is done at runtime in the middle of calculation. Runtime sourcing is best used when runtime information (modifier-related values like qualifiers, product, and pricing attributes) is needed to derive an accumulation value.

The following list outlines the general steps for implementing and using runtime sourcing:

1. Define and write function Get\_numeric\_attribute\_value for package QP\_RUNTIME\_SOURCE, following the specification (provided).
2. Compile the source into the database.
3. Define a pricing attribute under VOLUME context with Mapping Method RUNTIME SOURCE in Attribute Mapping.
4. Enable the profile QP: Accumulation Attribute Enabled.
5. Define a modifier price break with this attribute attached in the Accumulation Attribute field.

When the modifier is selected, runtime sourcing is called to obtain an accumulation value before the adjustment amount is calculated.

### To set up runtime sourcing:

The specification for the function Get\_numeric\_attribute\_value is provided in the package QP\_RUNTIME\_SOURCE (located in the file QPXRSRCB.pls). You must write the corresponding body with the following parameters:

- p\_list\_line\_id: the list line ID of the modifier being applied.
- p\_list\_line\_no: the list line number of said modifier.
- p\_order\_header\_id: the ID that is assigned to the order, if supported by the calling application. This value may be null.
- p\_order\_line\_id: the ID that is assigned to the request line of the order.
- p\_price\_effective\_date: the price effective date that is given in the order.
- p\_req\_line\_attrs\_tbl: a table of records of the request line attributes.

- `p_accum_rec`: a record structure pertaining to the accumulation attribute.

The `p_req_line_attrs_tbl` table structure parallels that of the one that is found in package `QP_FORMULA_PRICE_CALC_PVT`.

The `p_accum_rec` record contains three fields, but can be expanded for any future requirements:

- `p_request_type_code`: the request type code that is specified in the pricing request.
- `context`: the accumulation attribute context, currently hard-coded as `VOLUME`.
- `attribute`: the segment name designation of the accumulation attribute.

### To apply runtime sourcing:

The Runtime Sourcing API returns the accumulation value for each line being priced. The accumulation can be done per order or across orders. Therefore, you must write code logic that returns the correct value based on any or all of the input parameters.

This user-defined logic can be based on any number of input parameters. For example, to accumulate based on the order item, then you need to evaluate only the attribute corresponding to Item Number on the request line needs to be evaluated. If the requirements call for a complex derivation involving multiple input parameters, the API enables this too, provided that you correctly implement the logic in the PL/SQL code.

By design, accumulation is done by the calling application and not by the pricing engine. Accumulation values are stored in user-defined tables, so any SQL queries within `Get_numeric_attribute_value` should be performed on these tables to get the values. Furthermore, leave any `UPDATE`, `INSERT`, or `DELETE` statements outside of runtime sourcing, because those operations should be done within the calling application itself.

#### Sample Runtime Sourcing Code

The following example shows sample code for the function body. In this scenario, the user is accumulating based on a combination of customer class and order type. Here, we define customer class as context `CUSTOMER`, attribute as `PRICING_ATTRIBUTE31`, order type as context `ORDER`, and attribute as `PRICING_ATTRIBUTE40`. Also, for illustrative purposes, the accumulation values are stored in a user-defined table called `accum_val_tbl`. Performance considerations are not taken into account here.

```

FUNCTION Get_numeric_attribute_value(
    p_list_line_id          IN NUMBER,
    p_list_line_no         IN VARCHAR2,
    p_order_header_id      IN NUMBER,
    p_order_line_id        IN NUMBER,
    p_price_effective_date IN DATE,
    p_req_line_attrs_tbl   IN ACCUM_REQ_LINE_ATTRS_TBL,
    p_accum_rec            IN ACCUM_RECORD_TYPE
) RETURN NUMBER
IS
    v_cust_class  VARCHAR2(240);
    v_order_type  VARCHAR2(240);
    v_req         accum_req_line_attrs_rec;
    i            NUMBER;
    accum_value   NUMBER
BEGIN
    -- this loop extracts the customer class and the order type that is
    -- passed on the request line.  We only use the
    p_req_line_attrs_tbl
    -- input parameter here.
    i := p_req_line_attrs_tbl.FIRST;
    WHILE I IS NOT NULL LOOP
        v_req := p_req_line_attrs_tbl(i);
        IF (v_req.context = 'CUSTOMER' AND
            v_req.attribute = 'PRICING_ATTRIBUTE31')
        THEN
            v_cust_class := v_req.value;
        ELSIF (v_req.context = 'ORDER' AND
            v_req.attribute = 'PRICING_ATTRIBUTE40')
        THEN
            v_order_type := v_req.value;
        END IF;
        i := p_req_line_attrs_tbl.NEXT(i);
    END LOOP;
    -- supposing the customer class and order type are not null, now
    -- query the user-defined table for the stored accumulation value
    -- and return this value.
    SELECT value
    INTO accum_value
    FROM accum_val_tbl
    WHERE customer_class = v_cust_class
    AND order_type = v_order_type;
    RETURN accum_value;
END Get_numeric_attribute_value;

```

In this example, the Customer Class and Order Type are extracted from the request line and used in the query to `accum_val_tbl` which stores an individual accumulation value for every customer class/order type pair. Once the value is selected, the function returns it to the pricing engine to continue the calculation. Depending on how many distinct pairs exist, this function (using the way the data is stored in the table) will return a different and correct accumulation value for each pair that is passed in the request line.

You can write the runtime sourcing function in many ways, and the implementation depends on the scenario and requirement. The logic can be as simple or complex as required within the limits of the PL/SQL language. Regardless, the function must process that logic and query the appropriate data table for the corresponding accumulation value before returning the number.

---

## Archiving and Purging Pricing Entities

This chapter covers the following topics:

- Overview of Archiving and Purging Pricing Entities
- Using API to Archive Pricing Entities
- Using API to Purge Pricing Entities

### Overview of Archiving and Purging Pricing Entities

Archiving pricing data enables you to copy pricing data from the pricing application tables to archive tables for long-term data storage.

You can archive price list and modifier lines. The archiving process permanently removes the lines from the related price list or modifier list into the archive tables for storage.

When the archived data is no longer required (and does not need to be retained for legal requirements), then you can use the purge feature to purge the data from the archive tables. QP\_ARCHIVE\_ENTITY\_PUB

You can archive and purge price list lines and modifier list lines for the following price list and modifier list types:

- Agreement Price List
- Standard Price List
- Deal
- Discount List
- Freight and Special Charge List
- Promotion

- Surcharge List

Archiving and purging records reduces the number of pricing records that the pricing engine queries, potentially improving processing performance.

You can also use APIs to archive and purge multiple pricing entities. For information about the APIs, see: *Using API to Archive Pricing Entities*, page 11-2 and *Using API to Purge Pricing Entities*, page 11-4.

## Related Topics

*Oracle Advanced Pricing User's Guide*, Archiving and Purging Pricing Entities

## Using API to Archive Pricing Entities

You use the **Archive Pricing Entities** window to archive single modifier lines and price list lines. This action permanently removes the line information from the associated price list or modifier list.

To archive multiple pricing entities that are no longer active or infrequently used, you can use the Archiving Pricing Entity API, `QP_ARCHIVE_ENTITY_PUB`. When you archive the data, the data from the application tables is inserted into the archiving tables.

The API supports the validations of the pricing entities. If there are any errors in the archive process, then you can view these errors in the log file.

For information about archiving and purging, see: *Overview of Archiving and Purging Pricing Entities*, *Oracle Advanced Pricing User's Guide* and *Archiving Pricing Entities*, *Oracle Advanced Pricing User's Guide*.

Use the following API to archive data:

**Name of the API:** Archiving Pricing Entity

**Package:** `QP_ARCHIVE_ENTITY_PUB`

For more information about the Archiving Pricing Entities API, see: *Oracle Integration Repository*.

Refer to the sample code to archive the pricing entities:

```

Declare
    errbuf          VARCHAR2(200);
    retcode         NUMBER;

    archive_entity_tbl QP_ARCHIVE_ENTITY_PUB.Archive_Entity_Tbl_Type;
    out_archive_entity_tbl QP_ARCHIVE_ENTITY_PUB.Archive_Entity_Tbl_Type;

    l_Archive_Entity_rec          QP_ARCHIVE_ENTITY_PUB.Archive_Entity_Type;
Begin
    fnd_profile.PUT('QP_ARCH_HEADER_QUAL', 'Y');
    fnd_profile.PUT('QP_DEBUG', 'Y');

    archive_entity_tbl(1).archive_name := 'Ptest11';
    archive_entity_tbl(1).source_system_code := 'QP';
    archive_entity_tbl(1).entity := 568742;
    archive_entity_tbl(1).all_lines := 'Y'; -- To archive all the lines of
    the entity
    archive_entity_tbl(1).entity_type := 'Standard Price List';

    archive_entity_tbl(2).archive_name := 'Mtest11';
    archive_entity_tbl(2).source_system_code := 'QP';
    archive_entity_tbl(2).entity := 568744;
    archive_entity_tbl(2).all_lines := 'Y'; -- To archive all the lines of
    the entity
    archive_entity_tbl(2).entity_type := 'Discount List';

    QP_ARCHIVE_ENTITY_PUB.archive_entity(archive_entity_tbl,
    out_archive_entity_tbl);

    IF out_archive_entity_tbl.COUNT <> 0 THEN

        FOR I IN 1..out_archive_entity_tbl.COUNT LOOP
            dbms_output.put_line
            ('-----
            -');
            dbms_output.put_line('I --->  || I);

            l_Archive_Entity_rec := out_archive_entity_tbl(I);

            dbms_output.put_line('p_archive_name -  || l_Archive_Entity_rec.
            archive_name);
            dbms_output.put_line('p_archive_name -  || l_Archive_Entity_rec.
            entity_type);
            dbms_output.put_line('p_source_system_code -  || l_Archive_Entity_rec.
            source_system_code);
            dbms_output.put_line('p_entity -  || l_Archive_Entity_rec.entity );
            dbms_output.put_line('p_all_lines -  || l_Archive_Entity_rec.all_lines
            );
            dbms_output.put_line('start_date_active -  || l_Archive_Entity_rec.
            start_date_active);
            dbms_output.put_line('end_date_active -  || l_Archive_Entity_rec.
            end_date_active );
            dbms_output.put_line('creation_date -  || l_Archive_Entity_rec.
            creation_date );
            dbms_output.put_line('created_by -  || l_Archive_Entity_rec.created_by
            );

            dbms_output.put_line('errbuf -  || l_Archive_Entity_rec.errbuf);
            dbms_output.put_line('retcode -  || l_Archive_Entity_rec.retcode);

```

```
dbms_output.put_line('COMPLETED SUCCESSFULLY');  
  
    END LOOP;  
  
END IF;  
  
EXCEPTION  
WHEN OTHERS THEN  
dbms_output.put_line(' ERROR ' || sqlerrm);  
END;
```

## Related Topics

Purging Pricing Entities, *Oracle Advanced Pricing User's Guide*

Using API to Purge Pricing Entities, page 11-4

## Using API to Purge Pricing Entities

You use the **Purge Entity window** to permanently delete a single archived pricing entity.

To delete multiple archived pricing entities permanently, you can use Purge Pricing Entity API, QP\_PURGE\_ENTITY\_PUB API.

The API supports validations of pricing entities. If there are any errors, the log displays these errors.

For information about archiving and purging, see: Overview of Archiving and Purging Pricing Entities, *Oracle Advanced Pricing User's Guide* and Purging Pricing Entities, *Oracle Advanced Pricing User's Guide*

Use the following API to purge pricing entities:

**Name of the API:**Purge Pricing Entity API

**Package:**QP\_PURGE\_ENTITY\_PUB API

For more information about the Purge Pricing Entity API, see: *Oracle Integration Repository*.

Refer to the following sample code to purge the pricing entities:

```

declare

    errbuf                VARCHAR2(200);
    retcode               NUMBER;
    x_return_status       VARCHAR2(100);
    x_return_text         VARCHAR2(2000);
    purge_entity_tbl      QP_PURGE_ENTITY_PUB.Purge_Entity_Tbl_Type;
    out_purge_entity_tbl  QP_PURGE_ENTITY_PUB.Purge_Entity_Tbl_Type;
    l_Purge_Entity_rec    QP_PURGE_ENTITY_PUB.Purge_Entity_Type;
Begin

    fnd_profile.PUT('QP_DEBUG', 'Y');

    purge_entity_tbl(1).source_system_code := 'QP';
    purge_entity_tbl(1).archive_name := 'pv5';

    purge_entity_tbl(2).source_system_code := 'QP';
    purge_entity_tbl(2).entity_type := 'Discount List';
    purge_entity_tbl(2).entity := '566751';

    IF purge_entity_tbl.COUNT <> 0 THEN
        QP_PURGE_ENTITY_PUB.Purge_entity(purge_entity_tbl ,
        out_purge_entity_tbl );
    END IF;
    IF out_purge_entity_tbl.COUNT <> 0 THEN
        FOR I IN 1..out_purge_entity_tbl.COUNT LOOP

            dbms_output.put_line('archive_name --->  ' || purge_entity_tbl(I).
            archive_name);
            dbms_output.put_line('source_system_code --->  ' || purge_entity_tbl(I).
            source_system_code);

            dbms_output.put_line('errbuf --->  ' || out_purge_entity_tbl(I).errbuf);
            dbms_output.put_line('retcode --->  ' || out_purge_entity_tbl(I).
            retcode);
        END LOOP;

    END IF;
    dbms_output.put_line('COMPLETED SUCCESSFULLY');

EXCEPTION
WHEN OTHERS THEN
    dbms_output.put_line(' ERROR ' || sqlerrm);
END;

```

## Related Topics

Overview of Archiving and Purging Pricing Entities, *Oracle Advanced Pricing User's Guide*

Archiving Pricing Entities, *Oracle Advanced Pricing User's Guide*

Using API to Archive Pricing Entities, page 11-2



This chapter covers the following topics:

- Auditing
- Setting up Auditing
- Audit Setup

### Auditing

Oracle Advanced Pricing supports the auditing of price lists and modifiers. You can:

- Select the fields to audit.
- Generate the audit report.
- Record the history of changes on the header and lines.

To perform auditing you must set the profile option *QP: Enable Auditing* to **Yes** and set the *QP: Audit Pricing Tables* process parameters to **Yes**. To disable auditing, set the value profile option to **No** and set the values in the *QP: Audit Pricing Tables* process parameters to **No**.

When you select the fields to audit on the **Audit Setup** page, the application saves the information in the following audit tables for each base table: (QP\_LIST\_HEADERS\_HD, QP\_LIST\_HEADERS\_TL\_HD , and QP\_LIST\_LINES\_HD).

Run the *QP: Audit Trail Report* to generate the audit report for the tables that need to be audited.

See *QP: Audit Trail Report* and *QP: Audit Pricing Tables* in the *Oracle Advanced Pricing User's Guide*.

### Setting up Auditing

Oracle Advanced Pricing enables you to audit price lists and modifiers. Use the **Audit**

**Setup** page to select the headers and line fields for which to record the history of changes. You can then generate an audit report for the selected fields.

To set up auditing of price lists and modifiers, perform the following steps:

1. Use the System Administrator responsibility and navigate to the **Profiles** window. Query the profile option **QP: Enable Auditing**. Set the profile option to **Yes** at the site level.

For information on profile options, see Profile Options Setup Summary, page 5-3.

2. Switch to the **Oracle Pricing Administrator** responsibility and run the **QP: Audit Pricing Tables** process to enable the triggers on the base tables.

For information on this process, see *QP: Audit Pricing Tables* in *Oracle Advanced Pricing User's Guide*.

3. Navigate to the **Audit Setup** page to select the fields for which to audit.

For information on the fields, see Audit Setup, page 12-2.

4. Create or update a price list or a modifier list.

For information on how to create and updating price lists, see *Creating a Price List and Updating a Price List* in *Oracle Advanced Pricing User's Guide*.

*Creating a Modifier Lists and Updating a Modifier List Lines* in *Oracle Advanced Pricing User's Guide*.

5. Switch to the **Oracle Pricing Manager** responsibility to run the QP: Audit Trail Report to generate the audit report.

For information on the audit report, see *QP: Audit Trail Report* in *Oracle Advanced Pricing User's Guide*.

## Audit Setup

Use the **Audit Setup** page to select the fields to be audited.

**Note:** You cannot audit qualifiers and pricing attributes in the price lists and the modifiers.

### To audit price lists and modifiers:

1. Log in with the **Oracle Pricing Administrator** responsibility.
2. Click **Audit** to access the **Audit Setup** page.
3. Click the **Price Lists** tab to see headers and lines for the price lists.

4. Add the headers and lines fields that you want to audit.
5. Click the **Modifiers** tab to see headers and lines for the modifiers.
6. Add the headers and lines fields that you want to audit.

### Audit Setup Page

The screenshot shows the Oracle Advanced Pricing interface. The top navigation bar includes the Oracle logo, 'Advanced Pricing', a star icon, a notification bell with '59+', and 'OPERATIONS'. Below this, there are tabs for 'Price List Maintenance', 'Oracle Pricing Administrator Setup', and 'Audit'. The 'Audit' tab is active. On the left, there are sub-tabs for 'Price Lists' and 'Modifiers'. Under 'Price Lists', there are further sub-tabs for 'List Headers' and 'List Lines'. The main content area is titled 'Audit Setup' and contains a table with the following data:

Field	Description	Audit	Delete
PRODUCT_PRECED	Precedence	<input checked="" type="checkbox"/>	
DYNAMIC_FORMULA	Dynamic Formula	<input checked="" type="checkbox"/>	
OPERAND	Value	<input checked="" type="checkbox"/>	

Below the table, there is a 'Table Diagnostics' section with 'Apply' and 'Cancel' buttons.

### To add header or line fields for auditing:

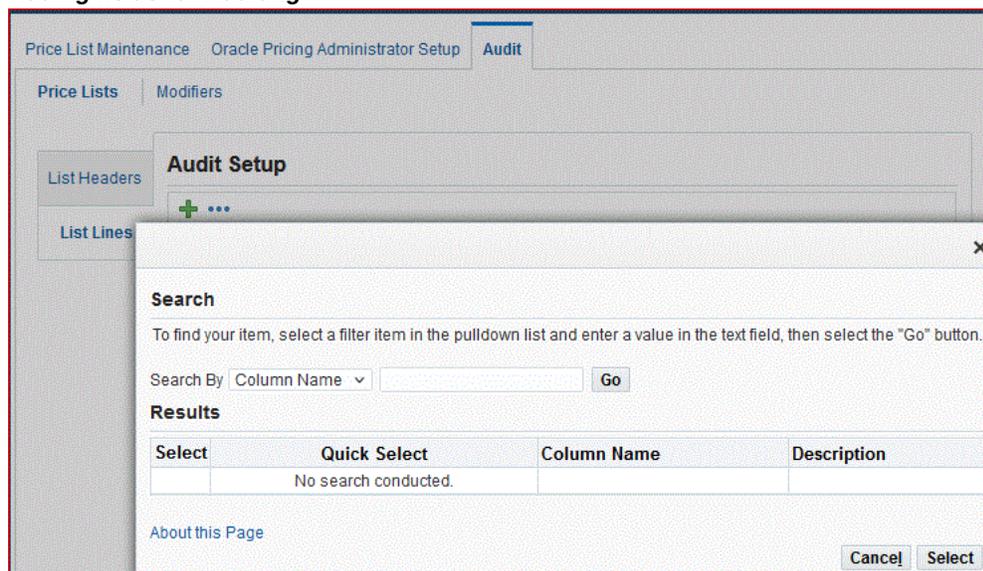
When you add the fields to audit on the **Audit Setup** page, the application saves the information in the following audit tables for each base table: QP\_LIST\_HEADERS\_HD, QP\_LIST\_HEADERS\_TL\_HD, and QP\_LIST\_LINES\_HD.

Perform the following steps to add a field for a header or list lines:

1. Click the **Add Another Row** icon (+) to add a new row.
2. Search for the field that you want to audit.
3. Select the field and click **Select**. The selected entity is added.
4. Select the **Audit** check box.
5. Click **Apply**.

The selected field is saved to the database.

### Adding fields for Auditing



### To remove header or line fields for auditing:

1. To remove a field from being audited, click the **Delete** icon.

### What's Next

After you add the fields on the **Audit Setup** page, you can create a price list or a modifier list. You can then make changes to the fields on the price list or the modifier list that need to be audited.

For information on price lists and modifier lists, see "Price Lists" and "Modifiers" in *Oracle Advanced Pricing User's Guide*.

---

# Multicurrency Price Lists and Agreements

This chapter covers the following topics:

- Overview of Multicurrency Price Lists and Agreements
- Example of Multicurrency Price List Usage
- Fresh Install Using Multiple Currency Price List
- Upgrading from Single Currency Price List to Multiple Currency Price List
- Implementation Decisions for Creating Multicurrency Conversion Lists
- Using Multiple Currency Price List with Other Oracle Products

## Overview of Multicurrency Price Lists and Agreements

Advanced Pricing supports both single currency price lists and multiple currency price lists. For single currency price lists, one currency is defined per price list. For multicurrency price lists, one base currency price list exists along with an attached currency conversion list defining conversion factors and rules for converting prices.

You need to determine which price list strategy your organization wants to support. Advanced Pricing provides a profile option and concurrent program to convert existing price lists from a Single Currency Price List to Multiple Currency price lists. However, once this profile is enabled, and price lists are converted, users should not return to non-multicurrency price lists. Changing the profile back to No may cause undesired results if conversion criteria were used. Oracle does not support changing the setting back to No.

## Single Currency Price Lists

Single currency price lists are the default setting for Advanced Pricing. These are used when your business requires that you maintain different prices for different currencies, and no relationship exists between prices that are defined.

## Multiple Currency Price Lists

Multiple Currency price lists enable businesses that have pricing strategies that are based on a single price for an item in a base currency and use exchange rates or formulas to convert that price into the ordering currency. At pricing engine run time, the pricing engine takes the currency from the order and search for a price list(s) with base or conversion currencies matching this currency. The price is converted from the base currency and calculates the ordering currency based upon the established conversion rules.

Multicurrency price lists use a specified base currency and the conversions for other currencies are applied to the values of the base currency price list. Additionally, multicurrency price lists allow some significant features such as markup conversions that can be applied to the values of the base price list without any changes to the list line values. This can significantly reduce the number of price lists that must be maintained and reduce data storage.

The Currency Conversion window is used to define the conversion criteria. Seeded conversion types include: fixed, formula, user defined, spot, EMU fixed, transaction and corporate.

**Note:** Some of these seeded conversion types require Oracle General Ledger to be installed. Users can still link to other non-Oracle stored conversion rate information by using the formula functionality

### Attributes

You can define multiple conversion criteria based upon certain attributes for the same base currency. For further information about attributes, see chapters in the *Advanced Pricing Implementation Guide*, Attribute Management and *Oracle Advanced Pricing User's Guide*.

### Markup Values and Formulas

You can define markup criteria per currency definition, including the base currency. This markup can be a fixed value of amount or percent, or based upon a formula. For further information on formulas, see: *Oracle Advance Pricing User's Guide*, Formulas.

### Rounding

- **Base Round To:** Rounds the selling price after applying the modifiers. In this field, you can enter a numeric-rounding factor that rounds the list price after the conversion and markup is applied.

**Note:** You cannot enter the Base Round To on the price list. Base Round To must be set in the currency conversion list as a Base Round To and will always default to the price list from the attached currency conversion list.

- **Conversion Rounding Factor:** Rounds the To Currency list price after the

conversion and markup is applied. This rounding factor could be different for the same To-Currency. This is entered in the Conversion Rounding Factor field in the currency conversion list.

- **Round To:** Rounds the selling price after applying the modifiers. This rounding factor will always be the same for a Currency To.

## Example of Multicurrency Price List Usage

The following example shows how a multicurrency conversion list calculates a price in U.S. dollars when an item is ordered in Canadian Dollars (CAD). The following multi-conversion list and base price list setups are used in the conversion scenarios:

### *Multicurrency Conversion List Setup*

Header Level Field Name	Value
Name of multicurrency list	Test Multicurrency Conversion List
Currency (base currency)	USD
Base Round To	- 2
Base Markup Value	%
Base Markup Value	10
Base Markup Formula	<p><i>My Base Markup Formula</i> with expression 1/2 (Step 1 divided by Step 2) where:</p> <ul style="list-style-type: none"> <li>• Step 1 is of type List Price</li> <li>• Step 2 is of type Modifier Value</li> </ul>
Line Level Field Name	Value
To-currency	CAD
Conversion Type	FORMULA

Header Level Field Name	Value
Formula (conversion)	<i>My Conversion Formula</i> with expression 1*2 where: <ul style="list-style-type: none"> <li>• Step 1 is of type Numeric Constant and value 25</li> <li>• Step 2 is of type List Price</li> </ul>
Markup Operator	AMT
Markup Value	50.12345
Conversion Rounding Factor	-3
Round To	-2

The following table describes the setup for the base price list that is used in this example:

**Base Price List Setup**

Field Name	Value
Name	Base PL
Currency	USD
Item	Item A
List Price	20
Dynamic Formula	<i>My Base Price List Line Formula</i> with expression 1*2 where: <ul style="list-style-type: none"> <li>• Step 1 is of type List Price</li> <li>• Step 2 is of type Numeric Constant with a value of 30</li> </ul>

**Scenario 1**

If 1 unit of Item A is ordered in the currency Canadian Dollars (CAD), then the price is determined in the following way:

1. Price of Item A from base price list Base PL is calculated using the dynamic formula  $1 * 2$ , where step 1 is List Price and step 2 is Numeric Constant. The list price on the base price list line for Item A is 20 and numeric constant is 30. So the list price now is  $20 * 30 = 600$ .
2. Because the Item A is ordered in CAD (which is not the base currency USD), the Base Markup Value, Base Markup Operator, Base Markup Formula, and Base Round To are not used in this calculation. Instead, the information at the conversion line level is used:
  - The conversion type is FORMULA.
  - The formula that is used is *My Conversion Formula* with expression  $1*2$ , where step 1 is of type Numeric Constant and value 25 and step 2 is of type List Price.

The List Price that is used is the result of step 1, which is 600.

So the new list price now is  $25 * 600 = 15,000$ .

3. Because a markup formula *My Conversion Markup Formula* exists with expression  $1/2 + 3$ , where:
  - Step 1 is of type List Price.
  - Step 2 is numeric constant 5,000.
  - Step 3 is of type Modifier Value.

The markup is calculated as

$$15,000/5,000 + 50.12345$$

$$= 3 + 50.12345$$

$$= 53.12345$$

Note that the List Price is taken from Step 2 and that the Modifier Value is the Markup Value on the conversion line. The final markup value is 53.12345, and the markup operator is AMT.

So the new list price now is  $15,000 + 53.12345 = 15053.12345$ .

4. If Item A is a non service item, then the conversion rounding factor - 3 is applied to the result of step 3, that is, after the markup, if any, has been applied to the post-conversion list price of step 2.

**Note:** This step is skipped for service items. So the new list price now is 15053.123

So the new list price now is 15053.123.

5. The conversion line Round To is applied to the result of step 4. This Round To is applicable to the list price, as well as to the net price but its application is subject to the value of the profile option QP: Selling Price Rounding Options.
  - If the profile option value is No, then the list price 15053.123, any adjustments, and the selling price are not rounded.
  - If the profile option value is Individual, then the list price, any adjustments, and the selling price are each individually rounded to two places after the decimal because the Round To on the conversion list is - 2.

**So the new list price would be 15053.123 and is rounded to 15053.12.**

### Comments

- Markup operator can apply to the Markup Value as well as to the Markup Formula. This is true for both Base Markup and Conversion Markup.
- Markup Value and Markup Formula can both be present in the same list. In that case, the Markup Formula is used; the Markup Value may be used as the value for any Modifier Value type of Formula Line in the Markup Formula.
- Modifiers is selected and applied only if either the currency on the modifier is null or the currency on the modifier matches the ordering currency.
- If a Conversion Formula has a List Price formula component, the List Price from the Base Price List (after dynamic formula calculation on the base price list if applicable) is used for this formula component.
- If a Markup Formula (at the conversion line level) has a List Price formula component, then the List Price after performing conversion is used for this formula component.
- Conversion Rounding Factor is applied on the marked up value of list price. For example, apply conversion rounding to the marked up conversion list price.
- Round To value on the conversion line is used to perform rounding of final list price, adjustments, and selling price subject to the QP: Selling Price Rounding Options.

### Scenario 2

If 1 unit of Item A is ordered in the base currency USD, then the price is determined in the following way:

1. Price of Item A from base price list is calculated using the dynamic formula  $1 * 2$  where step 1 is of type List Price and step 2 is of type Numeric Constant. The list price on the base price list line for Item A is 20 and numeric constant is 30.

So the list price now is  $20 * 30 = 600$ .

2. Because the Item A is ordered in the base currency USD, only the Base Markup Value, Base Markup Operator, Base Markup Formula and Base Round To are used in this calculation. Setup values at the conversion line level are not used.

Because there is a base markup formula *My Base Markup Formula* with expression 1/2 (step 1 divided by step 2), where step 1 is of type List Price and step 2 is of type Modifier Value, the base markup is calculated as  $600 / 10 = 60$  (where list price is taken from step 1) and step 2 is the Base Markup Value. Because the Base Markup Operator is %, the new list price now is  $600 + 60\% \text{ of } 600 = 600 + 360 = 960$ .

3. The Base Round To -2 is applied to the result of step 2. This Round To is applicable to the list price and the net price, and how it is applied is determined by the profile option QP: Selling Price Rounding Options:
  - If the profile option value is No, then the list price 960, any adjustments, and the selling price are not rounded.
  - If the profile option value is Individual, then the list price, any adjustments, and the selling price are each individually rounded to two places after the decimal. So in this case, the new list price is 960.00.

## Fresh Install Using Multiple Currency Price List

You can set the profile option QP: Multi-Currency Installed to control which type of price lists you will use:

- No (default setting): Can create and maintain multiple price lists with each price list that is defined in its own base currency (non-multicurrency enabled).
- Yes: Enables you to attach a multicurrency conversion list to each price list or agreement price list. Setting the value of the profile to Yes enables all Price List and Agreement windows with some window field and functionality changes. Once the profile is set, you must create at least one multiple currency conversion list. A currency conversion list is required when you are setting up a price list.

The following steps outline how to set up multiple currency price lists:

1. Navigate to System Administer responsibility >Profile > System> QP: Multi-Currency Installed > Select Yes.
2. Navigate to Oracle Pricing Manager responsibility >Price List Setup > Create a base currency master price list

**Note:** If single currency price lists were created prior to changing QP: Multi-Currency Installed to Yes, then you must run the concurrent program *Update Price Lists with Multi-Currency*

*Conversion Criteria.* This converts all existing price lists and agreement forms as Multi-Currency and activates the Multi-Currency Conversion Setup form.

## Upgrading from Single Currency Price List to Multiple Currency Price List

If you are already using single currency price lists and want to upgrade to multiple currency price lists and agreements, the following steps are required.

### **Step 1. Set profile option QP: multicurrency Installed**

The profile QP: Multi-Currency Installed controls which type of price lists you will use. Set this profile to Yes. This will enable you to run the Concurrent Request program that updates all existing price lists and agreements to the multiple currency forms and functionality.

### **Step 2. Run concurrent request Update Price Lists with Multicurrency Conversion Criteria**

To start using the multicurrency feature, run the concurrent program Update Price Lists with Multi-Currency Conversion Criteria only once. This program creates:

- A currency conversion list for each combination of price list currency and rounding factor. This conversion will have the same Base Currency with the Base Round To from the price list.
- Attach a currency conversion list to each price list and agreement.

This is a mandatory step because without this the pricing engine will not be able to use the current price lists as multicurrency price lists.

### **Example**

Five price lists are set up as outlined in the following table:

Currency	Round To
USD	-2
USD	-2
USD	-3
FRF	-1
CAD	NULL

The concurrent program will create four Currency Conversion Lists:

Currency	Base Round To
USD	-2
USD	-3
FRF	-1
CAD	NULL

After you run the concurrent program, all price lists and agreements will be multicurrency enabled. You can now add additional To Currencies and their conversions.

## Implementation Decisions for Creating Multicurrency Conversion Lists

Prior to using multicurrency price lists, you need to consider the following guidelines:

### Combining Price Lists

Before creating or merging price lists and their currency conversion lists you must determine whether you can use one single price list or whether you need to create more than one.

You can use a single base currency price list to replace price lists that are defined in various other currencies for conversions provided that all have the following identical attributes:

- Items
- Qualifiers
- Pricing attributes for price list lines

The basis of the price list line values (of the single currency defined price lists) should equal the base currency of the single multicurrency price list to which you attach the conversion list. Otherwise you will need to define multiple conversion criteria based upon attributes for the same To-Currency.

### Example of combining single currency price lists:

Qualifiers and Items are identical in the selected single currency price lists. All of the single currency price lists have list line values that are converted values based on the USD price list. Except for CAD, all the lists apply their conversion types, equally to all items except for the CAD defined price list. CAD Price List line values equal a Fixed conversion rate = 2, *except* Item C which is valued at a conversion rate of 1.5 as shown in the following table:

Price List = USD	Price	Price List = CAD	Price	= Conversion Rate
All Items	--	All Items	--	2
Item A	100	Item A	200	2
Item B	200	Item B	400	2
Item C	200	Item C	300	1.5

In the following table, the individual price lists for MXN, JPY, and Euro that previously were set up with the price list values equal to the conversion types have been transitioned in the setup. The CAD defined price list required an additional step to set up a conversion line for the item(s) that were not converted at the same conversion type of Fixed, with same Fixed Value.

Currency conversions will apply to the price list values on the base currency price list to which it is attached.

Base Currency (USD)	Convers. Type	Fixed Value	Attribute Code	Attribute Value	Start Date	End Date	Pr. <sup>1</sup>
To-Currency (CAD)	Fixed	2	--	--	01/01/02	12/31/02	2
To-Currency (CAD)	Fixed	1.5	Item Number	Item C	01/01/02	12/31/02	1
To-Currency (MXN)	Corporate	--	--	--	01/01/02	12/31/02	--
To-Currency (JPY)	Corporate	--	--	--	01/01/02	12/31/02	--

Base Currency (USD)	Convers. Type	Fixed Value	Attribute Code	Attribute Value	Start Date	End Date	Pr. <sup>1</sup>
To-Currency (Euro)	Spot	--	--	--	01/01/02	12/31/02	--

**Note:** <sup>1</sup>Pr. = Precedence

### Deactivating Price lists

You should deactivate a price list once you have merged it successfully with a multicurrency conversion list.

### Rounding

Three rounding profiles affect rounding in multicurrency price lists. These profiles are discussed in the *Oracle Advanced Pricing Implementation Guide*, Profile Options.

- QP: Unit Price Precision Type: This profile is used to determine the value for the rounding factor which is defaulted on the price list.
- QP: Price Rounding: If this profile option is set to *Enforce Currency Precision*, the Base Round To cannot be updated in the currency conversion list.
- QP: Selling Price Rounding Options: This profile has optional settings to determine how the converted list price and adjustments can be rounded.

### Currencies

The following must be defined in Oracle General Ledger:

- Base currency and To Currencies
- Seeded conversion types of user defined, spot, EMU fixed and corporate

### Markup and Conversion Formulas

Formulas can be created and used for Markup and for Conversion Type 'Formula'. For setup information, see the *Oracle Advanced Pricing User's Guide*.

Using formulas in multiple currency price lists has some limitations.

- Base Markup Formula: The value that is returned by the Base Markup Formula will be used as markup for the base currency. If the Formula uses the List Price (LP) component, the formula will use the list price after applying the conversion rate. A formula with a component type as PLL (Price List Line) is not allowed as a Base Conversion Formula. This is because the PLL line may have a different base currency than the price list to which this currency conversion criteria is attached.

- Conversion Type Formula: The value that is returned by the Formula selected, is the conversion rate.

A formula with a component type of PLL (Price List Line) is not allowed as a conversion formula because the PLL line may have a different base currency than the price list to which this currency conversion is attached.

A formula with a component type of MV (Modifier Value) is not allowed as a conversion formula because there is no modifier value in the currency conversion list.

### Attributes

You can use different attribute values for attribute types: Product, Pricing, and Qualifier, to specify more than one conversion for the same Currency To, start date and end date.

### Precedence

This precedence differs from attribute precedence.

This precedence, which is set up by the user, applies only to the currency conversion window. You must enter a precedence value in this field when setting up more than one conversion for the same Currency-To, start date, and end date and having different Attribute Value. When more than one attribute is passed from the calling application, the attribute with the lowest precedence value is selected by the pricing engine for conversion.

Currency-To	Conversion Type	Attribute Code	Attribute Value	Start Date	End Date	Prc.
JPY	Fixed	Item Number	AS54888	3/1/01	3/31/01	1
JPY	Daily	All Items	All	3/1/01	3/31/01	2

In this scenario, when item AS5488 is passed from the calling application the conversion that is defined for Item Number= AS54888 is used by the pricing engine because it since it has lower precedence value = 1 as compared to the precedence for All Items = All which is 2.

## Related Topics

*Oracle Advanced Pricing User's Guide*, Multi-Currency Conversion List chapter

Creating Context and Attributes to be used for Pricing Setup windows, page 17-3

Precedence and Best Price, page 16-1

## Using Multiple Currency Price List with Other Oracle Products

The Multiple Currency Price List feature is fully integrated with Oracle Order Management. Before using this feature, you should check with other Oracle products that integrate with Advanced Pricing to determine when they will support this feature. At the time of this writing, these products include Oracle iStore, Oracle Order Capture, Oracle Quoting and Oracle Contracts.

If Advanced Pricing is multicurrency enabled and you use Oracle products that do not yet support multiple currency price lists, when the calling application passes the price list, the pricing engine matches the base currency of the price list with the order currency. If no price list is passed, the pricing engine looks for price lists for which the base currency matches the order currency. The pricing engine does not look to the conversion currencies on the multiple currency price lists. This means that price lists still need to be defined and maintained for every single base currency used.



---

## Unit of Measure

This chapter covers the following topics:

- Overview of Unit of Measure

### Overview of Unit of Measure

As you analyze and understand a client's pricing scenario, key implementation decisions must be addressed to develop a logical pricing solution. These decisions, in relation to unit of measure, are addressed in this chapter.

#### **Key Implementation Decision: How do I adjust UOM setups to use Oracle Advanced Pricing?**

- You must define conversion rates between units of measure in order to price and discount in UOMs other than the primary UOM.
- All modifier UOMs must be consistent with the UOM on the price list. Price lists and modifiers must be constructed in the same unit of measure; this is what the pricing engine expects.
- Accruals require definition of UOM and UOM class.
- You must define UOMs if you use seeded qualifiers for line volume or line weight.

You must define appropriate setups in Oracle Service Contracts to price service items that span partial periods. These are used by pricing in service duration UOM conversions. For more information, see *Oracle Service Contracts* documentation.

### Defining UOM

You must define units of measure (UOM) for Oracle Advanced Pricing to:

- Price or discount items.
- Give non-monetary accruals as benefits. You must define a UOM class and UOM

that represent your non-monetary accruals. The profile option QP: Accrual UOM Class should be set to the UOM class that you define.

- Define qualifier rules that include the seeded qualifiers line volume or line weight.

**Note:** Defining UOM is not necessary if you have already installed and set up Oracle Inventory, or if you completed the common applications setup for another Oracle product. For more information, see *Oracle Inventory User's Guide, Defining Unit of Measure*.

### Defining UOM Conversions

To price and discount an item in a different (not primary) UOM, you must define the conversion rates between the base and other UOMs within the class. Oracle Advanced Pricing uses these rates to automatically convert transaction quantities to the primary pricing UOM. All price adjustments, benefits, and charges must be defined in the same UOM as on the price list.

**Note:** This step is not necessary if you have already installed and set up Oracle Inventory or if you completed the common applications setup for another Oracle product. For more information, see *Oracle Inventory User's Guide, Defining Unit of Measure Classes*.

### UOM Conversions for Service Items

Oracle Advanced Pricing evaluates UOM conversions from different sources when pricing service items:

- If service dates are specified on the transaction being priced, Oracle Advanced Pricing calls Oracle Service Contracts to compute or convert service duration to pricing UOM.

For more information on time period conversions, see Oracle Service Contracts implementation and user documentation.

- If service dates are not supplied on the pricing transaction, Oracle Advanced Pricing uses UOM conversion setups in Oracle Inventory for converting service duration to pricing UOM.

## Pricing Actions: Primary UOM and Pricing UOM

The primary UOM feature is used to price an item in different UOM without having to explicitly define prices for each UOM. The pricing UOM is the UOM in which the pricing engine prices the order line. The pricing quantity is the order quantity expressed in the pricing UOM. Invoicing shows information based on the ordered quantity and ordered UOM.

Example:

- An item is defined with "Each" declared as the primary UOM in a price list. You order one dozen. The price list does not have a price defined in dozen. The pricing engine uses the conversion factor to calculate a pricing quantity of 12 and pricing UOM of each.
- In the Price List window, you define a price for an item/UOM combination as a price list line. In a price list, you may specify only one as primary UOM for an item.

### **Defaulting UOM While Creating Price Lists**

When creating a new price list, the primary UOM defined in Oracle Inventory for an item defaults into the price list UOM. Users can change the defaulted UOM and select the primary flag on the price list line to make this the Pricing primary UOM.

Example: Item A can have:

- Primary UOM in INV: EA
- Primary UOM in QP: CASE

## **Pricing Controls: Profile Options**

Three profile options are critical for setting UOMs in Oracle Advanced Pricing:

- QP: Accrual UOM Class: Specifies the UOM class used to define accrual UOM. The modifier setup window displays all units of measure in this class when entering the benefit UOM for an accrual.
  - Default value: none
  - Required if your business gives non-monetary accruals as benefits
  - All UOM classes are defined to Oracle Applications
  - Visible and can be updated at the site and application level
- QP: Line Volume UOM Code: Specifies the UOM of the line volume qualifier. The attribute sourcing API converts the item on the request line to its primary UOM. It then uses the volume attributes of the item to derive the line volume of the item in the specified UOM.
  - Default value: none
  - Required if your business must define qualifier rules that include the seeded qualifier line volume
  - All UOM currently defined to Oracle
  - Visible and can be updated at the site and application levels

- QP: Line Weight UOM Code: Specifies the UOM of the line weight qualifier. The attribute sourcing API converts the item on the request line to its primary UOM, and uses the weight attributes of the item to derive the line weight of the item in the UOM specified in this profile option.
  - Default value: none
  - Required if your business needs to define qualifier rules that include the seeded qualifier line weight
  - Specifies UOM of the line weight qualifier (the attribute sourcing API converts the item on the request line to its primary UOM, and then uses the weight attributes of the item to derive the line weight of the item in the specified UOM)
  - All units of measure currently defined to Oracle
  - Visible and can be updated at the site and application levels

---

## Multiple Organizations

This chapter covers the following topics:

- Overview of Multiple Organization Model

### Overview of Multiple Organization Model

The Oracle Applications organization models define organizations and the relationships among them in arbitrarily complex enterprises. It determines how transactions flow through different organizations and how these organizations interact with each other.

### Operating Units

#### Operating Units and Pricing Security

Pricing security features are provided that enable you to:

- Assign or reassign a single operating unit to price lists and modifiers.
- Control which operating units can use pricing entities when pricing transactions.
- If Multi-Org Access Control (MOAC) is enabled, the operating unit that can be assigned to a price list or modifier is based on the MO: Security Profile option. The operating unit value, which is set in the MO: Default Operating Unit profile option, appears if you clear the Global check box.

#### Using Qualifiers to Assign Multiple Operating Units

By using qualifiers, you can create price lists and modifiers that are operating unit specific. Operating unit is not a seeded qualifier. It is necessary to set up the attribute and mapping rules. Once the qualifier attribute of operating unit is available for use, it can be applied to price lists and modifiers. When a call is made to the pricing engine for a transaction in an operating unit, only those price lists and modifiers for the specified operating unit and global price lists/modifiers (applicable to all operating units) are evaluated by the engine.

## Cross Order Volume Loader

The concurrent program accumulates all cross order totals for an operating unit, but does not summarize them across operating units. The engine considers only orders that a customer places with this sales organization; the engine does not consider orders that the customer places with other operating units. When running the cross order volume loader, you may specify that the load run for one operating unit. If specified as null, the engine summarizes for all operating units for which there are orders (assuming there is a multi-org install). If loading for multiple operating units, each summary is within an operating unit.

## Operating Unit Specific Seeded Qualifiers and Pricing Attributes

Some of the seeded qualifiers attributes and pricing attributes in Oracle Advanced Pricing are specific to an operating unit. The following is a list of some seeded qualifier and pricing attributes that are operating unit specific:

Type	Context	Qualifier Attribute
Qualifier	Customer	Site use
Qualifier	Customer	Bill to
Qualifier	Order	Line type
Qualifier	Order	Ship from
Qualifier	Volume	Period 1 order amount
Qualifier	Volume	Period 2 order amount
Qualifier	Volume	Period 3 order amount
Pricing	Volume	Period 1 item quantity
Pricing	Volume	Period 2 item quantity
Pricing	Volume	Period 3 item quantity
Pricing	Volume	Period 1 item amount
Pricing	Volume	Period 2 item amount
Pricing	Volume	Period 3 item amount

## Item Validation Organizations

### QP: Item Validation Organization

You must set the profile option QP: Item Validation Organization to the level in your organization hierarchy at which you set prices when using Oracle Inventory. This profile value indicates the Oracle Manufacturing organizations that items are validated and viewed against when entering price lists or modifiers. You should set this profile value should be set to the master organization.

**Note:** If MOAC is enabled, each responsibility can have access to multiple operating units, and each operating unit can have its own item validation organization. However, profile QP: Item Validation Org can be set only at the responsibility or site level. Therefore, you should assign only operating units that share a common pricing item validation organization should be assigned to one pricing responsibility.

### Modifier Type of Item Upgrade

When you define item relationships for item upgrade type of modifiers, the item relationship must belong to the same item organization as specified in QP: Item Validation Organization.

## Related Topics

Overview of Pricing Security, page 6-1

Pricing Security and Operating Units, page 6-4



---

## Precedence and Best Price

This chapter covers the following topics:

- Overview of Precedence and Best Price
- Default Precedence Numbers
- Matched Qualifiers for Modifiers/Price Lists
- Price List Incompatibility Resolution
- Modifier Incompatibility Resolution
- Incompatibility Resolution Examples

### Overview of Precedence and Best Price

Incompatibility occurs when the pricing engine finds more than one price or modifier to return, but the pricing engine and user-defined rules prohibit applying more than one of them. You can resolve incompatibilities by setting the Incompatibility Resolve Code field in the Event Phases window to either Precedence or Best Price:

- **Precedence:** When incompatibility is encountered between price lists or modifier lists and the incompatibility resolve code for the phase is precedence, the pricing engine attempts to resolve the incompatibility by ordering the qualifier attributes and item context attributes from highest to lowest priority (number 1 having the highest priority). The qualifier or item attribute with the highest priority has precedence over all other attributes, and the engine selects the price list line or modifier list line to which this attribute belongs.
- **Best Price:** Best price is the highest modifier value that calculates the highest discount value. When the pricing engine encounters incompatibility between modifier lists and the incompatibility resolve code for the phase is best price, the engine attempts to resolve the incompatibility by finding the modifier that gives the best price.

**Note:** Incompatibility resolution can be set for each pricing engine phase with the exception of Phase Sequence 0 - List Line Base Price. See Events and Phases for more information about phases.

## Default Precedence Numbers

When resolving incompatibilities by precedence, the pricing engine evaluates the precedence numbers that are assigned to pricing and qualifier attributes. Precedence numbers for both seeded and user-created attributes are defined in the Precedence field in the Context Setup window:

- Seeded attributes: Pre-assigned precedence numbers are assigned to seeded attributes.
- User-entered attributes: You can define precedence numbers when creating or updating attributes.

The attributes and their precedence values default to the appropriate price list, modifier, and qualifier windows. However, if you wanted to change the precedence value for a particular attribute in a price list, modifier, or qualifier line, you can manually override the precedence value only for the particular line. This only changes the precedence value for that particular line. However, to update the actual precedence number for an attribute so that it defaults with the new value, you must change the Precedence number in the Context Setup window.

**Warning:** To avoid problems with future upgrades, do not change the original precedence numbers for seeded attributes in the Context Setup window!

### Precedence and Pricing Attributes

The precedence numbers for attributes that are defined for the Pricing context (excluding item context) do not have significant meaning for pricing engine evaluation in determining precedence.

## Matched Qualifiers for Modifiers/Price Lists

The pricing engine only evaluates matched qualifiers for a price list or modifier only when ordering for precedence. A matched qualifier is described as those attributes that are evaluated as true. Because qualifiers can be defined as OR conditions, some qualifier attributes that exist on the setup of the price list or modifier might not be chosen by the engine, or in other words, are not qualified by the engine. The pricing engine uses matched qualifiers only when ordering the qualifier attributes to determine which has the highest priority.

The following table lists some seeded qualifier contexts and qualifier attributes with the attribute number for each attribute for a modifier list. The qualifier attributes of Agreement Type and Customer Class have the same grouping number, and Order type is in a different grouping number. For the engine to select the modifier, an order must have either Agreement Type and Customer Class be true or Order type be true. In this example, Order type is true and Agreement Type and Customer Class are false. Order type is a matched attribute, and only the value for order type is used to evaluate precedence.

**Note:** Effective precedence of a line = Minimum (product/qualifier precedence).

Grouping Number	Qualifier Context	Qualifier Attribute	Precedence	Matched?
1	Customer	Agreement type	240	No
1	Customer	Customer class	310	No
2	Orders	Order type	470	Yes

## Price List Incompatibility Resolution

The pricing engine is coded to calculate base price from the price list in phase sequence 0. Phase 0 is coded with the resolve incompatibility code as precedence. You cannot change this setting. When the pricing engine determines that a pricing request is eligible for a price from more than one price list, the engine attempts to resolve the incompatibility by ordering the qualifier attributes and item attributes that are described in the precedence resolution section of this chapter.

If two or more price list lines are matched and have the same precedence, the pricing engine selects the price list line with the most matching pricing attributes. If the incompatibility between price lists cannot be resolved, then an error message is returned to the calling application (that a price cannot be applied) and lists the names of the price lists where the incompatibility resides.

## Modifier Incompatibility Resolution

The incompatibility resolution codes for phases that are associated with modifier processing can be changed from the seeded values to reflect the business need. The exception to this is for Phase 0, in which the code cannot be changed.

## Best Price Resolution for Modifiers

Incompatible modifiers can be across modifier types, therefore when the engine resolves incompatibility by best price, the engine needs to evaluate the modifiers on a similar basis. The engine does this by calculating a common benefit percent for each modifier. The following table identifies the modifier value that is used for evaluating best price for each modifier type:

Type of Modifier	Modifier Value
Discount/surcharge: percentage	List price%
Discount/surcharge: amount	Amount
Discount/surcharge: new price	List price - new price
Discount/surcharge: lumpsum	Lumpsum amount/line quantity
Price Break	Best price comparison for price break is based on the value of the matching break modifier and its list price *%.
Terms substitution	Estimated discount value defined in the Comparison Value field on the modifier line. If not provided then best price is set to zero.
Item upgrade	Estimated discount value defined in the Comparison Value field on the modifier line. If not provided then best price is set to zero.
Coupon issue	Estimated discount value defined in the Comparison Value field on the modifier line. If not provided then best price is set to zero.
Other Item discount	Estimated discount value defined in the Comparison Value field on the modifier line.
Promotional goods	Estimated discount value defined in the Comparison Value field on the modifier line.
Freight Charges	Same as discount and surcharges.

The pricing engine does not consider modifiers with formulas for best price calculation.

## Calculating Common Benefit Percent

The pricing engine needs a common basis to estimate modifier types to determine which modifier yields the best price. The engine accomplishes this by calculating the common benefit percent. For some modifier types, no stated value exists that the engine can use to evaluate best price; thus, the engine uses the value that the user enters in the Comparison Value column on the modifier summary line. The engine calculates the benefit percent by taking the Comparison Value as the numerator divided by the list price of the item that the modifier will be applied as the denominators. This benefit percent is compared to the other benefit percents for the modifier lines that are being evaluated. The highest benefit percent number is the modifier that yields the best price. This is the modifier that the engine will apply.

**Note:** Comparison Value is a user-entered field. The pricing engine uses this value to calculate the common benefit percent.

## Best Price Calculation Ignores Buckets

The pricing engine ignores buckets when selecting modifiers and determining best price calculation. Best price is calculated off of the list price. For example, an item has a \$100 list price. The engine determines that the item is eligible for modifiers A, B and C. Modifiers B and C are incompatible and the engine must resolve the incompatibility by best price. For modifier B, the engine calculates:

- $100 - (100 - 75) = 75$

Modifier C is calculated as:

- $100 - (100 * 0.125) = 87.5$

Modifier B is selected because it gives the better price.

Modifier	Incompat.	Bucket	Application Method	Value	Calculate Best Price	Engine Selection
A	Not applicable	1	Percent	20	=	Yes
B	Level 1	2	New price	75	75	Yes
C	Level 1	2	Percent	12.5	87.5	No

If only eligible modifiers are sent to the calculation engine, these modifiers are calculated in the correct bucket. In this example, the calculation engine takes the list price, \$100, subtracts the bucket 1 discount, 20, for a selling price of \$80. Next the engine

calculates the bucket 2 discount off of the \$80. Because modifier B is a new price discount, a discount of \$5 is created and the new selling price is \$75.

## Header and Line Qualifiers for Modifiers

To determine precedence, the engine selects all qualified header level and line level qualifiers and item context for the modifier line. The engine then chooses the attribute with the lowest precedence number. This attribute is used for incompatibility resolution.

During precedence processing if two or more modifiers tie based on precedence processing, then the engine resolves using best price processing *without* going below the established precedence.

## Modifier Incompatible Precedence Resolution

When the incompatibility resolve code for a phase is precedence, and more than one modifier line is eligible, the engine resolves the incompatibility in the following way (for this example, assume that all qualifier attributes are matched):

Modifier	Context Type	Context	Attribute	Precedence Number	Select Precedence	Resolve Incompatibility
A	Qualifier	Customer	Agreement type	240	Yes	No
A	Qualifier	Customer	Customer class	310	No	No
A	Pricing	Item	Item number	300	No	No
B	Qualifier	Order	Order type	470	No	No
B	Pricing	Item	Item category	290	Yes	No
C	Qualifier	Order	Order amount	100	Yes	Yes
C	Pricing	Item	Item number	200	No	No

First, the engine will select the attribute with the lowest precedence number for each of the modifiers. For modifier A, agreement type is selected because it has the lowest

number. For modifier B, item category is selected and for modifier C, order amount is selected.

The engine resolves incompatibility by ordering the precedence numbers for the three modifiers. Because modifier C has the lowest number, it is selected by the engine as the highest priority and is the modifier that is returned by the engine.

When the incompatibility resolve code for a phase is precedence and the engine cannot resolve incompatibility between two or more modifiers through precedence, the engine resolves incompatibility using best price resolution. If two or more modifiers result in the same best price, the engine randomly selects one modifier to return to the calling application.

## Setting Up Incompatibility Groups

Incompatibility occurs when the pricing engine finds more than one modifier to return, but is permitted to apply only one of them. When incompatibility is encountered between modifier lines, the pricing engine attempts to resolve the incompatibility by evaluating the precedence value and selecting the modifier line with the highest priority (precedence number of 1 having the highest priority).

**Note:** The incompatibility resolve code, which determines incompatibility processing preference for the phase, must be set to Precedence.

The qualifier or item attribute with the highest priority has precedence over all other attributes, and the engine selects the modifier to which this attribute belongs.

### Notes:

Use the Find Incompatibility Groups window to query modifiers by Phase Sequence, Phase Name and Incompatibility Group name and to display them in the Incompatibility Groups window. The query is phase-specific and returns only the modifier lines that are assigned to a specific phase such as the List Line Adjustment phase. You can view all modifier lines for the selected phase, and change or assign an incompatibility group. To view and edit the related modifier and its line details click Modifiers.

- **Incompatible Group:** Displays the incompatibility group currently assigned to a modifier line.

Assigning modifier lines to an incompatibility group enables the pricing engine to use incompatibility processing to resolve incompatibility between multiple eligible modifier lines. If an incompatibility occurs, the pricing engine reviews the modifier lines in the incompatibility group (assigned for a specific phase), evaluates their product precedence value from highest to lowest priority (number 1 having the highest priority), then selects the modifier line with the highest priority. In the Incompatible Group field, you can change an existing incompatible group assignment or assign a new incompatibly group to a modifier line.

**Note:** You do not need to assign a modifier line to an Incompatibility Group.

- **Product Precedence:** This value identifies the product precedence for the modifier line. The pricing engine evaluates the product precedence value during incompatibility resolution to select the modifier line with the highest priority (product precedence number of 1 having the highest priority). If required, you can click Modifiers to open the modifier to change the product Precedence value.

**Note:** The precedence for the qualifier attribute does not appear. These attributes are evaluated by the engine during incompatibility resolution.

## Incompatibility Resolution Examples

The following examples illustrate incompatibility processing by precedence or best price for modifiers and price lists:

### Price List: Precedence Incompatibility Resolution

The following table lists several of the seeded qualifier contexts and qualifier attributes, and the item context and item number attribute with the precedence number for each attribute.

Context Type	Context	Attribute	Precedence Number	Price List
Qualifier	Customer	Agreement type	240	B
Qualifier	Customer	Customer class	310	A, B
Qualifier	Orders	Order type	470	A
Pricing	Item	Item category	290	A, B

Consider the following setup:

#### Price List A

- Qualifier: Customer Class = VIP Customer

#### Price List B

- Qualifier: Agreement Type = Yearly and Order Type = Special.

Both price lists contain prices for Item Category Z. Item X is part of Item Category Z. A price list is not defined for Item X.

An order is placed in Oracle Order Management for Item X. When Oracle Order Management sends the pricing request without a price list name to the pricing engine, the pricing engine must search for a price list for Item X. The pricing engine will not find a price for Item X, but will find that Item X belongs to Item Category Z. Item Category Z is on both Price List A and Price List B. Because the engine can return only one price for an item, the engine must determine which price list to select.

To resolve this, the engine first evaluates the qualifier attributes and item context attribute on Price List A. On Price List A, customer class has a precedence number of 310 and item category has a precedence number of 290; therefore the engine selects item category as the highest precedence on Price List A.

On Price List B, agreement type has a segment number of 240 and order type has a segment value of 470; therefore the engine selects agreement type with a number 240 to compare to the item context. Item category with a segment number of 290 is compared to agreement type with a number of 240. Agreement type has a lower number and has the highest precedence on Price List B. The engine orders the precedence numbers for Price Lists A and B in a sequence of the lowest number to the highest. The engine chooses the lowest number (the number with the highest precedence).

The engine will order the attributes as outlined in the following table:

Context Type	Context	Attribute	Attribute Number	Price List
Qualifier	Customer	Agreement type	240	B
Pricing	Item	Item category	290	A, B

Because agreement type has a precedence of 240, the pricing engine selects the price of Item X from Price List B and returns this information to the calling application.

**Note:** If the calling application sends a validated price list to the engine in the pricing request and is qualified to receive price from this price list, the engine does not perform precedence resolution; it returns the price from the designated price list. Using the example, if Oracle Order Management sends the pricing request to use Price List A and the eligibility rules are met, the engine returns the price from Price List A to Oracle Order Management and does not need to perform incompatibility processing.

The default precedence numbers can be changed by the user on the price list window at the time of setup.

## Modifier: Precedence Incompatibility Resolution

In the following example, the pricing engine selects the following modifiers based on precedence resolution (for this example, assume that all qualifier attributes are qualified):

For this List Line Adjustments phase:

- The Preferred Discount is applied because there are no other modifiers are in this incompatibility level.
- The XYZ Brand Discount and the Summer Promotion are in the same phase and same incompatibility group. All Items has the lowest precedence number for Summer Promotion - 315. The XYZ Brand Discount has an item category precedence of 290. The XYZ Brand Discount is selected over the Summer Promotion because the precedence for item category at 290 has the highest priority.

For line charges phase, the Repack Charge is applied because there are no other modifiers in this incompatibility level.

For header level adjustments phase, the New Site Discount is exclusive and therefore the only modifier that is applied in this phase.

For header level charges phase, the Handling Charge is applied because there are no other modifiers in this incompatibility level.

Pricing Phase	Modifier	Incompat. Group	Qualifier/Product Attributes	Precedence	Engine Selection
List line adjustments	Preferred discount	Level 1	Customer class Item number	310 220	Yes
List line adjustments	Summer promotion	Level 2	Sales channel All items	320 315	No
List line adjustments	XYZ brand discount	Level 2	Item category	290	Yes
Line charges	Repack charge	Level 1	Item number	220	Yes
Header level adjustments	New Site discount	Exclusive	Site use Item all	270 315	Yes

Pricing Phase	Modifier	Incompat. Group	Qualifier/Product Attributes	Precedence	Engine Selection
Header level adjustments	Order amount discount	Level 1	Customer name	260	No
			Item number	220	
Header level adjustments	Independence Day promotion	Level 1	Sales channel	320	No
			All items	315	
Header level charges	Handling charge	Level 1	None	None	Yes

### Modifier: Best Price Incompatibility Resolution

The pricing engine determines that Modifier A and Modifier B are incompatible. The incompatibility resolve code for the phase is set to best price. Modifier A is a 10% discount and Modifier B is an *other item discount* with a Comparison Value of 200. The list price for the item that the modifier will be applied to is \$1000. The engine calculates the common benefit percent to determine which modifier has the highest number.

Modifier	Value	Calculation	Common Benefit Percent	Highest Number
A	10%	None	10	No
B	200 Comparison Value	200 Comparison Value/1000 list price	20	Yes

Modifier B has the highest common benefit percent and is applied as the best price discount.

**Note:** Best Price processing cannot be used for the following order-level modifier types: Discount, Surcharge, Freight and Special charge.



---

# Attribute Management

This chapter covers the following topics:

- Overview of Attribute Management
- Creating Contexts and Attributes for Pricing Setup Windows
- Deleting Contexts and Attributes
- Linking Attributes to a Pricing Transaction Entity
- Mapping Attributes of Type ATTRIBUTE MAPPING
- Running the Build Attribute Mapping Rules Program (Attribute Mapping Only)
- Creating Source Systems
- Creating a New Pricing Transaction Entity
- Using Custom Sourced Attributes
- Restoring Seeded Data Using the Restore Defaults Button
- Troubleshooting While Setting Up Attributes
- Upgrading Considerations
- Upgrading Context and Attributes
- Mapping of Seeded Request Types and Source Systems
- Creating PTE and Attribute Links
- Upgrade Attribute Mapping Rules
- Assigning PTE to Existing Modifiers

## Overview of Attribute Management

Attribute mapping enables you to extend your pricing capabilities by using data from a wide variety of non-standard sources to drive your pricing actions. The data sources for the qualifiers and pricing attributes can be from within or outside of Oracle

Applications.

Using the attribute management feature, you can complete the following tasks:

- Create new contexts and attributes
- Update existing contexts and attribute properties
- Disable existing attributes.

Creating new contexts and attributes extends the ability of Oracle Advanced Pricing to provide user-defined data sources to drive pricing actions. The three methods to source data for an attribute are:

- User Entered: Attribute value is entered by user.
- Custom Sourced: Custom code is used to derive an attribute.
- Attribute Mapping: The pricing engine derives information from other Oracle Applications and non-Oracle data sources.

Qualifier and pricing attributes are used to define customer or product attributes:

- A *product* attribute defines product pricing characteristics such as a product item, item number, category, or brand.
- A *customer* attribute defines customer pricing characteristics such as customer name or customer number.

When the pricing engine gets a pricing request, attribute management retrieves all values for the qualifier and pricing attributes that are associated with the transaction. The pricing engine evaluates these values to determine which price lists and modifiers are eligible for the transaction.

For example, the following setup could define a discount for a specific day of the week:

Pricing Action	Pricing Rule	Data Source	Attribute Value
Discount Modifier	Qualifier	Day of the Week	Monday

**Note:** Upgrade Considerations

Use the attribute management windows in Oracle Advanced Pricing to create and maintain contexts and attributes and to map attributes. In releases earlier than 11i, qualifier and pricing attributes were created in the Flexfields and Value Set windows using the Oracle Flexfield setup features.

## Terminology for Attribute Management

### Pricing Transaction Entity (PTE)

A PTE is an ordering structure that has associated request types and source systems. Different applications may have different request structures when they make requests to the pricing engine.

### Source System

The source system is the application that captures the pricing setup data.

### Request Type

The request type identifies the type of transaction that is being priced. Different applications make requests to the pricing engine. Request types of these applications may be different. Some applications may share their request types.

For example, iStore and Order Capture share the same request type. On the other hand, Order Management and iStore have different request structures.

## Related Topics

Creating Context and Attributes for Pricing Setup windows, page 17-3

Linking Attributes to a Pricing Transaction Entity, page 17-11

Mapping Attributes of Type ATTRIBUTE MAPPING, page 17-18

Running the Build Attribute Mapping Rules program (Attribute Mapping only), page 17-21

Creating a New Pricing Transaction Entity, page 17-25

## Creating Contexts and Attributes for Pricing Setup Windows

The following sections describe how to create contexts and attributes. First you create a context, then create its attributes to define specific values that define pricing rules. For example, a context of Customer can include pricing attributes such as Customer Name or Customer Class. Using attribute management, you can set up the following contexts and their attributes:

- Qualifier Contexts

Used to create qualifiers that determine eligibility for a modifier (who is eligible to receive the modifier). Qualifiers can be attached to price lists (only in Advanced Pricing) and modifiers.

- Product Contexts

Items that are used in price lists and modifiers are defined using the Product Context: Item. Oracle Advanced Pricing supports price and modifier definitions at

the following levels (or *attributes* of the context):

- All Items
- Item Number
- Item Category

For example, if the attribute is All Items, then the pricing engine would evaluate all items. Oracle supports only *one* Product Context which is ITEM. If you wanted to create an item hierarchy that is not seeded from Oracle, you could set up additional attributes for the Product Context of ITEM. For example, Product Context of Items, and Product Attribute of Marketing Items.

**Note:** You cannot add new contexts to the Product Context type. However, you can add attributes to the existing Product context ITEM.

- Pricing Contexts

Define eligibility for a price list line or modifier and can be used for a price list line, as a formula component, or in modifiers.

For example, a product context such as Item can be further categorized by attributes such as Item Name and Item Number, while a qualifier context such as Customer can consist of attributes such Customer Name and Customer ID.

Context Type	Context Name	Attribute
Qualifier Context	Customer	Customer Name
Product Context	Item	Joe's Items
Pricing Context	Color	Blue

**Note:** The global data element context is not supported for pricing contexts.

Once you create the context, you can create its attributes to define specific values for your pricing rules, determine how the attributes are mapped, decide which attributes can be selected from a list in the pricing setup windows, and determine which ones are used in promotional limits.

The pricing engine evaluates attributes during engine run time to determine which price lists and modifiers are eligible for the transaction. Attribute values can be derived

at the order level, line level, or for both order and line levels.

**Warning:** Pricing attributes for a price list line are not sourced at the ORDER level. Therefore, when setting up a pricing attribute (Context Type: Pricing Context) you should select the LINE level rather than ORDER or BOTH otherwise you cannot view or select that pricing attribute from a price list line. Note: You select a level for the pricing attribute in the Link Attributes window.

To create contexts and attributes using the Attribute Management feature, you must complete one or more of the following setup steps. The steps may vary depending on your requirements and current setup:

1. Verify the request types and source systems for a given PTE. Otherwise create a new pricing Transaction Entity in QP Lookups.
2. Create a new context, and then define the attributes for that context.

Pricing rules such as qualifiers and pricing attributes are used to drive your pricing actions. You can create new contexts and attributes to help set up your pricing rules in the Context Setup window.

**Note:** Oracle Advanced Pricing supports the creation of pricing attributes with the same name if they are in different contexts. But the flexfield generation process does *not* support this and fails when the Build Attribute Mapping Rules program is run. Therefore, you may want to use unique names (not duplicated) when you create attribute names.

3. Link the attribute to a PTE, and then define properties of the attribute for the given PTE.

Once you create the context-attribute grouping, you can link it to a specific PTE. For mapped attributes, define Attribute Mapping Rules, and then run the Build Attribute Mapping Rules Program.

4. Run the Build Attribute Mapping Rules program.

This step applies only to attributes with the Attribute Mapping Method of Attribute Mapping. Whenever you create or update these attributes or change attribute mapping rules, you should run the Build Attribute Mapping Rules program.

**Tip:** If the Build Attribute Mapping Rules program is run when you create the attribute, you will be prompted to run it again when you use the attribute (such as in a formula or as a pricing attribute). Therefore, you may want to run it only once when you use the

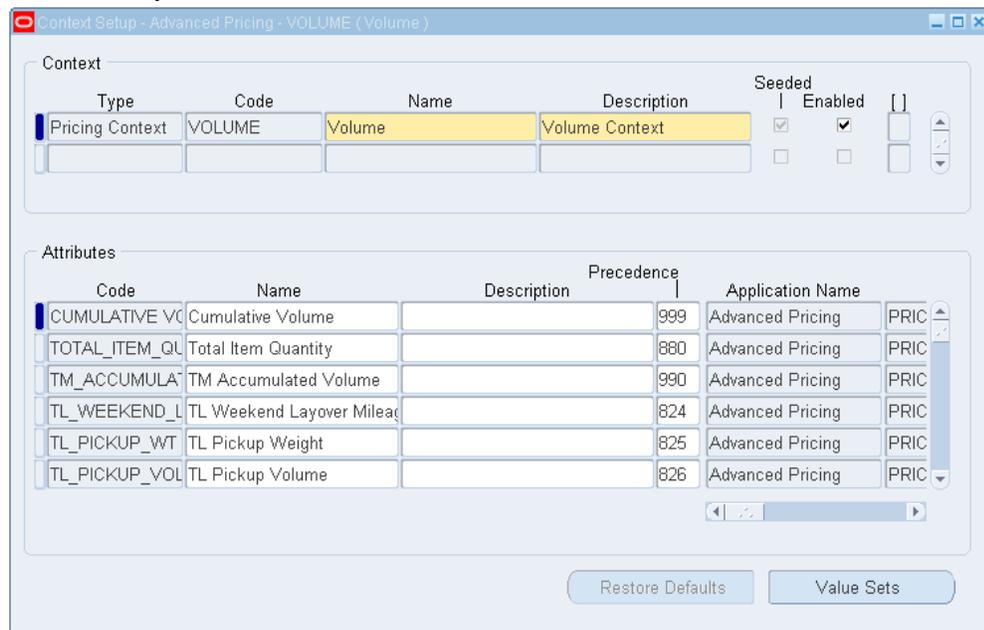
attribute.

5. Use the attribute in a valid pricing setup.
6. Enter an order. Verify that the mapped and user-enter attributes have been correctly passed to the pricing engine from the Pricing Engine Request Viewer.

**To create contexts:**

1. Navigate to the Context Setup window.

**Context Setup window**



2. Complete your entries as required:
  - Type: Select the context type you want to create, such as a Qualifier, Pricing, or Product Context.
  - Code: Enter a Code that is a short name for the context. Once you create the code, you cannot update it.
  - Name and Description: Enter a name and description for the context. The Name you create will be available from the list of values (for the contexts) in sales order, and from the pricing context fields in Price List and Modifier windows. Because new contexts can be introduced by different applications from Oracle Pricing, enter a meaningful description.

- Seeded box: The Seeded box is selected automatically if the context is a seeded value. It remains selected even if you overwrite a seeded context.
  - Enabled box: Select the Enabled box to make this context available for pricing setup windows. If this box is cleared, the context will not appear in the Context field on the pricing setup windows and all the attributes that are defined for the context will also be unavailable for setup.
3. When you have completed your entries, enter the attributes for this context in the Attributes region.

#### To create attributes

4. In the Context Setup window, select the context to which you want to add attributes.
5. Then, in the Attributes region, complete your entries for the selected attribute:
- Code: Enter a unique short name for the attribute (the code name is used internally). Once created, it cannot be updated.
  - Name and Description: Enter a display Name and Description for the attribute. Since new attributes in the attribute mapping manager can be introduced by different applications other than Pricing, entering an attribute description is informative.

**Note:** You can update seeded names and descriptions; however, if the original seeded values are changed, they can be restored using the Restore Defaults button.

- Precedence: Enter a numeric Precedence value that can be evaluated during incompatibility processing. For example, if two incompatible discounts qualify for the same item, the discount with the higher precedence is given (the lower the number the higher the precedence, so a precedence of 1 is selected before 2). Precedence numbers in the series of 5s and 10s are reserved for seeded qualifiers/pricing attributes and should not be used. The precedence is restricted to a maximum of 3 digits (any number between 1 and 999).
- Application Name: Enter the name of the application that created this attribute. If an application name is not entered, the system defaults Oracle Pricing as the creator of the attribute.
- Column Mapped: Select a value to map the attribute to a column in the pricing tables. You can select from the names of the unused columns only:
  - Columns QUALIFIER\_ATTRIBUTE1 through QUALIFIER\_ATTRIBUTE30

are reserved for seeded qualifier attributes.

- Columns 1 to 100 are available and you should use pricing attributes 1 to 30 as user-entered pricing attributes. Columns 1 - 30 are available only for user Datamerge.
- The Pricing Manager user's list has unused columns between 31 and 100.
- Value Set: Select a value set to define the valid values for an attribute. The Datatype field displays the format type of the selected value set such as numeric (Number) or alphabetic (Char). Alternatively, to create a new value set or view an existing one, click Value Sets. Once you have created a new value set, you can select the newly created value from the Value Set field.

**Note:** You cannot set up dependent value sets for any attributes. A value set for an attribute can be replaced only by a value set with the same datatype as the one being replaced.

- Seeded box: The Seeded box is selected automatically if the context is a seeded value. It remains selected even if you overwrite a seeded context.
- Required: Select Required to make the attribute a required value in pricing windows. For required attributes, a user must enter the specified attribute, for example, every time he or she creates a sales order line.

**Note:** Previously the Required box was updated in the Segments (Pricing Contexts) window, and reflected in the flexfield. However, these updates will not be updated in attribute management. Therefore the Required box should be selected in attribute management.

- Party Hierarchy Enabled box: Hierarchies consist of tiered relationships where one party is ranked above another, for example, a corporate hierarchy where different companies are related as parent, subsidiary, headquarters, division and so on. Select this check box if the qualifier attribute references Oracle TCA (Trading Community Architecture) parties and if price lists or modifiers with this qualifier attribute need to be made available to the party hierarchy. This enables you to cascade a discount or price list to subsidiaries and branches by setting up a qualifier at the top level rather than creating multiple qualifiers, one for each subsidiary or division.

If the Party Hierarchy Enabled check box is selected for an attribute that is *not* based on TCA Parties, a warning appears. This enables the check box to be set for user-defined attributes that use custom tables with a party\_id reference.

Setting this flag for any other attributes can yield incorrect pricing results if qualifiers reference an attribute with Applies To Party Hierarchy selected. Once this check box is selected, you cannot deselect it if any qualifier references an attribute that has Applies To Party Hierarchy selected.

The following seeded qualifier attributes are based on Oracle TCA parties:

- Customer Party (qualifier context = Party Information (ASOPARTYINFO))
- Party Id (qualifier context = Customer (CUSTOMER))
- Buying Groups (qualifier context = Customer Group (CUSTOMER GROUP))
- Supplier (qualifier context = Party Information (PARTY))
- Buyer (qualifier context = Party Information (PARTY))

**Note:** The profile option QP: Pricing Party Hierarchy Type governs the relationship type for the TCA Party seeded qualifier attributes and for user-defined qualifier attributes that are enabled for party hierarchy. The Party Hierarchy Enabled check box is available for Oracle Advanced Pricing users, not for Basic Pricing users. For more information, see *Oracle Advanced Pricing Implementation Guide, Profile Options*.

6. Once you have created the context and its attributes, you can link an attribute grouping to a specific Pricing Transaction Entity (PTE).

#### **Attribute Levels and Restrictions in Pricing Setup windows**

The following sections describes the attributes (and associated restrictions) that appear in the list of values based on the attribute level that is assigned to the attribute.

- Price Lists window
  - In the Qualifiers tab, the list of values displays qualifier attributes with the attribute level of LINE or BOTH. This occurs because an order line may qualify for a price list based on a qualifier attribute for the order line or both the order header and order line.
  - The values for the pricing attributes appear at the line attribute level.
- Modifier List Setup window
  - The values for qualifier attributes in the Qualifier - Line Level Qualifiers window displays qualifier attributes with attribute levels of ORDER or BOTH if the modifier level is Order.

- The values for the qualifier attributes in the Line Qualifiers window displays qualifier attributes with attribute levels at LINE or BOTH if the Modifier Level Code is Line or Line Group.
- The values for the qualifier attributes in the List Qualifiers window displays qualifier attributes with attribute levels at LINE, ORDER or BOTH.
- The values for pricing attributes display pricing attributes with attribute level at the LINE level.
- Qualifier Group Setup window
 

The values for the qualifier attributes on the Qualifier Group Setup window displays all levels of qualifier attributes. The following conditions apply to the forms-based user interface: A qualifier group cannot have one qualifier attribute with the level Order and another qualifier attribute with the level Line with the same qualifier grouping number. All the qualifiers within a qualifier grouping number must have the attributes with the level either as Order or Both or as Line or Both.
- Formula Lines and Factors List
 

Because a formula can be applied either at order level or at line level, you cannot at definition time restrict the attributes appearing in the lists based on the attribute level. The list of values has no level-based restrictions.

### **Pricing Attributes and Flexfields tables**

Oracle Order Management (OM) uses the flexfield structure for pricing attributes; therefore, for pricing attributes to display in Oracle Order Management UIs, these definitions need to be stored in the flexfield tables as well. All definitions of contexts and attributes are stored in pricing (QP) tables; however, the same data is also stored in flexfield tables if the context type is Pricing Attribute. So when a context of type Pricing Attribute is created using the Contexts and Attributes window, the context is also created as a flexfield pricing context in the flexfield tables.

However, you must set up pricing attributes from the Context and Attributes setup menu not in flexfields, because if a pricing attribute is set up in the flexfield tables, the setup data will not be available in the QP tables (because data is not copied from flexfield tables to QP tables). So even though the pricing attribute appears in Order Management windows, the pricing engine cannot view the attribute.

The following steps describe how the context of type Pricing Attribute is registered as Flexfield Pricing Contexts in the flexfield tables after being created in the Contexts and Attributes window:

1. A context of type Pricing Attribute is registered automatically if the mapped column is greater than PRICING\_ATTRIBUTE30.

2. Once the context of type Pricing Attribute is created and registered, the system automatically compiles the flexfield in the background. You can follow the stage of compilation by viewing your concurrent requests.
3. Descriptive flexfields in Advanced Pricing for Pricing Context gets mapped to Pricing Context and Product Context (context=ITEM only). All qualifier context gets mapped to Qualifier Context.

A context of Pricing Attribute and its related attributes will get updated and deleted in the flexfield (and the Context and Attributes window) if it meets the deletion criteria.

## Related Topics

Overview of Attribute Management, page 17-1

Linking Attributes to a Pricing Transaction Entity, page 17-11

Creating a New Pricing Transaction Entity, page 17-25

Restoring Seeded Data Using the Restore Defaults button, page 17-30

## Deleting Contexts and Attributes

You can delete contexts and attributes except under the following conditions:

### To delete contexts:

You cannot delete a context if it has one or more attributes.

### To delete attributes:

Attributes that are already used in pricing setup windows cannot be deleted.

## Linking Attributes to a Pricing Transaction Entity

A PTE consists of a group of applications that point to the same setup data and attributes, and includes request types and source systems. The source system is the application that captures the pricing setup data and the request type identifies the type of transaction that is being priced.

Once you create a context and its attributes, you link them to a specific PTE. For a given PTE, this combination becomes available in the pricing setup windows. Each PTE has its own unique combination of attributes.

The following table displays the seeded PTEs and their request types and source systems in Oracle Advanced Pricing:

<b>Pricing Transaction Entity (Code)</b>	<b>Source Systems</b>	<b>Request Types</b>
Complex MRO (CMRO)	AHL	AHL
Demand Planning (DEMAND)	QP	MSD
Intercompany Transaction (INTCOM)	INV, QP	IC
Logistics (LOGSTX)	FTE	FTE
Order Fulfillment (ORDFUL)	AMS, ASO, OKC, OKS, QP	ASO, OKC, OKS, ONT
Procurement (PO)	PO	PO

**To link attributes to a PTE:**

1. Navigate to the Advanced Pricing - Pricing Transaction Entity-Attribute Linking window.

### Pricing Transaction Entity-Attribute Linking window

Assigned to PTE	Code	Name	Description	Seeded	Enabled
<input type="checkbox"/>	AHL_PRICING	CMRO Pricing Attributes	CMRO Pricing Attributes	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
<input checked="" type="checkbox"/>	BREAK_UOM	Break UOM	Break UOM	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
<input type="checkbox"/>	Global Data Elemer	Global Data Elements	Global Data Element Context	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
<input type="checkbox"/>	LOGISTICS	Logistics	Logistics Exchange Context	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
<input checked="" type="checkbox"/>	OTA	OTA Pricing	OTA Pricing Context	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
<input checked="" type="checkbox"/>	PERIODICITY	Periodicity	Charge Periodicity	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
<input type="checkbox"/>	PO_PRICING_ATT	PO Pricing Attributes	PO Pricing Attributes	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
<input checked="" type="checkbox"/>	PRICING ATTRIBUT	Pricing Attribute	Pricing Attribute Context	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

2. Complete your entries in the Advanced Pricing - Pricing Transaction Entity-Attribute Linking window:
  - Pricing Transaction Entity: Select a PTE.
  - Context Type: Select the context type such as Pricing Context, Product Context, or Qualifier Context.
  - Show Linked Contexts check box: Select the Show Linked Contexts box to display only those contexts that are assigned to the selected PTE. The contexts that match the criteria for the selected PTE and context type appear in the Contexts region.
3. Select a context whose attributes you want to link to a PTE. If the Assigned to PTE box is not checked, attributes have not been created or selected for that context in the given PTE.
4. Click Link Attributes to display the Link Attributes window.

### Link Attributes window

Code	Name	Precedence	Level	Attribute Mapping Method	LOV	Enabled	Use
USAGE_UOM	Usage UOM	500	LINE	ATTRIBUTE MAPPING		<input type="checkbox"/>	
						<input type="checkbox"/>	
						<input type="checkbox"/>	
						<input type="checkbox"/>	
						<input type="checkbox"/>	
						<input type="checkbox"/>	
						<input type="checkbox"/>	
						<input type="checkbox"/>	

Restore Defaults    Attribute Mapping

- Code: Enter a short name for the attribute.
- Level: Select an attribute Level: ORDER, LINE, or BOTH (Order and Line levels).

For example, for an attribute agreement ID, if the level is ORDER then only the agreement ID at the header level will be attribute mapped. The agreement ID on order lines will not be attribute mapped. This level is also used to selectively display pricing attributes and qualifiers in the Pricing Setup list of values.

**Important:** Pricing attributes for a price list line are not sourced at the ORDER level. Therefore, when setting up a pricing attribute (Context Type: Pricing Context) you should select the LINE level rather than ORDER or BOTH otherwise you cannot view or select that pricing attribute from a price list line.

- Attribute Mapping Method: Select an attribute mapping method to define how you want the attribute to derive its value:

However, do not change the mapping method from USER ENTERED to ATTRIBUTE SOURCING for the seeded attributes Item Quantity and Item Amount (pricing context of Volume).

- ATTRIBUTE MAPPING: Attribute mapping is required. After you complete the remaining entries in the Link Attributes window, click Attribute Mapping to define the attribute mapping rules for the selected attribute.

**Note:** If ATTRIBUTE MAPPING is selected then the Attribute Mapping button is enabled. If USER ENTERED or CUSTOM SOURCED is selected, then the Attribute Mapping button is grayed out.

- CUSTOM SOURCED: User provides a code. For more information, see *Oracle Advanced Pricing Implementation Guide, Using Custom Sourced Attributes*.
- RUNTIME SOURCED: The pricing engine derives a runtime sourced value using a custom API. This method is used to source an accumulation attribute at runtime. During accumulated range break calculations, the pricing engine calls the Runtime Sourcing API to acquire an accumulation value for the accumulation attribute (defined as RUNTIME SOURCE in the Attribute Mapping setup). For more information about using runtime sourcing for accumulation attributes, see *Oracle Advanced Pricing Implementation Guide, Setting up Runtime Sourcing for Accumulated Range Breaks*.
- USER ENTERED: The attribute value is entered by the user; therefore attribute mapping is not required. However, if USER ENTERED is selected and the context type is Pricing, you must complete the following steps to view this attribute in the Order Management Sales Order pad:
  - i. Navigate to the Flexfields window.
  - ii. Select the Freeze Flexfield Definition box.
  - iii. Click Compile to compile the flexfield.

The attributes ITEM AMOUNT and ITEM QUANTITY are user-entered but are sourced internally by the pricing engine.

**Note:** If a change is made in the Flexfields window and you click the Compile button, the corresponding change will not be visible in the Flexfield window in integrating applications until a responsibility switch or relogin has been made.

For example, if you select the Required Flag for a pricing attribute in the attribute manager setup window, the attribute will not appear as a mandatory field (in yellow) in the Pricing Attributes flexfield window in the Sales Order pad, until a responsibility switch or relogin has been made. This occurs because the information is based on cached information. Therefore you need to log in again or switch

responsibility to refresh the cached information and update the values.

- LOV (list of values) Enabled: Select LOV Enabled to display the attribute in the list of values of modifier, qualifier, price list, and formula windows. If LOV Enabled is not selected, the attributes will not be available in these windows. However, existing modifiers that are defined using that attribute will act the same way:
  - If LOV Enabled is selected and the Attribute Mapping Enabled box is cleared, then even though the attribute is available for setup, a mapping rule is not created. So if the attribute is used in the qualifier, modifier, price list, and formula setup windows, an error occurs.
  - The LOV Enabled box is selected for all seeded attributes to make them available in the LOVs from price list, qualifier, modifier, formula and other setup windows. This box will be unaffected whenever data upgrade scripts are run.
- Use in Limits box: If Use in Limits is selected, the attribute can be selected from the Attribute list of values in the Limits setup windows (including the Other Attributes window).
- Attribute Mapping Enabled box: If you selected ATTRIBUTE MAPPING as the attribute mapping method, select the Attribute Mapping Enabled box so that the concurrent program Build\_context API can successfully map the attribute. If the box is cleared, the attribute will not be mapped. This box is cleared for all seeded attributes to prevent the mapping of unwanted attributes.  
The Column Mapped value is a default value.
- Used in Setup box: The Used in Setup box indicates whether the attribute is used in an active pricing setup such as a price list, modifier, formula, qualifier, or limit. The setting for the Used In Setup box (selected or cleared) depends on the profile option QP: Build Attributes Mapping Options and the Active box values:

<b>If the setting for QP: Build Attributes Mapping Options is:</b>	<b>Then...</b>
Yes (map attributes used in active pricing setup)	The Used In Setup box is set to Yes, only when an attribute is used in the active pricing setup.

If the setting for QP: Build Attributes Mapping Options is:	Then...
No (map all attributes)	The Used In Setup box is set to Yes when an attribute is used in both the active and inactive pricing setup.
No	The Used In Setup box is set to No when an attribute is used in the inactive pricing setup.

**Note:** The Used In Setup box may not always reflect the exact status of current attribute usage for some attributes. For example, if an attribute is deleted, the Used In Setup box may appear as "selected" even though the attribute has been deleted. This occurs because the status is not updated automatically.

However, when the concurrent program Build Attribute Mapping Rules is run, the Used In Setup box is updated to reflect the exact status of the attribute usage.

- Attribute Mapping Status check box: The Attribute Mapping Status check box is selected (or cleared) automatically by the concurrent program, which generates the Build\_Contexts API for mapped attributes. The Attribute Mapping Status check box is cleared in the following cases:
  - After you define the attribute mapping for new attributes (but before you run the concurrent program).
  - When an attribute mapping rule is modified.
  - If the Attribute Mapping Status box is cleared and 1) Attribute Mapping Method is ATTRIBUTE MAPPING and 2) Attribute Mapping Enabled is selected, then the concurrent program must be run to regenerate the Build\_Contexts api. The Attribute Mapping Status box is selected when the concurrent program is run successfully. The field is not navigable.

**Note:** A warning box displays if you try to use new attributes with the Attribute Mapping Method as ATTRIBUTE MAPPING and the Mapping Status box as No.

The Pricing Transaction Entity-Attribute Linking window displays the newly created context; the Assigned to PTE box indicates that the attributes have been

assigned to this PTE. The attributes that you created can be viewed in the pricing setup windows.

- For CUSTOM SOURCED, RUNTIME SOURCED, and USER ENTERED attributes, once you have completed your entries, the attribute is now linked to a Pricing Transaction Entity.
- For ATTRIBUTE MAPPING attributes, click Attribute Mapping to define the Attribute Mapping rules for the selected attribute.

Optionally, click Contexts to add a new context or to update, enable, or view an existing context.

## Related Topics

Mapping Attributes of Type ATTRIBUTE MAPPING, page 17-18

Using Custom Sourced Attributes, page 17-30

Running the Build Attribute Mapping Rules program, page 17-21

## Mapping Attributes of Type ATTRIBUTE MAPPING

This section applies *only* to attributes whose Attribute Mapping Method is ATTRIBUTE MAPPING.

After you have linked the context and attribute(s) to a PTE, you can complete the attribute mapping process in the Attribute Mapping window. Mapping an attribute enables the pricing engine to derive values for an attribute that is used in modifiers, price lists, or qualifiers.

Attribute mapping provides additional flexibility for the pricing engine: for example, the value for Customer Region can be derived from a OE\_ORDER\_PUB.G\_HDR.sold\_to\_org\_id in Order Management; however this same element can also be derived from ASO\_PRICING\_INT.G\_HEADER\_REC.cust\_account\_id in Order Capture.

Whenever you create or update these ATTRIBUTE MAPPING attributes or change attribute mapping rules, you should run the Build Attribute Mapping Rules program.

**Note:** You do not need to run the Build Attribute Mapping Rules program for attributes for which the Attribute Mapping Method is USER ENTERED or CUSTOM SOURCED.

### To map attributes of type ATTRIBUTE MAPPING:

1. In the Link Attributes window, click the Attribute Mapping button to display the Attribute Mapping window. Complete your entries as required:

### Attribute Mapping window

The screenshot shows the 'Attribute Mapping' window for 'Order Fulfillment - Party Information - Bill to Party Site'. It is divided into three main sections:

- Request Types:** A table with columns 'Application name', 'Request Type', and 'Description'.

Application name	Request Type	Description
Advanced Pricing	ASO	Order Capture
	OKC	Oracle Contracts Core
	ONT	Order Management Order
	OKS	Oracle Contracts for Service
- Header Level:** Configuration fields for the header level.
  - Global Object name: ASO\_PRICING\_INT.G\_HEA
  - Seeded Source Type: PL/SQL API
  - User Source Type: PL/SQL API
  - Seeded Value String: ASO\_PRICING\_INT.GET\_S
  - User Value String: ASO\_PRICING\_INT.GET\_S
  - Seeded:
  - Enabled:
- Line Level:** Configuration fields for the line level.
  - Global Object name: ASO\_PRICING\_INT.G\_LINE
  - Seeded Source Type: PL/SQL API
  - User Source Type: PL/SQL API
  - Seeded Value String: ASO\_PRICING\_INT.GET\_S
  - User Value String: ASO\_PRICING\_INT.GET\_S
  - Seeded:
  - Enabled:

A 'Restore Defaults' button is located at the bottom right of the window.

Application Name: Select the name of the application that created the mapping rule. If an application name is not selected, the system defaults Oracle Pricing as the creator of the mapping rule.

#### 2. Complete your entries in the Header Level region:

**Note:** The fields in the Header Level region are enabled only if the Attribute Mapping level (defined in the Assign Attributes window) for the attribute is Order or Both. The fields are grayed out if the level is Line.

- Global Object name (display-only): Defaults with the value defined in the Request Types tab of the Pricing Transaction Entity - Source System and Request Types window.
- Seeded Source Type: Appears if the record is seeded. This field is view-only and cannot be updated. To modify seeded mapping, you can enter data into the corresponding User Source Type.
- User Source Type:
  - PL/SQL API: Attribute can be sourced directly from a global structure that is defined for the given source system. For Oracle Order Management, all

the record structures are defined in the package `OE_ORDER_PUB`. For example, if you want to use the payment term ID as the source value for the new segment that you have defined, enter `G_LINE.payment_term_id` in the function name. You have two record structures available.

- `OE_ORDER_PUB.G_LINE` contains all the possible values of a sales order line.
- `OE_ORDER_PUB.G_HDR` contains fields from Order headers. The structure of the `Line_rec_type` and `header_rec_type` can be obtained from the Manufacturing Open Interface Manual. (For IStore/OC the equivalent global structures are defined in the package `ASO_PRICING_INT`.)
- **PL/SQL API Multi-Record:** You can write a custom API (must be a function) that returns multiple values. The output of your function can be only a table of `VARCHAR2`s, for example, to get the inventory categories for an item. For more information, see seeded sourcing for Item Categories or customer class as an example of multi-Value pl/sql API.
- **Constant Value: Constant:** Enter a constant value that will always be mapped to this attribute for the given condition.
- **Application Profile:** Select the profile option from where you want to get the default value for this attribute. A LOV will provide a list of valid profile options such as OE: Item Validation Organization.
- **System Variable:** Enter the system variable that will be mapped to the attribute for the given condition such as `SYSDATE`.

The Seeded Value String fields displays the seeded value string of the record. This field is view-only and cannot be updated. To modify a seeded value string, you can enter data into the corresponding user value string.

- **User Value String:** Enter a user value string.  
For example, the user value string for pulling the schedule ship date (Schedule Date) from Order Management is `OE_ORDER_PUB.G_LINE.SCHEDULE_SHIP_DATE`.
- **Seeded box:** Indicates whether the mapping rule is seeded.
- **Enabled box:** Select the Enabled box to enable attribute mapping rules for various Request types for qualifier, product, or pricing attributes. Alternatively, clear the Enabled box to disable the attribute mapping rules for the Request types for that pricing transaction entity (PTE)

**Note:** When you select or clear the Enabled box, a dialog box advises you to run the Build Attribute Mapping Rules program for the change to take effect.

3. In the Line Level region, complete your entries if the attribute level, which is defined in the Assign Attributes window, is LINE or BOTH. This region will be grayed out if the selected attribute level is ORDER. The field descriptions are the same as described for the Header level region.
4. After you have completed your entries in the Attribute Mapping window, run the Build Attribute Mapping Rules program.

## Related Topics

Running the Build Attribute Mapping Rules Program (Attribute Mapping only), page 17-21

Linking Attributes to a Pricing Transaction Entity (PTE), page 17-11

## Running the Build Attribute Mapping Rules Program (Attribute Mapping Only)

This section applies only to attributes with the following setup criteria:

- Attribute mapping method is ATTRIBUTE MAPPING.
- Attribute Mapping Enabled check box is selected.

**Note:** You do not need to run the Build Attribute Mapping Rules program for which the attribute mapping method is USER ENTERED or CUSTOM SOURCED.

You should run the concurrent program Build Attribute Mapping Rules program (attribute mapping method must be Attribute Mapping and the Attribute Mapping Enabled check box must be selected) every time you:

- Create or update a new attribute (one that has not been used by any other modifier, price list, or formula).
- Change any attribute mapping rules.

Any new attributes that are used *after the program was last run* are mapped dynamically to prevent any attributes from not getting mapped. Although the attributes are mapped dynamically and can be used in pricing setups, you should run the Build Attribute

Mapping Rules program for better performance.

**Warning:** Because the Build Attribute Mapping Rules program changes the status of database objects, do not run this program during peak hours.

### **What happens when the Build Attribute Mapping Rules program is run?**

When you run the concurrent program Build Attribute Mapping Rules, it dynamically generates the package QP\_BUILD\_SOURCING\_PVT which contains attribute mapping calls to build attribute mapping rules for attributes. This package contains only the rules of used attributes (Qualifiers and Pricing Attributes that are used in any pricing setup) that can be mapped at run-time.

The program generates the attribute mapping rules for all the attributes that have the Attribute Mapping Enabled box selected. When the Build Attribute Mapping Rules program runs successfully, the Attribute Mapping Status box is selected for those attributes for which an attribute mapping rule is generated.

The Build Attribute Mapping Rules program displays a status message to advise whether the program finishes successfully. The window automatically re-queries the database if the Build Attribute Mapping Rules process is successful and moves focus to the Attribute Mapping Status box.

**Note:** If an error in the build sourcing occurs, then all newly-created attribute mapping rules will fail and continue to fail until the error is resolved.

### **How attribute mapping works at run time**

The calling application calls QP\_Attr\_Mapping\_PUB.Build\_Contexts, which starts the attribute mapping routines and returns the attributes to the calling application.

1. The calling application appends the user entered attributes and "asked for" qualifiers to this request.
2. The calling application then calls the pricing engine with these mapped attribute values.
3. The pricing engine also appends a few internally mapped attributes to this request.
4. The pricing engine processes the request and returns the results to the calling application

The PTE helps narrow the data that the search engine evaluates. The search engine evaluates only the setup data that is generated by all the source systems that are defined for that PTE. It also makes contexts of one PTE unavailable to other pricing applications families.

## Checklist for building attribute mapping rules

After you successfully map an attribute, a message advises you that the attribute mapping rule was generated successfully. However, sometimes you may find that the new Attribute Mapping box for that attribute remains cleared when it should be selected. This may occur when you create a new attribute, link it to a PTE successfully, and then map it using the Build Attribute Mapping Rules from the Tools menu.

To ensure a successful mapping and to avoid the situation described previously, you must ensure that the following conditions are met:

- The Attribute Mapping Enabled box must be selected.
- The attribute mapping method must be ATTRIBUTE MAPPING. Attributes with mapping method USER ENTERED and CUSTOM SOURCED are never meant to be created by the Build Attribute Mapping Rules program.
- If the attribute is newly created, you must ensure that it is attached to at least one valid pricing setup such as a modifier, price list, or limit.
- Ensure that the PTE-Attribute link that was created has at least one attribute mapping rule.

## To run the Build Attribute Mapping Rules Program:

You can use this task flow for attributes with the following criteria:

- Attribute Mapping Enabled selected
- Attribute mapping method is attribute mapping
- Attribute Mapping Status box is cleared

When the Build Attribute Mapping Rules program is run, it generates a source code for a new qualifier or pricing attribute (for which attribute mapping is defined) and generates a source code for attributes for which attribute mapping rules are modified:

**Note:** Because this program changes the status of database objects, you should run this program during off peak hours.

1. Ensure that this attribute has been used in a valid pricing setup such as modifiers or price list. Set up a modifier and attach the newly created qualifier/pricing attribute.

For example, for this qualifier to be mapped by Oracle Order Management, you need to regenerate the attribute mapping package by running the program Build Attribute Mapping Rules

2. Confirm that the program has finished successfully.

This program needs to be run when a qualifier is used for the first time to setup a modifier or a price list. For example, if you are using Customer Class as a qualifier for the first time in your install, you need to run this program.

3. Navigate to the Pricing Transaction Entity - Attribute Linking window.
4. Find the PTE.
5. In the Context Type field, select the Context Type from LOV.
6. Navigate to Tools and select Build Attribute Mapping Rules to run the Build\_context api, which will generate a mapping rule code only for those attributes that have the Attribute Mapping Enabled box selected.
7. Check whether the program has finished successfully. If it has, the Attribute Mapping Status box is selected for the attributes for which a mapping rule code is generated.

**Note:** When the concurrent program Build Attribute Mapping Rules is run, the Used In Setup box is updated to reflect the status of the attribute usage. When the attribute is removed, the Used In Setup box will remain selected.

8. Enter the order
9. Verify through the Pricing Engine Request Viewer window that the attribute has been correctly sent to the pricing engine. You can also verify whether the pricing engine used the correct attributes while pricing. For more information, see: *Pricing Engine Request Viewer*.

**Note:** Before running the Build Attribute Mapping Rules program, set the profile option QP: Build Attributes Mapping Options to the desired setting.

- When the profile option is set to Yes, mapping rules can be generated for attributes that are used in active pricing setup.
- When set to No, mapping rules are generated for attributes in both active and inactive setups.

### **To run the Attribute Mapping Rules Error Report:**

If you run the Build Attribute Mapping Rules program and it generates attribute mapping messages, you can run the Attribute Mapping Rules Error Report to generate a list of all invalid attribute mapping rules and any warnings. Other status messages are

also provided. See *Oracle Advanced Pricing User's Guide, Reports and Concurrent Programs*, for more information.

1. Navigate to the Pricing Transaction Entity - Attribute Linking window.
2. Select Build Attribute Mapping Rules from the Tools menu to run the Build Attribute Mapping Rules program. This generates a mapping rule code only for those attributes that are used in the pricing setup and have the Attribute Mapping Enabled box selected.

If the program runs successfully, the Attribute Mapping Status box is selected for attributes that were successfully mapped.

## Creating Source Systems

In the Source Systems window, you can view request types and source systems, and add or update associations between a request type and its source system. A request type refers to the type of transaction that is being priced, such as an Order Management Order or Intercompany Invoicing that requests pricing or other information from a source application such as Oracle Pricing or Oracle Inventory. A source system defines the application that generates setup data such as Oracle iStore, Oracle Pricing (pricing setup data includes modifiers, formulas, and so on) and Oracle Marketing. A request type could be paired with one or more source systems depending on your business requirements. For example, the request type Order Management Order can be associated with source systems Oracle Marketing, Order Capture, and Oracle Pricing.

Request types and source systems can be grouped and associated with a PTE. PTEs are useful for narrowing the data that the search engine examines because it needs to evaluate only the setup data that is generated by the source systems that are defined for that pricing transaction entity. The same PTE ensures that the applications sharing the same data always give the same price for an item irrespective of the request type. All applications belonging to the same pricing transaction entity have the same set of attributes available to them. Attributes (see the attributes section) can be linked to one or more pricing transaction entities.

## Creating a New Pricing Transaction Entity

A *pricing transaction entity* (PTE) contains a group of applications that share the same setup data and attributes. For example, Oracle Istore and Oracle Order Management belong to the same pricing transaction entity because both are Order Fulfillment systems and share the same setup data. Conversely, Oracle Transportation Execution resides in a different pricing transaction entity and has its own attribute and setup information.

Using PTEs provides the following advantages:

- The data that the search engine evaluates is reduced because only the setup data

that is generated by the source systems defined for that PTE needs to be evaluated. The same PTE ensures that the applications sharing the same data always give the same price for an item regardless of the request type.

- All applications belonging to the same pricing transaction entity have the same set of attributes available to them. Attributes can be linked to one or more pricing transaction entities.

**Important:** The profile option, QP: Pricing Transaction Entity, defines the current PTE in use. Set the profile option to the appropriate value before creating or querying any setup data in the pricing application because only those contexts and attributes that are assigned to the current PTE can be selected from the pricing setup windows. When you query setup data, only those contexts and attributes that are assigned to the current PTE appear.

Oracle Advanced Pricing provides the following seeded PTEs:

- Complex MRO (CMRO)
- Demand Planning (DEMAND)
- Intercompany Transaction (INTCOM)
- Logistics (LOGSTX)
- Order Fulfillment (ORDFUL)
- Procurement (PO)

If required, additional PTEs can be created.

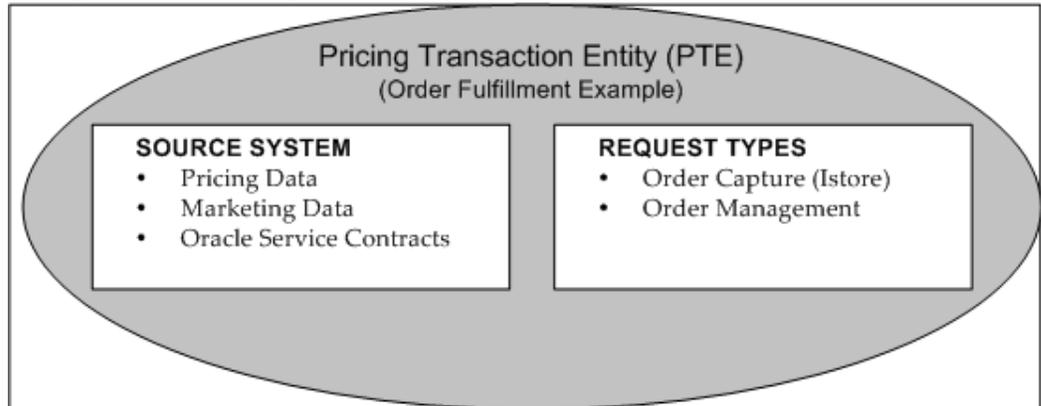
#### **When to define a new PTE**

Although Oracle provides seeded PTEs, you can set up additional PTEs to group source systems and request types. When a new request type is created - for example, a new ordering structure is added - your business must decide if the request type will use the same set of source systems in the existing PTE. A new PTE needs to be created only if the new request type uses a different ordering structure and a different set of source systems. Request types have to be unique across PTEs.

#### **Example of a PTE**

The following graphic displays the Order Fulfillment PTE. Within this PTE, the request types such as Oracle iStore and Oracle Order Management evaluate pricing data.

**Pricing Transaction Entity - Order Fulfillment**



**Global Structures**

The following table shows examples of the global structures at the order level and line level for the different request types of the Order Fulfillment PTE.

PTE	Request Type	Order Level Global Structure	Line Level Global Structure
Order Fulfillment	Order Capture	ASO_PRICING_INT. G_HEADER_REC	ASO_PRICING_INT. G_LINE_REC
Order Fulfillment	Oracle Contracts Core	OKC_PRICE_PUB. G_CONTRACT_INF O	OKC_PRICE_PUB. G_CONTRACT_INF O
Order Fulfillment	Order Management Order	OE_ORDER_PUB. G_HDR	OE_ORDER_PUB. G_LINE
Intercompany Transaction	Intercompany Invoicing	INV_IC_ORDER_PU B.G_HDR	INV_IC_ORDER_PU B.G_LINE

**To create a new PTE:**

1. Navigate to the Pricing Lookups window.
2. Query on QP\_PTE\_TYPE as lookup type to display the existing PTEs.
3. Enter a code (short name), meaning, and description for the new PTE.
4. Select Enabled.

5. Save the record.

### **Pricing Transaction Entity - Source System and Request Types window**

6. Navigate to the Pricing Transaction Entity - Source System and Request Types window to complete your entries. In this window, you can review, update, or create new request types, source systems, and functional areas for a given PTE:

- **Name:** Displays the short name for the PTE, for example, ORDFUL (Name) and Order Fulfillment (Description).
- **PTE Parameters button:** Click PTE Parameters to search and update parameter values for the PTE. For more information about pricing parameters, see Overview of Pricing Parameters, page 5-43.
- **Code (Source Systems tab):** Short name for the source system application.

A source system defines the application that generates setup data such as Oracle iStore, Oracle Pricing and Oracle Marketing. The Source Systems tab lists the related source system(s) for the selected PTE.

**Note:** A source system cannot be deleted if it is used in a setup for a given PTE.

### **Request Types tab**

A request type identifies the type of transaction that is being priced. For example, Oracle Order Management is a request type that requests a price from the pricing application (the source).

- **Header or Line Structure, Header and Line View, or both:** For you to map the data, a request type can have a global record structure or a view defined. You can either enter global structures for a header and line or view names for header and line. If data is entered in any of the following fields, a list of values (LOV) based on this data will appear in the ValueString field of the Attribute Mapping setup window. If no data is entered in any of the following fields, no LOV will be provided in the Value String field of the Attribute Mapping setup window.
  - **Header Structure:** The global structure for the header.
  - **Line Structure:** The global structure for the line.
  - **Header View name:** View name for the header.
  - **Line View name:** View name for the line.
- **Enabled:** Select to activate the request type.

- Request Type button: Click to view or update parameter values for the selected request type. For more information about pricing parameters, see Overview of Pricing Parameters, page 5-43.

**Note:** A request type cannot be created for more than one PTE. A request type cannot be deleted if a mapping rule is defined for it.

### Functional areas for Source System region

Each PTE must have at least one enabled functional area and related category set. You can use the category set and its related hierarchy of categories to define price list lines or modifiers. For example, the functional area of Order Management has a category set of inventory items, so you could either define a modifier at the category set level or for a category within the category set. Other examples of category sets include *Inventory* for Oracle Inventory, and the category set *Purchasing* for Oracle Purchasing (these products must be installed for their category sets to be available). The following example shows the functional areas and category combination for the Advanced Pricing source system in the Order Fulfillment PTE:

#### ***Advanced Pricing source system***

<b>Functional Area</b>	<b>Category Set</b>
Product Reporting	Product
Purchasing	Purchasing
Order Management	Inv. (Inventory) Items

- Functional Area: Advanced Pricing provides seeded functional areas for each source system in a PTE. You can also add new functional areas. Each seeded functional area is assigned a seeded category set that cannot be updated.
- Category Set: A default category set is provided for each functional area. For example, if the functional area is Order Management, then the default category set is Inventory Items. This is a read-only value. The category set for the associated functional area can be updated from the Default Category Set window.
- Enabled: You can select or deselect the Enabled check box to activate or inactivate a functional area. If the functional area is disabled, the pricing engine will not evaluate pricing data for the functional area being disabled (unless this functional area is attached to more than one source system). If this functional

area is attached to more than one source system for a given PTE, the pricing engine will still evaluate the pricing data.

## Related Topics

Viewing Parameter Definitions, page 5-44

Viewing Parameter Values, page 5-45

## Using Custom Sourced Attributes

For attributes with CUSTOM SOURCED defined as the attribute mapping method, the user must provide the code package procedure for the attribute. Attribute mapping is not required because the attribute will be sourced by Get Custom attributes. The user code is written in the package procedure QP\_CUSTOM\_SOURCE.

Get\_Custom\_Attribute\_Values. The attribute manager API program (Build\_Contexts) calls this procedure to pick up custom-sourced attributes if the profile option QP\_CUSTOM\_SOURCED is set to Yes. The input parameters to QP\_CUSTOM\_SOURCE are Request Type code and Pricing Type. Typical values of Request Type codes that can be passed are ONT, ASO, OKC, IC, FTE, and MSD. By using the Request\_Type\_Code, you can control how the attributes are sourced based on the PTE of the calling application.

Confirm that the profile option QP: Custom Sourced is set to Yes. The Build Contexts program builds the contexts from the dynamic package that is generated by the Build Attribute Mapping Rules program and from the custom package that is created by the customers. Attributes can also be passed to the pricing engine directly without an attribute mapping rule. In such cases, the Attribute Manager API calls a custom API, QP\_CUSTOM\_SOURCE, where the user has manually defined the attributes being passed and coded the sourcing of their values.

The pricing type can be H (Header) or L (Line) which defines the level of the attribute mapping rule. These attributes and their values are passed to the pricing engine in the same manner as the attributes that are sourced through attribute mapping rules.

### Example using QP\_CUSTOM\_SOURCE

The following example explains how you could code the body of QP\_CUSTOM\_SOURCE for a particular case. In this case, two segment mapping columns, QUALIFIER\_ATTRIBUTE31 and PRICING\_ATTRIBUTE31, which belong to contexts CUST\_SOURCE\_QUAL\_CON and CUST\_SOURCE\_PRIC\_CON respectively and are linked to PTE Order Fulfillment, will have Custom Sourced values as 10 for ORDER as well as LINE attribute mapping levels. You must ensure that the attribute mapping method for both these PTE-Attribute links is CUSTOM SOURCED in the Link Attributes window.

```

CREATE or REPLACE PACKAGE body QP_CUSTOM_SOURCE AS
/*Customizable Public Procedure*/
PROCEDURE Get_Custom_Attribute_Values
( p_req_type_code IN VARCHAR2
, p_pricing_type_code IN VARCHAR2
, x_qual_ctxts_result_tbl OUT QP_ATTR_MAPPING_PUB.
CONTEXTS_RESULT_TBL_TYPE
, x_price_ctxts_result_tbl OUT QP_ATTR_MAPPING_PUB.
CONTEXTS_RESULT_TBL_TYPE
) is
Begin
If p_req_type_code = 'ONT' and p_pricing_type_code in ('L','H') then
x_qual_ctxts_result_tbl(1).context_name := 'CUST_SOURCE_QUAL_CON';

x_qual_ctxts_result_tbl(1).attribute_name :=
'QUALIFIER_ATTRIBUTE31';
x_qual_ctxts_result_tbl(1).attribute_value := '10';
x_price_ctxts_result_tbl(1).context_name :=
'CUST_SOURCE_PRIC_CON';
x_price_ctxts_result_tbl(1).attribute_name :=
'PRICING_ATTRIBUTE31';
x_price_ctxts_result_tbl(1).attribute_value := '10';

end if;
end Get_Custom_Attribute_Values;
END QP_CUSTOM_SOURCE;
/

```

## Restoring Seeded Data Using the Restore Defaults Button

A seeded context is a system value that is not created by the user. A Seeded check box next to a context indicates whether the context is seeded or not (when the check box is selected, the context is seeded). When a Seeded context is selected, the Name and Description fields display the seeded values in the Context Setup window.

However, if you change the seeded values in the Name and Description fields, the new values are displayed, but the original seeded values are preserved in the hidden Seeded Name and Seeded Description fields.

To restore the original seeded values for a context, click Restore Defaults. The Name and Description fields will display the original seeded values rather than the changed values.

**Note:** The Restore Defaults button replaces values in all the fields with seeded values. The Restore Defaults button is grayed out for non-seeded attributes and for seeded attributes for which seeded values have not been changed by the user so far.

You can restore the default seeded settings in the following windows:

- Context Setup
- Link Attributes

## Context Setup

If a seeded context has been changed, you can restore the original seeded value by clicking the Restore Default button. The seeded context values appear in the Name and Description fields; however, if you change the seeded values, the new values appear but the original seeded values are preserved in hidden seeded name and seeded description fields respectively.

For *attributes*, the Restore Default button restores any changed values to the original seeded values of an attribute. So if a seeded attribute has been changed, the values can be restored to their original condition. Precedence, name, valueset, and datatype window fields display the seeded values in place of user entered values.

For seeded attributes, the Precedence, Name, Valueset, Datatype, and Required fields display the seeded values. However, if you change the seeded values, these fields will show the new user-entered values. The seeded values will be preserved in separate fields that are named as seeded precedence, seeded name, seeded valueset, seeded datatype, and seeded precedence. These seeded fields will always be hidden.

**Note:** Please note that this button will replace values in all the fields with seeded values.

## Link Attributes

For seeded attribute, the Attribute Mapping Method column will have its seeded value. However, if the user changes the attribute mapping method, this field will show the user entered value and the seeded value will be preserved in a separate field named as seeded Attribute Mapping Method. These seeded Attribute Mapping Method fields will always be hidden.

The Restore Default button restores the original seeded attribute mapping method of an attribute. If you overwrite a seeded attribute but later want to restore the seeded values, you can do so by clicking this button.

This button will be grayed out for non-seeded attributes. This button will continue to remain gray for seeded attributes, if the user-entered attribute mapping method remains the same as the seeded attribute mapping method.

**Note:** The attribute Total Item Quantity is seeded only for Oracle Transportation Execution (FTE). However, you can manually link an attribute to the Order Fulfillment PTE, and create a corresponding sourcing rule for attributes such as Total Item Quantity.

## Attribute Mapping

For all seeded Attribute Mapping rules, the Restore Defaults button restores the seeded

Source type and the seeded Value string as User source type and User value string respectively. If the seeded source type and the seeded value string are the same as the user source type and user value string, the Restore Defaults button will be grayed out.

## Troubleshooting While Setting Up Attributes

### Scenario 1

This warning message will appear on setup windows when the user tries to use in the setup, any newly added attribute having Attribute Mapping Method = ATTRIBUTE MAPPING for which the Attribute Mapping Enabled flag is not Y.

**Message:** This Attribute (with Context: &CONTEXT and Attribute: &ATTRIBUTE) will not be mapped since it is not Attribute Mapping Enabled.

### Scenario 2

This warning message will appear on setup windows when the user tries to use in the setup, any newly added attribute having Attribute Mapping Method =ATTRIBUTE MAPPING for which Attribute Mapping Status flag is not Y.

**Message:** Warning: This Attribute (with Context: &CONTEXT and Attribute: &ATTRIBUTE) has not been mapped yet. Please Build Attribute Mapping Rules.

### Scenario 3

The Attribute Mapping level for this attribute was defined as BOTH. In such cases, the user is expected to enter an Attribute Mapping rule, one each for the header and line level. If the user enters only one level, pricing inconsistencies will occur.

**Message:** This attribute must have an attribute mapping rule both at Header as well as Line levels.

### Scenario 4

For a given PTE, the user must enter Attribute Mapping rules for all the request types that belong to that PTE. In case the user does not enter for all of them, different Request types for the same PTE may not be able to fetch the same price.

**Message:** You must map this attribute for all the Request types.

## Troubleshooting in Pricing Setup windows Related to Attribute Management

### Scenario 1

You try to attach a combination of qualifier with the attributes that have Attribute Level of LINE and ORDER within the same qualifier grouping number of a qualifier rule on the Qualifier Rules window.

**Message:** Qualifier Attribute with an attribute level of LINE cannot be present in combination with a Qualifier Attribute with attribute level of ORDER, within a Qualifier Grouping No of a Qualifier Rule.

### Scenario 2

This error message will appear on the Qualifiers tab of the Modifiers window when a user tries to attach a combination of qualifier attributes that have attribute level of LINE

and ORDER within the same qualifier grouping number.

**Message:** There is a mix of LINE and ORDER qualifier attribute level for list line ID &LIST\_LINE\_ID and qualifier grouping number &QUALIFIER\_GRP\_NO. Ensure that all the qualifier attributes should be either of level LINE/BOTH or ORDER/BOTH for a given list line ID and qualifier grouping number.

## Troubleshooting During Pricing Setup

While you are defining Qualifier or Pricing attributes in the Pricing Setup windows, only those attributes with the LOV Enabled box selected appear in the LOVs on these windows. If basic pricing is installed, only those attributes that have been set up with the AVAILABLE\_IN\_BASIC box selected appear in the LOVs on the various Pricing Components setup windows. However, all attributes that are set up are available to Advanced Pricing users, subject to meeting the other attribute level conditions.

You need to set the profile QP: Pricing Transaction Entity to the correct value before creating or querying any setup data in the pricing application. You should not change this profile often because you will see the other context-attributes combinations for a different PTE when querying on the pricing setup windows. If this happens, you will see the internal ID code.

## Troubleshooting During Integration or Runtime

The following section provides answers to common troubleshooting questions that you may have during integration or runtime.

### Viewing Attributes

#### Question 1

Why do I not see the pricing contexts and attributes that I defined in the pricing setup windows?

**Answer:** The following conditions must be met:

- The contexts and attributes that you defined using the Attributes Manager windows are assigned to the correct PTE.
- The contexts and attributes are enabled.
- System Profile Option QP: Pricing Transaction Entity Code points to the correct Pricing Transaction Entity Code.
- The attribute levels are correct. Only attributes with certain levels may be visible in certain windows.

#### Question 2

What happens if I do not set the Profile QP: Pricing Transaction Entity correctly?

**Answer:** This profile indicates the current PTE in use. Only those contexts and

attributes that are assigned to the current PTE will be available in the LOVs on the setup windows.

Querying up setup data displays the description to be shown only for those contexts and attributes that are assigned to the current PTE.

## Entering Orders

### Question 1

When you enter an order line in the Sales Order Pad, the following error appears:

```
FND_AS_UNEXPECTED_ERROR (PKG_NAME=oe_order_adj_pvt)
(PROCEDURE_NAME=oe_line_adj.calculate_adjustments)
(ERROR_TEXT=calculate_adjustments#130 ORA-06508: PL/SQL: could not find
program unit being called).
```

**Answer:** Check in dba\_errors for this package in or to determine which attribute mapping API is causing the error. If this is a custom API then correct the API. If this is the seeded API then determine whether a correction patch is available.

### Question 2

The context of an attribute that was successfully mapped did not show up in the Pricing Context list of values on the Order Management Sales Pad.

**Answer:** One of the following solutions may resolve the issue:

- The only contexts that will show up in the OM Sales Pad window are the ones that have at least one attribute that is USER ENTERED. If the context has all its attributes as mapping method ATTRIBUTE MAPPING, this context will not show up Pricing Context List of values.
- You must create all new attributes using the new Attribute Manager Context and Attributes window. Using this method, all attributes of type Pricing Attribute will get created in the OM Flexfield and the flexfield will get registered (if required). Its definitions will freeze and then get compiled automatically. Remember, the converse is not true. An attribute that you created using the Flexfield windows will not get created in the Attribute Manager tables. Columns that are updated in Flexfield windows are also not supported in pricing.
- If the attribute that you just created was of the type Pricing Attribute, the system creates the same attribute in the OM flexfield tables and then compiles the flexfield. Check the Concurrent Manager requests to determine whether the request was completed as Normal.
- Attributes having Column Mapped values PRICING\_ATTRIBUTE31..100 must get registered. Although this is done automatically by the system, you should confirm that this has occurred.

## Additional Attribute Management Considerations

The following table outlines some problems and solutions that are related to attribute management:

Probable Cause	How to Debug
QP_Attr_Mapping_PUB.Build_Contexts package is invalid due to incorrect mapping of data attributes mapping.	Check in dba_errors for this package in or to determine which attribute mapping API is causing the error. If this is a custom API, then correct the API. If this is the seeded API then determine whether a correction patch is available.
Concurrent Program Build Attribute Mapping Rules failed with error.	Run the following statement and examine the output: select text from dba_errors where name =QP_BUILD_SOURCING_PVT. Verify that custom sourcing causes this error.
Getting error while running Build Attribute Mapping Rules concurrent program ORA-06502: PL/SQL: numeric or value error: character string buffer too small ORA-06512: at APPS.QP_ATTR_MAPPING_PUB, line 1445 ORA-20000: ORA-04021: timeout occurred while waiting.	This occurs when someone makes a pricing call while the concurrent program is running. Do not run the Build Attribute Mapping Rules concurrent program when active users are calling pricing engine.
While entering the order line in the Sales Order Pad, you receive the following error: END_AS_UNEXPECTED_ERROR (PKG_NAME=oe_order_adj_pvt) (PROCEDURE_NAME=oe_line_adj. calculate_adjustments) (ERROR_TEXT=calculate_adjustments#130 ORA-06508: PL/SQL: could not find program unit being called).	Run the following statement and examine the output: select text from dba_errors where name = QP_BUILD_SOURCING_PVT; Determine whether any custom sourcing causes the errors. If the seeded sourcing rule causes this error, determine whether a patch is available to correct the seeded rule. If the error is "Encountered the symbol _ when expecting..." then determine the relevant patch to be applied.

## Upgrading Considerations

When upgrading from releases before 11.5.7/Patch G, the upgrade program will do the following:

1. Upgrade Contexts and Attributes
2. Upgrade source systems and request types
3. Create PTEs and attribute links
4. Upgrade attribute mapping rules

5. Assign PTEs to existing modifiers

Pre H Release	Post H Release
Request Types and Source Systems common for all applications	Request Types and Source Systems grouped by PTE
Context and Attributes created and maintained in Flexfields	Context and Attributes created and maintained in pricing (QP) tables
All attributes available to all applications for pricing setup	Attributes available to applications based on PTE. Attributes can now be enabled for LOV and limits
Attributes have one sourcing rule used by all request types	Attributes may have one or more attribute mapping rules for different request types
Only one ordering structure, for example, Order Fulfillment.	At least four seeded PTEs, with provision for expanding.

**Table Upgrade**

Contexts	fnd_descr_flex_contexts	qp_prc_contexts_b
Context	fnd_descr_flex_contexts_tl	qp_prc_contexts_tl
Attributes	fnd_descr_flex_column_usages	qp_segments_b
Attributes	fnd_descr_flex_col_usage_tl	qp_segments_tl
Source System and Request Types	qp_price_req_sources	qp_pte_source_systems
Source System and Request Types	qp_price_req_sources	qp_pte_request_types_b/tl
Attribute Linking	oe_def_attr_condns	qp_pte_segments
Attribute Linking	ak_object_attributes	qp_pte_segments
Attribute Linking	oe_def_condn_elems	qp_pte_segments

Contexts	fnd_descr_flex_contexts	qp_prc_contexts_b
Attribute Mapping Rules	oOe_def_attr_def_rules	qp_attribute_sourcing

## Upgrading Context and Attributes

This section describes how to upgrade contexts and attributes from the current flexfield structure to a set of new normalized pricing (QP) tables.

### Upgrading Contexts

If your contexts are currently stored in qualifier contexts and pricing contexts flexfields, the upgrade program will copy all the contexts from these flexfields to the new pricing (QP) tables. All the Qualifier Contexts flexfield qualifiers will be copied as qualifier contexts.

All the Pricing Attribute flexfield qualifiers except ITEM will be copied as pricing attribute contexts. ITEM Qualifier will be copied as Product context. All contexts that are created in the flexfields by user 1 will be treated as seeded data. All other information, such as name and description in various languages, and enabled flag are available in the flexfields.

### Upgrading Attributes

For every context that is created in the new system from the flexfields, attributes are available in the flexfield usage tables. These attributes will be copied under the corresponding context in the new system. Some attributes in the flexfield usage tables do not have value sets, because some attributes use key flexfields for their valid values. All other information such as names, mapping columns, and precedence are available in flexfield usage tables.

### Upgrading Source System and Request Types

Currently, all the source systems and request types are mapped as many-to-many relationships. Seeded request types are provided to include additional relationships between request types.

Attribute management uses the pricing transaction entity feature which consists of an ordering structure comprised of associated request types and source systems. When you are upgrading, a new set of PTEs are created in the target system. In addition to the seeded PTEs, you can create more PTEs depending on the customer's own request types and source systems.

## Mapping of Seeded Request Types and Source Systems

A pre-determined process is available to associate request types and source systems with these PTEs. These request types and their associations will act as seeded data. Please refer to the previous sections on Seeded Pricing Transaction Entity, Source

Systems, and Request Types.

Whenever a request type is created in the target system by the upgrade program, the default global structure that is associated with them will be as shown in the following table:

<b>Request Types</b>	<b>Global Structure Name: Header Level</b>	<b>Global Structure Name: Line Level</b>
ASO	ASO_PRICING_INT.G_HEADER_REC	ASO_PRICING_INT. G_LINE_REC
OKC	OKC_PRICE_PUB.G_CONTRACT_INFO	OKC_PRICE_PUB. G_CONTRACT_INFO
IC	INV_IC_ORDER_PUB.G_HDR	INV_IC_ORDER_PUB.G_LINE
ONT	OE_ORDER_PUB.G_HDR	OE_ORDER_PUB.G_LINE
FTE	None	None
MSD	None	None

## Creating PTE and Attribute Links

Currently, all the attributes that are available in the existing flexfield may already be attribute mapped. In the new data model, each attribute must be linked to one or more PTEs, before it can have attribute mapping rules:

### **1) Attributes that are Attribute Mapping enabled or already mapped**

Such attributes will be associated with a source system in the current system as per defaulting rules. At this point, the upgrade program has already created all the PTE-Request Types-Source Systems associations (with data) in the target system. The upgrade program will determine the PTE(s) that this source system is associated with and will create as many PTE(s) and attribute links in the new PTE-Attribute mapping table.

For example, suppose Segment-1 is associated with QP Source System, as per the defaulting condition templates. The upgrade program will create three links for this attribute, represented as three rows in the new PTE-Attribute mapping table as shown in the following table:

Pricing Transaction Entity	Attribute
1 Unassigned-2	Segment-1
2 Order Fulfillment	Segment-1
3 Intercompany Transaction	Segment-1

The three rows get created because Source System QP is attached to PTEs Unseeded-2, Order Fulfillment and Intercompany Transaction.

## 2) Attributes that are not Attribute mapped

Some attributes may not have an attribute mapping rule. These are attributes for which there are no transactions in price lists, modifiers, and formulas. Such attributes will be linked to all the PTEs that you created so far (including the seeded ones) in the PTE Attribute Mapping Table.

## Upgrade Attribute Mapping Rules

When you are upgrading, all the attributes that have default Attribute Mapping rules defined in the current system will have the same Attribute Mapping rules redefined in the new model. Every attribute that is present in the PTE Attribute Mapping table may have an Order level mapping, a Line level mapping, or both, depending on how the attributes were defined in the current system, for example, Header only, Line only or Header and Line.

**Note:** If the Attribute Manager Loader or the Attribute Manager Upgrade program encounters an existing attribute in one of the following situations, it will bypass the upload or upgrade of that attribute:

- At customer location that uses the same segment mapping column that is used by a seeded attribute (in case of Attribute manager Loader program) or
- A user attribute was created in the flexfield (in case of Attribute Manager Upgrade program).

However, unlike the current system, every attribute that will have an attribute mapping rule will ideally have as many sets of attribute mapping rules as the number of Request types that are associated with the PTE for that attribute in the PTE attribute mapping table. At this point, the upgrade program may not create all the attribute mapping rules for all the request types. Depending on the attribute, given the application that created it, the upgrade program will create attribute mapping rules for the request types as

shown in the following table:

Source Systems	Request Types
QP	ONT
OKC	OKC
ASO	ASO
AMS	ONT, ASO

Only the seeded PTE-Attribute combinations from the PTE attribute mapping table that were created for use in basic pricing can have default attribute mapping rules. Also, all the attributes that have defaulting rules and belong to Source System FTE, will not have attribute mapping rules.

## Assigning PTE to Existing Modifiers

Attribute mapping identifies the PTE from which the related modifier lists, price lists, and formulas were created (in addition to the source system, which was used so far). A new column PTE\_CODE has been added to the QP\_LIST\_HEADERS\_B table, which stores the PTE. The upgrade program updates the PTE\_CODE for every modifier list, price list, or formula based on the following logic:

After the upgrade program creates all the PTE-Request Types-Source Systems associations (with data) in the target system, the upgrade program determines the PTE (s), for every modifier from its source system. If the source system is associated with one PTE only, the PTE\_CODE will be updated with that PTE. If the source system is associated with more than one PTE, Order Fulfillment will be chosen over other PTEs.

## Sample Code

```
-- Please note, the cursor cs_item_cost may need to be modified
-- before use in your environment.

CREATE OR REPLACE
PACKAGE MY_CUSTOM_SOURCING AUTHID CURRENT_USER AS
-- please see package body for version, history and notes.
-- Package globals...G_Organization_id NUMBER := FND_PROFILE.VALUE
('QP_ORGANIZATION_ID');
-- Cached in and out values for each sourcing routine.
TYPE Item_Info_Rec_Type IS RECORD
( inventory_item_id VARCHAR2(240)
, item_cost NUMBER
);
G_Item_Info Item_Info_Rec_Type;
FUNCTION Get_Item_Cost (p_item_id IN NUMBER) RETURN VARCHAR2;
PRAGMA RESTRICT_REFERENCES (Get_Item_Cost,WNDS);
END MY_CUSTOM_SOURCING;
/
-----CREATE OR REPLACE
PACKAGE BODY MY_CUSTOM_SOURCING AS
/* Package Body version 1.01 - 30th January 2001 */
-- This is the package body for Simon's example of
-- Advanced pricing lli's custom sourcing routines used for
-- retrieving additional information from the apps tables into pricing
-- data structures.
-- standard functionality is not feasible.
-- To marginally improve performance on successive pricing calls,
-- we cache the most recent details from the sourcing functions
-----
-- so that the cursors are not run every time if the requests
-- are the same. These cached values are stored in package-wide
-- variables.
-----
FUNCTION Get_Item_Cost (p_item_id IN NUMBER) RETURN VARCHAR2 IS
-- Function to retrieve an item cost from cst_item_costs
-- note, this returns null is a cost is not found
-- so that the calling app can handle this.
-- Note, Use your own cost type id from cst_cost_types.
-- I've used 1 just for testing.
CURSOR cs_item_cost(cp_item_id IN NUMBER) IS
SELECT cic.item_cost
FROM cst_item_costs cic
AND cic.cost_type_id = 1
AND cic.organization_id = G_organization_id;
v_cost cst_item_costs.item_cost%TYPE := NULL;
BEGIN
IF p_item_id = G_Item_Info.inventory_item_id THEN
-- if the requested item is already cached then do nothing yet.
NULL;
ELSE
```

---

## Get Custom Price

This chapter covers the following topics:

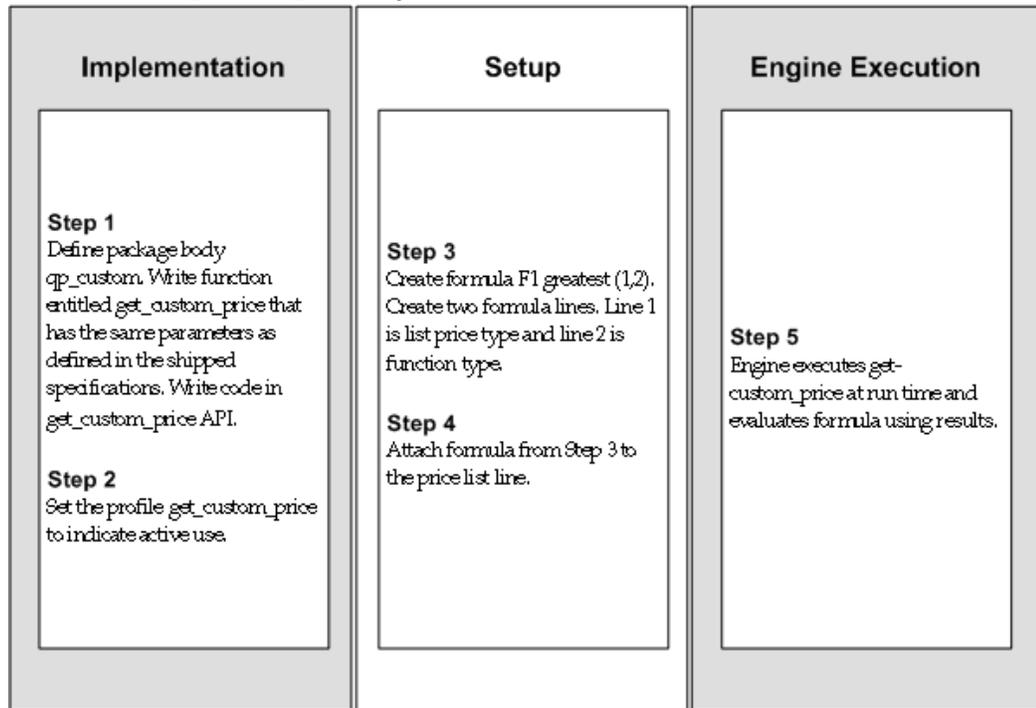
- Overview of Get\_Custom\_Price Implementation
- Implementing Get\_Custom\_Price
- Get\_Custom\_Price\_Customized

### Overview of Get\_Custom\_Price Implementation

The actual value of a modifier or price is calculated using the Get\_Custom\_Price function. The following is an example of the Get\_Custom\_Price function is: my price is 5% less than the competitor's price (your price is 95% of competitor's price), where the competitor's price is drawn from another custom database or a PL/SQL call.

An overview of the tasks that are involved in the implementation of Get\_Custom\_Price is depicted in the following image.

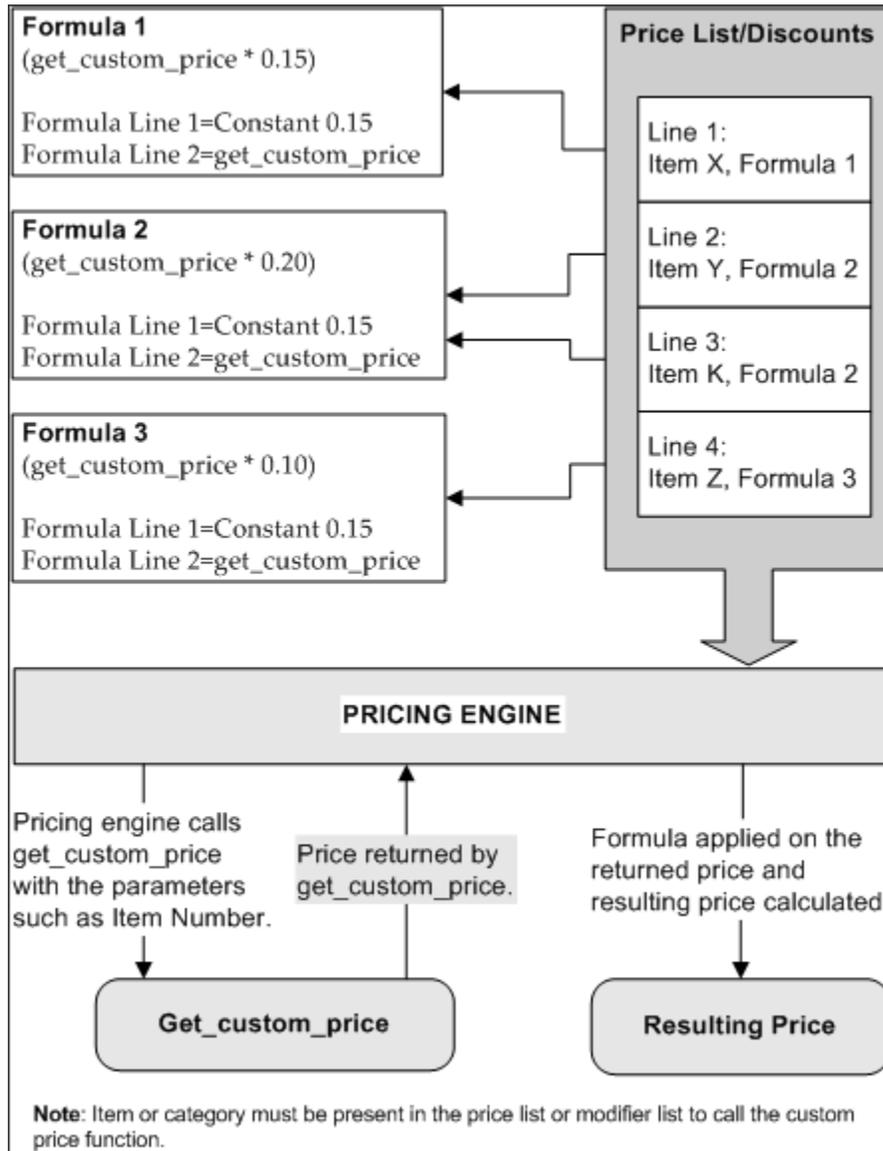
### Overview of Get\_Custom\_Price Implementation



The following diagram illustrates the GET\_CUSTOM\_PRICE function. It shows several formulas that reference price lists and discounts. When the pricing engine evaluates a formula, it passes the formula, price list, and discount information to GET\_CUSTOM\_PRICE. GET\_CUSTOM\_PRICE returns a value, and the pricing engine uses that value to calculate the formula and returns the formula value to the calling application.

The text following the diagram explains the GET\_CUSTOM\_PRICE function in more detail.

**Get Custom Price**



To use the Function formula type, modify the function GET\_CUSTOM\_PRICE. Write custom code in the function body that uses the standard input parameters. You can only write a function body that returns a number.

You can use the GET\_CUSTOM\_PRICE function in more than one formula. The formula ID is an input parameter that you can use to differentiate code logic among formulas.

To use the return value in a formula, do the following:

- Set up a formula having a formula line of type Function.

- Set the system profile option QP: Get Custom Price Customized to *Yes* at the Site level. If the pricing engine evaluates custom code within the GET\_CUSTOM\_PRICE function and the profile option is No, the formula calculation fails, and the calling application processes the failure.

GET\_CUSTOM\_PRICE is in the package QP\_CUSTOM and its specification is QPXCUSTS.pls. QP\_FORMULA\_PRICE\_CALC\_PVT calls QP\_CUSTOM.  
Get\_Custom\_Price().

The pricing engine passes the parameters to the function. You may not change the GET\_CUSTOM\_PRICE parameters that are:

- p\_price\_formula\_id: Primary key of the formula that uses the Get\_Custom\_Price function.
- p\_list\_price: List price on the price list line to which the formula using Get\_Custom\_Price is attached. May be null.
- p\_price\_effective\_date: Pricing Effective Date, the date for which the pricing engine is evaluating the formula.
- p\_req\_line\_attrs\_tbl: PL/SQL table of records containing context, attribute, and attribute value records for product and pricing attributes and a column indicating the type (Product Attribute or Pricing Attribute). The engine passes only the pricing and product Attributes of the price list line to which the formula is attached.

**Note:** The step numbers for a formula are available in the GET\_CUSTOM\_PRICE API.

## Implementing Get\_Custom\_Price

1. Write the extension code in qp\_custom.get\_custom\_price. If you use get\_custom\_price, then you must create the package body for qp\_custom and create a function get\_custom\_price. The pricing engine calls the application programming interface (API) qp\_custom.get\_custom\_price with the following set of parameters:
  - P\_price\_formula\_id: IN NUMBER  
The formula ID identifies the formula from which the API is called.
  - 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)

P\_req\_line\_attrs\_tbl provides information such as such as product, pricing attributes, qualifiers and special attributes and contains the following fields:

- Line\_index: Line index of the price request line
- Attribute\_type: Qualifier, product, pricing
- Context: Context name (for example, Item)
- Attribute: Attribute name (for example, Pricing\_attribute1)
- Value\_from: Attribute value (for example, 149)

### Step Numbers

Special attributes provide other useful information, such as step numbers for the formula line. A step number is important if more than one step exists in a formula using Get\_Custom\_Price. The step numbers for a formula are available in the GET\_CUSTOM\_PRICE API. The step number information is passed as a value in p\_req\_line\_attrs\_tbl as a last record for attribute\_type=QP\_GLOBALS.G\_SPECIAL\_ATTRIBUTE\_TYPE, context=QP\_GLOBALS.G\_SPECIAL\_CONTEXT, attribute=QP\_GLOBALS.G\_SPECIAL\_ATTRIBUTE1.

A DML (Insert/Update/Delete) operation is not supported in the get\_custom\_price routine. The pricing engine does not guarantee upgrade possibility if any engine variables are referred to the get\_custom\_price function.

**Note:** The step number information is passed as a value in p\_req\_line\_attrs\_tbl as a last record for attribute\_type=QP\_GLOBALS.G\_SPECIAL\_ATTRIBUTE\_TYPE, context=QP\_GLOBALS.G\_SPECIAL\_CONTEXT, attribute=QP\_GLOBALS.G\_SPECIAL\_ATTRIBUTE1.

### GET\_CUSTOM\_PRICE example

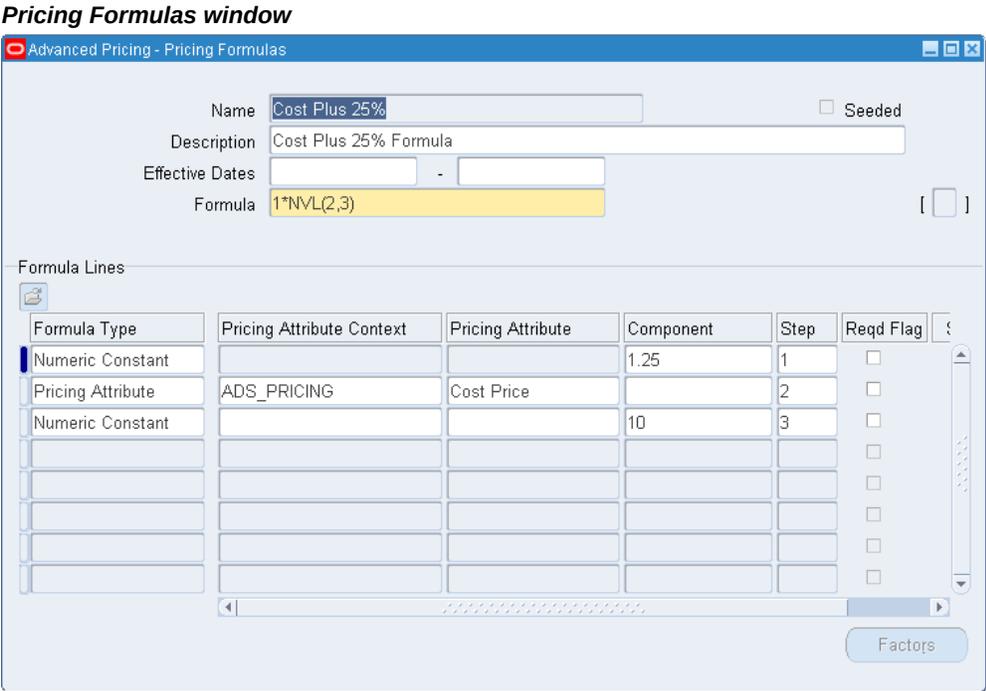
```
CREATE or REPLACE PACKAGE BODY QP_CUSTOM AS
FUNCTION Get_Custom_Price (p_price_formula_id IN NUMBER,
    p_list_price IN NUMBER,
    p_price_effective_date IN DATE,
    p_req_line_attrs_tbl IN QP_FORMULA_PRICE_CALC_PVT.
REQ_LINE_ATTRS_TBL)
RETURN NUMBER
is
BEGIN
if p_price_formula_id = 7538 then
    return 14.01;
end if;
end get_custom_price;
END QP_CUSTOM;
```

2. Set the value of profile QP: Get Custom Price Customized.

The engine calls the `get_custom_price` function only if profile QP: Get Custom Price Customized is set to Y. If you set up a formula but not a profile, a runtime error appears. Set the profile at site level.

3. Create a formula to use the `get_custom_price` function.

The following image depicts the Pricing Formulas window:



Remember to note the `formula_id` by using Help, Examine. Use this formula ID in the `get_custom_price` function to identify the formula.

4. Attach this formula to the price list line.

The following image shows the Price Lists window in Oracle Advanced Pricing.

### Price Lists window

Product Value	Product Description	UOM	Primary UOM	Dynamic Formula
AS54888	Sentinel Standard Desktop TPD	Ea	<input type="checkbox"/>	Cost Plus 25%
			<input type="checkbox"/>	
			<input type="checkbox"/>	
			<input type="checkbox"/>	

## Get\_Custom\_Price Customized

The following sample code shows how the body of the `Get_Custom_Price` function is coded in the file `QPXCUSTB.pls`. You 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 cannot be customized. You can use the input parameters that are passed by the pricing engine in their custom code. The function returns a number. You 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 with the 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. The value that is returned by the `QP_CUSTOM.Get_Custom_Price` function is 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. 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 previously must also be set to Yes to use this `Get_Custom_Price` functionality).

Assume that `Get_Custom_Price` is customized as `Package Body Qp_custom`. The `Get_Custom_Price` function name and parameters cannot be customized but the body can be customized. The parameters are:

- `p_price_formula_id`: Primary key of formula that uses `Get_Custom_Price` function.
- `p_list_price`: List price of the price list line to which the formula using `Get_Custom_Price` is attached. This can have null value.
- `p_price_effective_date`: Date of formula evaluation by the pricing engine.
- `p_req_line_attrs_tbl`: A PL/SQL table of records containing context, attribute, attribute value records for product and pricing attributes and a column indicating type (product attribute or pricing attribute). The engine passes the pricing attributes and product attributes of 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;
l_step_number 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
    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_attr      rs_tbl(i).value;
END IF;
        IF p_req_line_attrs_tbl(i).attribute_type = PRICING
    AND
    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;
        For this combination of Special Context and Attribute, the
Attribute Value is the Step Number of the formula line
        IF p_req_line_attrs_tbl(j).attribute_type=QP_GLOBALS.
G_SPECIAL_ATTRIBUTE_TYPE
        and p_req_line_attrs_tbl(j).context=QP_GLOBALS.G_SPECIAL_CONTEXT
        and p_req_line_attrs_tbl(j).attribute=QP_GLOBALS.
G_SPECIAL_ATTRIBUTE1 THEN
            l_step_number:=p_req_line_attrs_tbl(j).value;
        END IF;
    END LOOP; For Loop
RETURN v_weight;
    EXCEPTION
        WHEN OTHERS THEN
            RETURN NULL;
END Get_Custom_Price;
END QP_CUSTOM;

```

If v\_weight has a value 1.2 then Get\_Custom\_Price returns a value of 1.2. The pricing engine evaluates the formula as 200\*1.2 = 240.



---

## Events and Phases

This chapter covers the following topics:

- Overview of Events and Phases
- What Are Pricing Events?
- What Are Pricing Phases?
- Assigning Pricing Phases

### Overview of Events and Phases

Pricing events and phases enable you to configure Oracle Advanced Pricing so that transaction pricing occurs when it is required by the application process flow. Pricing events and phases also enable you to define which pricing data is considered for application to a request at the pricing point in your transaction process flow. You can separate the pricing of your transaction, rather than pricing a whole transaction at once. Events and phases allow for the implementation of the following types of pricing business rules:

- Freight and special charges are calculated at time of shipping.
- Cross order volume discounts apply at end-of-day (once total order volumes have been derived) in a high volume batch environment.
- Coupons are awarded only after all items in the shopping cart are priced and the user proceeds to final checkout.

### What Are Pricing Events?

A pricing event is a point in the transaction life cycle when you want to price the transaction (or certain transaction lines), or when you want to apply price adjustments, benefits, or charges to the whole transaction or specific transaction lines.

**Note:** The calling application must pass the events. The pricing engine searches the set of data belonging to the phases for the events that are passed. Multiple events can be concatenated in a single pricing engine call.

The following table outlines the action for each of the seeded events in Oracle Advanced Pricing:

<b>Event Code</b>	<b>Meaning</b>	<b>Function Which Calls the Pricing Engine</b>
BATCH	Batch Processing	Order is imported.
BOOK	Book Order	Order is booked.
FTE_APPLY_MOD	FTE: Apply Modifiers to Price	This event is used only by the Oracle Transportation Execution application to price transactions.
FTE_PRICE_LINE	FTE: Price a Transportation Line	This event is used only by the Oracle Transportation Execution application to price transactions.
ICBATCH	INV: Batch Processing for Intercompany Transfer Pricing	This event is used only by the Oracle Inventory application to price transactions.
LINE	Enter Order Line	User exits the order line.
ORDER	Save Order Line	User saves order.
PO_BATCH	PO Batch Processing	Note: If you include the freight charge phases or some custom phases in the PO: BATCH event, the freight charges will not impact the unit selling price on the purchase order, but will not compute the freight charges on the purchase order.
PRICE	Fetch List Price	User enters item quantity and unit of measure.
PRICE_LOAD	Price a Logistics Load	This event is used to price a logistics load.
REPRICE_LINE	Reprice line	Order is shipped, prior to invoice.

Event Code	Meaning	Function Which Calls the Pricing Engine
RETROBILL	Retrobill	Event occurs during retrobilling.
SHIP	Enter Shipments	Order is ship confirmed.

**Note:** The Price a Logistics Load event is only used by the Oracle Transportation Execution application for pricing their transactions.

## What Are Pricing Phases?

A pricing phase controls which list types (prices and modifiers) are considered by the search engine and the sequence in which they are applied to a pricing request. The attributes of a pricing phase enable you to control which modifiers are placed in a phase. When you assign a modifier to a pricing phase, the Modifier Setup window matches the attributes of the modifier with the attributes of the available pricing phases to validate which pricing phase or phases a modifier can be placed in. A modifier can be assigned only to one phase.

The following table summarizes selected seeded pricing phases in Oracle Advanced Pricing (you can review *all* the event phases in the Event Phases window):

Phase Sequence	Name	Level	List Type	Incompatibility Resolve Code	Freeze Override
0	List Line Base Price	Line	Standard Price List	Precedence	NA <sup>1</sup>
5	Across All Phases	NA	NA	Precedence	NA
10	List Line Adjustment	Line	NA	Best Price	NA
30	All Lines Adjustment	NA	NA	Best Price	NA
40	Header Levels Adjustments	Order	NA	Precedence	NA

Phase Sequence	Name	Level	List Type	Incompatibility Resolve Code	Freeze Override
50	Line Charges	Line	Freight and Special charge List	Precedence	Yes
60	Line Charges - Manual	Line	Freight and Special charge List	Precedence	Yes
70	Charges: Header/All lines	NA	Freight and Special charge List	Precedence	Yes
80	Modifiers for BOOK Event	NA	NA	Precedence	NA

**Note:** <sup>1</sup>NA = Not applicable

## Assigning Pricing Phases

### Key Implementation Decision: What phases and events meet my business requirements?

Pricing events and phases help you exercise control over what pricing actions are taken and when they occur. Pricing events separate the order cycle into points that are associated with pricing actions, for example, the PRICE event (Fetch List Price) and LINE event (Enter Order Line). When setting up a modifier line such as a discount or promotions modifier line, you can select a pricing phase to control when the modifier adjustment is applied. When the pricing event occurs, the modifier adjustment is applied.

You can map a pricing event to several pricing phases, and each pricing phase can have one or more pricing events. This helps to define which pricing phases are to be processed and in which pricing event.

**Note:** Do not add any modifier phase to the pricing event. Do not assign Group of Lines and Other Item Discount modifier phase to a line level event; the pricing engine may not have all the necessary order lines. If you add any phase to line or order events, those phases must be added to the batch event.

When you assign the pricing phase as Across All Phases, the system applies only this exclusive modifier even though there exist other eligible modifiers. The pricing engine calculates the price based on the Across All Phases modifier line.

When assigning a pricing phase to a modifier line, remember to:

- Place line level discounts in List Line Adjustment phase.
- Place modifiers that span multiple lines in the All Lines phase.
- Place modifiers that apply to shipping in Ship Event phase.

**Warning:** You cannot set up a Promotional Goods (PRG) type of modifier in the User Freeze Override phase.

### Event Phases window

The screenshot shows the 'Event Phases' window with the following details:

- Sequence:** 1
- Name:** List Line Base Price
- Level:** Line
- List Type:** Standard Price List
- Seeded:**  Seeded
- Additional Buy Products Exist for PRG:**
- OID Exists:**
- Line Group Exists:**
- Seeded Freeze Override:**
- Incompatibility Resolve Code (Seeded):** Precedence
- User Freeze Override:**
- Incompatibility Resolve Code (User):** [ ]

Pricing Event	Start Date	End Date	Seeded Flag	Seeded Search Flag
Fetch List Price			<input checked="" type="checkbox"/>	No
Enter Order Line			<input checked="" type="checkbox"/>	No
Batch Processing			<input checked="" type="checkbox"/>	No
Price a Logistics Load			<input checked="" type="checkbox"/>	No
FTE:Price a Transportation Line			<input checked="" type="checkbox"/>	No

The following list describes the fields in the Event Phases window:

- **Sequence:** This field is required. Its numerical value is equivalent to the phase number. The pricing engine uses this number to determine the starting order of the phases if there are multiple phases in an event.
- **Name:** The phase Name field also appears in Modifier Setup windows when you assign a modifier to a phase. The field name should describe the timing and contents of a phase.

- **Level:** This field is optional. To restrict the modifiers in a particular phase to modifiers of a particular modifier level, enter the level in this field.
- **List Type:** This field is optional. To restrict the modifiers in a phase to modifiers on a modifier list, enter the list type in this field.
- **OID Exists:** This check box (view-only) is selected if any modifiers are of the type Other Item Discount for a particular phase.
- **Additional Buy Products Exist:** This check box (view-only) is selected if you defined modifiers for Promotional Goods or Other Item Discounts and you will define Additional Buy Products.
- **Line Group Exists:** This check box (view-only) is selected if modifiers are in the level Group of Lines for a particular phase.

### Seeded and User regions

These two regions differentiate the seeded values from the user-entered values. In the User section, you can update the Freeze Override check box and Incompatibility Resolve Code.

- **Freeze Override check box:** Provides additional control over freezing order lines on your transaction. When a calculate price call is sent from the calling application, the pricing engine evaluates the Freeze Override check box that is set up for the associated phase. If the Freeze Override check box is selected, then the pricing engine applies eligible modifiers in this phase to the request line. If it is not selected, the modifiers in this phase are not considered for application to the request line. This check box can also be updated on seeded pricing phases.

**Important:** Optionally, to prevent order level modifiers from being applied unintentionally, place Order level adjustments in a phase in which the Freeze Override check box is deselected. Otherwise, if you have an order line with Calculate Price Flag as *Freeze*, an order level discount will still be applied on Reprice if the Freeze Override check box for the phase is selected.

- **Incompatibility Resolve Code:** Enables a pricing engine to evaluate which modifier to select if multiple modifiers in the same exclusivity or incompatibility group are eligible to be applied to the same pricing request line:
  - **Best price:** The modifier that gives the lowest price (most advantageous) to a customer on the given pricing request line is applied. For Freight and Special charges, the modifier that gives the *highest* price to a customer on the given pricing request line is applied, not the lowest.
  - **Precedence:** The modifier with the lowest precedence (the lower the number the

higher the priority) on the given pricing request line is applied

The incompatibility resolve code value can be updated for seeded pricing phases.

### Event Phases region

- Pricing Event: Select one or more pricing events to be linked to the selected phase. Select from the available values (these values are defined as *pricing event* in the Lookups window).

**Note:** In Event Phases window, if you select the PRICE event (Fetch List Price) for a phase, and the phase is already attached to a promotional modifier type such as Coupon Issue, Item Upgrade, Other Item Discount, Promotional Goods, or Term Substitution, then the following error message appears:

The PRICE event cannot be attached to this phase because one or more of the following modifiers types has already been defined for this phase: Promotional Good, Other Item Discount, Item Upgrade, Term Substitution, or Coupon.

- Seeded Search Flag: Determines whether the pricing engine should search for price or modifier lists in addition to those that the calling application has requested. The search flag should be set to No when the calling application has already identified which price and modifier lists it will use to price the request. The pricing engine then uses the lists that are passed and does not attempt to find any other lists. For example, a number of discounts have been negotiated and recorded on a customer's service contract. When applying discounts to the service order, the application already knows which discount list should be used to price the order.

The search flag should be set to No only in the circumstances just described; in all other cases, the search flag should be set to Yes. If you set the flag to No and the calling application does not pass all the required pricing information, the prices and modifiers may not be applied to the transaction.

### Price List Search Based on Search Flag (Extended Search)

The pricing engine first tries to find the price from a price list after doing all the qualifications that are necessary to qualify for this passed price list. If the price is not found, the pricing engine attempts to get the price from a secondary price list (if present).

If the price is not found on the secondary list, the pricing engine searches for the price across all available price lists and tries to give the price from a price list with highest precedence. For the pricing engine to do this extended search across all price lists in the system, the Search Flag on the pricing phase under ALL the Pricing Events (which include the pricing phase sequence = 0) needs to be consistently set to Yes. You can do this by setting the User Search Flag to Yes.

### Example

**Step 1:** The pricing engine searches for the price from the Corporate price list. If the price is not found, the pricing engine tries to find a price from all the secondary price lists from the Corporate price list. If the price is still not found, the pricing engine completes Step 2.

**Step 2:** If your business requires that the pricing engine searches across all the price lists in the system, set the User Search Flag to Yes on all the pricing phases in all pricing events.

---

## Pricing Engine Request Viewer Window

This chapter covers the following topics:

- Overview of Pricing Engine Request Viewer
- Setting Up the User Profiles
- Regions in the Pricing Engine Request Viewer
- Pricing Engine Requests Region
- Pricing Engine Request Lines Region
- Pricing Engine Request Line Details Region
- Attributes Window
- Related Lines Window
- Formula Step Values Window
- Debug Log Window
- Analyzing Error Messages

### Overview of Pricing Engine Request Viewer

The Pricing Engine Request Viewer window captures and displays the inputs and outputs of pricing calls from calling applications such as Order Management, iStore, Order Capture, and Oracle Contracts Core. Information about the latest pricing request appears and is updated each time the pricing engine captures a new transaction. You can review this information to see which lines were selected or rejected by the pricing engine and to determine why certain prices and adjustments were or were not applied.

Using the Pricing Engine Request Viewer window, you can:

- View the controls such as Events, Rounding flag, Search and Calculate Flag, GSA flag passed by the calling application to the pricing engine.
- View the price request line passed in by the calling application.

- View which modifier lines the pricing engine applied or rejected for benefit adjustments along with the details of the modifier line.
- View the pricing, qualifiers, and product attributes passed to the pricing engine along with the other data generated by the pricing engine.
- View the relationship between order lines for promotional modifiers, price breaks, and service lines (OID, PRG, PBH, Service Items).
- View the formula step values generated by the pricing engine used in formula calculation.
- View and query fields in the Pricing Debug Log.

## Setting Up the User Profiles

Set the following profile options to control the behavior of the Pricing Engine Request viewer:

### **QP: Debug**

To start the Pricing Engine Request viewer, set the profile QP: Debug to "Request Viewer On." Alternately, select Request Viewer Off to turn the Request Viewer off. This profile option can be updated at the user level. The pricing engine request viewer is only active for the transactions of the user who set this profile option; other users' transactions are not affected. For additional information about the profile option settings, see QP: Debug, page 5-20.

### **QP: Set Request Name**

Set the value of the profile option QP: Set Request Name to append the value of the profile with Order ID to be stored in the Request Name field. The default value is Null.

**Note:** The Pricing Engine Request Viewer window is available from within Oracle Order Management. The navigation path is: Sales Order window > Tools > Pricing Engine Request Viewer. It is also available on the menu for Pricing Manager Responsibility.

## End-to-End Process for the Pricing Engine Request Viewer

The following series of activities occurs when a pricing call is made:

1. The calling application makes a call to the Build Attribute Mapping Rules package to generate the attributes defined by the attribute mapping function.
2. The calling application then calls the pricing engine with the attributes generated by attributes mapping.

3. The pricing engine processes the request and then searches for and evaluates eligible price list and modifier lines.
4. If the profile option QP: Debug is set to Request Viewer On, then the pricing engine inserts records into the permanent pricing debug tables and generates a unique request ID, storing the information from the calling application.
5. The pricing request information can then be viewed by querying the request in the Pricing Engine Request viewer from the OM Sales Order Pad or through the Pricing Manager responsibility menu.

## Regions in the Pricing Engine Request Viewer

The pricing engine request details appear in one or more of the following regions in the Pricing Engine Request window:

- Pricing Engine Requests region
- Pricing Engine Request Lines region
- Pricing Engine Request Line Details region.

### Pricing Engine Request Viewer window

The screenshot shows the 'Pricing Engine Request Viewer' window with the following data:

Order No.	Req Name	Req Id	Pricing Event	Created By	Creation Dat
66728	Order ID-160065	452709	LINE	OPERATIONS	15-AUG-200

Line No.	Status Code	Line Id	Status Text	Line Index	Rela
2	Updated	309067	PRODUCT_ONLY	1	1
66728	Updated	160065		2	2

Priced	Applied	Status Code	ier Level Code	Operand Calculation Name	Operand Value
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	New record created		Unit Price	100
<input checked="" type="checkbox"/>	<input type="checkbox"/>	New record created		New Price	100
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	New record created		New Price	90
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	New record created		Percent	10

Buttons at the bottom: View Debug Log, Attributes, Related Lines, Step Values

## Pricing Engine Requests Region

This region maps to the QP\_DEBUG\_REQ table and displays the following information about the pricing engine requests with associated controls sent by the calling application:

Field Name	Description
Order No	For Request Type ONT only, the order number associated with the request appears.  <b>Note:</b> Depending on the version of Oracle Pricing installed, the order and line numbers for orders created in previous releases may not appear in the Pricing Engine Request window. However, order and line numbers created in subsequent releases can be viewed.
Req Name	Order Id. of the calling application is appended with the value of the profile QP: Set Request Name and is stored in this field.

Field Name	Description
Req Id	Request Id is the sequence number, which the window provides to uniquely identify the Pricing Engine requests.
Pricing Event	A point in the transaction life cycle of your transaction at which you want to price it. The pricing event determines which phases the search engine processes according to the mapping in QP_EVENT_PHASES.
Created By	Name of the user who created this request.
Creation Date	Standard WHO Column. Contains the Date+ Time Stamp at which the record was created.
Calculate Flag	Refers to the Calculation_Flag in the control record, which is passed to the pricing engine. From all the requesting systems the value for this flag is always passed as Y. Eligible values are: <ul style="list-style-type: none"> <li>• N: Search engine: You do not want the engine to calculate the selling price.</li> <li>• C: Calculate engine: You are passing the adjustment records to the engine and you want the engine to recalculate the selling price, without retrieving new adjustments.</li> <li>• Y: Both calculate and search engine: Regular engine call. Retrieves new adjustments and calculates the selling price</li> </ul>
Simulation Flag	When selected, indicates that the call is for a pricing simulation. For simulations, the pricing engine should not make permanent record changes nor issue or redeem coupons.
Request Type	Identifies the transaction system making the pricing request.

Field Name	Description
Rounding Flag	<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.</p> <ul style="list-style-type: none"> <li>• If set to Y , the engine will apply the rounding factor defined in the price list.</li> <li>• If set to N, unrounded figures will be returned.</li> <li>• If set to Q, then refer to the value of the profile QP: Selling Price Rounding Options.</li> </ul>
GSA Check Flag	<p>Indicates whether the pricing calculation engine should test for GSA Violations:</p> <p>The evaluation is performed if a request is for a non-GSA customer, and GSA rules are violated if the selling price of an item is calculated to be less than the price of the item on any GSA price list. Allowable values are:</p> <ul style="list-style-type: none"> <li>• <i>Yes</i>: Price Calculation engine tests for GSA violations, and violating request lines are returned to the calling application with a status of GSA violation.</li> <li>• <i>No</i>: Do not test for GSA violations.</li> </ul> <p>The value of this field is controlled by profile in the requesting systems.</p>
GSA Dup Check Flag	<p>Indicates that the engine should perform GSA duplicate check. The calling application sets this flag to Yes to have the pricing engine test for GSA violations or No so that the pricing engine does not test for GSA violations.</p>
Temp Table Insert Flag	<p>This flag is set to Y, if the calling application directly inserts data into the pricing temporary tables. It is set to No if the pricing engine inserts into Temporary tables.</p>

Field Name	Description
Manual Discount Flag	<p data-bbox="678 310 1430 401">Indicates the setting of the profile QP: Return Manual Discounts which controls how the pricing engine should perform incompatibility processing for manual discounts. The values for this profile are:</p> <ul data-bbox="678 428 1430 688" style="list-style-type: none"> <li data-bbox="678 428 1430 527">• <i>Yes</i>: 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.</li> <li data-bbox="678 562 1430 688">• <i>No</i>: 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.</li> </ul>
Source Order Amount Flag	<p data-bbox="678 751 1463 842">If set to Y, indicates to the pricing engine to source the order amount. The calling application will provide the order amount. If set to N, the pricing engine should calculate the order amount.</p>
Public API Call Flag	<p data-bbox="678 890 1414 915">Indicates if the public API is being used to call the pricing engine.</p> <p data-bbox="678 942 1414 1035">If set to Yes, the public API, QP_PREQ_PUB is used to call the pricing engine. If set to N, the group API QP_PREQ_GRP is used to call the pricing engine.</p>

Field Name	Description
Manual Adjustments Call Flag	<p>Indicates if the pricing engine will search and return modifiers from phases that have manual adjustments or automatic adjustments. These values are set up in the profile option QP: Return Manual Discounts:</p> <ul style="list-style-type: none"> <li>• Yes: Apply the manual discounts. The unit price is not calculated by the pricing engine.</li> <li>• No: Do not apply the manual discounts. This means that the new unit price is calculated by the pricing engine.</li> </ul> <p>When the Manual Discount Flag check box is selected, it indicates that the profile option QP: Return Manual Discounts is set to Yes. The profile QP: Return Manual Discounts determines which modifier is evaluated by the pricing engine. The following table shows the setup of two modifiers with different precedence levels:</p> <p><b>Modifier Name: Manual _1</b></p> <ul style="list-style-type: none"> <li>• Incompatibility Level: Incompatibility 1</li> <li>• Precedence: 100</li> </ul> <p><b>Modifier Name: Manual _2</b></p> <ul style="list-style-type: none"> <li>• Incompatibility Level: Incompatibility 1</li> <li>• Precedence: 200</li> </ul> <p>When QP: Return Manual Discounts = Yes, then the LOV will show both Manual_1 and Manual 2.</p> <p>When QP: Return Manual Discounts = No, then the LOV will show Manual_1 since it has the highest precedence (determined by the lower precedence number).</p>
Check Cust View Flag	Used by the internal engine.
Currency Code	Currency in which the pricing engine priced. The value for this field should be same across all lines.

## Pricing Engine Request Lines Region

This region maps to QP\_DEBUG\_REQ\_LINES table and displays the following information about the lines being priced, including unit price and adjusted unit price.

You can also view information related to service and serviceable lines in this region.

---

<b>Field Name</b>	<b>Description</b>
Line No	Unique identifier of the request line in the calling application. For example, Order Number/Quote Number/Contract Number.
Status Code	Returned status  Allowable values are: <ul style="list-style-type: none"><li>• N: New record created [All N (No) records are returned back from the pricing engine. These are success records.]</li><li>• X: Unchanged (The default status when the line is passed to the pricing engine for processing)</li><li>• D: Deleted</li><li>• U: Updated</li><li>• IPL: Invalid price list (When passed in price list is not found, then an error is given)</li><li>• GSA: GSA violation</li><li>• FER: Error processing formula</li><li>• OER: Other error</li><li>• CALC: Error in calculation engine</li><li>• UOM: Failed to price using unit of measure</li><li>• INVALID_UOM: Invalid unit of measure</li><li>• DUPLICATE_PRICE_LIST: Duplicate price list</li><li>• INVALID_UOM_CONV: Unit of measure conversion not found</li><li>• INVALID_INCOMP: Could not resolve incompatibility</li><li>• INVALID_BEST_PRICE: Could not resolve best price.</li></ul>
Line Id	Unique identifier of the request line in the calling application. For example, Order Number/Quote Number/Contract Number.

---

---

Status Text	Returned message from Pricing Engine.
Line Index	PL/SQL unique identifier for request line.
Line Type Code	Type of line within the request. Eligible values are: <ul style="list-style-type: none"> <li>• ORDER</li> <li>• LINE</li> </ul>
Pricing Date	Date and Time for which the pricing engine calculates the prices.
Line Qty	Pricing request line quantity.
Line UOM Code	Pricing request line unit of measure.
Unit Price	Unit price of the item that is expressed in Priced UOM Code.
Adjusted Unit Price	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.
UOM Qty	This holds service duration expressed in Line UOM Code. Unit of measure quantity, for example, in service pricing, LINE_UOM_CODE is Months and UOM_QUANTITY is 2.This field is used for service item pricing.
Priced Qty	Quantity of pricing request line that pricing engine has priced.
Priced UOM Code	Unit of measure in which the pricing engine priced.
Currency Code	Currency in which the pricing engine priced. The value for this field should be same across all lines.

---

---

Price Flag	<p>Indicates the degree to which the price is frozen. Allowable values, based on lookup type CALCULATE_PRICE_FLAG are:</p> <ul style="list-style-type: none"> <li>• Y (Calculate Price): Apply all prices and modifiers to the request line.</li> <li>• 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.</li> <li>• P (Partial Price): Apply prices and modifiers in phases whose freeze override flag is Y.</li> </ul>
Percent Price	Price calculated as a percentage of the price of another item.
Parent Price	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 Qty	When the pricing engine determines the price of an item from the price of another item, the quantity of the related item.
Parent Uom Code:	When the pricing engine determines the price of an item from the price of another item, the unit of measure of the related item.
Processing Order	This field is used for service pricing. It indicates the order in which pricing will be done for order lines related to service pricing.
Processed Flag	<p>Indicates whether line has been processed by engine or not. Possible values:</p> <ul style="list-style-type: none"> <li>• <i>No</i>: Not processed</li> <li>• <i>Yes</i>: Processed</li> </ul> <p>Used by the internal engine.</p>
Processed Code	Internal code that indicates the stage of engine processing when an error occurred.
Active Date First Type	The date type of ACTIVE_DATE_FIRST based on lookup type EFFECTIVE_DATE_TYPES. Default value is date Ordered.

---

---

Start Date Active First	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_LINES. START_DATE_ACTIVE_FIRST and QP_LIST_LINES. END_DATE_ACTIVE_FIRST.
Active Date Second Type	The date type of ACTIVE_DATE_SECOND based on lookup type EFFECTIVE_DATE_TYPES. Default value is Requested Ship Date.
Start Date Active Second	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_LINES. START_DATE_ACTIVE_SECOND and QP_LIST_LINES. END_DATE_ACTIVE_SECOND.
Group Qty	Sum of the quantity of group of lines. Used by the internal engine.
Group Amount	Sum of the price of group of lines. Used by the internal engine.
Line Amount	Price for the line quantity. Used by the internal engine.
Rounding Factor	If ROUNDING_FLAG = Y and the pricing event does not include the base price phase, the rounding factor that the pricing engine should use.
Updated Adjusted Unit Price	To update the adjusted unit price or to manually override the selling price, the calling application writes the new manual updated price into this field and calls the pricing engine. The pricing engine tries to apply the eligible manual adjustments and calculate the new price. If it cannot apply the manual adjustments, then it will raise an error at that line.
Price Request Code	Unique identifier for order line used by limits processing. It has the structure Request Type Code-Order Id-Line Id. Used by the internal engine.
Hold Code	Whenever the limit is adjusted or exceeded and limit_hold_flag is Y, the engine sets the value of this field to LIMIT. Used by the internal engine.

---

---

Hold Text	Whenever the limit is adjusted or exceeded and limit_hold_flag is Y, the engine sets the value of the field HOLD_CODE to LIMIT and an appropriate message is set to HOLD_TEXT. Used by the internal engine.
Price List Header	Name of the list header used to create or update the pricing line.
Validated Flag	This field is related to PRICE_LIST_HEADER_ID. If set to Y, indicates that the price list is validated and no qualification check is necessary. If set to N, indicates that the price list is not validated.
Qualifiers Exist Flag	This field is related to PRICE_LIST_HEADER_ID. If set to Y, indicates that the qualifiers exist for the price list. If set to N, indicates that the qualifiers does not exist for the price list.
Pricing Attrs Exist Flag	This field is related to PRICE_LIST_HEADER_ID. If set to Y, indicates that pricing attributes exist for the price list. If set to N, indicates that pricing attributes does not exist for the price list.
Primary Qual Match Flag	This field is related to PRICE_LIST_HEADER_ID. If set to Y, indicates that qualifiers exist for primary price list. If set to N, indicates that qualifiers does not exist for primary price list.
Usage Pricing Type	Indicates the usage pricing type. Allowable values are: <ul style="list-style-type: none"> <li>• Regular</li> <li>• Billing</li> <li>• Authoring</li> </ul>

---

The lines region can be used to locate the source of a problem. For example, from the lines region of the Pricing Engine Request window, a specific line in the lines region shows the expected adjusted unit price (map to unit selling price). From the Sales Order window, we observe that the unit-selling price is blank and does not display an expected unit-selling price. The problem is in the pricing integration code. A price is generated, but Oracle Order Management does not display it.

## Pricing Engine Request Line Details Region

This region maps to the QP\_DEBUG\_REQ\_LDETS table. The Req Id + Line Index column maintains the master-detail relationship between lines and line details. This region shows information regarding processed price list lines and modifiers lines selected and/eliminated by the engine.

The Priced box indicates which lines were finally selected for pricing by the pricing engine. The Applied box indicates which lines were considered in calculating the selling price. This region also displays the information for item upgrades, coupon issue, term substitution, freight and special charges, and relationships between price breaks. The information that appears includes the following:

---

Field Name	Description
Priced	This value indicates whether the pricing engine successfully selected all the adjustments (Both manual and automatic). If selected, the line is considered for pricing by the pricing engine. If cleared, the line is rejected by the pricing engine.
Applied	The lists or list lines that this pricing event or a prior pricing event applied. Allowable values are: <ul style="list-style-type: none"><li>• Yes: Applicable when the attribute context is a list or list line</li><li>• No: Not applicable when the attribute context is a list or list line</li></ul>

---

---

Status Code	<p>Indicates returned status. Possible Values are:</p> <ul style="list-style-type: none"> <li>• New record created [All N (No) records are returned back from the pricing engine. These are success records.]</li> <li>• Unchanged (Default status when the line is passed to the pricing engine for processing)</li> <li>• Deleted</li> <li>• Updated</li> <li>• Invalid price list (When passed in price list is not found, then an error occurs.)</li> <li>• GSA violation</li> <li>• Error processing formula</li> <li>• Other error</li> <li>• Error in calculation engine</li> <li>• Failed to price using unit of measure</li> <li>• Invalid unit of measure</li> <li>• Duplicate price list</li> <li>• Unit of measure conversion not found</li> <li>• Could not resolve incompatibility</li> <li>• Could not resolve best price</li> </ul>
Status Text	Returned message from Pricing Engine.
Parent Line Detail Index	PL/SQL unique identifier. Unique identifier of request line detail in calling application.
Line Detail Index	PL/SQL unique identifier. Unique identifier of request line detail in calling application.
List Type Name	List type of the line used. Possible values can be found from the lookup type LIST_TYPE_CODE from qp_lookup table.

---

---

Price List Name	Price List Name of the line used.
Modifier No.	Modifier list number  <b>Additional Information:</b> Manual modifiers: For versions 11.5.9 and higher, records are not created in the Pricing Engine Request Viewer until the manual modifier is applied.
Modifier Name	Modifier list name
List Line Type	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.
List Line No	Modifier list line number
Modifier Level Code	The level at which the list line qualified for the transaction. Based on lookup type MODIFIER_LEVEL_CODE.
Operand Calculation Name	Type of operand. Allowable values are: <ul style="list-style-type: none"> <li>• Adjustment percent (for discounts)</li> <li>• Adjustment amount (for discounts)</li> <li>• Adjustment New Price (for discounts)</li> <li>• UNIT_PRICE (for price lists)</li> <li>• PERCENT_PRICE (for price lists)</li> <li>• LUMPSUM</li> </ul>
Operand	Value of pricing request detail line.
Adjustment Amount	It indicates the dollar value of the adjusted amount. It holds the value of the bucketed adjusted amount for line types like PLL, DIS, and 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.

---

---

Automatic	If Automatic is selected, it indicates that the pricing engine automatically applied the request line detail to the request line. The engine derives the value from the list line.
Bucket	Indicates the pricing bucket in which the pricing engine applied this list line. If MODIFIER_LEVEL is ORDER or if AUTOMATIC_FLAG is set to N, the value in this field cannot be modified.
Pricing Phase	The pricing phase that created the request line detail.
Price Formula	Formula attached to the price list line or modifier line.
Incompatibility Group Code	This specifies that the discount is incompatible with all other discounts in this incompatibility group. Incompatibilities can be specified for discounts across Modifier types.
Override	Indicates if a user in the calling application can override the modifier value. No restriction in place to modify the OPERAND_VALUE irrespective of value in this flag.
Charge Type	Indicates the type of charge based on lookup type FREIGHT_CHARGES_TYPE. Used for Freight/Special Charge-type modifiers.
Charge Sub Type	Indicates the type of charge based on lookup type CHARGE_TYPE_CODE.
Item Upgrade Value From	Original Item.
Item Upgrade Value To	Upgraded Item. The Item and its upgrade item must be related and the relationship is defined in Oracle Inventory screen.
Ask for	Used by the internal engine. If selected, it indicates that the selected modifier is an ASK FOR modifier.
Processed	Used by the internal engine. Indicates whether line has been processed by engine or not. Possible values: <ul style="list-style-type: none"> <li>• No: Not processed</li> <li>• Yes: Processed</li> </ul>

---

---

Created From SQL	<p>Indicates which cursor was used by the engine to select the modifier. Used by the internal engine. Possible values:</p> <ul style="list-style-type: none"> <li>• PRODUCT_ONLY</li> <li>• EXCLUDED_PRODUCT_ONLY</li> <li>• QUALIFIER_ONLY</li> <li>• PRODUCT_QUALIFIER_ONLY</li> <li>• PRODUCT_PRICING_ONLY</li> <li>• PRODUCT_QUALIFIER_PRICING_ONLY</li> <li>• INSERTED IN SECONDARY LIST HEADER SEARCH</li> <li>• INSERTED IN VALIDATED LIST_HEADER_SEARCH1</li> <li>• INSERTED IN NOT VALIDATED LIST_HEADER_SEARCH1</li> <li>• INSERTED IN VALIDATED ASKED FOR PROMOTION SEARCH</li> <li>• INSERTED IN NOT VALIDATED QUAL_LIST_HEADER_SEARCH</li> <li>• INSERTED BY CREATE_QUALIFIER_FROM_LIST</li> </ul>
Line Quantity	<p>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.</p>
Product Precedence	<p>It indicates the rank of preference given for the Qualifiers/Pricing Attributes. For the same item if there are more than one incompatible discounts qualifying then the discount with the higher precedence is given.</p>
Best Percent	<p>Modifier percentage that gives the best price. Used by the internal engine.</p>
Primary UOM	<p>If set to Yes it indicates that if the price cannot be found for a product in the UOM passed from the calling application, the pricing engine will convert the transaction quantity to the primary UOM specified. Applicable only to price lists.</p>

---

---

Benefit Qty	The accrual quantity for non-monetary accruals or, for promotional goods, item quantity
Benefit UOM Code	The accrual unit of measure for non-monetary accruals, or for promotional goods, item unit of measure.
Accrual	Indicates whether the discount is an accrual.
Accrual Conversion Rate	The rate to use when converting a non-monetary accrual to a monetary value.
Estimated Accrual Rate	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.
Rounding Factor	If ROUNDING_FLAG = Y and the pricing event does not include the base price phase, the rounding factor that the pricing engine should use. This value is passed in by the calling application.
Secondary Price list Ind:	Indicates that the pricing used a secondary price list instead of the price list that the calling application requested. Applicable only to price lists.
Group Qty	Sum of the quantity of group of lines. Used by the internal engine.
Group Amount	Sum of the price of group of lines. Used by the internal engine.
Process Code	This is set by the engine and used for selecting lines for calculation. Used by the internal engine. Possible values: <ul style="list-style-type: none"> <li>• N: New</li> <li>• D: Deleted</li> <li>• X: Unchanged</li> </ul>
Updated Flag	This value is passed in by the calling application. Used by the internal engine.
Limit Code	The value of this field is set to ADJUSTED when limit is adjusted, set to EXCEEDED when the limit is exceeded. Used by the internal engine.
Limit Text	Returned message from Pricing Engine whenever limit is Exceeded or Adjusted.

---

---

Header Limit Exists	Selected if a Header Limit exists.
Line Limit Exists	Selected if the Line Limit exists

---

## Attributes Window

The line attributes region maps to QP\_DEBUG\_REQ\_LINE\_ATTRS table. This region displays information about the pricing attributes that the attribute mapping function passed to the pricing engine. The pricing engine uses these attributes to qualify a line or an order for price and adjustments. From the Pricing Engine Request Viewer window select the Attributes button to display all attributes for a selected line or line detail. If you select the Attributes button from Request Lines region, the attributes displayed will be attributes passed to the pricing engine.

If you select the Attribute from the Request Line Details region, the attributes displayed will be the attributes related to the selected price list lines and modifier lines.

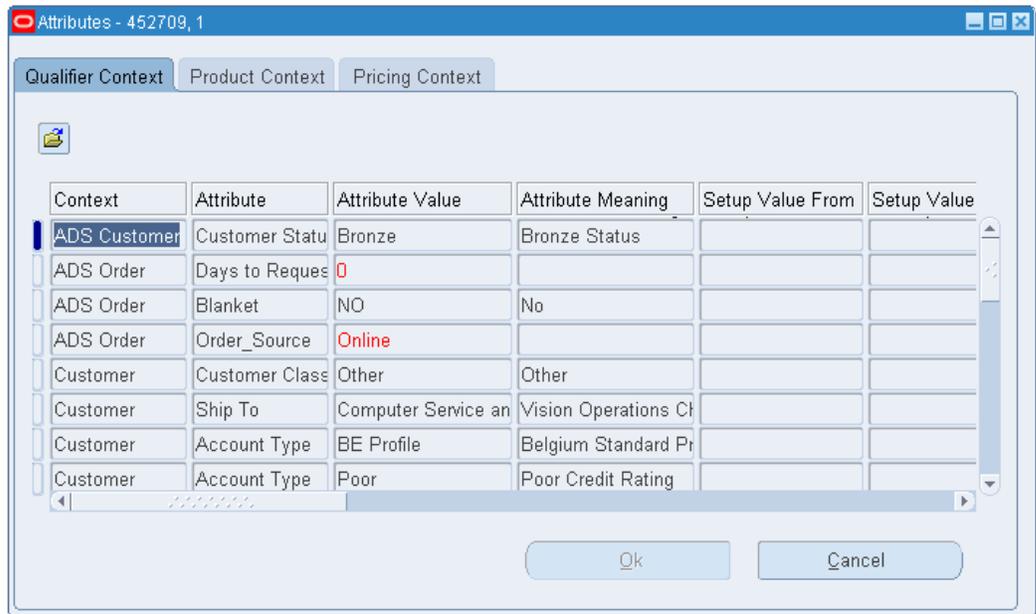
Use this table when Oracle Order Management does not return an expected discount or adjustment and the list line ID do not appear in line details region.

The Attributes window consists of three tabs: Qualifier Context, Product Context and Pricing Context.

### **Qualifier Context tab**

The following image depicts the Qualifier Context tab from the Attributes window:

**Qualifier Context tab: Attributes window**



Columns in this tab include:

Field Name	Description
Context	Context for a product or pricing attribute, for example, Product Hierarchy.
Attribute	Product or pricing attribute, for example, PRICING_ATTRIBUTE11: Customer Item ID.
Value From	Passed in value for product or pricing attribute.
Setup Value From	Setup value for product or pricing attribute.
Setup Value To	Setup value for product or pricing attribute.
Grouping Number	It indicates the qualifier grouping number which is used to group qualifiers together to create AND/OR relationships.
Validated Flag	If set to Y, Indicates that the price list is validated and no qualification check is necessary. If set to N, Indicates that the price list is not validated.

---

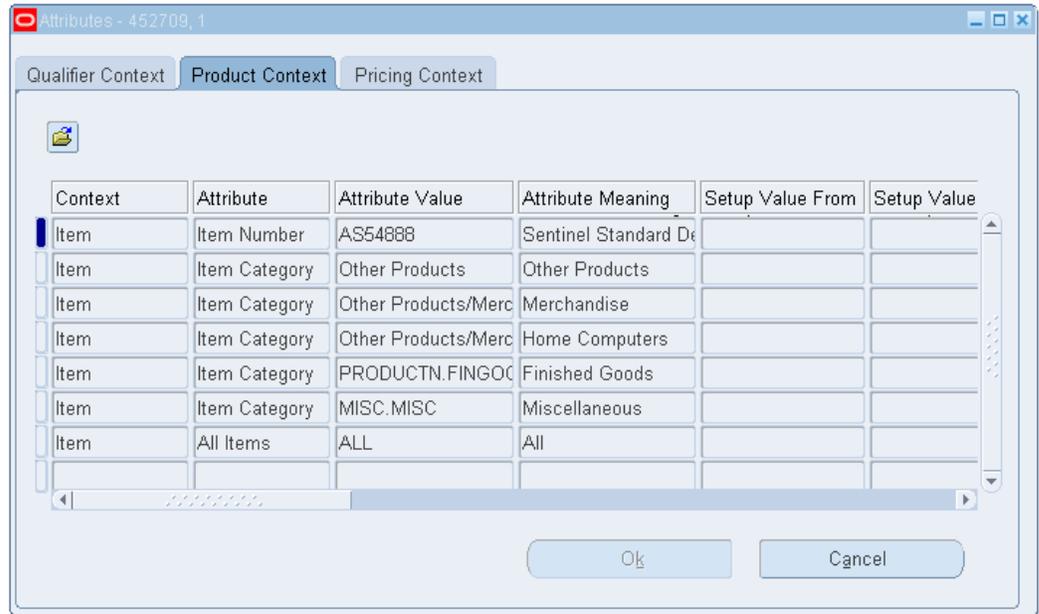
Comparison Operator Type	The relational operator code used to define how the pricing engine should evaluate the pricing attributes or qualifier attributes, based on lookup type COMPARISON_OPERATOR.
Applied Flag	The lists or list lines that this pricing event or a prior pricing event applied. Allowable values are: <ul style="list-style-type: none"> <li>• <i>Yes</i>: Applicable when the attribute context is a list or list line</li> <li>• <i>No</i>: Applicable when the attribute context is a list or list line</li> </ul>
Qualifier Precedence	The precedence number, or selectivity of the qualifier attribute in the Qualifier Descriptive Flex field. It is used by the pricing engine for incompatibility resolution.
Data Type	Indicates the data type of the pricing attribute value or qualifier attribute value.
Processed Code	Set by the engine and used for selecting lines for calculation. Used by the internal engine. Possible values are: <ul style="list-style-type: none"> <li>• N: New</li> <li>• D: Deleted</li> <li>• X: Unchanged.</li> </ul>
Distinct Qualifier Flag	To determine qualification engine sets this flag to indicate that this is unique qualifier.
Primary UOM Flag	If set to Yes, indicates that if the price cannot be found for a product in the UOM passed from the calling application, then the pricing engine will convert the transaction quantity to the primary UOM specified.
Modifier Number	Modifier list number.
Modifier Name	Modifier list name.
List Line Number	Modifier list line number.

---

### Product Context tab

The following image shows the Product Context tab:

**Product Context tab: Attributes window**



Columns in this tab include:

---

Field Name	Description
Context	Context for a product or pricing attribute, for example, Product Hierarchy.
Attribute	Product or pricing attribute, for example, PRICING_ATTRIBUTE11: Customer Item ID.
Value From	Passed in value for product or pricing attribute.
Setup Value From:	Setup value for product or pricing attribute.
Setup Value To:	Setup value for product or pricing attribute.
Applied Flag:	The lists or list lines that this pricing event or a prior pricing event applied. Allowable values are: <ul style="list-style-type: none"> <li>• Yes: Applicable when the attribute context is a list or list line</li> <li>• No: Applicable when the attribute context is a list or list line</li> </ul>

---

---

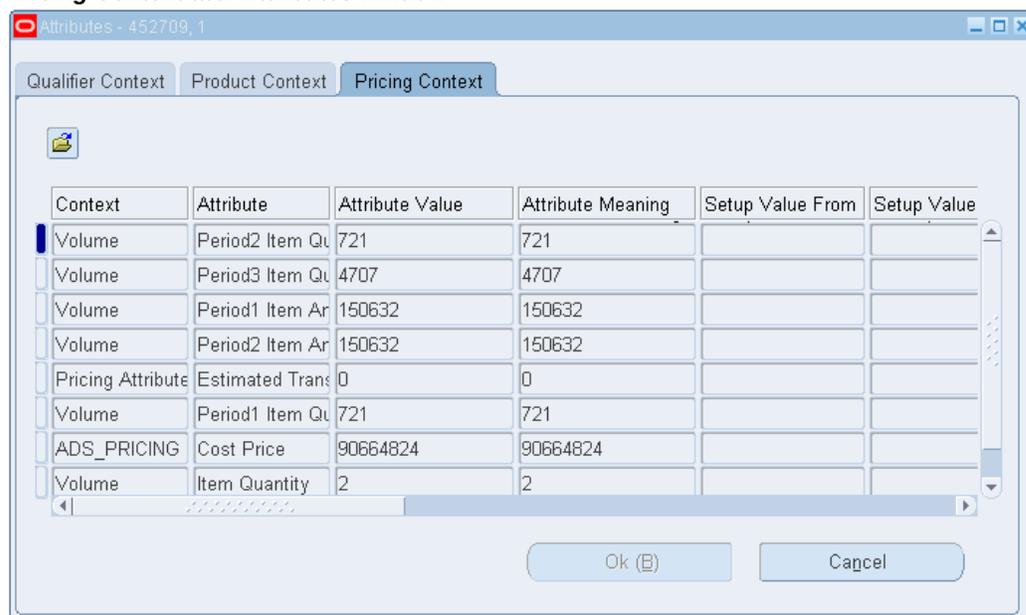
Qualifier Precedence:	The precedence number, or selectivity of the qualifier attribute in the Qualifier Descriptive Flex field. It is used by the pricing engine for incompatibility resolution. This field cannot be updated by the user.
Data Type	Indicates the data type of the pricing attribute value or qualifier attribute value.
Product UOM	Unit of measure of the item, product group, and so on for which the price or modifier is defined.
Processed Code	This is set by the engine and used for selecting lines for calculation. Used by the internal engine. Possible values are: <ul style="list-style-type: none"> <li>• N: New</li> <li>• D: Deleted</li> <li>• X: Unchanged.</li> </ul>
Excluded Flag	If set to Yes, indicates that the product value was defined as an excluded item on the modifier line.
Group Qty	Sum of the quantity of group of lines. Used by the internal engine.
Group Amount	Sum of the price of group of lines. Used by the internal engine.
Primary UOM Flag	Primary UOM Flag: If set to Yes it indicates that if the price cannot be found for a product in the UOM passed from the calling application, the pricing engine will convert the transaction quantity to the primary UOM specified.
Modifier Number	Modifier list number.
Modifier Name	Modifier list name.
List Line Number	Modifier list line number.

---

### Pricing Context tab

The following image shows the Pricing Context tab:

**Pricing Context tab: Attributes window**



Columns in the Pricing Context tab are described in the following table:

Field Name	Description
Context	Context for a product or pricing attribute, for example, Product Hierarchy.
Attribute	Product or pricing attribute, for example, PRICING_ATTRIBUTE11: Customer Item ID.
Value From	Passed in value for product or pricing attribute.
Setup Value From	Setup value for product or pricing attribute.
Setup Value To	Setup value for product or pricing attribute.
Comparison Operator Type	The relational operator code used to define how the pricing engine should evaluate the pricing attributes or qualifier attributes, based on lookup type COMPARISON_OPERATOR.

Field Name	Description
Applied Flag	The lists or list lines that this pricing event or a prior pricing event applied. Allowable values are: <ul style="list-style-type: none"> <li>• Yes: Applicable when the attribute context is a list or list line</li> <li>• No: Applicable when the attribute context is a list or list line</li> </ul>
Data Type	Indicates the data type of the pricing attribute value or qualifier attribute value.
Processed Code	Set by the engine and used for selecting lines for calculation. Used by the internal engine. Possible values: <ul style="list-style-type: none"> <li>• N: New</li> <li>• D : Deleted</li> <li>• X: Unchanged.</li> </ul>
Primary Uom Flag	If set to Y it indicates that if the price cannot be found for a product in the UOM passed from the calling application, the pricing engine will convert the transaction quantity to the primary UOM specified.
Modifier Number	Modifier list number.
Modifier Name	Modifier list name.
List Line Number	Modifier list line number.

The line attributes window of the Pricing Engine Requests Viewer window can be used to locate the source of a problem. For example, customer 1006 receives a discount. If Oracle Order Management does not return the discount and the line detail region does not display the list line ID in the Pricing Engine Requests Viewer window, the solution would be as follows:

In the line attributes region, query to find the record with the following values:

- Context = CUSTOMER
- Attribute = QUALIFIER\_ATTRIBUTE2

- Value = 1006

If you do not find the attribute, this qualifier was not passed to the pricing engine. It is possible that the Build Attribute Mapping Rules program was not run after the first use of new type of qualifier (the qualifier is not sourced by the attribute mapping). If you are using your customer-defined qualifier, then make sure that the attributes mapping setup is properly defined.

If you do find the attribute, the problem occurred in the pricing engine. Contact Oracle Support for a pricing engine debug script.

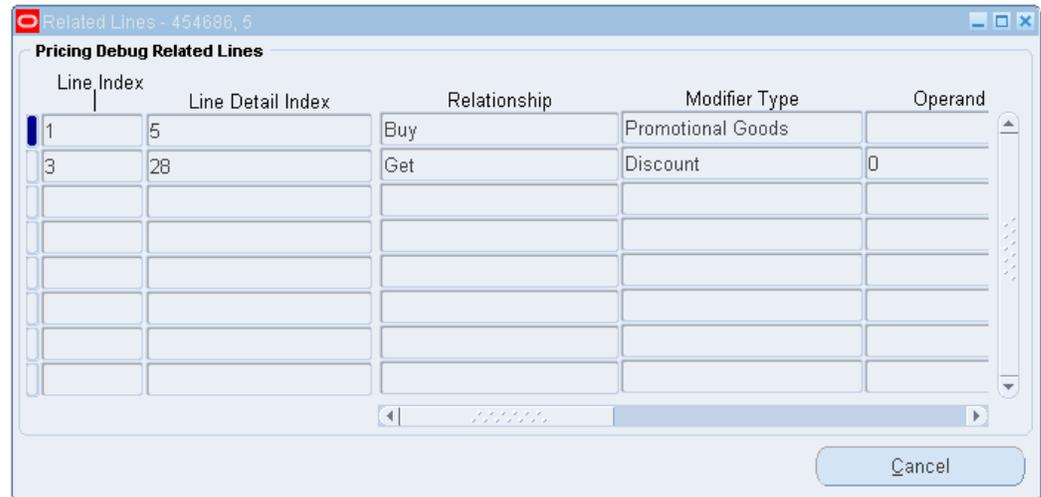
## Related Lines Window

This window maps to the QP\_DEBUG\_REQ\_RLTD\_LINES table.

Select the Related Lines button from the Pricing Engine Request Viewer window to display the Pricing Debug Related Lines window. The cursor needs to be in the Request Line Details region. You can view the relationship between the Buy and Get items for Other Item Discounts and Promotional Goods.

The following image shows the Related Lines window:

**Related Lines window**



Columns in this window are described in the following table:

Field Name	Description
Line Index	PL/SQL unique identifier for request line.

Field Name	Description
Line Detail Index	PL/SQL unique identifier for request detail line.
Relationship	Type of relationship between pricing lines. Allowable values are: <ul style="list-style-type: none"> <li>• BUY</li> <li>• GET</li> </ul>
Modifier Type	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 the qp_lookups table.
Operand	Value of pricing request detail line, for example, 10 currency unit list price with 3 percent discount.
Modifier number	Modifier list number.
Modifier Name	Modifier list name.
Application Method	Type of operand. Eligible values are: <ul style="list-style-type: none"> <li>• Adjustment percent (for discounts)</li> <li>• Adjustment amount (for discounts)</li> <li>• Adjustment New Price (for discounts)</li> <li>• Unit Price (for price lists)</li> <li>• Percent Price (for price lists)</li> <li>• Lumpsum</li> <li>• Block Price</li> <li>• Break Unit Price</li> </ul>

## Formula Step Values Window

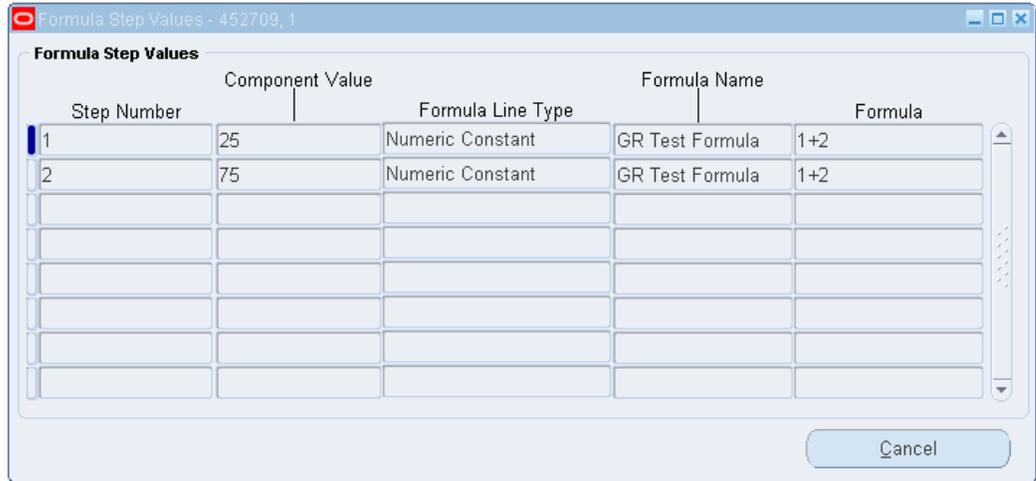
This window maps to QP\_DEBUG\_FORMULA\_STEP\_VALUES table. This region shows information about the formula step values, which are inserted into the table

during the evaluation of formula attached to the price lists. The pricing engine inserts step values into QP\_FORMULA\_STEP\_VALUES only if the profile QP: Insert Formula Step Values into Temp Table is set to Y.

Select the Step Values button from the Pricing Engine Request Viewer window to display the Formula Step Values window. The cursor needs to be in the Request Line Detail region.

The following image shows the Formula Step Values window:

**Formula Step Values window**



Columns in this window are described in the following table:

Field Name	Description
Step Number	Step number corresponding to a formula line.
Component Value	Evaluated value of a formula step.

Field Name	Description
Formula Line Type	Type of the formula line. Possible values are: <ul style="list-style-type: none"> <li>• FUNC: Function</li> <li>• LP: List Price</li> <li>• ML: Factor List</li> <li>• MV: Modifier Value</li> <li>• NUM: Numeric Constant</li> <li>• PLL: Price List Line</li> <li>• PRA: Pricing Attribute</li> </ul>
Formula Name	Name of the pricing formula.
Formula	Mathematical formula for a rule.

## Debug Log Window

The Debug Log window, which maps to the `qp_debug_text` table, displays the contents of the debug log file.

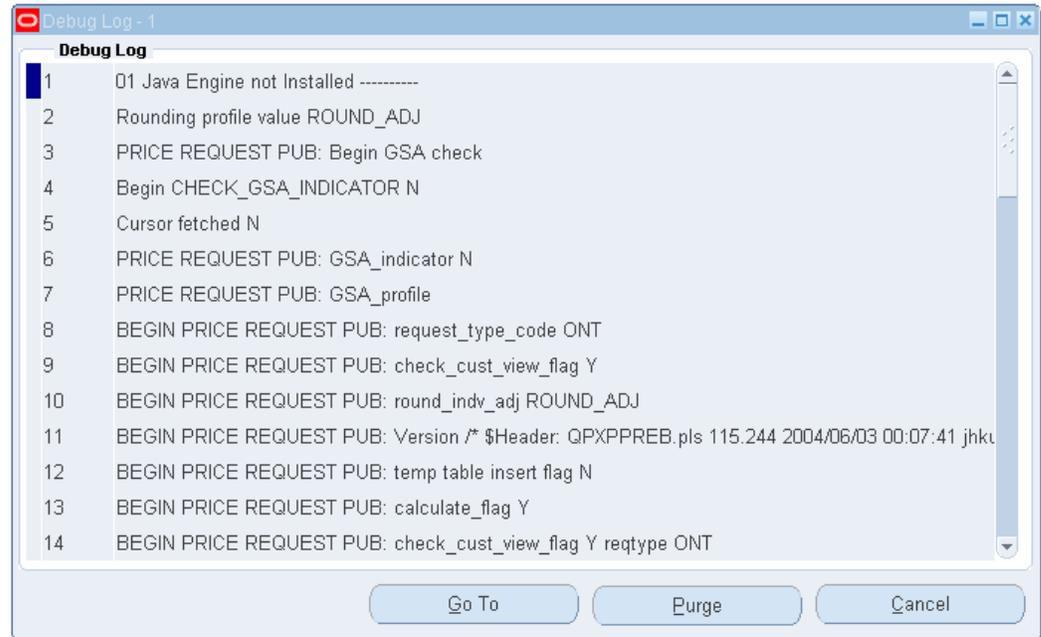
### Searching the Debug Log

You can search for a string in the Debug Log to find related records and lines. Select a returned record, and click the Go To button to view ten previous records from that line number and the remaining lines.

### Purging the Debug Log

Click Purge to delete all the records in the Debug Log.

### Debug Log



## Analyzing Error Messages

From the Line Details region of the Pricing Engine Requests Viewer window, you can analyze error messages to troubleshoot and resolve problems. For example, if Oracle Order Management does not return an expected adjustment (list line), then you could use the Pricing Engine Requests Viewer window to find the problems as described in the following example:

In the Pricing Engine Requests Viewer window, look for the list line ID in the `created_from_list_line_id` column. If you find the expected list line ID in the window and it has a `pricing_status_code` of `D_GRP`, the grouping operation of the pricing engine deleted it.

Check your grouping condition in the Modifier window to see what other conditions must be met to receive the adjustment. If the list line ID has a `pricing_status_code` of `N` (accepted by engine) and it is not reflected in pricing integration, then the problem occurs in pricing integration.

Other pricing status codes that may appear are listed in the following tables:

This table lists the three success codes for a line:

<b>Pricing Status Code</b>	<b>Error</b>
N	New record created
U	Updated
X	Unchanged

This table lists the available internal processing status:

<b>Pricing Status Code</b>	<b>Error</b>
D	Deleted
T	Transient
B	Best price evaluation
OTHER_ITEM_BENEFITS	Deleted in PRG processing
P_UOM_FLAG	Removed due to primary UOM conversion
I	Deleted during incompatibility
G	Deleted in grouping

This table lists codes that require action from the calling application:

<b>Pricing Status Code</b>	<b>Error</b>
IPL	Invalid price list
GSA	GSA violation
NMS	Item not found in primary or secondary price list
FER	Error processing formula

<b>Pricing Status Code</b>	<b>Error</b>
OER	Other errors
S	System generated message
CALC	Error in calculation engine
UOM	Failed for price unit of measure
INVALID_UOM	Invalid unit of measure
DUPLICATE_PRICE_LIST	Duplicate price lists
INVALID_UOM_CONV	Unit of measure conversion not found
INVALID_INCOMP	Unable to resolve incompatibility
INVALID_BEST_PRICE	Unable to resolve best price
LIMIT	Put limit on hold
EXCEEDED	Limit exceeded

### **Viewing Service and Serviceable Lines**

This Pricing Engine Request Lines region has information related to the Service Lines relationship. The fields Related Line Index and Line Index in the Pricing Engine Request Lines region are related. By comparing the values of the Related Line Index with the Line Index, you can identify if the lines are related, for example, Service Item and Serviceable Item.

For example, Line Index 2 could be a serviceable item (such as Oracle 8i) and Line Index 3 could be a service item (such as Gold Support for 8i).

### **Viewing Price Break lines**

This Pricing Engine Request Line Details region has information related to Price Break Lines Relationship. The Parent Line Detail Index field and Line Detail Index field in the Pricing Engine Request Line Details region are related. By looking at the values of Parent Line Detail Index and Line Detail Index, you can identify if a Price Break exists. Price Break setup has a Price Break Parent Record, which has a line type called PBH. This PBH record can have more than one child record that actually defines the breaks.

For example, you could create a price break setup where the Price Break Parent Record has a line type called Price Break Header. This PBH record could have three child records (Line Detail Index 10, Line Detail Index 11, and Line Detail Index 12) that define

the breaks.

### **Viewing Other Item Discounts (OID)**

In this case, two request lines, such as Buy ITEM1 and get \$500 off on ITEM2, are passed to the pricing engine. In this example, both ITEM1 and ITEM2 need to be ordered on two order lines so that two request lines are created and passed to the pricing engine. When the engine processes the other item discounts, it creates a discount line for -\$500 on the second request line. It also creates a relationship record, and this relationship is shown in the Related Lines window. Line Detail Index 9 is the actual Discount Line, which is the OID line, and Line Detail Index 10 is the actual benefit line, which is the \$500 off line.

### **Viewing Promotional Goods Discount (PRG)**

In this case, only the original buy item request line, such as Buy ITEM1 and get ITEM2 for free, is passed to the pricing engine. In this example, only the request line/order line with ITEM1 is sent to the pricing engine. ITEM2 need not be ordered. The pricing engine selects the PRG Modifier because of purchase of ITEM1 and creates a Line Detail Record (Line Index 1 - Line Detail Index 2). Then it tries to give the benefit, which is a free item (ITEM2). The process engine does the following:

- Creates a new request line (LINE RECORD) is created (Line Index 3).
- Creates a new relationship (RELATED LINES RECORD) between the ITEM1 line and ITEM2 line is created. It is a Line-Line relationship (Line Index 1 - Line Index 3).
- Creates a new Price List Line (LINE DETAIL RECORD) is created for the new request line (Line Index 3 - Line Detail Index 3).
- Creates a new adjustment line for a 100 percent discount is created for the new request line (LINE DETAIL RECORD) (Line Index 3 - Line Detail Index 4).
- Creates a new relationship line (RELATED LINES RECORD) between the original PRG Line Detail Line and the new 100 percent off line detail is created (Line Detail Index 2 - Line Detail Index 4).
- Creates a new record (LINE ATTRIBUTE RECORD) is created for the new ITEM2. (Line Index 2 - PRICING\_CONTEXT = ITEM, PRICING\_ATTRIBUTE = PRICING\_ATTRIBUTE1, PRICING\_ATTR\_VALUE = ITEM2).

### **Purging Pricing Engine Requests**

The concurrent program Purge Pricing Engine Requests will purge the pricing engine requests. You should run this program on a regular basis to purge the historical data from the pricing debug tables or if you observe a marked deterioration in the performance of the Pricing Engine Request Viewer window. Periodically purging the historical data from the pricing debug tables improves the performance of the Pricing Engine Request Viewer window.

See the *Oracle Advanced Pricing User's Guide* for more information on running concurrent programs.

**Important:** This concurrent program does not affect any of the user procedures on Pricing Engine Request Viewer UI.

### **Deleting Pricing Engine Requests**

Follow these steps to delete a previously saved pricing engine request:

1. From the Pricing Engine Requests Viewer window, choose View > Find to find the Pricing Engine Request to delete.
2. Choose Edit > Delete.



---

## Integrating with Oracle Advanced Pricing

This chapter covers the following topics:

- Overview of Integrating with Oracle Advanced Pricing
- Integration Steps Required for Pricing
- Pricing Engine Interaction Details
- Changed Lines API
- Oracle Service Contracts (OKS) Integration: Proration and Price List Locking
- Pricing Features to Support Telecommunications Industry Flows [Oracle Telecommunications Service Ordering (TSO)]

### Overview of Integrating with Oracle Advanced Pricing

Oracle Advanced Pricing is an API-based engine that can be called by any integrating application. This chapter provides certain essential information needed for the successful integration with Advanced Pricing.

For more information, see:

- *Oracle Advanced Pricing Implementation Guide, Attributes Mapping and Technical Considerations*
- *Oracle Order Management Suite APIs and Open Interfaces Manual* for Pricing API and example scripts
- *Oracle Advanced Pricing User's Guide*

### Integration Steps Required for Pricing

The following steps describe the process required to integrate with Oracle Pricing.

### **Step 1. Determine your end to end pricing business needs:**

See the Oracle Advanced Pricing User's Guide and the *Oracle Advanced Pricing Implementation Guide* to determine which pricing features need to be supported by your application.

1. Evaluate your transaction variations and choose the pricing events to be called at what phase of your transaction cycle. To create new pricing phases or learn more about pricing phases and events, see: Oracle Advanced Pricing User's Guide.
2. Determine your business need to choose a pricing transaction entity (PTE). If a new PTE is required, see: Creating a New Pricing Transaction Entity, page 17-25 to learn about how to create a PTE. When you create a new PTE, the seeded contexts and attributes are not available. New contexts and attributes need to be created for the newly created PTE, or existing attributes need to be linked to the PTE.
3. The pricing engine needs to be called with a request type. Each request type is linked to a PTE. Request type determines two things: the price lists and modifiers to be searched during pricing engine call and the attribute mapping rules to be run when build\_context API is called. Note that each modifier/price list is associated with a source system and hence with a PTE. The pricing engine will look at modifiers or price lists belonging to the source systems linked with the PTE.
4. Attribute Mapping rules are seeded by individual transaction systems for their respective PTEs. The PTE is linked to a set of request types and source systems. Determine if your transaction data corresponds to the attribute mapping rules defined already. If not, refer to attribute mapping to learn how to define a custom mapping.

### **Step 2. Determine your pl/sql global structure:**

Before calling the API to build the contexts, it is necessary to populate a global record structure corresponding to the request type with the information pertaining to the summary line (order header) or the request line (order line) for which the qualifier/pricing attribute information needs to be built. Request type ONT uses a global structure oe\_order\_pub.g\_line for order line and oe\_order\_pub.g\_hdr for order header. The request type ASO uses a global structure aso\_pricing\_int.g\_line\_rec and aso\_pricing\_int.g\_header\_rec.

For more information about the oe\_order\_pub.g\_line/g\_hdr structures and the Order Capture API doc for the aso\_pricing\_int.g\_line\_rec and aso\_pricing\_int.g\_header\_rec, see the *Oracle Order Management Open Interfaces, API, & Electronic Messaging Guide*.

Determine if your request can be mapped to one of these structures. Attribute mapping rules need to be defined specific to the structure that will be used by the calling application. If you find that your request structure cannot be mapped to one of these, then you need to define a new structure and write attribute mapping rules using this new structure.

### Step 3. Determine the tables to hold the adjustment information:

If you need to hold the information returned by pricing engine, for showing the breakup of all the benefits given to your order/quote/pricing request, determine if you can use the price adjustment tables used by Order Management/Order Capture.

OE\_PRICE\_ADJUSTMENTS/ASO\_PRICE\_ADJUSTMENTS

Holds the price adjustments

OE\_PRICE\_ADJ\_ATTRIBS/ASO\_PRICE\_ADJ\_ATTRIBS

Holds the qualification conditions applied to give the adjustments

OE\_PRICE\_ADJ ASSOCS/ ASO\_PRICE\_ADJ\_RELATIONSHIPS -

Holds relationships between multiple adjustment records for price breaks, promotional goods and other item benefits

If you need to create your own tables, please use the table definition of the above tables as a guideline and create your tables.

### Step 4. Populate the pricing request structures:

The following is the API structure for QP\_PREQ\_PUB.PRICE\_REQUEST:

```
(p_control_rec IN  
QP_PREQ_GRP.CONTROL_RECORD_TYPE,  
x_return_status OUT VARCHAR2,  
x_return_status_text OUT VARCHAR2  
);
```

1. Populate the control record p\_control\_rec.

For details on the parameters of the control record, see the *Order Management Suite APIs and Open Interfaces Manual, Pricing APIs*. The control record determines the way the pricing engine returns the price.

2. Call the API: QP\_Price\_Request\_Context.Set\_Request\_Id();

Set the pricing request\_id to enable the pricing engine to identify the data in the pricing temporary tables that belong to the current pricing engine call. The pricing engine will not delete the temporary table data before each pricing engine call. Instead, the data from the previous pricing engine call may remain in the pricing temporary tables. The calling application needs to set the request\_id each time the pricing engine call is made. This will be the first step.

The API QP\_Price\_Request\_Context.Set\_Request\_Id() call sets a unique request\_id for every pricing engine call made in the same session without commits or rollbacks. This negates the need to delete the data in the pricing temporary tables between calls. Also, if the request\_id is not set, then the calling application needs to delete the data in the pricing temporary tables before making the next pricing engine call without a commit or rollback. Remember that a commit or rollback purges the data in the temporary tables so that the records do not need to be deleted.

3. Populate the global structure used by the attribute mapping. For example if you are using request type ONT, populate the PL/SQL structure OE\_ORDER\_PUB.G\_LINE.
4. Call the API: QP\_ATTR\_MAPPING\_PUB.Build\_contexts

```
QP_Attribute_Mapping_PUB.Build_Contexts
(
    p_request_type_code      IN      VARCHAR2,
    p_line_index            IN      NUMBER,
    p_pricing_type_code     IN      VARCHAR2
);
```

This API evaluates the public record structures listed in the attribute mapping rule; therefore, it is necessary that the request line information for each line is loaded into the request structure and the API is called for each line. The build context API inserts the mapped attributes into the pricing temporary tables with the passed line\_index. Therefore, you must use the same line\_index that is used for the request lines.

If no attribute mapping rules or record structure are found, then the qualifiers and pricing attributes can be inserted into the qp\_preq\_line\_attrs\_tmp. For example, if you want the pricing engine to fetch the price from a particular price list, pass the price list in qp\_preq\_line\_attrs\_tmp with qualifier\_context=MODLIST.

The qualifier\_attribute=QUALIFIER\_ATTRIBUTE4 and qualifier\_attr\_value\_from holds the price\_list\_id.

Remember that the inventory\_item\_id is passed in the temporary table qp\_preq\_line\_attrs\_tmp with pricing\_context=ITEM pricing\_attribute=PRICING\_ATTRIBUTE1 and pricing\_attr\_value\_from holds the item\_id.

The build context API needs to be called with a p\_pricing\_type\_code of L for the request lines.

5. For more information about the Copy\_Line\_to\_request to load the request lines, see: *Oracle Order Management Open Interfaces, API, & Electronic Messaging Guide, Sample Code using Order Management Structure*. Call the API QP\_PREQ\_GRP.Insert\_Lines2 to insert the request lines to the pricing temporary table qp\_preq\_lines\_tmp.

For more information about API structures and parameter descriptions, see: *Oracle Order Management Open Interfaces, API, & Electronic Messaging Guide*.

6. Append any user-entered pricing attributes and Asked for promotions/deals or coupons to the temporary table qp\_preq\_line\_attrs\_tmp. Set the validated\_flag of qp\_preq\_line\_attrs\_tmp to Y for "Asked For" promotions, if you do not want pricing to check for the qualifications, before applying the promotion.

For more information about the Append\_ask\_for procedure, see the *Oracle Order Management Open Interfaces, API, & Electronic Messaging Guide, Sample Code using Order Management Structure*. These attributes can be inserted into the temporary table qp\_preq\_line\_attrs\_tmp using the API QP\_PREQ\_GRP.insert\_line\_attrs2.

For more information about API structures and parameter descriptions, see the *Oracle Order Management Open Interfaces, API, & Electronic Messaging Guide*.

**Note:** When passing the keys to the pricing interface and temporary tables, the keys such as `line_index`, `line_detail_index` and so on need to be within the range for a `pls_integer` datatype.

7. To apply manual adjustments or to have the pricing engine consider any other adjustments, insert those records into the pricing temporary table `qp_preq_ldets_tmp` with the `applied_flag` and `updated_flag` set to Y (Yes). Pass adjustments or modifiers to the pricing engine using the `QP_PREQ_GRP.insert_ldets2` API.

8. Ensure that the Summary Line is populated.

Every request to the pricing engine must contain a summary record in `qp_preq_lines_tmp` with information from order header. The pricing engine will apply the order level modifiers against the summary line passed. If the summary line is not passed, the pricing engine will not apply any order level adjustments.

Set the `line_type_code` to ORDER for the summary line. For order header, repeat steps # 1 to 6, and use `p_pricing_type=H` while calling `QP_ATTR_MAPPING_PUB.build_context()` for the summary line.

For more information on how to populate the summary record using `copy_Header_to_request()`, see *Oracle Order Management Open Interfaces, API, & Electronic Messaging Guide*, Sample Code using Order Management Structure.

9. For Service Lines with Percentage Price, ensure that the Parent Line is passed. For more information about how to price service items, see: Service Item Pricing.

### Step 5. Call `Price_request()`:

Call the API:

```
QP_PREQ_PUB.PRICE_REQUEST  
(p_control_rec IN QP_PREQ_GRP.CONTROL_RECORD_TYPE,  
x_return_status OUT VARCHAR2,  
x_return_status_text OUT VARCHAR2);
```

### Step 6. Interpreting the results of price request:

1. Handling errors:

The pricing engine can return hard errors and soft errors. The pricing engine call is a success if the value of `x_return_status` is `FND_API.G_RET_STS_SUCCESS`.

The soft errors can indicate the line level exceptions while pricing. These errors are populated in `qp_preq_lines_tmp.pricing_status_code`. These are the three success codes for a line:

G\_STATUS\_NEW  
G\_STATUS\_UPDATED;  
G\_STATUS\_UNCHANGED

These codes require some action from the calling application:

G\_STATUS\_DELETED  
G\_STATUS\_TRANSIENT  
G\_STATUS\_GROUPING  
G\_STATUS\_INVALID\_PRICE\_LIST  
G\_STATUS\_GSA\_VIOLATIONG\_STS\_LHS\_NOT\_FOUND  
G\_STATUS\_FORMULA\_ERROR  
G\_STATUS\_OTHER\_ERRORS  
G\_STATUS\_SYSTEM\_GENERATED  
G\_STATUS\_BEST\_PRICE\_EVAL  
G\_STATUS\_INCOMP\_LOGIC  
G\_STATUS\_CALC\_ERROR  
G\_STATUS\_UOM\_FAILURE  
G\_STATUS\_PRIMARY\_UOM\_FLAG  
G\_STATUS\_OTHER\_ITEM\_BENEFITS  
G\_STATUS\_INVALID\_UOM  
G\_STATUS\_DUP\_PRICE\_LIST  
G\_STATUS\_INVALID\_UOM\_CONV  
G\_STATUS\_INVALID\_INCOMP  
G\_STATUS\_BEST\_PRICE\_EVAL\_ERROR  
G\_STATUS\_LIMIT\_HOLD  
G\_STATUS\_LIMIT\_EXCEEDED  
G\_STATUS\_LIMIT\_ADJUSTED  
G\_STATUS\_LIMIT\_CONSUMED

The GSA violation G\_STATUS\_GSA\_VIOLATION is a special case. Refer to the GSA Violation topic in the following section detailing the GSA Violation behavior in detail.

For more information about the different status codes, see: *Oracle Advanced Pricing Implementation Guide*, Troubleshooting.

Price List fetched for a request line.

- The price\_list\_id is populated on the price\_list\_id column in the qp\_preq\_lines\_tmp.

The List price (undiscounted base price) is returned in unit\_price and the discounted price (after applying all the discounts/surcharges) is in adjusted\_unit\_price. Remember that these are per unit price expressed in unit of measure pricing\_uom\_code. Pricing\_uom\_code could be different from the line\_uom\_code (Ordered UOM). Therefore, if the price list is set in a unit of measure EACH and has been marked as primary, and if the order is in Dozen and there is no price list line for Dozen, then the pricing engine would return the price in EACH.

Priced\_quantity holds the line\_quantity (Order Line quantity) expressed in pricing\_uom\_code. If the price list line has a percent price set, the percentage is returned in percent\_price and the price of the parent line used in price calculation is in the parent\_price.

- Modifiers fetched for an order line.

The modifiers/adjustments are returned in the temporary table qp\_preq\_ldets\_tmp. Pricing has a view qp\_ldets\_v that the calling application needs to look at to

insert/update the adjustment details into the transaction tables. If the `created_from_list_line_type` for this record is of type OID (Other Item Discounts), PRG (Promotional Goods) CIE (Coupon Issue), or PBH (Price Breaks), the `Qp_preq_rtd_lines_tmp` would hold the relationship. For example, you can find the new line created for a free good by looking for an adjustment line of type PRG in `qp_ldets_v`, and then look for the `line_detail_index` in `qp_preq_rltd_lines_tmp` and fetch `related_line_detail_index`. Now search `qp_ldets_v`, with `line_detail_index = related_line_detail_index`. The `line_index` corresponding to this would be the free goods line.

If you find a record with `qp_ldets_v.list_line_type_code = IUE`, the order line has a free upgrade of the item given to the order line.

If `qp_ldets_v.list_line_type_code=TSN`, the adjustment is of type terms upgrade.

If `qp_ldets_v.accrual_flag` is set to Y, the adjustment is of type accrual is not included in calculating the `adjusted_unit_price`. `Benefit_qty` is populated for non-monetary accruals.

All the adjustments with `automatic_flag=Y` have been applied by the engine along with any manual adjustments with `automatic_flag=N` passed in by the user and having `applied_flag=Y` and `updated_flag=Y`.

`qp_ldets_v.operand_calculation_code` holds `qp_list_lines.arithmetic_operator (% ,AMT ,LUMPSUM ,NEWPRICE)`.

The value of this is held in `qp_ldets_v.operand_value`. The \$ value of `operand_value` is contained in `qp_ldets_v.adjustment_amount`. This `adjustment_amount` is per unit expressed in pricing unit of measure (UOM).

Freight charges have `qp_ldets_v.list_line_type_code=FREIGHT CHARGE` and they have `charge_type_code` and `charge_subtype_code` populated. At present, pricing returns only one freight charge (of highest monetary value) for a `charge_type_code` and `sub_type_code` combination.

- **Qualifiers and attributes**

The table `qp_preq_qual_tmp` holds the qualifiers that were matched for a price adjustment. This information can help in tracking why a particular adjustment was given for any given order line. The structure `qp_preq_line_attrs_tmp` holds the product and pricing attributes matched for an adjustment record.

## Pricing Engine Interaction Details

This section provides an overview of the features supported by the pricing engine and how the pricing engine processes them. This information includes what the pricing engine returns for each feature and how the request information needs to be passed on subsequent calls for the same request lines to get the same results back again.

## Passing adjustments/modifiers to the pricing engine

If the calling application needs to apply specific modifiers to the pricing engine in order for it to apply them against a request line, it needs to be inserted into `qp_preq_ldets_tmp` with `pricing_status_code = QP_PREQ_GRP`. `G_STATUS_UNCHANGED`. You can pass adjustments or modifiers to the pricing engine using the `QP_PREQ_GRP.insert_ldets2` API.

### Manual adjustments

All the automatic modifiers (`automatic_flag = Y`) 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 calling application if the profile QP: Return Manual Adjustments is set to Y. In addition to these discounts, all the qualified manual modifiers of type Discounts. Surcharge discounts are also returned to the calling application, unapplied.

**Note:** When manual adjustments are applied or automatic adjustments are overridden, then changes in the pricing setup for those modifiers are not applicable anymore. The pricing engine will not do a qualification check for those modifiers. For example, if the qualifications change (for example, a change is made to the modifier eligibility criteria), the manually overridden modifiers will continue to get applied even if the request line does not meet the qualification. For example, If you have applied a manual modifier on an order quote and then make changes so that the quote does not qualify for it, the pricing engine does not remove the modifier. This condition also applies to changing qualifiers on the modifier.

You can try the same with an automatic overrideable modifier. If you override the automatic modifier, and change the UOM, the modifier will not be removed from the quote.

### Applying manual adjustments

Manual adjustments can be applied in two ways:

- The calling application can pass the manual adjustment to the pricing engine with `applied_Flag = Y` and `updated_Flag = Y`. The pricing engine will apply this manual adjustment. The calling application can override the manual adjustment by passing the new operand `qp_preq_ldets_tmp.operand_value`. Manual adjustments can be passed to the pricing engine by inserting the adjustment into `qp_preq_ldets_tmp` against the request line to which it needs to be applied.

For example, the calling application makes a pricing engine call with three request lines. A manual adjustment of 10 percent is set up to be applied to the second request line. Then, the calling application 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 request 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.

- The calling application can override the selling price by passing the new selling price in the `qp_preq_lines_tmp.updated_adjusted_unit_price`. The engine will then pick up a suitable manual overrideable modifier that the request line has been qualified for 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. The back calculation is done after applying all the automatic and overridden manual adjustments. Therefore, the calculated selling price will reflect the desired selling price.

For example, if the calling application passes three request lines to the pricing engine; the unit selling price on the second request line is 80 USD and the unit list price is 100 USD and the user wants to override the selling price to 90 USD. In this case, the user has to pass 80 USD in `qp_preq_lines_tmp.updated_adjusted_unit_price` on the request line in the record corresponding to the second request line. In this case, the pricing engine applies all the automatic adjustments and any manual overridden adjustments passed by the user.

The pricing engine registers that the selling price has been overridden. It picks up all the qualified 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 overrideable surcharge, back calculates the adjustment amount, \$10 surcharge in this case, and returns it with `calculation_code` as `BACK_CALCULATE`.

If no surcharge adjustment applies, it tries to apply a manual overrideable discount adjustment. If no qualified manual overrideable adjustments apply, it returns an error status on the second request line and the `pricing_status_text` indicates that there are no manual adjustments apply. If an error occurs during the back calculation, the engine returns an error status on the second request line. The `pricing_status_code` on the second request line has the error code `QP_PREQ_PUB.G_BACK_CALCULATION_STS` in case of an error.

### **Deleting manual adjustments that are applied**

To delete manual adjustments that are applied, they need to be deleted from the transaction table on the calling application's side, which stores the applied adjustments returned by the pricing engine. This means that the calling application also needs to delete the attributes and the associations (relationships) for the adjustment. In case the manual adjustment is a price break, the child lines of the price break, the attributes of the price break and the child lines and the relationships between the price break and the child lines also need to be applied.

Once they are deleted from the transaction tables, they will not be passed to the pricing engine in any consequent calls to the pricing engine and will not get applied.

### **Deleting automatic overridden adjustments that are applied**

In case your business supports deleting automatic overridden adjustments, the calling application needs to insert these adjustments as `applied_flag = N` and `updated_flag = Y` and these need to be passed to the pricing engine during any consequent pricing engine calls. The pricing engine will not apply this automatic adjustment even if the request line qualifies for it.

### **Apply automatic overrideable modifiers**

In order to apply automatic overrideable modifiers, pass the modifiers to pricing engine by inserting the modifier into `qp_preq_ldets_tmp` with `updated_flag = Y` and `applied_flag = Y` and `pricing_status_code = QP_PREQ_GRP.G_STATUS_UNCHANGED`. The pricing engine will apply this modifier.

### **Manual/Overrideable price breaks**

To apply manual overrideable price breaks, pass the price break header adjustment and its child lines with the relationships between the price break modifier and the child break lines with the overridden operand. The pricing engine evaluates the break and applies the overridden price breaks. Also, remember to pass the volume attributes that the pricing engine returned previously along with the price break modifier and the child lines. The volume attributes are necessary because the pricing engine does not look at the pricing setup to derive the item quantity/amount information on the price break modifier. To insert relationships, use the `QP_PREQ_GRP.insert_rltd_lines2` API.

## **GSA Violation**

Pricing setup allows users to create GSA price lists. A GSA violation occurs when the price of an item goes below the GSA price for the item for a non-GSA customer. The pricing engine checks for GSA violation at the end of the pricing engine call. If a GSA Violation has occurred, then the `pricing_status_code` on the request line is set to `QP_PREQ_GRP.G_STATUS_GSA_VIOLATION`. The calling application handles the GSA violation by raising a warning or an error message. For GSA customers, the pricing engine gives the price on the GSA price list. This request line can qualify for further discounts for the GSA customer.

The pricing engine does GSA violation checks only if the `GSA_CHECK_FLAG` on the control record is passed as `Y` and if the GSA qualifier (Context = CUSTOMER, Attribute = QUALIFIER\_ATTRIBUTE15) is passed on the request line with a value `N` indicating that the customer is a non-GSA customer. The attribute mapping API returns a GSA qualifier based on the `gsa_indicator` flag checked on the customer in AR or the invoice location has the `gsa_indicator` set to `Y`.

In Order Management, if the GSA Violation profile is set to Warning, a warning message is raised. If the profile is set to Error, the application throws an error message.

## **Bucketing**

The pricing engine applies bucketing rules after getting all modifiers to calculate the unit selling price.

## Pricing Formulas

The formulas are processed and the operand is evaluated based on the attributes or factors passed to the pricing engine.

## Rounding

Rounding refers to the rounding of the list and selling price. Rounding is controlled by the `rounding_flag` on the `control_record`, which is entered into the pricing engine and the value of the profile QP: Selling Price Rounding Options. For more information, see: *Oracle Advanced Pricing Implementation Guide*, Profile Options and for information on the values for the rounding flag, see *Oracle Order Management Open Interfaces, API, & Electronic Messaging Guide*.

## Freight Charges

The freight charges that the order line qualified for are evaluated and the pricing engine applies the maximum freight charge for each charge type(`charge_type_code` and `charge_subtype_code`) for the request line. The freight charge does not affect the selling price. Manual and overridden charges that need to be applied, must be passed to the pricing engine with `applied_flag` set to Y and `updated_flag` set to Y, just like any manual adjustment (discount and surcharge).

## Coupon Issue

When a coupon is issued, the engine generates a unique coupon number. The pricing engine inserts a record into QP\_COUPONS table with this unique coupon number. This number may be displayed in the list to show the available coupons for the user to pick. This number is unique. This is the number that can be used to redeem the discount later. One coupon number equals one redemption. Once the coupon number is used (redeemed) it cannot be used again. The user should know the coupon number they were given in order to redeem the coupon.

The next time an order is placed, if the user enters the coupon number, the engine qualifies the discount the coupon is eligible for and applies the discount. The coupon is deleted once the coupon has been redeemed.

**Note:** For Coupon Issue modifiers, the Benefit Quantity, Benefit UOM, and Coupon Conversion Rate fields are not supported.

The coupons are stored in QP\_COUPONS table and are marked redeemed when they are redeemed.

To redeem a coupon, the coupon needs to be passed as a qualifier to the order line to the pricing engine (context = MODLIST, attribute =QUALIFIER\_ATTRIBUTE3, value = <coupon number>). The coupon may be stored as an attribute of the line along with the

ask\_for\_promotions. For the pricing engine to give the coupon discount on consequent reprice calls, the coupon needs to be passed as an attribute.

It is possible that the Coupon Issue modifier has qualifiers in the setup. When the coupon is redeemed, the pricing engine does a qualification check to see if all qualifiers are passed and if the qualifiers are not passed, the pricing engine does not qualify the request line for the modifier associated with the coupon. To prevent the pricing engine from doing this qualification check, pass a value of Y to the validated\_flag in the qp\_preq\_line\_attrs\_tmp against this coupon qualifier record.

In Order Management, if an order qualifies for a coupon, the coupon is not deleted when the order is cancelled or when the item is returned. During a reprice, the pricing engine keeps issuing new coupons.

## Item Upgrade

If an order line qualifies for an item upgrade, the item upgrade modifier is applied against that line and can be found in the qp\_ldets\_v temporary table. In the temp table qp\_ldets\_v, the column related\_item\_id has the inventory\_item\_id of the upgraded item and the column inventory\_item\_id has the inventory\_item\_id of the original item. The calling application can replace the originally ordered item on the line with the upgraded item. During reprice, the calling application needs to pass back the original item on the line so that the order is repriced the same way. This can be done by loading at the original inventory\_item\_id from the item upgrade modifier to the public record structure before attribute mapping.

To set up an item upgrade modifier, an item relationship must be defined between the buy item and the upgrade item in Inventory item relationships with a relationship type of Promotional Upgrade.

## Promotional Goods

If an order line qualifies for a promotional goods modifier (PRG), the pricing engine creates a new order line for the free good line. The PRG usually is setup as buy item A get item B at x percent off. In this case, if item A is ordered and if it qualifies for the PRG, the pricing engine creates the new order line with item B and it also applies the x percent discount to this new line. In case of buy 1 get 1 free, this discount is 100 percent.

The new line that is created by the pricing engine can be identified by the value of processed\_code in the temp table qp\_preq\_lines\_tmp which is set to QP\_PREQ\_GRP.G\_BY\_ENGINE.

The calling application needs to create a new order line to represent the free good line. Remember that the free good item is passed as an attribute in the qp\_preq\_line\_attrs\_tmp to that line. The discount on the free good line is passed in the temp table qp\_ldets\_v. The order line that qualified for the PRG modifier is the parent modifier, and the discount on the free goods line is the child line.

The engine creates a parent child relationship in the temp table qp\_preq\_rltd\_lines\_tmp

with `relationship_type_code` as `QP_PREQ_GRP.G_GENERATED_LINE`, the `line_detail_index` is the `line_detail_index` of the parent line and `related_line_detail_index` is the `line_detail_index` of the child line. The above information returned by the pricing engine needs to be stored by the calling application in its transaction tables.

During a reprice, the pricing engine public API `QP_PREQ_PUB` compares the existing free goods line with the promotional goods modifier in the pricing setup. If the promotional goods modifier has not changed, it populates a value of `QP_PREQ_GRP.G_STATUS_UNCHANGED` on the column `process_status` on `qp_preq_lines_tmp`. In this case, the calling application need not make any changes to the free good line. If the pricing setup has changed, then the `process_status` is populated with a value `QP_PREQ_GRP.G_STATUS_UPDATED`. For a fresh pricing engine call, where one of the request lines qualifies for the promotional goods line, the new line is created in `qp_preq_lines_tmp` with `process_status = QP_PREQ_GRP.G_STATUS_NEW`. In this case, the calling application may insert this line into its transaction system.

When the free goods line is created with `process_status = QP_PREQ_GRP.G_STATUS_NEW/G_STATUS_UPDATED`, the calling application needs to make another call to calculate freight charges for the free goods lines. For more information about the implicit call for freight charges for promotional goods, see: Freight Charges for the free good line, page 21-13.

When users do not want the free goods line, they can remove it from the order. To make the pricing engine recognize that it does not need to create the free goods again, complete the following steps:

1. Delete the free goods line, its attributes, its adjustments, and the associations between the buy line and the free good line from the system. Retain the PRG adjustment on the buy line, but mark it as updated by setting the `updated_flag = QP_PREQ_GRP.G_YES`.
2. If some other free goods line results because of the same PRG (if the promotional goods modifier is set up as buy A, get B and C free) on the order/quote, mark the discounts against the remaining free good lines as updated by setting the `updated_flag = Y`.

Now, the pricing engine will not re-create the deleted free goods lines. The same approach needs to be followed if the calling application wants to substitute the free goods item with some other item.

If the buy line is deleted, the calling application needs to delete all the free goods associated with this buy line, the adjustments on the buy line, the adjustments on the free goods lines, and the associations between the buy line and the free goods line.

### **Freight Charges for the free goods line**

The pricing engine does not apply freight charges on the free goods line. To get freight changes for the free goods item, a second pricing engine call needs to be made, which will insert just the free goods line with `price_flag QP_PREQ_GRP.G_PHASE` and the pricing engine will only search for freight charges for the free goods line. The freight

charges do not affect the price on the free goods line. The reason the pricing engine cannot apply the freight charges (if any) on the free goods line is that the pricing engine does not have any attributes mapped for the free goods line to look for freight charges. The pricing engine inserts the free goods line with the pricing information, and the product is the only attribute inserted against this line. To get the freight charges for free goods lines, the calling application needs to make another implicit call to the pricing engine. The freight call needs to be made only when there are lines in `qp_preq_lines_tmp` with `process_status` `QP_PREQ_GRP.G_STATUS_NEW / G_STATUS_UPDATED`. In this case, the calling application can make this implicit call to pass all the free goods lines to the pricing engine. If any line group-based modifiers exist, the order or quote can qualify for a free goods line when the order or quote is repriced. For example, if there are two promotional goods modifiers, if the user adds A and then gets B free, then

- PRG1: buy A, get B free
- PRG2: buy A and B, get C free

When freight charges for B are calculated, the pricing engine could have qualified the order or quote for the free good C if A had also been passed. For this to happen, the calling application needs to pass all the lines on the quote or order in the implicit call to evaluate the freight charges. Make sure the `QP_UTIL_PUB.Get_Order_Line_Status` is called and pass all the lines only when the output of `Get_Order_Line_Status` passes `all_lines_flag = QP_PREQ_GRP.G_YES`; otherwise, pass the free good lines only. `Get_Order_Line_Status` passes `all_lines_flag = QP_PREQ_GRP.G_YES` if there are line group modifiers or other item discount modifiers. By passing all lines, you are making sure that the quote/order will qualify for all the applicable modifiers and that the price will not change on subsequent reprice.

## Other Item Discount

Other item discount (OID) is very similar to the free goods or promotional goods as explained previously except that the engine does not create a new order line. Other Item Discount is set up as buy item A and get x percent discount on item B where both item A and item B are ordered.

Therefore, to qualify for the OID, both item A and item B need to be on the request lines (lines with item A and item B need to be passed to the pricing engine). In this case, the pricing engine applies the OID modifier on the request line with item A, and it applies the discount on the line with item B.

The OID modifier can be found in the `qp_ldets_v` with `created_from_list_line_type = QP_PREQ_GRP.G_OTHER_ITEM_DISCOUNT` with `line_index` of the request line with item A which is the primary item on the OID modifier. The OID discount is inserted with `line_index` of request line having item B. The pricing engine also creates a relationship record between the OID modifier and the discount adjustment on the other line in `qp_preq_rltd_lines_tmp` with `relationship_type_code = QP_PREQ_GRP.G_GENERATED_LINE` with `line_detail_index` as the `line_detail_index` of the OID

modifier and the `related_line_detail_index` as the `line_detail_index` of the discount adjustment. All of this information needs to be stored in the transaction tables of the calling application.

## Pricing from Configurator

The configurator in Order Management (OM) and Order Capture (OC) uses OM and OC APIs to call the pricing engine. For included items and config items, the `unit_price` is zero by default. In case the pricing engine is called, the order lines with included items or config items are passed to the pricing engine with `price_flag` as *N* or *P*, depending upon the OM profile, OM: Charges for Included Item.

If the profile is set to *Y*, then freight charges need to be calculated for the included item. In this case, the `price_flag` is *P*, otherwise it is *N*. The pricing engine does not calculate the `unit_price` in either case.

## Service Item Pricing

Requests for pricing service items can be passed from integrating applications such as Oracle Order Management, Oracle Quoting, Oracle Istore, and Oracle Service Contracts.

If an item has a price list line set as a percentage of a price of the parent line (for example, the price of a service item called Gold Support is 10 percent of the price of a parent serviceable item called Platinum Services), then the parent line must be passed in `qp_preq_lines_tmp` and the relationship between the parent and the child must be passed in `qp_preq_rltd_lines_tmp`. To pass service relationships and other relationships to pricing, use the API `QP_PREQ_GRP.insert_rltd_lines2` to set the following parameters:

- `relationship_type Line_Index:= <parent line index>;`
- `Related_Line_Index:= <Child line>;`
- `Relationship_Type_Code:= QP_PREQ_GRP.G_SERVICE_LINE;`

If the item needs to be priced over a period of time, such as pricing a service item for a duration of 12 months, the duration must be passed as `uom_quantity` in the structure `qp_preq_lines_tmp`. If the service start and end dates are known, they can be passed as `contract_start_date/contract_end_date` in the structure `qp_preq_lines_tmp`.

The pricing engine retrieves a price list line for the service item and calculates the percent price based on the parent line's list price. Ideally, the parent line can belong to a different order/quote. The service line must be deleted or passed when the parent line is deleted or updated.

### Order Management Attributes passed to Pricing Engine

The following order management attributes are passed by the pricing engine application program interface (API) when pricing service items:

- **Ordered Quantity** (API: P\_Line\_Tbl.Line\_Quantity): The order quantity of the service item expressed in the in the serviceable item unit of measure.
- **Ordered UOM Code** (API: P\_Line\_Tbl.Line\_Uom\_Code): The unit of measure in the time scale.
- **Service Duration and Service Period** (API: P\_line\_Tbl.UOM\_Quantity): The duration of the service being ordered, for example, to order Computer Maintenance for one year, set Service Duration to 1 and Service Period to Year. These values are available in the Service tab when you enter the item in the order. In the API, P\_line\_Tbl.UOM\_Quantity is the service duration expressed in Ordered UOM Code service period.

### **Service Duration and Pricing**

When pricing service items, Oracle Advanced Pricing evaluates unit of measure (UOM) conversions to determine the service duration using the following criteria:

- If service dates are specified on the transaction being priced, Oracle Advanced Pricing calls Oracle Service Contracts to compute or convert service duration to pricing UOM. The Service Contracts' API is OKS\_OMINT\_PUB . get\_target\_quantity.
- If service dates are not supplied on the pricing transaction, Oracle Advanced Pricing uses unit of measure conversion setups in Oracle Inventory for converting service duration to pricing UOM.

For more information on time period conversions, see Oracle Service Contracts implementation and user documentation.

The following table compares how pricing evaluates the extended price when service duration is passed and when it is not. The following terms are used in the table:

- **Product:** Product that the service is covering; for example, a laptop model covered under the warranty.
- **Service Duration:** Duration of the service provided; for example, a One Year warranty or a Five Year warranty.
- **Extended Price:** unit selling price multiplied by quantity

--	Price List Found in Request UOM, NO UOM Conversion	UOM Conversion Required Between Price List UOM and Request UOM	Extended Price Formula <sup>1</sup>
Service duration is passed (Order Management, Quoting and Istore)	<ul style="list-style-type: none"> <li>Unit Price is for entire Service Duration</li> <li>Pricing Quantity same as Product Quantity</li> </ul>	<ul style="list-style-type: none"> <li>Unit Price same as on Price List</li> <li>Pricing Quantity is Product Quantity * Service Duration</li> </ul>	Unit Price * Pricing Quantity
Service duration is <i>not</i> passed (Service Contracts)	<ul style="list-style-type: none"> <li>Unit Price same as on Price List</li> <li>Pricing Quantity same as Service Duration, Product Quantity not included</li> </ul>	<ul style="list-style-type: none"> <li>Unit Price same as on Price List</li> <li>Pricing Quantity is Service Duration in Price List UOM, Product Quantity not included</li> </ul>	Unit Price * Pricing Quantity * Product Quantity

**Note:** <sup>1</sup> The application calling the price request (not the pricing engine) calculates the extended price.

### Example of Calculating Service Duration and Extended List Price for a Service Item

The following example shows how pricing calculates service duration to determine the extended price for a service item. The tables display these details: the price list setups, the inputs (such as the dates and service duration that pricing receives from the calling application), and the outputs that pricing calculates (such as the unit price). The following price list setup is used in the example:

- Price list 1 (PL1): \$120/Year
- Price list 2 (PL2): \$10/Month

**Note:** Note that the setups for PL1 and PL2 are equivalent: \$10/Month is equivalent to \$120/Year. The list price is expected to be calculated the same regardless of which price list is used.

The following table shows the price request inputs such as the service dates, UOM, and service duration that the calling application sends to pricing. Pricing evaluates these

inputs to calculate the extended selling price.

**Price Request Inputs**

Service Start Date	Service End Date	Service UOM	Service Duration	Product Quantity
01-JAN-2006	31-DEC-2007	Year	2	10

The following tables show how pricing calculates the extended list price, unit price, and pricing quantity (the output) when service duration is passed and when it is not. For both scenarios, the extended price is \$2400. (The extended price is calculated by the application calling Oracle Advanced Pricing, for example, Oracle Order Management or Oracle Service Contracts.)

**Service Dates and Service Duration are Passed**

--	Unit Price	Pricing Quantity	Extended Price <sup>1</sup> (Unit Price * Pricing Quantity)
Using PL1 (No UOM Conversion)	\$240 (list price * service duration: 120 * 2)	10 (same as product quantity)	240*10 = \$2400
Using PL2 (UOM Conversion)	\$10 (same as list price on price list)	240 (product quantity * service duration in PL UOM - Month: 10 * 12 * 2)	10*240 = \$2400

**Service Dates are Passed but not Service Duration**

--	Unit Price	Pricing Quantity	Extended Price <sup>1</sup> (Unit Price * Pricing Quantity * Product Quantity)
Using PL1 (No UOM Conversion)	\$120 (same as list price on price list)	2 (service duration in PL UOM - Year)	120*2*10 = \$2400

--	Unit Price	Pricing Quantity	Extended Price <sup>1</sup> (Unit Price * Pricing Quantity * Product Quantity)
Using PL2 (UOM Conversion)	\$10 (same as list price on price list)	24 (service duration in PL UOM – Month: 12 * 2)	10*24*10 = \$2400

<sup>1</sup>The calling application, not the pricing engine, calculates the extended price after the price request. Note that the extended price is the same in all scenarios.

**Note:** Service dates and duration are passed in the following situations:

- When no UOM Conversion occurs, service duration is considered in **unit price**:

$$\begin{aligned} \text{unit price} &= (\text{list price from price list}) * \text{service duration} \\ &= 120 * 2 = \$240 \end{aligned}$$

- When UOM Conversion occurs, service duration is considered in **pricing quantity**:

$$\begin{aligned} \text{pricing quantity} &= \text{product quantity} * \text{uom conversion factor} * \\ &\text{service duration} \\ &= 10 * 12 * 2 = 240. \end{aligned}$$

Service Dates are passed but Service Duration is not:

- Unit price is always the list price on the price list.
- Pricing quantity is always the service duration in PL UOM.

## Ask for promotions

The calling application can request that an ask for promotion be applied to the request line by passing the promotion/modifier as a qualifier to the request line. If the promotion is passed as a qualifier with context = MODLIST, attribute = QUALIFIER\_ATTRIBUTE1, value\_from = list\_header\_id of the ask for promotion, the pricing engine applies all the modifiers under the ask for promotion to the request line. The calling application can also ask for a specific modifier line to be applied in which case, the modifier line needs to be passed with context = MODLIST, attribute = QUALIFIER\_ATTRIBUTE2, value\_from = list\_line\_id of the modifier line that is asked for. This can be done by inserting the ask for promotion/modifier as an attribute in qp\_preq\_line\_attrs\_tmp as a part of append ask for procedure.

Remember that the pricing engine does not look at the qualifiers set up on the asked for promotion/modifier if the validated\_flag is passed as Y in the qualifier record in qp\_preq\_line\_attrs\_tmp. Make sure the validated\_flag is passed appropriately. Also, remember to store the asked for promotion into the transaction tables and pass them during every reprice call so that the pricing engine applies it consistently.

**Deleting ask for promotions that are applied:**

If the ask for promotion needs to be deleted, it needs to be deleted from the transaction table and a pricing engine call needs to be made so that the pricing engine does not apply it any more. Also, if the ask for promotion has a modifier that has been overridden, the pricing engine will not delete it. The calling application needs to delete it from the transaction table. When the asked for promotion is deleted, the adjustments with the list\_header\_id of the ask for promotion need to be deleted. In case the calling application asked for a specific modifier line, the adjustment with the list\_line\_id as the value\_from of the asked for modifier line needs to be deleted.

## **Evaluation of passed in modifier**

The calling application can request that a modifier be applied to the request line by passing the modifier line as a qualifier to the request line. If the modifier is passed as a qualifier with context = MODLIST, attribute = QUALIFIER\_ATTRIBUTE7, value\_from = list\_line\_id of the modifier, the pricing engine applies only the passed in modifier to the request line. Remember that the pricing engine consider the qualifiers of the passed in modifier.

This functionality is available for both the Pricing Search Engines - the Pattern Engine as well as the Traditional Engine.

For pattern engine you must set the profile QP: Pattern Search at Site level (as it is executed via API) and the QP: Pattern Upgrade concurrent program must be run before executing it.

**Deleting ask for promotions that are applied:**

If the ask for promotion needs to be deleted, it needs to be deleted from the transaction table and a pricing engine call needs to be made so that the pricing engine does not apply it any more. Also, if the ask for promotion has a modifier that has been overridden, the pricing engine will not delete it. The calling application needs to delete it from the transaction table. When the asked for promotion is deleted, the adjustments with the list\_line\_id of the ask for promotion need to be deleted. In case the calling application asked for a specific modifier line, the adjustment with the list\_line\_id as the value\_from of the asked for modifier line needs to be deleted.

## **Term Substitution**

If a request line qualifies for a term substitution (TSN) modifier, the pricing engine inserts a line detail record in qp\_ldets\_v with created\_from\_list\_line\_type = QP\_PREQ\_GRP.G\_TERMS\_SUBSTITUTION.

The following three types of terms upgrade are supported by pricing:

If `qp_ldets_v.substitution_attribute=QUALIFIER_ATTRIBUTE1`, `substitution_to` holds `payment_term_id`.

If `qp_ldets_v.substitution_attribute=QUALIFIER_ATTRIBUTE10`, `substitution_to` holds Freight Term Code.

`qp_ldets_v.substitution_attribute=QUALIFIER_ATTRIBUTE11`, `substitution_to` holds Shipment Method.

If a request line qualifies for a TSN modifier, based on the `substitution_attribute` which is the term type, the corresponding term (for example, payment/shipment/freight term on the order/quote/request) needs to be replaced by the value in the `qp_ldets_v.substitution_to`. Remember to populate the record structure with the old term before any repricing calls so that the old term is returned in the event of an attribute mapping rule. Alternatively, make sure the old term is passed to the pricing engine if no attribute mapping rules exist and if the attributes are passed directly.

Also, if the pricing engine does not pass the TSN modifier on subsequent reprice calls because the request line no longer qualifies for it, make sure to replace the old term before deleting the TSN modifier from the transaction table.

## Cross Order Volume Based Modifiers

Oracle Advanced Pricing provides seeded cross-order based volume modifiers that can be set up with pricing or qualifier attributes; for example, pricing attributes such as Period1 Item Quantity and Period1 Item Amount (pricing attributes). These attributes are operating unit specific and are seeded under the Order Fulfillment PTE. The cross order volume amount (such as Period 1 Item Amount) is calculated using list price and not selling price. Therefore, the calculated amounts would differ from order totals displayed in Order Management, which is based on selling price.

To find the cross order volume total for a given operating unit, you need to run the concurrent program Cross Order Volume Loader. This program reviews the Order Management (OM) tables for the order information, and populates the OM tables `OE_ITEM_CUST_VOLS_ALL` and `OE_ITEM_CUST_VOLS_ALL`. For more information, see *Oracle Advanced Pricing User's Guide, Reports and Concurrent Programs*.

**Note:** Modifier Level: When setting up a modifier with cross order volume attributes, select the Line modifier level rather than the Group of Lines modifier level. This is recommended because the cross order volume attribute value is 1) not based on the lines of the current order (it is based on previously-booked orders) and 2) the sum of lines across previous orders is already considered when using the cross order volume load attribute.

Recurring charges: Cross order volume loader excludes recurring

charges in its calculations. Only order lines with charge periodicity of NULL are included in this calculation.

Also, the seeded attribute mapping rules get the cross order volume total for specific attributes from the OM cross order volume tables. The pricing engine qualifies the request lines for modifiers based on the cross-order volume attribute value the build\_contexts API returns. If the calling application uses a new request\_type\_code or a different global structure, then the following steps are required to make the pricing engine process the cross order volume based discounts:

1. Write a concurrent program to populate the cross order volume total into the above tables or new tables for each cross order volume attribute used in the modifier setup.
2. Write attribute mapping rules to extract the cross order volume total for each cross order volume attribute used in modifier setup.

For more information about the cross order volume, see *Oracle Advanced Pricing User's Guide*, Cross Order Volume Load and Cross Order Volume Report.

## Promotional Limits

Promotional limits are set up against modifiers. When a pricing engine call is made, the pricing engine qualifies the request line for suitable modifiers. If limits are set up against those modifiers, then the modifiers applied are consumed from this limit. The pricing engine calls the limits engine, which maintains the limits balance details. To use the limits functionality, the user needs to pass a price\_request\_code to uniquely identify each request line.

For example, the price\_request\_code may be built by concatenating the request\_type\_code||'-'||header\_id||'-'||line\_id. The limits consumption is updated in the pricing limits tables for each request line and the price\_request\_code is used as the key to identify the request line. After the pricing engine call, make sure to process any errors related to limits. In case errors occur due to limits, the pricing engine populates error status in qp\_preq\_lines\_tmp.pricing\_status\_code for each request line. Make sure you handle the errors appropriately. In case the pricing engine call is to price an order, if the pricing engine throws an error status on one of the lines due to limits, then the order or that specific line may be placed on hold, or a warning message may be raised depending upon the business requirement.

### Limit Consumption

The limit consumption does not evaluate the manual modifiers or overridden automatic/manual modifiers and the out-of-phase modifiers. The out-of-phase modifiers are those applied in previous pricing events, which are in pricing phases that do not belong to the current pricing event. For example, an order line was entered and saved and the price was \$80 for that order line. A limit is defined for \$20 on a bucketed new price modifier that, for example, consumes only \$10. The limits API will apply the

newprice modifier with an adjustment amount so that it can consume only \$10 limit. If a modifier has been overridden in the previous bucket in such a way that the limit can consume the entire \$20, the limits engine will still not look at the overridden modifier and will allow only \$10 to be consumed. This issue is specific to limits defined on bucketed modifiers where there are out-of-phase or manual or overridden modifiers exist in the previous buckets.

The limit balances need to be updated when an order line is returned, amended or cancelled. If the calling application makes a pricing engine call, the pricing engine does this against the price\_request\_code passed. If no pricing engine call is made, pricing provides an API to do this. Ensure that you call the following API to update the limits consumption details:

```
QP_UTIL_PUB.Reverse_Limits (p_action_code           IN  VARCHAR2 ,
    p_cons_price_request_code IN  VARCHAR2 ,
    p_orig_ordered_qty       IN  NUMBER   DEFAULT NULL ,
    p_amended_qty           IN  NUMBER   DEFAULT NULL ,
    p_ret_price_request_code IN  VARCHAR2 DEFAULT NULL ,
    p_returned_qty          IN  NUMBER   DEFAULT NULL ,
    x_return_status         OUT VARCHAR2 ,
    x_return_message        OUT VARCHAR2 ) ;
```

For more information about this API, see: *Oracle Order Management Open Interfaces, API, & Electronic Messaging Guide*.

## Multi-Currency

In multi-currency functionality, the price list passed must exist in the currency passed and vice versa. In case the currency code passed is not a part of the multi-currency price list, then the pricing engine will not be able to find a price. In order to ensure this, pricing provides an API to validate the passed in currency and price list. This validation needs to be done in the integration code. The API looks like:

```
QP_UTIL_PUB.Validate_Price_list_Curr_code
(
    l_price_list_id           IN NUMBER
    l_currency_code          IN VARCHAR2
    l_pricing_effective_date  IN DATE
    l_validate_result        OUT VARCHAR2
);
```

For more information, see *Oracle Order Management Open Interfaces, API, & Electronic Messaging Guide*. The pricing engine will look for multi currency price lists if the profile QP: Multi-Currency Installed is set to Y to derive a price.

Also, pricing provides an API to return the rounded selling price or adjustment amount and uses the rounding factor based on the multi-currency price list and the order currency. This API can be used if there is a requirement to round the price apart from the rounding that the pricing engine does. Call the API,

```

QP_UTIL_PUB.Round_price(p_operand          => null
                        ,p_rounding_factor  => null
                        ,p_use_multi_currency => p_use_multi_currency
                        ,p_price_list_id     => p_price_list_id
                        ,p_currency_code    => p_currency_code
                        ,p_pricing_effective_date => p_pricing_effective_date
                        ,x_rounded_operand   => l_rounding_factor
                        ,x_status_code       => l_status_code
                        ,p_operand_type      => 'R'
);

```

If the price list or the currency changes on the order or quote that was priced earlier, remember to reprice the request because the pricing engine might return a different price based on the different currency.

Sample scripts are available to make pricing engine calls. The sample scripts, which are found in \$QP\_TOP/patch/115/sql, are

- `qp_engine_testing.sql`: Demonstrates making a pricing engine call by populating the price request pl/sql structure
- `qp_manual_adjustments.sql`: Demonstrates how to apply manual adjustments
- `qp_override_selling_price.sql`: Demonstrates overriding the selling price
- `qp_direct_insert.sql`: Demonstrates inserting price request information into pricing temporary tables for performance improvement.
- `qp_test_service.sql`: Demonstrates Service Item Pricing Setup
- `qp_test_oid.sql`: Demonstrates Other Item Discount Setup

### **Uptake Requirements for Multi-Currency functionality by other Oracle Applications**

In Oracle Advanced Pricing, the multi-currency price lists feature allows you to attach multiple currencies to the same price list or agreement. This reduces the volume of data processed and saves significant maintenance work for users. Additionally, you can set up different ways of deriving the conversion rate such as GL daily rate, fixed rate, user entered, and function call.

The following steps are used by the calling applications to support installations using multi-currency price lists:

1. Install Order Management Minipack-H or application release 11.5.8 or higher.
2. Set the Site level profile option QP: Multi-Currency Installed to Yes.
3. Run the concurrent program: Update Price Lists with Multi-Currency Conversion Criteria. After running the concurrent program, all price lists and agreement price lists are converted to multi-currency price lists.
4. If a user has converted to multi-currency price lists, applications calling the pricing engine must pass the control record variable `use_multi_currency` as Yes (Y) in

the pricing engine call for the currency conversion to occur. This variable is the deciding factor for the calling application to start using the Multi-currency functionality.

5. The calling application LOV (list of values) for the price list name at header and lines must be modified to show the multi-currency price lists and currencies.
  - When the user first enters the order currency and clicks the price list, the list of values displays only those price lists whose Currency Conversion's Currency-To is the same as order (transaction) currency. Also, the pricing effective date (if entered) on the sales order must be within the Currency-To effective dates. This is applicable for both the header and line level list of values. Pricing provides a view QP\_PRICELISTS\_LOV\_V for calling applications to display the list of values for price lists for the given transaction.
  - When the user first enters the price list and clicks the Currency, the list of values displays all the Currency-To in its Currency Conversion. Also, the pricing effective date (if entered) on the sales order (transaction) must be within the Currency-To effective dates. This is applicable for both the header and line level list of values. The calling application to call an API provided by pricing is called QP\_UTIL\_PUB.Get\_Currency.
  - The Process Order API currently validates the currency and price list passed to it. Now, the currency will have to be one of the Currency-To of the currency conversion criteria attached to this price list. The calling application needs to call the pricing API called QP\_UTIL\_PUB.Validate\_Price\_List\_Curr\_Code.
6. Uptake new rounding API for price list. For re-price processing, the calling application needs to call QP\_UTIL\_PUB.round\_price for price list rounding during re-price processing. This will use the Round To value to round the price.
7. For Conversion Type of Transaction, the calling application integration needs to pass the conversion rate and conversion type entered in the Sales Order header (if any) to the pricing engine.
8. The calling application integration needs to pass the functional currency to the pricing engine control record variable - function\_currency.

**Sample Code using Order Management Structure**

procedure copy\_Line\_to\_request(

```

p_Line_rec OE_Order_PUB.Line_Rec_Type
,px_req_line_tbl in out nocopy QP_PREQ_GRP.LINE_TBL_TYPE
,p_pricing_event varchar2
,p_Request_Type_Code varchar2
)
is
l_line_index pls_integer := nvl(px_req_line_tbl.count,0);
l_uom_rate NUMBER;
v_discounting_privilege VARCHAR2(30);
begin
l_line_index := l_line_index+1;
px_req_line_tbl(l_line_index).Line_id := p_Line_rec.line_id;
px_req_line_tbl(l_line_index).REQUEST_TYPE_CODE :=
p_Request_Type_Code;
px_req_line_tbl(l_line_index).LINE_INDEX := l_line_index;
px_req_line_tbl(l_line_index).LINE_TYPE_CODE := 'LINE';
If p_Line_rec.pricing_date is null or
p_Line_rec.pricing_date = fnd_api.g_miss_date then
px_req_line_tbl(l_line_index).PRICING_EFFECTIVE_DATE :=
trunc(sysdate);
Else
px_req_line_tbl(l_line_index).PRICING_EFFECTIVE_DATE :=
p_Line_rec.pricing_date;
End If;
px_req_line_tbl(l_line_index).LINE_QUANTITY :=
p_Line_rec.Ordered_quantity ;
px_req_line_tbl(l_line_index).LINE_UOM_CODE :=
p_Line_rec.Order_quantity_uom;
px_req_line_tbl(l_line_index).CURRENCY_CODE :=

OE_Order_PUB.g_hdr.transactional_curr_code;
If (p_Line_rec.service_period = p_Line_rec.Order_quantity_uom) Then
px_req_line_tbl(l_line_index).UOM_QUANTITY :=
p_Line_rec.service_duration;
Else
INV_CONVERT.INV_UM_CONVERSION(From_Unit =>
p_Line_rec.service_period
,To_Unit => p_Line_rec.Order_quantity_uom
,Item_ID => p_Line_rec.Inventory_item_id
,Uom_Rate => l_Uom_rate);
px_req_line_tbl(l_line_index).UOM_QUANTITY :=
p_Line_rec.service_duration * l_uom_rate;
End If;
px_req_line_tbl(l_line_index).Active_date_first_type := 'ORD';
px_req_line_tbl(l_line_index).Active_date_first :=
OE_Order_Pub.G_HDR.Ordered_date;
If p_Line_rec.schedule_ship_date is not null then
px_req_line_tbl(l_line_index).Active_date_Second_type := 'SHIP';
px_req_line_tbl(l_line_index).Active_date_Second :=
p_Line_rec.schedule_ship_date;
End If;
px_req_line_tbl(l_line_index).PRICE_FLAG :=
nvl(p_Line_rec.calculate_Price_flag,'Y');
end copy_Line_to_request;

```

### procedure copy\_attributes\_to\_Request

```
p_line_index number
,p_pricing_contexts_Tbl
QP_Attr_Mapping_PUB.Contexts_Result_Tbl_Type
,p_qualifier_contexts_Tbl QP_Attr_Mapping_PUB.Contexts_Result_Tbl_Type
,px_Req_line_attr_tbl in out nocopy
QP_PREQ_GRP.LINE_ATTR_TBL_TYPE
,px_Req_qual_tbl in out nocopy
QP_PREQ_GRP.QUAL_TBL_TYPE
)
is
i pls_integer := 0;
l_attr_index pls_integer := nvl(px_Req_line_attr_tbl.last,0);
l_qual_index pls_integer := nvl(px_Req_qual_tbl.last,0);
begin
i := p_pricing_contexts_Tbl.First;
While i is not null loop
l_attr_index := l_attr_index +1;
px_Req_line_attr_tbl(l_attr_index).VALIDATED_FLAG := 'N';
px_Req_line_attr_tbl(l_attr_index).line_index := p_line_index;
-- Product and Pricing Contexts go into pricing contexts...
px_Req_line_attr_tbl(l_attr_index).PRICING_CONTEXT
:=
p_pricing_contexts_Tbl(i).context_name;
px_Req_line_attr_tbl(l_attr_index).PRICING_ATTRIBUTE :=
p_pricing_contexts_Tbl(i).Attribute_Name;
px_Req_line_attr_tbl(l_attr_index).PRICING_ATTR_VALUE_FROM :=
p_pricing_contexts_Tbl(i).attribute_value;
i := p_pricing_contexts_Tbl.Next(i);
end loop;
-- Copy the qualifiers
i := p_qualifier_contexts_Tbl.First;
While i is not null loop
l_qual_index := l_qual_index +1;
If p_qualifier_contexts_Tbl(i).context_name = 'MODLIST' and
p_qualifier_contexts_Tbl(i).Attribute_Name
='QUALIFIER_ATTRIBUTE4' then
If OE_Order_PUB.G_Line.agreement_id is not null and
OE_Order_PUB.G_Line.agreement_id <>
fnd_api.g_miss_num then
px_Req_Qual_Tbl(l_qual_index).Validated_Flag := 'Y';
px_Req_Qual_Tbl(l_qual_index).Validated_Flag
:= 'N';
End If;
Else
px_Req_Qual_Tbl(l_qual_index).Validated_Flag := 'N';
End If;
px_Req_qual_tbl(l_qual_index).line_index := p_line_index;
px_Req_qual_tbl(l_qual_index).QUALIFIER_CONTEXT :=
p_qualifier_contexts_Tbl(i).context_name;
px_Req_qual_tbl(l_qual_index).QUALIFIER_ATTRIBUTE :=
p_qualifier_contexts_Tbl(i).Attribute_Name;
px_Req_qual_tbl(l_qual_index).QUALIFIER_ATTR_VALUE_FROM :=
p_qualifier_contexts_Tbl(i).attribute_value;
i := p_qualifier_contexts_Tbl.Next(i);
end loop;
end copy_attributes_to_Request;
procedure copy_Header_to_request(
p_header_rec OE_Order_PUB.Header_Rec_Type
,px_req_line_tbl in out nocopy QP_PREQ_GRP.LINE_TBL_TYPE
--,p_pricing_event varchar2
,p_Request_Type_Code varchar2
,p_calculate_price_flag varchar2
)
is
l_line_index pls_integer := px_req_line_tbl.count;
```

```

begin
l_line_index := l_line_index+1;
px_req_line_tbl(l_line_index).REQUEST_TYPE_CODE
:=p_Request_Type_Code;
--px_req_line_tbl(l_line_index).PRICING_EVENT :=p_pricing_event;
--px_req_line_tbl(l_line_index).LIST_LINE_LEVEL_CODE
:=p_Request_Type_Code;
px_req_line_tbl(l_line_index).LINE_INDEX := l_line_index;
px_req_line_tbl(l_line_index).LINE_TYPE_CODE := 'ORDER';
-- Hold the header_id in line_id for 'HEADER' Records
px_req_line_tbl(l_line_index).line_id := p_Header_rec.header_id;
if p_header_rec.pricing_date is null or
p_header_rec.pricing_date = fnd_api.g_miss_date then
ptrunc(sysdate);
x_req_line_tbl(l_line_index).PRICING_EFFECTIVE_DATE :=
Else
px_req_line_tbl(l_line_index).PRICING_EFFECTIVE_DATE :=
p_header_rec.pricing_date;
End If;
px_req_line_tbl(l_line_index).CURRENCY_CODE :=
p_Header_rec.transactional_curr_code;
px_req_line_tbl(l_line_index).PRICE_FLAG := p_calculate_price_flag;
px_req_line_tbl(l_line_index).Active_date_first_type := 'ORD';
px_req_line_tbl(l_line_index).Active_date_first :=
p_Header_rec.Ordered_date;
end copy_Header_to_request;

```

Refer to the sample script qp\_direct\_insert.sql in \$QP\_TOP/patch/115/sql to call the insert APIs of the pricing engine.

## Changed Lines API

This API permits only the modified and related lines to be passed to the pricing engine. This reduces unnecessary pricing processing and improves performance.

In typical pricing, there are multiple order lines or quote lines per order or quote. Calling applications such as Order Management and Order Capture that use Oracle Advanced Pricing for pricing their transactions, make pricing engine calls with one or more order lines in a single pricing request. Instead of passing all the order lines in every order, it is efficient to just pass the modified lines (newly entered lines or existing lines with update) to the pricing engine so that the pricing engine will not have to process the unnecessary lines that will result in the same prices anywhere.

The calling applications make calls to QP\_UTIL\_PUB.Get order line status to determine what to pass. The output record of this API has three flags:

### **All\_Lines\_Flag (Yes or No)**

This flag tells the calling applications whether all lines should be passed to the engine. The value of this flag is determined by the pricing setup. For these modifiers, all lines information is required in order for the pricing engine to evaluate the eligibility of discounts.

### **Changed\_Lines\_Flag (Yes or No)**

When this flag is set to 'Y', only the modified lines should be passed to the engine. This flag is 'Y' when there are only Line Level Modifiers.

### Summary\_Line\_Flag (Yes or No):

The value of this flag depends on the modifier setup at the order level. For these modifiers, this flag will be set to Y (Yes) so that only the Order Summary Line is passed to the engine.

In addition to the Changed Lines API, the QP\_ATTR\_MAPPING\_PUB.Build\_contexts API can be used with the signature Build\_Contexts:

```
(      p_request_type_code,          p_line_index,
  p_check_line_flag,
  p_pricing_event,
  p_pricing_type_code,
  p_org_id,
  x_pass_line)
```

If the changed lines API passed All\_lines\_flag as Y (Yes), you can use the build\_contexts API for attribute mapping. For the actual lines that have changed, the p\_check\_line\_flag may be passed as N (No) and for the remaining lines that did not change, the p\_check\_line\_flag should be passed as Y (Yes). For the lines for which p\_check\_line\_flag is passed as Y, this API will check if there are any active advanced line group or other item discount type modifier or promotional goods modifiers with additional buy products defined based on the product attributes sourced for that line.

If such modifiers exist, the Build contexts API will source all the product/pricing/qualifier attributes and return the x\_pass\_line as Y to indicate that the caller needs to pass this line to the pricing engine because it can affect the price on the remaining lines on the order/quote although the line did not undergo any changes. If such active modifiers are not found, the attributes are not sourced for the line and the x\_pass\_line is returned as No so that the caller need not pass this line to the pricing engine to avoid performance overhead.

## Scenarios

### Case 1

One hundred-line order with five changed lines. Setup: All\_Lines\_Flag = 'N', Changed\_Lines\_Flag = 'Y', Summary\_Line\_Flag = 'N' Result: Only five lines are passed to the pricing engine.

### Case 2

One hundred-line order with all lines changed.

Setup: All\_Lines\_Flag = 'Y', Changed\_Lines\_Flag = 'Y/N', Summary\_Line\_Flag = 'N'

Result: All 100 lines are passed to the pricing engine.

### Case 3

One hundred-line order with five changed lines and either Other Item, Promotional Goods, or Group of Lines discount.

Setup: All\_Lines\_Flag = 'Y', Changed\_Lines\_Flag = 'Y/N', Summary\_Line\_Flag = 'N'

Result: All lines are passed to the pricing engine.

**Case 4**

One hundred-line order with five changed lines and an order level modifier.

Setup: All\_Lines\_Flag = 'N', Changed\_Lines\_Flag = 'Y', Summary\_Line\_Flag = 'Y'

Result: Five lines and a summary line are passed to the pricing engine.

**Case 5**

One-hundred-line order with no changed lines.

Setup: All\_Lines\_Flag = 'N', Changed\_Lines\_Flag = 'N', Summary\_Line\_Flag = 'N'

Result: No lines are passed to the pricing engine.

**Case 6**

Two new lines are added to a one hundred-line order.

Setup: All\_Lines\_Flag = 'N', Changed\_Lines\_Flag = 'Y', Summary\_Line\_Flag = 'N'

Result: Only the two new lines are passed to the pricing engine.

## Oracle Service Contracts (OKS) Integration: Proration and Price List Locking

This section describes the usage proration and price list locking features available when Oracle Service Contracts (OKS) is integrated with Oracle Advanced Pricing (QP). The OKS API is used for calculating service duration and determining start and end dates. Usage proration is available to any calling application making usage calls. Price list locking is available only for use by OKS. Price list locking refers to:

- The creation of a copy (locked price list) of the specified source price list with a different price list name and without effective dates (this enables OKS to control the effective dates on the associated Service Contract).
- The creation of a copy (locked price list line) of the source price list line (belonging to the source price list) as a child of the locked price list. The locked price list line is created without any effective dates to enable OKS to control price effectivity. Further, a pricing attribute (with Context = QP Internal, Attribute = List Line Id and value = <list\_line\_id of the newly created locked price list line) is created for the locked list line so that the locked price list line is available to the caller of the pricing engine only when this list\_line\_id pricing attribute is sourced appropriately.

For example, if Oracle Service Contracts wants to lock the price list line with list\_line\_id = 201 on the Corporate price list, the following occurs:

The locked price list will be created with the name OKS LOCKED Corporate. The naming convention followed is:

```
<Source System Code> || 'LOCKED' || <Source Price List>.
```

The locked list line is copied from the price list line with list\_line\_id = 201 and will be created under the locked price list OKS LOCKED Corporate. The locked line with, for example, list\_line\_id = 301, will have a pricing attribute with the following setup:

- Context = QP Internal
- Attribute = List Line Id
- Value = 301

**Note:** The Locked price list is created only once for a given source price list and source system code. For subsequent lock requests the new locked price list lines are created under the already existing locked price list.

### Locking Price Lists and Price List Lines that are Already Locked

In previous releases, you had to create a locked price list and price list line from the user interface (UI). However, in the current release, an API is called programmatically from the OKS code to lock the price list and price list line without your intervention. You can now also lock an already locked price list and line.

For example, suppose you want to lock an already locked price list that is named: `<Source System Code> || 'LOCKED' || <Source Price List Name>`. However, since the Source Price List name already has already been assigned a "locked" name, for example, `<Source System Code> || 'LOCKED' prefix` then a new name is automatically assigned to the locked price list using the following format `<Source System Code> || ' LOCKED' || <version no> || ' ' <Original Source Price List Name>` where the version number starts from 2. For example, if the source price list is named Corporate, then the locked price list will be named QP LOCKED Corporate. However, if the source price list name is QP LOCKED Corporate, then the locked price list name will be QP LOCKED2 Corporate.

### Price Line Locking

You can also lock an already locked price list line. Price line locking occurs when the price list line of the source price list is copied (now the locked price list line) as a child of the locked price list. For example, suppose Oracle Service Contracts (OKS) wants to lock the locked price list line with `list_line_id = 201` on the OKS LOCKED Corporate price list. Since the list line is already locked, it will have a pricing attribute with `context = 'QP Internal'`, `attribute = 'List Line Id'` and `value = 201`.

Then the following steps occur: the new locked price list line will be a copy of the price list line with `list_line_id = 201` and created under the locked price list OKS LOCKED2 Corporate price list. The new locked line with, for example, `list_line_id = 301`, will have a new pricing attribute with `context = 'QP Internal'`, `attribute = 'List Line Id'` and `value = 301` attached to it (in addition to the other pricing attributes copied over from the source list line with `list_line_id 201`).

However, the pricing attribute with `context = 'QP Internal'`, `attribute = 'List Line Id'` and `value = 201` that is attached to the source list line with `list_line_id = 201` is not copied over from the source to the newly locked list line having `list_line_id = 301`

**Note:** The Locked price list is created only once for a given source price list and source system code. For subsequent lock requests, the new locked price list lines are created under the already existing locked price list.

### Effective Dates for Copied Price List

No effective dates are copied to the locked price list and price list lines--instead the effective dates can be controlled by OKS using the effectivity dates on the associated Service Contract.

## Integration Flow for Price List Locking

The following steps outline the process flow for price list locking:

1. The application Oracle Service Contracts (OKS) makes an authoring call to the pricing engine with an authoring UOM (unit of measure) which can be an item uom such as Quarter, Month, Year.
2. The pricing engine returns the price list line, price breaks (no price break calculation as quantity not passed).
3. OKS displays the price list line/breaks (for example, locked breaks) to OKS users in the OKS user interface, enabling updates. Product should be same for the price list line as returned by the pricing engine in step 2.

4. OKS launches the price list window to support creation of locked price list, locked price list line and its breaks. It passes the following related values to window parameters to create and query a new price list with name = 'OKS ' || 'LOCKED ' || <Source Price list name>:

LOCK\_MODE (= 'Y')

SOURCE\_PRICE\_LIST\_ID

SOURCE\_LIST\_LINE\_ID

ORIG\_SYS\_HEADER\_REF (= Contract Number)

STARTUP\_MODE (= OKS),

STARTUP\_MODE = OKS

START\_DATE\_ACTIVE and END DATE ACTIVE columns set to null

5. A pricing attribute is created for the locked PBH (price break header) line with the value equal to its list\_line\_id. This enables the pricing engine to select this specific line. The effective dates (START\_DATE\_ACTIVE and END\_DATE\_ACTIVE) on the locked PBH price list line are set to null.

6. OKS stores the foreign key of the price list line id, and passes the price list id while calling pricing engine at the time of billing.
7. For further updates to the locked price breaks, OKS will launch the price list form with appropriate parameters (LOCK\_MODE='N', LOCKED\_PRICE\_LIST\_ID, LOCKED\_LIST\_LINE\_ID, STARTUP\_MODE = 'OKS') to query the locked price list and locked list line specified.
8. OKS calls the pricing engine at the time of billing and provides the UOM for proration. The assumption is that pricing engine returns the exact same price.
9. The price list created by OKS is prevented from being updated from pricing window if LIST\_SOURCE\_CODE = 'OKS' and orig\_system\_header\_ref is not null.
10. A price list with locked price breaks can be deleted using Purge API. Only inactivated price lists should be purged. OKS will use the pricing public API to inactivate the price list.
11. During billing call, OKS will pass contract id and price list as qualifiers. OKS will also pass price list line id as pricing attribute.

#### **Example of Integration Flow for Price List Locking**

The following steps outline the integration flow for price list locking when Oracle Service Contracts (OKS) is integrated with Oracle Advanced Pricing (QP).

1. Define the contract header and select a price list name (example Corporate).
2. Enter a contract line, then select an item.

A pricing engine call is made with the price list at the contract header (for example, Corporate). The pricing engine returns the price break information, and displays the information in the Contracts window.

3. To lock the price for the item, click the Lock button.
4. At this point, OKS populates the following window parameters:
  - Startup\_mode (OKS)
  - Lock\_mode (Y/N)
  - Source\_price\_list\_id (if Lock\_mode='Y')
  - source\_list\_line\_id (if Lock\_mode='Y')
  - locked\_price\_list\_id (if Lock\_mode = 'N')
  - locked\_list\_line\_id (if Lock\_mode = 'N').

Then it calls Price List setup window.

5. The pricing application then completes the following actions:
  1. Checks if this call is from OKS (if Startup\_Mode = 'OKS').
  2. If yes, then it checks if Lock\_mode = Y. If no, then it goes to the Update step.
  3. If yes, then it checks if a record exists in qp\_list\_headers\_b where the:
    4. Source\_system\_code = profile option value of QP: Source System Code
    5. Locked\_from\_list\_header\_id = Source\_price\_list\_id
  6. If the record does not exist, that means a new price list needs to be created. Hence, it populates the record structure for the new price list. Ensure that the source system code is OKS and the Start Date Active and End Date Active columns are null.
  7. If the record exists, that means a new line needs to be created for the locked price list, and you need to populate the line structures.
  8. Copy the PBH line, its attributes and price break child lines. Ensure that the:
    - Start Date Active and End Date Active columns for the copied (locked) PBH line record are set to null
    - list\_line\_id of the newly created PBH line is used to add a pricing attribute to the PBH line. Post the records.
  9. Query the newly inserted/locked price list.
  10. In the pre-query of the list lines block, set the "where" clause to query the newly created/locked PBH line.
  11. Navigate to the List lines tab. Click the Price Breaks tab to update the price breaks.
  12. Set the global variables (set in the .pld files) to pass back the last created/modified price\_list\_id, list\_line\_id to OKS.
  13. In the post-query, clear the "where" clause to its original status.
  14. If the mode is Update then perform steps from step g).
6. If you are querying an existing PBH line which is already locked, then OKS needs to make a pricing engine call with List\_line\_id as a pricing attribute. To do this,

navigate to the Service Contracts Authoring window.

7. To delete or unlock, OKS needs to directly call the Pricing API to delete the price list line.
8. Oracle Advanced Pricing (QP) should change seed data to default QP: Source System Code profile to OKS when application is OKS. Also, add OKS to Order Fulfillment PTE.
9. QP needs to seed List\_Line\_Id as a pricing attribute. Do not use the PRICING ATTRIBUTE' pricing context for this pricing attribute. Since a generic pricing context is needed to assign the above pricing attribute, a new pricing context called 'QP INTERNAL' will also be created/seeded.

The prefix <Source System Code> || ' LOCKED ' || <Source Price List Name> should be the name of the locked price list; for example, 'OKS LOCKED Corporate'.

### Changes Related to the Price List Locking feature

The following parameters are added to the Price List window. They are not visible to the end-user but are required for integration with the Price List Locking feature.

Parameter	Datatype	Meaning/Values
LOCK_MODE	CHAR(1)	Valid values are Y/N. Indicates if Lock Mode is Yes or No. Appropriate value to be passed by OKS while launching the price list window.
LOCKED_PRICE_LIST_ID	NUMBER	If Lock_Mode = N then list_header_id of a locked price list to be passed by OKS while launching price list window.
LOCKED_LIST_LINE_ID	NUMBER	If Lock_Mode = N then list_line_id of a locked price list line to be passed by OKS while launching price list window.
SOURCE_PRICE_LIST_ID	NUMBER	If Lock_Mode = Y then list_header_id of source price list to be passed by OKS while launching price list window.

Parameter	Datatype	Meaning/Values
SOURCE_LIST_LINE_ID	NUMBER	If Lock_Mode = Y then list_line_id of a source list line to be passed by OKS while launching price list window.

The following global variables pass information back to OKS:

Parameter	Meaning/Values
LOCKED_PRICE_LIST_ID	Has the LIST_HEADER_ID of the last locked price list created or modified in the price list window launched by OKS.
LOCKED_LIST_LINE_ID	Has the LIST_LINE_ID of the last Locked list line created or modified in the price list window launched by OKS.

The following changes are also related to the price list locking feature:

- New Source System Code 'OKS' under the PTE code 'ORDFUL'.
- New Request Type Code 'OKS' under PTE code 'ORDFUL'
- New Lookup Code for 'OKS' under SOURCE\_SYSTEM lookup type
- New Lookup Code for 'OKS' under REQUEST\_TYPE lookup type
- Set the default Value for the System Profile Option 'QP: Source System Code' to 'OKS' at the Application level for the OKS Application.
- New Pricing Context 'QP Internal' (Code = QP\_INTERNAL)
- New Pricing Attribute 'List Line Id' (Code = 'LIST\_LINE\_ID', Column = 'PRICING\_ATTRIBUTE1') under the Pricing Context 'QP Internal'.
- New Pricing Attribute linked to the 'ORDFUL' PTE.

## Integration Flow for Proration

Oracle Service Contracts (OKS) supports prorated billing for Oracle Advanced Pricing customers. The following steps outline the changes required to the calling application and the behavior of the pricing engine:

1. The calling application submits a billing call to the pricing engine with a Volume-Quantity (for example, 500) and (usage\_break\_UOM) such as MONTH. If the calling application does not pass usage\_break\_UOM attribute then the conversion is not needed and the engine proceeds with normal price break evaluation.
2. The pricing engine evaluates the following:
  - PBH (price break header) line
  - Break UOM context (BREAK\_UOM)
  - Break\_UOM\_attribute (USAGE\_UOM)

If Break UOM is not set up then the engine proceeds with normal break evaluation.
3. Pricing calls the Contracts API for unit-of-measure conversion. The service\_start\_date and end\_date from qp\_preq\_lines\_temp as well as setup Break UOM and passed\_Break\_UOM is passed to the Contracts API.
4. The API returns the conversion factor for proration. If the contracts profile is not set, then the standard UOM conversion results. For example, if the conversion = 1/3, then the Break From/To values will be converted as follows:

Original From/To	Modified From	Modified To
0 - 1000	0	333.333.....3
1000 - 2000	333.333...3	666.666...6
2000 - 3000	666.666...6	1000

**Note:** The values shown in the table will not be truncated.

5. The pricing engine evaluates the preceding Break From/To, and returns the setup Break UOM to the calling application.
6. OKS needs to pass the following values to the pricing engine to calculate the proration: Context = BreakUOM, Attribute = Pricing\_Attribute1, Value = Billing UOM.

**Note:** New continuous prorated price breaks that are billed based on fixed or actual usage may have a different monetary value than pre-R12

prorated price breaks. Prorated continuous price break tiers will no longer be "floored" in calculation (that is, set to previous whole number).

### **Changes Related to the Proration feature**

The Price List window displays two additional columns for entering price breaks in the List Lines tab:

- The first column defines the Volume Break UOM.
- The second column defines the attribute in which the calling application is going to pass the Volume Usage UOM. For OKS, this will be hard-coded to context the BreakUOM attribute "Pricing\_attribute1" (Usage UOM).
- User sets up the break, for example, 0-1000 BreakUOM = QTR.

To display these columns, the profile option QP: Break UOM Proration Allowed, must be set to Yes. The valid values are Yes or No. This can be set at both the Site and Application levels.

### **Duration and Partial Period Pricing of Service Items**

Duration defines the time period for a particular service. Partial period pricing is required to change the price when the contract duration changes. For example, your customer, ABC Applications Software (currently, on a 2 year service program), wants to update their service contract by adding another 10 months to their Extended Notebook PC Service Program.

You can use Oracle Advanced Pricing with Oracle Service Contracts to calculate duration and partial period pricing. With Service Contracts, the service duration of a contract is calculated from the contract start date/end date by calling the Service Contracts conversion routines (with/without UOM Conversion) to derive line quantity.

**Note:** The inventory conversion routine is used if dates are not passed.

To ensure the consistency of partial period pricing across all calling applications, Oracle Advanced Pricing calculates partial period pricing for a services line depending on whether the uom quantity or contract start date is passed to the pricing engine:

- If uom quantity/duration is passed and contract dates are passed, Advanced Pricing will ignore the passed dates and use the passed duration to compute the unit price.
- If the pricing engine is not able to find the price in the passed service UOM (line\_uom\_code) and when a UOM conversion is needed, Advanced Pricing will call the new OKS conversion API, OKS\_OMINT\_PUB.get\_target\_duration, to compute the duration in price list UOM and derive the unit price for the duration. When the duration is passed, Advanced Pricing will calculate the unit price for the

entire service duration and price breaks will be evaluated based on line quantity (parent quantity)

- If uom quantity/duration is passed and dates are not passed, Advanced Pricing will use the passed duration to compute the unit price for the entire service duration and price breaks will be evaluated based on line quantity. If the line\_uom\_code and the price list UOM do not match, the pricing engine will use the Inventory conversion API to compute the duration.

## Examples of Usage Proration with Oracle Service Contracts (OKS)

The following are some examples of usage proration with Oracle Service Contracts (OKS).

### Example 1

Profile QP: Break UOM Proration Allowed = Y

Set up a price break price list line for an item, for example, AS999:

- Break UOM = QTR (Quarter)
- Break Type = POINT
- Break lines for Volume Attribute Item Quantity:
  - 0 - 1000, Value = 100
  - 1000 - 2000, Value = 90
  - 2000 - 3000, Value = 80

Call pricing engine for the same item AS999 with quantity 500, usage\_pricing\_type = REGULAR passing the Price List as qualifier and also with pricing context = BREAK\_UOM, attribute = PRICING\_ATTRIBUTE1 and value = MTH (Month)

Expected Results:

The unit\_price returned from pricing engine should be 90 (for line quantity 500) based on following usage proration for price break child lines:

- 0-333.333.....3, Value = 100
- 333.333.....3-666.666...6, Value = 90
- 666.666...6-1000, Value = 80

### Example 2

Profile QP: Break UOM Proration Allowed = Y

Set up a price break price list line for an item, for example, AS999:

- Break UOM = QTR (Quarter)

- Break Type = RANGE
- Break lines for Volume Attribute Item Quantity:
  - 0–1000, Value = 100
  - 1000–2000, Value = 90
  - 2000–3000, Value = 80

Call pricing engine for the same item AS999 with quantity 910, usage\_pricing\_type = REGULAR passing the Price List as qualifier and also with pricing context = BREAK\_UOM, attribute = PRICING\_ATTRIBUTE1 and value = MTH (Month)

**Expected Results:**

The unit\_price returned from the pricing engine should be 90.98 (for line quantity 910) based on following usage proration for price break child lines:

- 0–333.333.....3, Value = 100
- 333.333.....3–666.666...6, Value = 90
- 666.666...6–1000, Value = 80

**Example 3**

Profile QP: Break UOM Proration Allowed = Y

Set up a price break price list line for an item, for example, AS999:

- Break UOM = QTR (Quarter)
- Break Type = POINT
- Break Lines for Volume Attribute Item Quantity:
  - 0 - 1000, Value = 100
  - 1000 - 2000, Value = 90

Call the pricing engine for the same item AS999 with quantity 500, usage\_pricing\_type <> REGULAR passing the price list as qualifier and also with the following:

- Pricing Context = BREAK\_UOM
- Attribute = PRICING\_ATTRIBUTE1
- Value = MTH (Month)

**Expected Results:**

The unit\_price returned from pricing engine should be 100 (for line quantity 500) with no proration as usage\_pricing\_type passed is not REGULAR.

#### **Example 4**

Profile QP: Break UOM Proration Allowed = Y

Set up a price break price list line for an item, for example, AS999:

- Break UOM = QTR (Quarter)
- Break Type = POINT
- Break Lines for Volume Attribute Item Quantity:
  - 0 - 1000, Value = 100
  - 1000 - 2000, Value = 90

Call the pricing engine for the same item AS999 with quantity 500, usage\_pricing\_type = REGULAR passing the Price List as qualifier but without passing the following pricing attributes for the line:

- Pricing Context = BREAK\_UOM
- Attribute = PRICING\_ATTRIBUTE1
- Value = MTH (Month)

#### **Expected Results:**

The unit\_price returned from pricing engine should be 100 (for line quantity 500) with no proration as Break UOM attribute is not passed to the engine.

#### **Example 5**

Profile QP: Break UOM Proration Allowed = Y

Set up a price break price list line for an item, for example, AS999:

- Break UOM = null
- Break Type = POINT
- Break Lines for Volume Attribute Item Quantity:
  - 0 - 1000, Value = 100
  - 1000 - 2000, Value = 90

Call the pricing engine for the same item AS999 with quantity 500, usage\_pricing\_type = REGULAR, passing the price list as qualifier and the following:

- Pricing Context = BREAK\_UOM
- Attribute = PRICING\_ATTRIBUTE1
- Value = MTH (Month)

**Expected Results:**

The `unit_price` returned from pricing engine should be 100 (for line quantity 500) with no proration as Break UOM is null for the price break header line.

## Pricing Features to Support Telecommunications Industry Flows [Oracle Telecommunications Service Ordering (TSO)]

Some services, such as wireless phone services, charge their customers a fixed recurring service charge for certain services; for example, a monthly charge for a Local or Long Distance Calling Plan or other services such as Caller ID or Call Waiting. The billing frequency of these charges, such as every month or quarter, is the *charge periodicity*. Service providers may choose to offer customers the same service in different bundles (plans) with each plan having different charge periodicities. For example, a wireless phone company may offer the same phone service with either a monthly price or a quarterly price.

These recurring charges are typically set up when the customer signs up for the service. When a customer orders an item with a recurring charge, the periodicity such as a month or year must be specified on the order, quote, or "shopping cart" line to get the correct price for the service.

**Note:** Order Amount qualifier value and Cross Order Volume computations aggregate only one-time charge lines; recurring charge lines are excluded.

**Charge Periodicity attribute:** Select this seeded context/attribute when setting up charge periodicities for recurring charges for either a price list line or modifier line.

**Note:** Charge Periodicity values are not seeded in pricing. The list of values for Charge Periodicity pricing attribute is based on the unit of measure (UOM) class registered in the profile OM: Charge Periodicity UOM Class. This UOM class can have units of measure such as MONTH, QUARTER, and YEAR.

### Related Topics

For more information on setting up recurring charges for telecommunications services, see the *Oracle Telecommunications Service Ordering Process Guide*.

---

## High Volume Order Processing

This chapter covers the following topics:

- Overview of High Volume Order Processing

### Overview of High Volume Order Processing

High Volume Order Processing (HVOP) enables you to import large volume orders. HVOP supports both Basic Pricing and Oracle Advanced Pricing.

In Order Management, HVOP is optimized when used in pricing setups with basic modifiers. However, HVOP is not optimized if any of the following exist in the pricing setup:

- Coupon Issue modifier
- Item Upgrade modifier
- Promotional Goods modifier
- Promotional limits
- Terms Substitution modifier

To see if your pricing setup is compliant with the optimized pricing path, check the profile option, QP: High Volume Order Processing Compliance for the following conditions:

- If no active modifiers of the preceding type are found in the pricing setup, this profile will have a value of Yes, which indicates that HVOP takes the optimized path which improves the performance.
- If there are active modifiers of the above types, HVOP is still supported, but the pricing is not optimized. If any unused advanced modifier is made Inactive, the profile will get updated automatically, and pricing will use the optimized path.

To update the QP: High Volume Order Processing Compliance profile option, run the concurrent program QP: Maintain Denormalised data in QP Qualifiers with "Update High Volume Order Processing profile" selected as the Update Type. Then review the profile option to confirm if the pricing setup is compliant with the optimized pricing path.

## Attribute Mapping Rules Restrictions

HVOP uses the same attribute mapping rules (including the custom mapping rules used by Order Import or Sales Order pad) to source attributes to the pricing engine.

However, one restriction applies to the custom attribute mapping rules: any references to the OM global record structure inside the API must be removed and instead, must be passed as input parameters to the API. The Build Sourcing API automatically maps these references to the global record structure elements to the appropriate memory structure elements.

Also, in the optimized pricing path, the order lines are not posted to the database before pricing. So any mapping rules that review the OE tables to derive custom attributes need to look at the memory structures during the HVOP path. For more information, see the QP\_SOURCING\_API\_PUB.Get\_Order\_Amount API in the Integration Repository or sample implementation for details.

**Note:** The Oracle Integration Repository is a compilation of information about the service endpoints displayed by the Oracle E-Business Suite of applications. It provides a complete catalog of Oracle E-Business Suite business service interfaces. The tool enables you to easily discover and deploy the appropriate business service interface for integration with any system, application, or business partner. The Oracle Integration Repository is shipped as part of Oracle E-Business Suite.

If the custom mapping rules which reference the OE global record structure elements in the custom API or if custom mapping logic looking at the OE tables are not changed, the attributes will not be sourced correctly.

Until they are changed, the HVOP profile may be manually set to No to price using the non-optimized path, so that the attributes are sourced correctly. Once the mapping rules are fixed, the HVOP concurrent program should be run to reset the HVOP profile to use the optimized pricing path.

## Related Topics

*Order Management User's Guide*

*Oracle Order Management Implementation Manual*

---

## Technical Considerations

This chapter covers the following topics:

- Basic versus Oracle Advanced Pricing
- Oracle Advanced Pricing Engine Processing
- Extensibility Features

### Basic versus Oracle Advanced Pricing

Oracle Advanced Pricing and Oracle Order Management, with its basic pricing component, share a common code set. Licensed users of Oracle Order Management are authorized to use the basic features of the set, while licensed users of Oracle Advanced Pricing are authorized to use the full set of features.

The following SQL statement can be issued from within SQL\*Plus to obtain the installation status for Oracle Advanced Pricing: `SQL: select qp_util.get_qp_status from dual;`

The following table describes the values of `get_qp_status`:

---

Value	Meaning
S	Shared installation. Only basic pricing features are available.
I	Full installation. All Oracle Advanced Pricing features are available.

---

When pricing functionality is installed SHARED, the setup forms contain restricted function. This is accomplished using several methods, including:

- Restricting the list of values a user may select from. For example, the list type code shows only discount, surcharge, and freight list types.

- Disabling fields. Fields that are not available to basic pricing users appear grey in the window, or do not display at all.
- Defaulting fields not available to basic pricing users. For example, the field incompatibility group automatically defaults to LVL 1: Level 1 Incompatibility when using the basic feature set.
- Modifying regions and tabs. For example, Oracle Advanced Pricing modifier tabs (coupons, promotional upgrade) are not shown on the modifier window when Oracle Advanced Pricing is not installed.

The pricing engine and pricing setup APIs are authorized for use only if Oracle Advanced Pricing is installed.

**Note:** Although the pricing engine is a common component of both basic and Oracle Advanced Pricing, it does not execute the API `Get_qp_status`. The pricing engine returns user setups as allowed by the APIs and setup forms.

## Architectural Overview

Oracle Advanced Pricing significantly increases the pricing functionality of Oracle Order Management. Both Oracle Advanced Pricing and basic pricing are designed to:

- Support the complexities of pricing while providing an intuitive user interface.
- Accommodate new inventions in promotion management and e-business pricing.
- Provide flexibility that meets the individual pricing needs of customers in different industries.

### Architectural Changes and Innovations

Some of the major architectural features and innovations of Oracle Advanced Pricing include:

- A new data model: all internal pricing data, including price lists, discounts, and pricing rules are kept in a common set of tables internal to Oracle Advanced Pricing. This data model includes a variety of constructs necessary to support Advanced e-business needs.
- Oracle Advanced Pricing integrates with Oracle Order Management and Oracle Customer Relationship Management products such as Oracle iStore. The pricing engine is called on-demand, passing appropriate parameters at the time of call. These parameters are passed in a pricing request structure, which may be a single or a multi-line call. Parameter and results passing are performed strictly via APIs.
- Oracle Advanced Pricing receives a pricing request structure when invoked from a

calling application. This incoming pricing request references internal tables based on the parameters passed at invocation, and returns an answer set in a defined API.

- Multilingual Support (MLS) is provided for price lists, modifiers, and formula tables.

## Oracle Advanced Pricing Engine Processing

The Oracle Advanced Pricing Engine is composed of a search engine and a calculation engine.

### Search Engine

The search engine uses qualifier and pricing attributes to select a list of price and modifier list lines. These may be applied to the pricing request according to rules of eligibility, incompatibility, exclusivity, and precedence.

### Determining Eligibility

The search engine selects lists and list lines based on effectivity dates. It compares the pricing effective date supplied by the calling application against the following dates to determine if a list or list line is effective for the given pricing date:

- Price Lists
  - Price list: effective date range
  - Price list line: start and end date
  - Price list qualifiers: start and end date
- Modifier Lists
  - Modifier list: effective date range
  - List qualifiers: start and end date
  - Modifier list line: start and end date
  - Line qualifiers: start and end date
- Formulas
  - Pricing formula: effective date range
- Modifier List Additional Dates Ranges

Modifier lists have two date ranges and a date type for each range. This allows additional date eligibility to be defined on a modifier list. Seeded values for date type are:

- Order date
- Requested ship date

For example, a summer promotion is available between June 1 and August 31. To receive this promotion the customer must order before July 31. Effective dates of the promotion are June 1 through August 31, with an additional date range defined by the order date type that has an end date of July 31.

**Note:** If both of the above date ranges are defined for a modifier list, the criteria of both must be met. For example, promotion XYZ has an order date type with a range of December 1 through December 31. It has a second date type of requested ship date with a range of December 15 through December 31. Both the order date and requested ship date must fall within the range to receive the promotion. If you wish to make this an OR condition, a qualifier should be used.

### **Active/Inactive Lists**

To be selected by the search engine, a list must be marked as Active. If the active flag is set to no (inactive), the search engine does not consider the list, even if it is effective for the pricing date.

### **Pricing Currency**

For single currency price lists, the search engine matches the currency of the transaction to the currency of the price and/or modifier lists. Only lists defined in the same currency as the transaction are selected. The currency of the transaction must be included in the pricing request structure.

To price transactions in multiple currencies, set up a price list and modifier list for each currency for which a transaction may be priced. Formula functionality enables you to convert the base currency price and modifier lists to the transaction currency, passing the transaction currency as a formula pricing attribute.

*For Multiple Currency price lists:* The search engine matches the currency of the transaction to the currency of the price list(s) with the base or conversion currencies matching this currency. The pricing engine converts the price from the base currency and calculates the transaction currency based on the established conversion rules. The search engine matches the currency of the transaction to the currency of the modifier lists. Only modifiers in the same currency as the transaction are selected. The currency of the transaction must be included in the pricing request structure.

To price transactions in multiple currencies, set up a Multiple Currency price list with

attached multiple currency conversions, and a modifier list for each currency where a transaction may be priced.

**Identifying Appropriate Transaction Pricing Data**

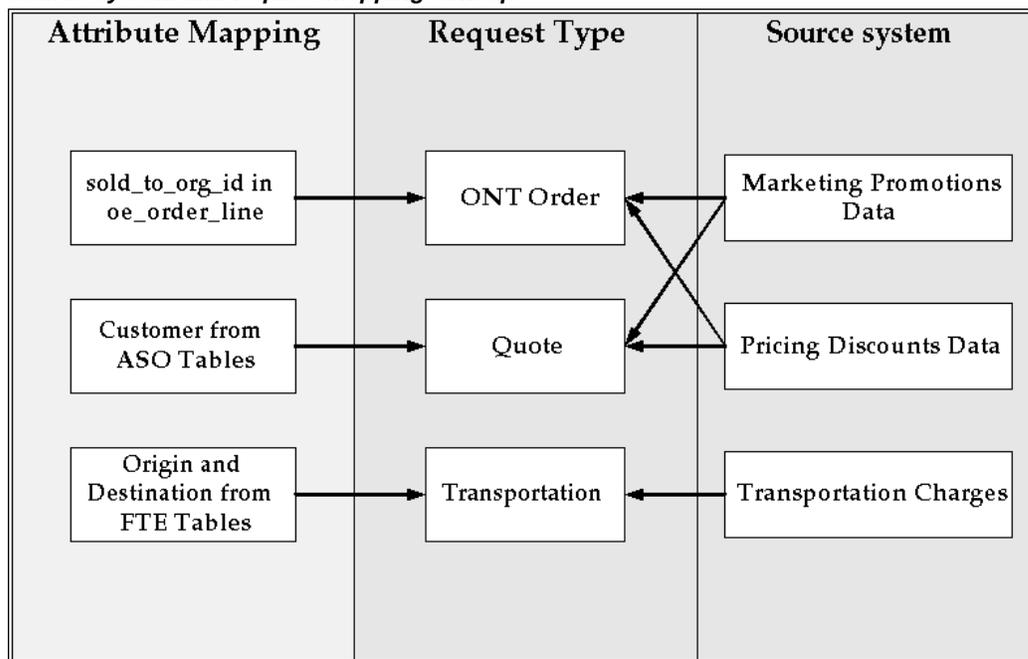
The search engine must know the data source to determine whether it is appropriate to price a transaction. For example, only price lists set up to price contracts should be considered when deriving a price for a contract line.

A source system identifier, which identifies the application that created the pricing data, is recorded on all price and modifier lists. Examples of this include Oracle Order Management, Oracle Marketing, and Oracle Service Contracts.

The request type identifies the type of transaction that is being priced. Each pricing request line must be identified with a request type. Examples of this include Oracle Order Management and Oracle Order Capture.

The mapping of a request type to one or more source systems defines the source(s) of pricing data that are used to price a transaction. For example, if a pricing request line has a request type of Order, the search engine considers only data from a source system that is mapped to this type of request. The following figure illustrates the source system and request mapping concept.

**Source system and request mapping concept**



**Phasing the pricing of a transaction**

A user may configure a pricing transaction, when the source system process flow requires it to occur, through pricing phases and pricing events. Pricing phases and events also define the pricing data that should be considered for application to the request at that point in the transaction process flow. Rather than pricing an entire

transaction at once, you may break up pricing into discrete activities.

A pricing event occurs when a specific point in the process flow of the sourcing application requires service by the pricing engine. Pricing events in the source system trigger base price determination for the transaction, certain transaction lines, and price adjustments, benefits, or charges to be applied to the whole transaction or to specific transaction lines.

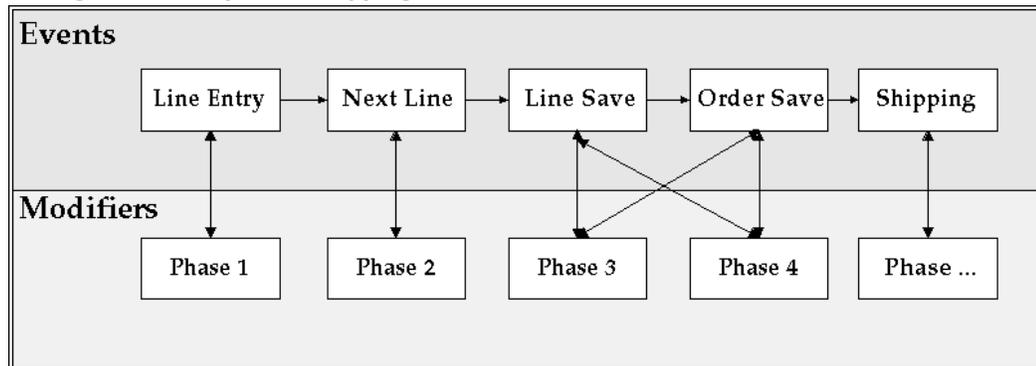
A pricing phase controls which modifiers are considered by the search engine and in what sequence they should be applied to the request. Some pricing phase attributes determine which modifiers can be placed in a phase. For example, if the list type of the pricing phase is charges, only list lines of this type may be placed in the phase. Phase attributes that are used to control the modifiers placed in the phase are:

- Modifier level code
- List type code
- List line type code

**Note:** All price lists are automatically placed in the seeded Phase 0 list line base price and cannot be assigned to any other phase.

A pricing event can be mapped to one or more pricing phases, and a pricing phase can be assigned to more than one pricing event. The following figure illustrates this how pricing events and phases can be mapped.

**Pricing events and phases mapping**



The concept illustrated in the previous figure allows the following pricing business rules types to be implemented:

- All freight and special charges should be calculated at the time of shipping.
- Once total order volumes have been derived in a high volume batch environment, all cross order volume discounts are applied at the end of the day.

- Coupons are given only after all items in the shopping cart have been priced and the user has proceeded to final checkout.

### Selecting Price Lists and Modifier Lists based on Qualifiers

A qualifier is a user-defined rule that specifies one or more conditions under which the pricing engine applies a price list, price adjustment, or other benefits or charges. If a condition described in a qualifier evaluates as true when the search engine is processing, then the pricing object(s) tied to that rule are considered eligible by the search engine.

Many companies apply prices and discounts across and at different levels of their customer hierarchy. Each level at which businesses set their pricing and discounting can be conditionally tested in a qualifier.

When creating a price list, discount, or promotion, pre-defined qualifiers can be used to define who is eligible for the price or modifier.

Qualifier attributes are combined with operators to create qualifying conditions for a list header or modifier. The following operators are available:

- =
- Between (includes  $\geq$  and  $\leq$ )
- Not =

Qualifying conditions such as the following may be used to define eligibility for prices, promotions, etc.

- Customer = XYZ
- Sales territory = Northeastern Region
- Order amount  $>$  \$100
- Contract signed date between December 1 and December 31

When pricing a transaction line, the search engine compares the qualifying conditions on the list header with the value of the qualifiers on the pricing request line. Qualifiers for a pricing request line are populated using dimension sourcing. For example, if eligibility for a price list is qualified by Market Sector = Consumer Products, and if the value for the market sector qualifier is consumer products, then the price list is available to price that transaction line. If the qualifiers on the list header are met for modifier lists, the search engine also ensures that any qualifiers at the line level are satisfied.

Qualifiers may be grouped to create AND/OR conditions using a grouping number. For example, if a 10 percent discount is given if a customer is a preferred customer *and* the customer spends more than \$150, the qualifying condition may be defined by creating qualifiers in the same qualifier group:

Qualifier Group	Qualifier Attribute	Operator	Value From	Value To
1	Customer class	=	Preferred	Not applicable
1	Order amount	Between	150	9999999999

If the requirement is to give a 10 percent discount if a customer is a preferred customer or the customer spends more than \$150, the qualifying condition may be defined by creating qualifiers in different qualifier groups:

Qualifier Group	Qualifier Attribute	Operator	Value From	Value To
1	Customer class	=	Preferred	Not applicable
2	Order amount	Between	150	9999999999

Multiple qualifiers may be included in a qualifier group.

If a qualifier is mandatory for all qualifying conditions and must be included in all qualifier groups, a -1 qualifier group number can be given. The search engine always ensures that this condition is met before proceeding to check all other groups. For example, the conditions to receive a 10 percent discount on an order are that a customer is a preferred customer OR the customer must spend more than \$150 AND must place the order on a United Kingdom website, and the customer must always pay with a Visa card. This is defined as follows:

Qualifier Group	Qualifier Attribute	Operator	Value From	Value To
-1	Credit card type	=	Visa	Not applicable
1	Customer class	=	Preferred	Not applicable
2	Order amount	Between	150	9999999999
2	Website domain	=	.co.uk	Not applicable

#### Pricing Attributes: Selecting What is Priced

Pricing attributes are user-defined attributes that define what is being priced or modified. Attributes may include factors that affect the price of the item, provide

additional definition without the need to create an item, or manage pricing and discounting at a level higher than in the product hierarchy.

Examples of pricing attributes are:

- Item X:
  1. Grade A is priced at \$60
  2. Grade B is priced at \$55
  3. Grade C is priced at \$50
  
- Servicing of a photocopier is priced based on:
  1. Distance from the service center
  2. Age of the photocopier
  3. Environment
  
- Training course discounts are given based on the number of attendees:
  1. 1 - 5 students, 10 percent discount
  2. 6 - 10 students, 12 percent discount
  3. 10 or more students, 15 percent discount
  
- All contact lenses are priced at \$2.25 per pair.

Pricing attributes are defined under contexts and attributes. Creating pricing attributes in different contexts allows attributes to be grouped according to their business use. The context of Item is reserved for defining product pricing hierarchy; every level in the product hierarchy at which pricing and discounting is set should be defined as a segment in this context.

---

Pricing category:	Milled goods
Pricing sub-category:	Flour
Margin group:	Branded flour
Margin sub group:	Homepride
Product group:	White

---

---

Size	1.0kg
Configuration	12*2
Stock keeping unit:	SKU1234

---

### **Product Pricing Hierarchy Example**

In the product pricing hierarchy example given, each of the eight levels are defined as attributes in the item context in the pricing context defined in contexts and attributes.

The attributes are ordered to reflect the structure of the product pricing hierarchy, with the lowest level in the hierarchy having the lowest sequence number. The search engine uses the sequence to select the most specific price or discount. For example, if no price is found for a promotion product, use the configuration price. If no price is found for the configuration, use the pack size price. If no price is found for the pack size, use the flour type. The item context of the pricing attribute represents a flattened view of the product pricing hierarchy.

As with qualifiers, the search engine compares pricing attributes on the transaction or pricing request line with pricing attributes on the price or modifier list line. This determines if the base price or modifier can be applied to the request line. Product and pricing attributes for a pricing request line can be either user entered or populated using attribute mapping.

Exclusion enables a price adjustment, benefit, or charge to be given at a level in the product hierarchy but it excludes lower levels in the hierarchy from that modifier. In the product pricing hierarchy defined previously, if a 10 percent discount is given on all flour apart from the Homepride brand, then a discount can be created with a product attribute of Oracle Advanced Pricing Subcategory = Flour, excluding Margin Sub Group = Homepride. The search engine eliminates any pricing request lines with the margin sub group pricing attribute of Homepride.

The search engine returns only those modifiers that are defined in the same UOM as pricing (which is determined when the base price is derived as described) or where there is no product UOM is specified. The latter may be the case if the modifier is not product specific.

### **UOM Conversion Logic**

The search engine only returns modifiers which are defined in the same unit of measure as the pricing unit of measure, or when no unit of measure specified. The latter may be the case if the modifier is not product specific, or if the type of modifier does not require a unit of measure.

### **Modifier Level Code**

Modifier level code determines which qualifiers and pricing attributes are considered by the search engine when deciding if a request line qualifies for a modifier. This code also determines at what level a modifier should be applied to the request.

### Line Level Code

Only qualifiers and pricing attributes of an individual request line are considered by the search engine. For volume related pricing attributes only the quantity of the request line is considered for qualification. Modifier application is at the request level.

### Group of Lines Level Code

The quantity in the pricing UOM and amount spent on an item is summed across all qualified request lines. The total item quantity and amount on the request or total quantity and amount at a level in the product hierarchy is considered by the search engine when deciding whether a modifier is qualified. Modifier application is at the request line level.

### Order Level Code

Only qualifiers or pricing attributes of the summary request line or header are considered by the search engine when deciding whether a modifier is qualified. It is not possible for a header level modifier to be qualified by a request line. Modifier application is at the summary request line/header level.

### Freezing Pricing Request Lines

The Calculate Price flag allows the calling application to fully or partially freeze the price on a pricing request line. Price may be completely frozen with no additional modifiers applied, or additional modifiers may be applied in certain phases, depending on the value of the flag. Possible values are as follows:

Flag Value	Action
Y (calculate price)	Applies all prices and modifiers to the request line.
N (freeze price)	Does not apply any prices or modifiers to the request line.
P (partial price)	Only applies prices or modifiers in certain phases.

When the calculate price flag is set to partial price, the search engine observes the freeze override flag on the phase. When the calculate price flag is set to yes the search engine applies eligible modifiers in the phase to the request line. When the calculate price flag is set to no the modifiers in the phase are not considered for application to the request line. This enables, for example, the price of a line to be fixed but still allows freight charges to be applied to the line.

### Other Qualifying Items

Some types of modifiers allow multiple buy items to be specified as a qualification for the benefit or charge. For example:

- If you buy a set of six chairs and a coffee table, or two standard lamps, get \$400 off a dining table.

Using this example, the search engine determines whether the pricing request contains all qualifying items in the specified quantity or amount before the modifier is applied to the pricing request line for the benefit item (the \$400 discount on the dining table). The modifier must always contain a primary item, which can be combined with groups of other items. In the previous example, chair is the primary item combined with two other qualifying items, coffee table and standard lamp. The OR condition is achieved by giving additional buy items different grouping numbers. If the condition were six chairs, a coffee table, and two standard lamps, the grouping numbers would have been the same.

**Note:** The only pricing attributes that can be used with the additional qualifying items are those in the volume context.

Modifiers that allow the definition of additional qualifying items are:

- Other Item Discount
- Promotional Goods
- Coupon Issue

## Resolving Incompatibility and Exclusivity Between Modifiers

Once the search engine locates all modifiers eligible for application to a pricing request line, it must be determined whether the modifiers are exclusive or incompatible.

Additional modifiers may not be applied to a pricing request line if an exclusive modifier is applied to the request line in the same pricing phase. For example, a customer receives a 15 percent discount on an item and is not eligible for any other discount on the item.

An incompatible modifier may not be applied to the same pricing request line as any other incompatible modifier within a pricing phase. For example, a 5 percent discount on a tennis racquet may not be given if the customer is also eligible for "Summer Sports Promotion" that gives a 6 percent discount on all sporting goods.

Incompatibility between discounts is defined between modifiers at the same level in a discount application hierarchy. This following image illustrates this concept. In the image, Level 1 is base level discounts, Level 2 is specially negotiated discounts, and Level 3 is retrospective discounts/accruals.

If the discounts of Levels 1 through 4 in the following image are set for the same product family (and therefore applied to the same pricing request line) a customer entitled to receive all of these discounts only receives one discount from each incompatibility group. For example, the customer could be eligible for a 5 percent discount, a 1 percent special discount, and a \$100 off the invoice.

If the pricing phase is Across All Phases, then the system applies only this modifier even though there are other eligible modifiers. The pricing engine calculates the price based on the Across All Phases modifier line and does not apply any other modifier. When you select this pricing phase, the Incompatibility Code field is automatically populated with the value Exclusive and gets disabled.

***Incompatibility Groups for a Pricing Phase***

<b>Pricing Phase</b>	
Level 1 Incompatibility Group	5% discount 6% promotion
Level 2 Incompatibility Group	\$200 advertising allowance \$100 off invoice
Level 3 Incompatibility Group	1% special discount
Level N	

If the customer negotiates a 15 percent exclusive discount for the same product family and the search engine determines that this discount can be applied to the pricing request line, then only this discount is applied to the line in the current pricing phase. None of the discounts in Levels 1 through N is considered. This concept is illustrated in the following image.

**Incompatibility Groups and Exclusivity rules**

Pricing Phase	
Level 1 Incompatibility Group	5% discount 6% promotion
Level 2 Incompatibility Group	Advertising allowance \$100 off invoice
Level 3 Incompatibility Group	1% special discount
Level 4 Incompatibility Group	

**Exclusive**  
15% negotiated discount

**Note:** Incompatibility and exclusivity rules only apply to modifiers that are in the same pricing phase and that are eligible to be applied to the same pricing request line.

**Price List Lines**

Price list lines are always treated as exclusive; all price lists lines are automatically assigned to EXCL: Exclusive Incompatibility Group. When multiple prices are found for the same pricing request line in the ordered UOM or in the pricing UOM the search engine selects the price list line using precedence resolution. If the price list lines are of equal precedence the search engine returns an error for the request line.

The following image depicts price list search/incompatibility processing, part A. It contains diamond boxes, which represent decisions, and rectangular boxes, which represent processes.

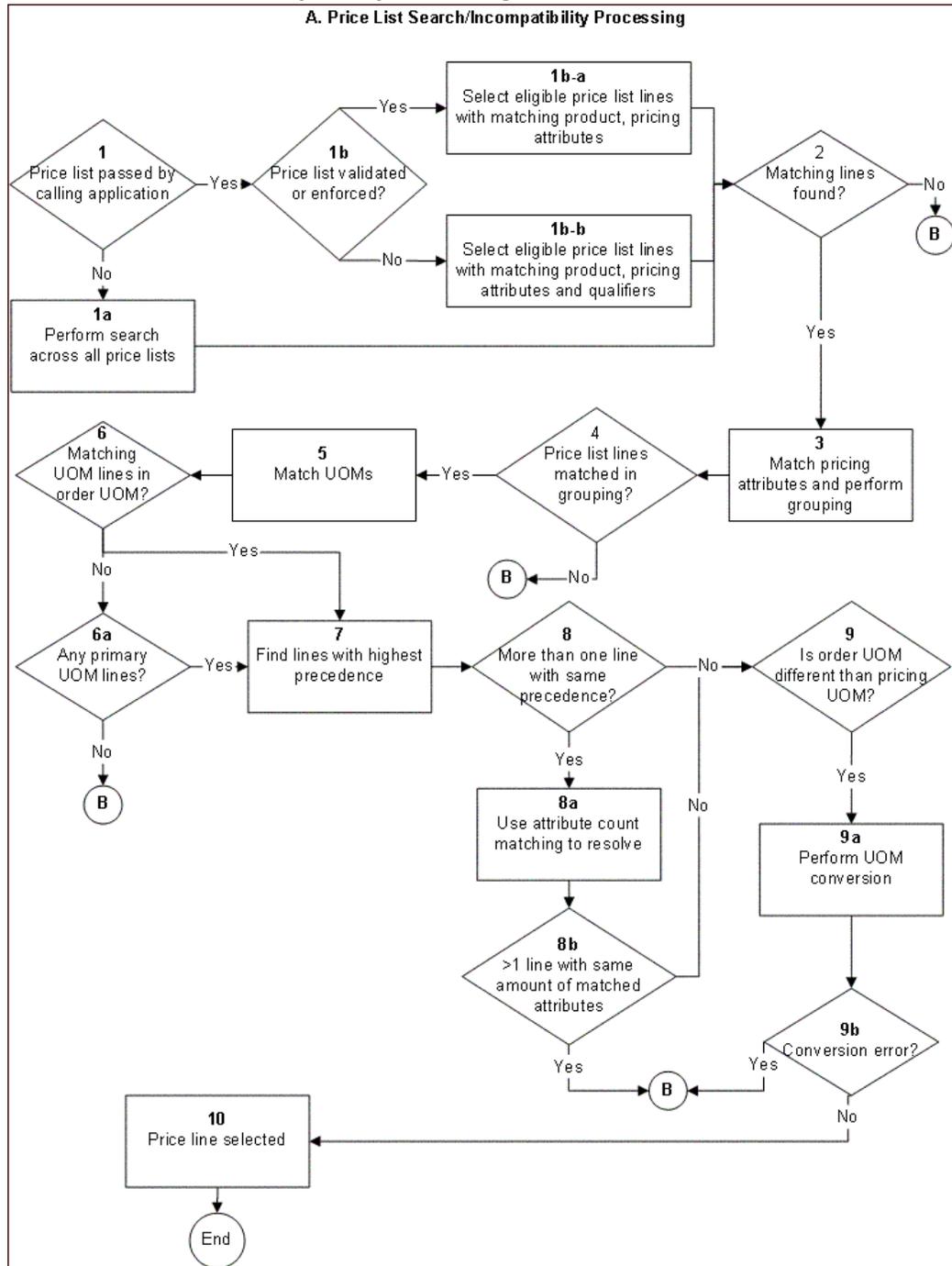
Box 1 asks whether the price list is passed by the calling application:

- Yes leads to Box 1a: Perform search across all price lists. Box 1a then progresses to Box 2.
- A no result for Box 1 leads to Box 1b, which asks whether the price list has been validated/enforced. Yes leads to Box 1b-a: Select eligible price list lines with matching product and pricing attributes. No leads to Box 1b-b: Select eligible price list lines with matching qualifiers, product and pricing attributes. Both of these boxes lead to Box 2.

Box 2 asks if matching lines are found. *No* leads to part B. *Yes* leads to Box 3: Match

pricing attributes and perform grouping. Proceed to Box 4, which asks whether any price list lines are matched in grouping. *No* leads to part B. *Yes* leads to Box 5: Match UOMs. Proceed to Box 6, which asks if matching UOM lines exist in the order UOM. *Yes* leads Box 7. *No* leads to Box 6a, which asks whether any primary UOM lines exist. *No* leads to part B. *Yes* leads to Box 7: Find the line with the highest precedence. Proceed to Box 8. Box 8 asks if more than one line has the same precedence. *Yes* leads to Box 8a: Use attribute count matching to resolve the same precedence issue. Proceed to Box 8b, which asks if more than one line has the same number of matched attributes. *Yes* leads to part B; *No* leads to Box 9. A *No* result to Box 8 leads to Box 9, which asks whether the order UOM is different than the pricing UOM. *No* leads to Box 10. *Yes* leads to Box 9a: Perform UOM conversion. Proceed to Box 9b, which asks if a conversion error occurred. *Yes* leads to part B. If *No*, proceed to Box 10: Price list line selected. This ends the process.

### Price List Search and Incompatibility Processing



The following image depicts price list search and incompatibility processing, part B. It contains diamond boxes, which represent decisions, and rectangular boxes, which represent processes.

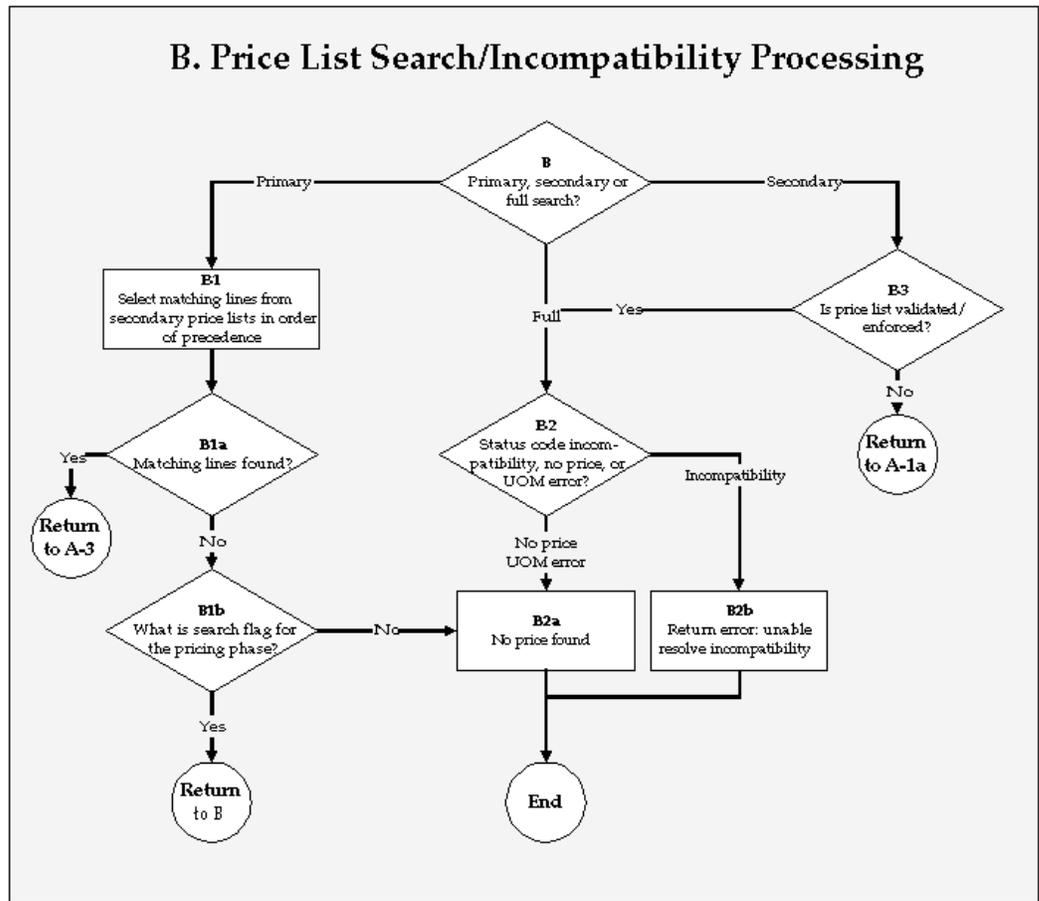
Box B asks if the search is primary, secondary or full.

If the search is primary, proceed to Box B1: Select matching lines from secondary price lists in order of precedence. Proceed to Box B1a, which asks if any matching lines are found. If *Yes*, return to part A, Box 3. If *No*, proceed to Box B1b, which asks if the search flag for the pricing phase is *Yes* or *No*. If *Yes*, return to Box B. If *No*, Proceed to Box B2a: No price found.

If the search is secondary, proceed to Box B3, which asks whether the price list is validated and enforced. If *No*, return to part A, Box 1a. If *Yes*, proceed to Box B2.

If the search is full, proceed to Box B2, which asks if the status code is incompatibility, nor price, or UOM error. No price and UOM error lead to Box 2a. Incompatibility leads to Box B2b: Return error: unable to resolve incompatibility. Boxes B2a and B2b end the process.

**Price List search and incompatibility processing**



**Modifiers**

When multiple modifiers in an incompatibility group are eligible for application to a pricing request line, the search engine must determine which modifier should be applied. The search engine uses two methods to determine this: precedence and best price. Which method the engine chooses depends on the incompatibility resolution

method which is set for the pricing phase. If the incompatibility resolution code for the phase is precedence, then the search engine selects the most specific modifier. If this fails (the modifiers have equal precedence), then the search engine uses best price resolution. If the incompatibility resolution method on the phase is best price, the search engine attempts to find the modifier that gives the greatest discount. If this fails, the search engine returns an error for the request line.

The following image shows modifiers search and incompatibility processing. It contains diamond boxes, which represent decisions, and rectangular boxes, which represent processes.

Box 1 asks whether any modifier/header line is passed as certified asked for. *Yes* leads to Box 1a: Select modifiers for matching products without qualification. *No* leads to Box 1b: Select modifiers by matching qualifiers, products, and attributes. Both boxes lead to Box 2: Perform price break evaluation. Proceed to Box 3: Perform line group processing. Proceed to Box 4: Perform qualifier group and attributes group matching.

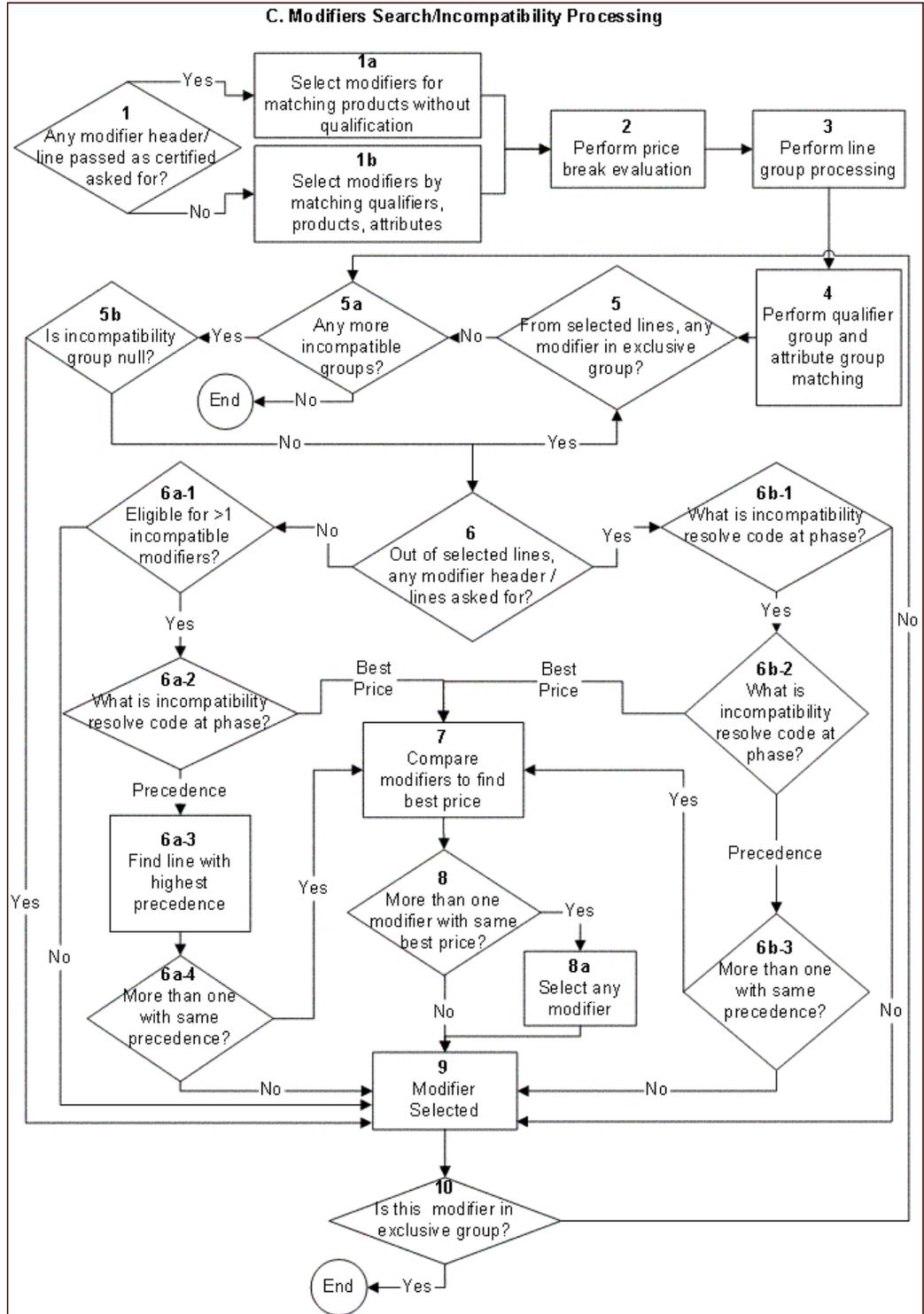
Box 4 leads to Box 5, which asks whether any modifiers of the selected lines are in an exclusive group. *Yes* leads to Box 6. *No* leads to Box 5a, which asks if there are any more incompatible groups. If not, then the process is complete. *Yes* leads to Box 5b which asks if the incompatibility group is null. *Yes* leads to Box 9. *No* leads to Box 6.

Box 6 asks whether any of the modifiers in the selected lines are asked for. Each answer leads to a different case. *No* leads to the first case, which begins with Box 6a-1. This box asks whether any of the selected lines are eligible for more than one incompatible modifiers. *No* leads to Box 9. *Yes* leads to Box 6a-2, which asks for the incompatibility resolve at phase. Best price leads to Box 7. Precedence leads to Box 6a-3: Find lines with highest precedence. Proceed to Box 6a-4, which asks if more than one line has the same precedence. *Yes* leads to Box 7; *No* leads to Box 9.

A *Yes* result to Box 6 leads to the second case, which begins with Box 6b-1. This box asks whether the lines are eligible for more than one asked for in the same incompatibility group. *No* leads to Box 9. *Yes* leads to Box 6b-2 which asks for the incompatibility resolve code at phase. Best price leads to Box 7. Precedence leads to Box 6b-3. which asks whether more than one line has the same precedence. *Yes* leads to Box 7; *No* leads to Box 9.

Box 7: Compare modifiers to find best price leads to Box 8, which asks if more than one modifier has the same best price. *No* leads to Box 9. *Yes* leads to Box 8a: Select any modifier. Proceed to Box 9: Modifier selected. Proceed to Box 10, which asks if this modifier is in the exclusive group. *No* leads back to Box 5a; *Yes* ends the process.

**Modifier search and incompatibility processing**



## Precedence and Specificity

The search engine always selects the modifier that is the most specific; the modifier with the lowest precedence number is selected.

For example, all priority customers get a 5 percent discount on Product Family A. Customer Boomerang Emporium, although a priority customer, has negotiated a 10 percent discount on the same product family and is therefore not eligible to receive the 5 percent discount.

Both discounts are defined in the same incompatibility group and phase. The search engine determines that both are eligible to be applied to a Boomerang Emporium order for an item in Product Family A. Because the discounts are incompatible and both can not be applied to the order line, the search engine must determine which discount is more specific. In this case, the Boomerang Emporium negotiated discount is more specific than the general customer class discount. The search engine derives modifier specificity by selecting the lowest precedence number from each of the qualifiers and any product attributes on the modifier. The search engine then applies the modifier with the lowest precedence to the pricing request line. In the example shown, if the precedence of the customer qualifier is 1, the precedence of the customer class qualifier is 2, and product family has a precedence of 3, then the specificity would be calculated as follows:

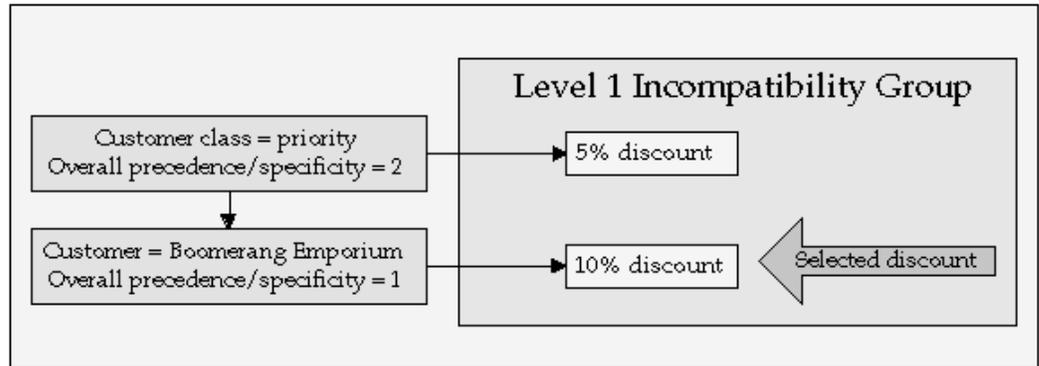
---

Type of Discount	Qualifier Precedence	Pricing Attribute Precedence	Overall Precedence/ Specificity
5 percent discount	2 (customer class)	3	2
10 percent Boomerang Emporium discount	1 (customer)	3	1

---

The search engine selects the 10 percent Boomerang Emporium discount because it has the lowest precedence (the most specific discount). This concept is shown in the following image.

### ***Precedence and incompatibility***



At time of setup, the qualifier or product attribute precedence defaults from the value defined in attribute management. The sequence of the qualifier and product attributes in and across contexts determine which qualifiers have priority when the selection engine is forced to choose between multiple prices, incompatible benefits, or charges that are eligible for application to request line. The sequence number of each segment should be unique across the qualifier descriptive contexts, the item context, and the most specific qualifiers and product attributes having the lower sequence numbers.

#### **Best Price**

Best price is an alternative method of precedence for selecting which modifier should be applied to the pricing request line when multiple incompatible modifiers are eligible. Using this method, the modifier that gives the greatest discount to the customer is selected. For more information, see Best Price Resolution for Modifiers, page 16-3.

## **Calculation Engine**

The calculation engine takes a pricing request line and its associated pricing request line details to calculate the base price, adjusted price, and/or the extended price.

#### **Fixed Minimum Price (GSA Pricing)**

Oracle Advanced Pricing enables you to set a minimum price below which the item price cannot be discounted for all, or a subset of, customers. This setting is commonly used to manage General Service Administration (GSA) contracts, which must ensure that commercial customers do not receive discounts equal to or greater than those fixed for customers on GSA contracts. GSA Violation is checked if the customer is a GSA violation or if the invoice location is GSA. GSA Violation is checked if the customer is a GSA violation or if the invoice location is GSA.

The minimum or GSA price for an item is set on a special GSA minimum price list; a discount list with the minimum price specified as a new (fixed) price type of discount. A GSA discount is created when the regular item price is reduced to the GSA price for a GSA customer, enabling the discount cost to be recognized. The GSA discount list is available only to GSA customers and automatically qualified by GSA = YES when created. Using attribute mapping, all orders for GSA customers are sourced with GSA =

YES to ensure that these orders receive the minimum price for the item or items.

If GSA pricing is enabled and the customer is not a GSA customer, then the calculation engine determines whether the selling price for an item violates the minimum allowed price for this item. If the selling price violates the minimum price then the calculation engine returns a status of GSA violation to the request line. The calling application is responsible for error handling such as determining whether to place the order line on hold.

**Note:** The calculation engine does not consider pricing attributes when comparing the selling price on the request line with the GSA price, therefore the GSA price can only be set for a product attribute.

In a non-GSA environment this functionality can be used to prevent pricing below the allowable minimum price. In this case the GSA price list would list all items and their possible selling price. No customers would be defined as GSA so the calculation engine would verify that the selling price of an item had not fallen below the minimum for all customers.

### **Calculating Price Breaks**

A price break is a series of quantity delimiters to which associated prices or discounts by range of quantity, volume, value, or weight are ordered by the customer. A price break is modeled in Oracle Advanced Pricing as a set of discount, surcharge, or charge modifiers grouped through related modifiers under a price break header modifier. The price break header contains all qualifiers, product attribute, any pricing attributes (apart from those in the volume context), unit of measure, date effectivity, and other attributes that are common elements of the price break. The price break lines contain only the break unit. The break unit must be pricing attributes in the volume context of the pricing attribute descriptive flexfield.

The search engine finds any price breaks eligible for application to the pricing request line. The search engine only considers elements of the price break header, (qualifiers, product and pricing attributes), incompatibility, and whether a price break line exists to match the break unit (quantity, amount, and so on) of the pricing request line.

The calculation engine takes the price break lines and determines a base price or discount value for the modifier. The value depends on the type of price break:

- **Point:** Volume break in which each volume of break unit receives a base price modifier in the break range within which the total volume falls.
- **Range:** Volume break in which each volume of break unit gets price/discount in the break range into which it falls.
- **Recurring:** Volume break in which the modifier is given for each volume of break unit that falls into the break range. This type of price break is used only for modifiers.

## Applying manual adjustments

The pricing engine applies any overridden manual or automatic adjustments passed in by the calling application. For more information, see: *Oracle Advanced Pricing Implementation Guide*, Integration chapter.

## Pricing Engine Flow

During a pricing engine call, the calling application:

- Compares the returned modifiers with existing adjustments and performs an update if necessary.
- Calls the local calculation engine to get the selling price.

As an aid to understanding, the following is a simplified pseudocode flow of the Oracle Advanced Pricing engine call. This flow may vary in future releases.

### Invoking Application Integration Code

```
engine call preparation
populate global record as needed in attributes mapping
call build_Contexts for header and the line (or for every line if it is
save or book event)
populate engine PL/SQL record structure
call the engine (QP_PREQ_PUB.Price_Request) within engine call
clear the temporary tables
populate temporary tables from input PL/SQL structure
for every phase of the event loop
if pricelist phase then
select pricelist list line in the pricelist provided by the call
if not found then perform the secondary search
if not found in secondary search perform big search (see Event Phases)
end if
if modifiers phase then
select and insert matching modifiers into temp tables
perform grouping, incompatibility, breaks processing
end if
end loop
call calculation engine
populate output PL/SQL structure from temp tables
return back to the invoking application
```

## Extendibility Features

Oracle Advanced Pricing contains a number of extendibility features.

### Oracle Advanced Pricing APIs

Oracle Advanced Pricing contains several application programming interfaces (APIs). For setup and Oracle Advanced Pricing engine calls, PL/SQL APIs are available. Sample working API calls can be downloaded by ARU and are available from My Oracle Support. These can be studied and modified for specific customer needs. For a complete listing of Oracle Advanced Pricing public APIs, see: *Oracle Order Management Open Interfaces, API, & Electronic Messaging Guide*.

## Related Topics

[Optimal Performance, page D-1](#)

---

## Diagnostics and Troubleshooting

This chapter covers the following topics:

- Overview of Diagnostics and Troubleshooting
- Summary of Pricing Engine Messages and Diagnosis
- Common Troubleshooting Problems in Pricing Windows
- Attribute Management Troubleshooting

### Overview of Diagnostics and Troubleshooting

This section contains information about the diagnosing and troubleshooting of problems in Oracle Advanced Pricing. The following table provides a summary of various methods of diagnosing and troubleshooting the results of the pricing engine.

**Important:** For additional troubleshooting information, see the *Oracle Advanced Pricing Troubleshooting White Paper* (docNote 373269.1) available from My Oracle Support.

## Diagnostics and troubleshooting

---

Diagnosing Method	When to Use
<p>Pricing Engine Request Viewer window in Sales Order Pad.</p> <p>How to use: The Pricing Engine Request Viewer window is available from within Oracle Order Management. The navigation path is: Sales Order window &gt; Tools &gt; Pricing Engine Request Viewer.</p>	<p>This method provides a subset of information as compared to the debug output file. It provides information such as the list line selected and deleted by the engine during processing, and the reason for deletion. It does not provide information on why the list line is not selected by the engine (qp_list_line_detail.sql is a useful mechanism for this).</p> <p>This is a quick method to verify the data passed to the pricing engine and the data that was returned by the pricing engine.</p> <p>This method determines whether the expected qualifiers and pricing attribute are sourced.</p> <p>Since the Pricing Engine Request Viewer requests are stored in the permanent pricing debug tables, users can query previous pricing debug requests.</p>
<p>Debug output</p> <p>How to for Oracle Order Management users: Set OM: Debug Directory to a database directory defined for PL/SQL file I/O. Set OM: Debug_Level to at least 5. See My Oracle Support Knowledge Document 2525754.1, Using UTL_FILE_DIR or Database Directories for PL/SQL File I/O in Oracle E-Business Suite Releases 12.1 and 12.2.</p> <p>How to for other users: Set OM:Debug Directory to a database directory defined for PL/SQL file I/O. Search the output file in the directory mentioned in above profile based on time-stamp. See My Oracle Support Knowledge Document 2525754.1, Using UTL_FILE_DIR or Database Directories for PL/SQL File I/O in Oracle E-Business Suite Releases 12.1 and 12.2.</p>	<p>Use this method when diagnosing why a line is not selected by the engine. This is done to compare the output of the debug with the output of qp_list_line_detail.sql. This is also useful if developers do not have access to the online windows. This also provides Oracle Order Management Integration debug messages, and provides extended debug messages for development.</p> <p><b>Warning:</b> Because the Pricing Engine Request Viewer consumes a large amount of system resources, ensure it is turned off in the production system. For Oracle Order Management users make sure that the Debug Level is set to 0 and QP:Debug is No to turn it off.</p>

---

Diagnosing Method	When to Use
<p>script: qp_list_line_detail.sql</p> <p>How to Use:</p> <p>Get the price list line ID/modifier line ID from the Price List/ Modifier setup window.</p> <p>Open the Price List/Modifiers window. Select the price list line/modifier line:</p> <p>Help &gt; Diagnostics &gt; Examine &gt; Pick the LIST_LINE_ID field from the Field LOV</p> <p>Note the value. This list_line_id value must be provided as input to the script qp_list_line_detail.sql.</p> <p>Login to apps/apps@sid.</p> <p>Run the script \$qp/patch/115/sql/qp_list_line_detail.sql to get all the price list line/modifier line information, which takes list line ID as input.</p> <p>Make sure that the script outputs a &lt;list line id&gt;.lst file.</p> <p>Alternately, you can run this script as a concurrent program and view the output in the concurrent request output file:</p> <p>From the Oracle Pricing Manager responsibility, navigate to Reports, submit a request for the concurrent program Diagnostics: List Line Details, and enter the list line ID for the parameter.</p>	<p>Use this method when the price list line or modifier is not selected by the engine. The script provides information about how the price list line or modifier is setup.</p> <p>Use this method when the user knows the specific price list line or modifier line that should be selected by the engine, but it is not selected.</p> <p>Use this method when the possibility exists that the denormalized columns are not being properly updated due to unusual user exceptions.</p>

Diagnosing Method	When to Use
Verify setup using Oracle Advanced Pricing set up windows	<p>Certain setup information affects the pricing engine process. Verify that the following columns have appropriate values:</p> <p>Incompatibility Resolve Code in event phase</p> <p>Search_flag in event phase</p> <p>Automatic_flag, active_flag in modifiers and price lists</p> <p>Pricing Phase in modifier</p> <p>Certain profile values affect pricing engine processes. Verify that the following profiles are appropriately set:</p> <ul style="list-style-type: none"> <li>• QP: Get Custom Price Customized</li> <li>• QP: Blind Discount Option</li> <li>• QP: Verify GSA Violations</li> <li>• QP: Return Manual Discounts</li> </ul>
Verify engine control record and other record structure	For power users: If you call the pricing engine directly, refer to the <i>Oracle Manufacturing Suite APIs and Open Interfaces Manual</i> for examples of a pricing engine call.

## Summary of Pricing Engine Messages and Diagnosis

### Price Lists Messages and Errors

The follow table summarizes pricing engine messages and provides an explanation and potential solutions.

#### Price not found on price list for item and UOM

Probable Cause	How to Debug
The price list header is inactive.	Select Active box on price list header.

Probable Cause	How to Debug
The order line initially prices when you enter the item, UOM, and quantity. However, when you click Save, the unit selling price is set to NULL and an error message advises that the item is not available on the price list.	<ul style="list-style-type: none"> <li>• Select the Enforce List Price on the order type so the error does not occur.</li> <li>• Alternatively, creating an active modifier also resolves the issue.</li> </ul>
The price list header is ineffective as of the pricing date.	Select active dates on the price list header window. Blank dates mean no restriction (pricing effectivity date can be checked in the Pricing tab of Sales Order window).
The source system code on the price list is not correct.	Verify request type, source system code mapping to make sure that appropriate source system codes are attached to the request type code.
The price list line is ineffective as of the pricing date.	Select active dates on the Price List Lines window. Blank dates mean no restriction (pricing effectivity date can be checked in the Pricing tab of Sales Order window).
The qualifiers for the price list are not met or not passed to the pricing engine (attributes mapping).	Occasionally, a qualifier is unintentionally created for a price list, which prevents the use of that price list and results in an error. If the price list has a qualifier, verify that the qualifier is passed to the engine.
The pricing attributes for price list are not met.	Using the Pricing Engine Request Viewer window, verify that the pricing attributes are mapped.
The price break conditions are not met based on item quantity and item amount.	Make sure that context volume and attribute line quantity is sourced properly for the descriptive flexfield Pricing Contexts. Use Pricing Engine Request Viewer window to verify this.
The product UOMs do not match.	Check the Pricing Engine Request Viewer window to make sure that the correct UOM is passed.

<b>Probable Cause</b>	<b>How to Debug</b>
Pricing engine call is made before the pricing attribute in the database is saved. Verify that the line is saved before making a call to the pricing engine.	You may get an item not found on the price list error if the engine is not able to find the price list due to unavailability of pricing attribute.
An unusual error causes pricing performance related columns to be out of sync.	<ol style="list-style-type: none"> <li>1. Qp_list_line_detail.sql shows that the columns are not properly updated. Run the QP: Maintain de-normalized data concurrent program to correct this situation.</li> <li>2. Price list header currency is different from the order currency. Check currency.</li> </ol>

### **Cannot Resolve Incompatibility Between Price List X and Price list Y**

<b>Probable Cause</b>	<b>How to Debug</b>
Engine could not use the passed-in Price List and found multiple matching price list lines while attempting to search other price lists.	In the price list window, determine if the passed-in price lists have qualifiers. Also, verify that the price list is active. If the user intends for the engine to use the passed in price list, then debug this issue based on suggestions in the previous table.
Engine found multiple matching price list lines with the same precedence.	Using the Pricing Engine Request Viewer window or the debug script, find the list line information of the selected list lines. Determine if one of the lines is selected unnecessarily due to missing pricing attributes. If not, update the precedence appropriately.

### **UOM is invalid**

<b>Probable Cause</b>	<b>How to Debug</b>
The pricing engine can not find the price list in the ordered UOM.	Check the UOM on the order. Open the price list window and search the price list to find out if the price is defined in the ordered UOM.

<b>Probable Cause</b>	<b>How to Debug</b>
No other matching price list line has the primary UOM flag selected.	If user intends to define the price list in primary UOM and expects the engine to convert the pricing quantity, then verify that the primary flag of the price list line is selected.

### Invalid UOM Conversion

<b>Probable Cause</b>	<b>How to Debug</b>
The ordered UOM does not match the UOM on the price list line.	If the user does not expect the pricing engine to do the conversion, then verify that the primary flag on the price list line is not set. Evaluate the reasons why the price list line with the ordered UOM is not being selected based on the previous table: Item and UOM not on Price List.
No conversion is defined in mtl_uom_conversions between the ordered UOM and the primary UOM.	If the user expects pricing to convert the pricing quantity from ordered UOM to the pricing UOM, verify that the correct conversion is defined in Oracle Inventory.

### Invalid Formula, Error Returned by QP\_FORMULA\_PRICE\_CALC\_PVT.Calculate

<b>Probable Cause</b>	<b>How to Debug</b>
User does not use NVL in the expression and a step number (formula line) has null value.	Verify that the pricing engine is selecting the expected price list line by using the debug window or debug script. Verify that the required pricing attributes are passed to the engine. If you use get_custom_price, verify that the that the function does not return a null value. If you use a factor list, verify that appropriate pricing attributes are being sourced, and that a matching factor line exists.
Formula is not a valid mathematical expression supported by the database sql.	Verify that the formula is a valid expression.

Probable Cause	How to Debug
Pricing engine call is made without passing all relevant pricing attributes. Dynamic formula is attached to the price list line. However, the pricing attributes are not entered before making the pricing engine call.	Verify that the relevant pricing attributes are entered.
Formula is either not effective for the specified date or does not exist.	Verify that the formula exists and is effective as of the pricing date.
Formula does not have at least one component.	Verify that the formula has at least one component.
QP_CUSTOM.Get_Custom_Price( ) function does not exist or is invalid in database.	Verify that function Get_Custom_Price( ) has been custom-coded in the package body of QP_CUSTOM and compiled successfully.
One of the pricing attributes is expecting a numeric value but is receiving a non-numeric value.	<p>a) Ensure that the pricing attributes used in the formula calculation always have the number valueset attached to it. The steps to verify are:</p> <ol style="list-style-type: none"> <li>1. Go to Oracle Pricing Responsibility.</li> <li>2. Go to Setup &gt; Attribute Management &gt; Context and Attributes.</li> <li>3. Query the context and attribute used in the formula setup.</li> <li>4. For that attribute, select if any number valueset is attached.</li> <li>5. 5. If none is attached, attach any number valueset to it.</li> </ol> <p>Another way to check is to change the formula by making use of the TO_NUMBER function.</p> <p>This method works only if the attribute value returned has only numeric characters. It will fail if any alphabetic characters are present in the attribute value.</p>

Probable Cause	How to Debug
The formula could contain undefined step numbers.	Verify that all of the step numbers in the formula are defined.

## Modifier Messages and Errors

### Expected Modifier Not Selected by the Engine

Probable Cause	How to Debug
Similar to message: Item and UOM not found.	Refer to previous table: <i>Item and UOM not on Price List</i> .
Qualifiers for discount list and line are not met. Qualifiers in -1 group are added to all groups.	Make sure that qualifiers in the -1 group are satisfied.
Modifier is eliminated in incompatibility.	As a test, temporarily remove the incompatibility group from the modifier line, and then run the engine call and verify that the modifier is selected. If yes, then check which other modifier is selected from the same incompatibility group. Determine if the precedence must be changed. Also, determine if the exclusive group contains a modifier.
Modifier UOM is different from Pricing UOM.	Check modifier UOM. Blank UOM means that any UOM is allowed.
Modifier header currency is different from the order currency.	Check modifier currency.
Modifier is manual; it is not automatically set.	Check automatic flag on the modifier.
Asked_For flag is Y and the modifier is not asked for.	Check Asked For flag on the modifier. If the user has asked for the modifier then use debug window/output to verify that Asked For is passed to the engine. If passed, then determine whether Asked For was validated. If not passed, determine whether the qualifiers are matched. Refer to the incompatibility processing flowchart for more details.

<b>Probable Cause</b>	<b>How to Debug</b>
Pricing phase for the modifier line is not attached to the appropriate pricing event.	Verify that the pricing phase on the modifier is attached to the appropriate event. Use caution with the event-phase setup because it can impact the pricing of the entire organization.
The qualifier, sourced item attribute, sourced pricing attribute are setup recently, however, QP Build Sourcing concurrent program does not run.	Determine in the Pricing Engine Request Viewer window/debug file whether the qualifier/pricing/item attributes are sourced. If this is a new type of attribute then run the concurrent program.
There is no record in the qp_list_header_phases.	An unusual user error can cause the qp_list_header_phases to populate incorrectly to include all phases. Run the QP: Maintain Denormalized Data concurrent program for this header to resolve the issue.

#### **Modifiers: Incompatibility does not consider best price/precedence**

<b>Probable Cause</b>	<b>How to Debug</b>
Incompatibility resolution code is set incorrectly.	Refer to previous table: <i>Item and UOM not on Price List</i> .
Customer has not licensed Oracle Advanced Pricing.	Oracle Advanced Pricing customers can choose to resolve the incompatibility processing by best price or by precedence. Oracle Order Management (basic pricing) customers only have the best price option.

#### **Modifiers: Unable to override selling price/manual adjustments**

<b>Probable Cause</b>	<b>How to Debug</b>
Manual discounts are not available.	Save the order line or move the cursor out of the line and back; this action causes Oracle Order Management to fire the pricing engine modifier phase.

Probable Cause	How to Debug
Order level adjustments are not applied.	Order level adjustments are not applied if any lines in an order has a calculate price flag of partial price or freeze price.
The unit selling price and modifier LOVs only show unapplied manual adjustments.	Overtyping the unit selling price and increasing the price to apply overrideable surcharges. Decreasing the price applies overrideable discounts.

### Calculation: Back Calculation Error

This error happens when user tries to override the selling price on a quote and the pricing engine is not able to find a suitable manual overrideable adjustment to give the overridden selling price. Refer to the Integration chapter under Manual adjustments for more details.

Probable Cause	How to Debug
No overrideable Manual adjustments available.	Check if there are any active manual overrideable adjustments.

### Concurrent Program: QP: Maintains the denormalized data in QP qualifiers

Probable Cause	How to Debug
FDPSTP failed due to ORA-06502: PL/SQL: numeric or value error: character to number conversion error.	This error occurs because of a mismatch in the parameters sequence. Check with Support for an ARU to correct this.
Program has been running for a long time.	This program updates rows in the qp_qualifiers table. If the user has selected ALL headers, then the program requires time to run.

## Integration and Attributes Mapping Messages and Errors

### Unexpected Error In Calculate\_adjustments#130 User\_defined Exception

Probable Cause	How to Debug
QP_Attr_Mapping_PUB.Build_Contexts package is invalid due to incorrect sourcing data attributes mapping.	Check dba_errors for this package in or to determine which attribute sourcing API is causing the error. If this is a custom API, then correct the API. If this is the seeded API, then determine whether a correction patch is available.
Concurrent Program Build Sourcing Rules failed with error.	Run the following statement and examine the output:  select text from dba_errors where name = 'QP_BUILD_SOURCING_PVT'  Verify that custom sourcing causes the error.
Getting error while running Build Sourcing Rules concurrent program ORA-06502: PL/SQL: numeric or value error: character string buffer too small ORA-06512: at APPS.QP_ATTR_MAPPING_PUB, line 1445 ORA-20000: ORA-04021: timeout occurred while waiting.	This error occurs if you make a pricing call while the concurrent program is running. Do not run the Build Sourcing Rules concurrent program when active users are calling pricing engine.
While entering the order line in sales order PAD receives an error: FND_AS_UNEXPECTED_ERROR (PKG_NAME=oe_order_adj_pvt) (PROCEDURE_NAME=oe_line_adj. calculate_adjustments) (ERROR_TEXT=calculate_adjustments#130 ORA-06508: PL/SQL: could not find program unit being called).	Run the following statement and examine the output:  select text from dba_errors where name = 'QP_BUILD_SOURCING_PVT' ;  Determine whether any custom sourcing causes the errors.  If the seeded sourcing rule causes this error, determine whether a patch is available to correct the seeded rule.  If the error is "Encountered the symbol '_' when expecting..." then determine which patch to apply.

## Freight Charges Integration Messages and Errors

### Cost to Charge Conversion is not Returned by the Engine

Probable Cause	How to Debug
Qualifiers may be attached to the freight charge header or line.	Run qp_list_line_detail.sql script for the expected freight modifier and verify that all the qualifiers are being passed to the engine.
Oracle Shipping Execution passed a different charge type than that set up in the engine.	Verify that the same charge type/cost type is used in Oracle Shipping Execution and Oracle Advanced Pricing.
Pricing engine does not return a specific freight charge	Pricing engine now returns only the maximum freight charge modifier for every charge name. Make sure that the freight charge that you expect is the maximum freight charge, deactivate other freight charges higher than this freight charge, or see if you can put this freight charge in a different charge name.

## Common Troubleshooting Problems in Pricing Windows

### Price List Problems

Potential price list problems and tips about how to solve them are outlined below.

#### Problem 1

The price lists have a NULL value in the Multi-Currency Conversion field after the upgrade.

Suggested Action:

Ensure that the concurrent program Update Price Lists with Multi-Currency Conversion Criteria has been run. Before running the program, set the profile QP: Multi Currency Installed to Y (Yes). If the concurrent program is not run, the value of the Multi-Currency Conversion field will be NULL.

When the profile QP: Multi Currency Installed is set to Y and the Price List window is open, a default multi-currency conversion record is created if no record is found for the defaulted currency code and rounding factor. This occurs when the defaulted currency is USD and the rounding factor is -2.

#### Problem 2

Problems such as no values in LOVs occur while running pricing reports in Oracle Order Management.

Suggested Action:

Determine if obsolete reports are running. There are currently five reports related to

Oracle Advanced Pricing:

- QPPRCST.rdf for price lists
- QPXPRFOR.rdf for pricing formulas
- QPXPRQFS.rdf for qualifier grouping
- QPXPRMLS.rdf for modifier details
- QXPACRL.rdf for accrual details

All other reports related to pricing are no longer used.

To add a pricing report to the Oracle Order Management responsibility request group, login under System Administrator responsibility. Navigate to:

- Security > Responsibility > Request

Query your request group.

To determine which request group is attached to the Oracle Order Management responsibility, navigate to:

- Security > Responsibility > Define

Query the Oracle Order Management responsibility. Add the request from the Application Oracle Advanced Pricing.

### **Problem 3**

Problems occur while running copy price list, adjust price list, add items to price list, or update formula prices through Standard Request Submission.

Suggested Action: Each of the mentioned operations is a concurrent program that has its own window that submits a request. Requests must be submitted through those forms and not Standard Request Submission. The forms have checks for mandatory parameters which are not found in the Standard Request Submission window. You can locate these forms in the sub menus for Oracle Advanced Pricing.

### **Problem 4**

LOV for product attribute value on the price list window does not display any values (items).

Suggested Action:

Verify that the value for profile option QP: Item Validation Organization (Oracle Order Management SuperUser responsibility, Setup > Profiles) is Vision Operations. Also verify under Setup > Parameters that organization Vision Operations.

### **Problem 5**

Product attribute values (when product attribute is item category) change to X automatically on list line in price list/modifier window after querying a price list/modifier window.

Suggested Action:

The view mtl\_categories\_kfv is not properly regenerated. Re-compile the flex field for item categories.

## Promotional Limits Troubleshooting

### Problem

Limit Balance record not created.

Suggested Action:

Check if the limit setup has the each organization check box selected. This feature is not currently supported and therefore, balance records are not created. The Limit with Limit Id as indicated in the message has multiple balances records. Keep the Organization box selected or change the limit setup. Check if the modifier list for which the limit is setup is active and automatic.

For any other issues, please look at the engine debug file and investigate in the limits engine code.

## Promotional Limits Integration Messages

### Problem 1

Limit Exceeded for Promotion Number &PROMOTION\_NUMBER and Limit Number &LIMIT\_NUMBER by the amount of &LIMIT\_EXCEEDED\_BY units.

Probable Cause:

The soft limit setup for the Modifier List as indicated in the message has been exceeded and current limit balance is negative. This is an informational message.

How to Debug:

User can increase the Limit available for the Modifier List in the Limits setup window.

### Problem 2

Limit Exceeded for Modifier Number &MODIFIER\_NUMBER and Limit Number &LIMIT\_NUMBER by the amount of &LIMIT\_EXCEEDED\_BY units.

Probable Cause:

The soft limit setup for the Modifier as indicated in the message has been exceeded and current limit balance is negative. This is an informational message.

How to Debug:

User can increase the Limit available for the Modifier in the Limits setup window.

### Problem 3

Modifier &OPERATOR &OPERAND for Limit Number &LIMIT\_NUMBER and Promotion Number &PROMOTION\_NUMBER adjusted to &PERCENT.

Probable Cause:

The hard limit setup for the Modifier List as indicated in the message has been adjusted

and current limit balance is zero. This is an informational message. This means that the modifier to which this limit is attached will no longer be available unless the limit is increased.

How to Debug:

User can increase the Limit available for the Modifier List in the Limits setup window.

#### **Problem 4**

Modifier &OPERATOR &OPERAND for Limit Number &LIMIT\_NUMBER and Modifier Number &MODIFIER\_NUMBER adjusted to &PERCENT.

Probable Cause:

The hard limit setup for the Modifier as indicated in the message has been adjusted and current limit balance is zero. This is an informational message. This means that the modifier to which this limit is attached will no longer be available unless the limit is increased.

How to Debug:

User can increase the Limit available for the Modifier in the Limits setup window.

#### **Problem 5**

Limit Id &LIMIT has multiple balance records. A limit with no limit attributes or specific limit attributes (not 'Each' type) must have one and only one limit balance record.

Probable Cause:

The Limit with Limit Id as indicated in the message has multiple balances records.

How to Debug:

User must delete dubious or duplicate balance records and leaving only the correct balance record in database.

#### **Problem 6**

The variable QP\_PREQ\_GRP.G\_ORDER\_PRICE\_REQUEST\_CODE cannot be null. This can corrupt Promotional Limit Balances. So limits will not be consumed. Please investigate.

Probable Cause:

The variable QP\_PREQ\_GRP.G\_ORDER\_PRICE\_REQUEST\_CODE is being passed null value by the OM Integration code.

How to Debug:

Please investigate OM/QP integration code with the help of the engine debug file.

## **Pricing Formula Problems**

### **Problem 1**

Get custom price function in a pricing formula returns null.

Suggested Action:

Add customized code for the get custom price function in the QP\_CUSTOM package body and not in another package. The profile option QP: Get Custom Price Customized must be set to yes (using System Administrator responsibility) if get custom price function has been customized and used in any formula.

**Problem 2**

When setting up a formula with price list line formula line, querying all values in the LOV for price list lines and scrolling through it causes the server to disconnect.

Suggested Action:

This LOV has a large number of values. Issuing a query for all values in the LOV and scrolling through the values causes the server to disconnect. Use a more specific or reduced search.

**Problem 3**

When entering an order line with an item that has a formula-based price, an error message is received: use NVL around potential null formula components.

Suggested Action:

This error occurs if any component in a formula evaluates to a null when the pricing engine determines the price of an item during order entry. A formula component may have a null value when it is a pricing attribute type and the value for the pricing attribute must be entered on the sales order. In this case, enter pricing attributes on the order and save. Always enter pricing attributes that are expected in the formula before proceeding to use NVL.

## Pricing Organizer Problems

**Scenario 1**

User wants to query on modifiers that are effective from March 1, 2002 and enters the following query criteria:

---

Field Name in Header tab	Value
Type	Discount List
Exact Effective Date Match	Selected
Exact Effective Date Match	Selected
Effective From	01-MAR-2002
Effective To	Blank

---

**Problem**

The modifier lists on the Headers tab in the Modifiers Organizer are not returned.

Suggested Action:

Since Exact Effective Date Match is checked, the query will match those modifiers with the exact effectivity dates as those in the modifier setup. The query will only return the modifiers lists (defined in the Modifier setup) which have Start Date =01-MAR-2002 and End Date = <Blank>. For Modifier Lines and Qualifiers, the Exact Effective Date Match will compare the query criteria to the modifier setup.

### Scenario 2

User wants to query on line level modifiers in the Lines tab and enters the following query criteria:

Field Name in Lines tab	Value
Level	Line
Formula	<ANY FORMULA>

### Problem

The modifiers are not returned in query results even though there are modifier lines of Level 'Line'.

Suggested Action:

By adding the query criteria of Formula=Any Formula, the query will only return those modifier lines that have any formula attached to the line AND is of Modifier Level =Line. These are 'AND' conditions. To query on just line level modifier lines, leave the Formula field blank in the query.

### Scenario 3

User wants to query on modifiers in USD currency and has entered Product Attributes=Products as in the following example:

Tab Name in Pricing Organizer window	Field Name	Value
Header tab	Currency	USD
Product Attributes tab	Product Attributes	Products

### Problem

The modifiers returned in query results include all modifier lines that have a product attribute value.

Suggested Action:

By giving Products Attribute=Product, this becomes an additional query criteria and the

results will include modifier lines that have a product attribute value. If Product Attributes=Not Specified, then the query will return all modifier lists with Currency=USD.

**Scenario 4**

User wants to query only on Promotions but has entered Qualifiers=No Qualifiers.

Tab Name in Pricing Organizer window	Field Name	Value
Header tab	Type	Promotion
Qualifiers tab	Qualifiers	No Qualifiers

**Problem**

The modifiers returned in query results are those promotions that do not have qualifiers attached.

Suggested Action:

If 'No Qualifiers' is selected for Qualifiers in the Qualifiers tab, only modifier lists that do not have header level qualifiers will be returned ( on the Headers Tab of the Modifier Organizer.) The query will also return modifier lines that do not have any line level qualifiers attached (on the Lines Tab of the Modifier Organizer). If Qualifiers=Not Specified, then the query will return all modifier lists that are promotions, and not modifier lines.

**Scenario 5**

A customer wants to query on discount where General Technologies is used as a customer name qualifier

Tab Name in Pricing Organizer window	Field Name	Value
Header tab	Type	Discount
Qualifiers tab	Qualifiers	Qualifiers
Qualifiers tab	Qualifier Context1	Customer
Qualifiers tab	Qualifier Attribute1	Customer Name
Qualifiers tab	Operator1	=
Qualifiers tab	Value From1	General Technologies

## Problem

The query results show only Modifier Lists and not Modifier Lines.

Suggested Action:

The qualifier Customer Name=General Technologies is only attached as a list level qualifier, thus the query will return modifiers list with Modifier Lists=Discount. If this qualifier is used as a line level qualifier, then the query will show all the modifier lines that have this qualifier (in the Lines tab of the Modifier Organizer).

## Scenario 6

When viewing an order level charge in Order Management, the Charges window shows only the Charge Name, Type, and the charge value. The Modifier Name or modifier number is not on the window. The following example demonstrates a query for modifier lines with this Charge Name:

Tab Name in Pricing Organizer window	Field Name	Value
Header tab	Type	Freight and Special Charges

The query returns all modifier list=Freight and Special Charges and no modifier lines. The User still cannot find the modifier line.

Suggested Action:

In order to get the specific modifier lines (in the Lines tab of the Modifier Organizer), you have to include Charge Name as a additional query criteria.

## Multi-Currency Scenarios

### Scenario 1: Resolving Error "For TRANSACTION conversion type, Base Currency and Functional Currency are not same"

For conversion type as 'TRANSACTION', Base Currency of the Price List and Functional Currency must be same.

In Price List window:

- Name = PL1
- Currency = AUD
- Multi-Currency Conversion = MC1

In Multi-Currency Conversion window:

- Base Currency Code = AUD
- Name = MC1

- To Currency Code = CAD
- Conversion Type = TRANSACTION

In Order Management Sales Order pad:

- Price List = PL1
- Currency = CAD
- Conversion Type = User
- Conversion Rate = 1.522
- Set of Books Currency (functional currency) = USD

### **Problem**

While placing an order, user is getting the error - For TRANSACTION conversion type, Base Currency AUD and Functional Currency USD are not same.

Suggested Action:

Use a price list that has currency as USD and the multi-currency conversion attached to it has a record for To Currency Code CAD and Conversion Type TRANSACTION. This occurs because the Base currency of the price list and functional currency must be same for conversion type 'TRANSACTION'.

### **Scenario 2: Resolving Error "No conversion rate found"**

Set up the currency conversion rate in Oracle General Ledger before using conversion type of Oracle General Ledger.

In Price List window:

- Name = PL2
- Currency = USD
- Multi-Currency Conversion = MC2

In Multi-Currency Conversion window:

- Base Currency Code = USD
- Name = MC2
- To Currency Code = CAD
- Conversion Type = TRANSACTION

In Order Management Sales Order pad:

- Price List = PL2
- Currency = CAD
- Conversion Type = Corporate
- Pricing Effective Date = 25-JUN-2002

### **Problem**

While placing an order, user is getting the error - No conversion rate found: From Currency USD, To Currency CAD, Conversion Date 25-JUN-2002, Conversion Type Corporate.

Suggested Action:

Set up the conversion rate in Oracle General Ledger for the From Currency USD, To Currency CAD, Conversion Date 25-JUN-2002 and Conversion Type Corporate. As the conversion type "Corporate" is being used in Sales Order pad, which is one of conversion types defined in Oracle General Ledger, the necessary set up must be done before placing an order.

### **Scenario 3: No conversion type is passed from OM**

Conversion type must be passed from Sales Order pad to use multi-currency conversion of type TRANSACTION .

In Price List window:

- Name = PL2
- Currency = USD
- Multi-Currency Conversion = MC2

In Multi-Currency Conversion window:

- Base Currency Code = USD
- Name = MC2
- To Currency Code = CAD
- Conversion Type = TRANSACTION

In Order Management Sales Order pad:

- Price List = PL2
- Currency = CAD
- Conversion Type =

- Conversion Rate =

**Note:** Conversion type, rate and date can be entered in sales order only when the functional currency is the same as the price list base currency.

### **Problem**

While placing an order, user is getting the error. No conversion type is passed from OM.

Suggested Action:

Provide the value for Conversion Type/Conversion Rate in Sales Order pad. As the order currency CAD is set up as conversion type TRANSACTION in multi-currency conversion list MC2, it is mandatory to pass either conversion type or conversion rate from Sales order pad.

### **Scenario 4: Formula Calculation Failure**

In Price List window:

- Name = PL2
- Currency = USD
- Multi-Currency Conversion = MC2

In Multi-Currency Conversion window:

- Base Currency Code = USD
- Name = MC2
- To Currency Code = GBP
- Conversion Type = FORMULA
- Formula = GBPconversion

In Pricing Formulas window:

- Header:
  - Name = GBPconversion Formula = 1\*2
- Formula Lines:

Formula Type	Pricing Attribute Context	Pricing Attribute	Component	Step
Pricing Attribute	Pricing Attribute	Export Cost	--	1
Numeric Constant	--	--	1.2	2

In Order Management Sales Order pad:

- Price List = PL2 Currency = GBP  
Value of Pricing Attribute "Export Cost" passed to pricing engine = Null

### Problem

While placing an order, user is getting the error - Formula calculation failure.

Suggested Action:

Pass a numeric value for Pricing Attribute "Export Cost". While using formula, make sure all the necessary information is available to correctly evaluate the formula.

### Scenario 5: Formula Calculation Failure

How the effective dates work for multi-currency setup.

In Price List window:

- Name = PL2
- Currency = USD
- Multi-Currency Conversion = MC2
- Item = AS54888
- Unit Price = 1000

In Multi-Currency Conversion window:

- Base Currency Code = USD  
Name = MC2

To Currency Code	Effective From	Effective To	Conversion Type	Fixed Value
GBP	01-JAN-2002	31-MAR-2002	TRANSACTION	--
GBP	01-APR-2002	--	FIXED	.667

**In Order Management Sales Order pad:**

- Price List = PL2Currency = GBP
- Conversion Rate = .7
- Pricing effective date = 25-JUN-2002
- Item = AS54888

**Problem**

How the effective dates work for multi-currency setup?

Suggested Action:

As per scenario V above, the unit price returned by pricing engine for item AS54888 will be 667 (1000 \* 0.667) because pricing effective date is 25-JUN-2002. Conversion type .7 passed from Sales Order pad is ignored by pricing engine because the effective dates of TRANSACTION conversion type in multi-currency setup does not satisfy the pricing effective date passed from Sales order pad. Instead, the pricing engine chooses the FIXED conversion type record from multi-currency setup as it satisfies the pricing effective date.

**Scenario 6: Markup with Multi-currency setup**

The sequence of conversion, markup, and rounding operations.

In Price List window:

- Name = PL2
- Currency = USD
- Multi-Currency Conversion = MC2
- Item = AS54888
- Unit Price = 1000

In Multi-Currency Conversion window:

- Base Currency Code = USD

Name = MC2

To Currency Code	Conversion Type	Fixed Value	Markup Operator	Markup
GBP	FIXED	.667	AMT	Value

In Order Management Sales Order pad:

- Price List = PL2
- Currency = GBP
- Item = AS54888

### Problem

How the markup works for multi-currency setup?

Suggested Action:

From the preceding scenario, the unit price returned by pricing engine for item AS54888 will be 673  $((1000 * 0.667) + 6)$  because the markup will also be applied after conversion. In the process of multi-currency conversion, the list price is first converted to the order currency, then the markup is applied (if applicable), and finally, the rounding is done.

## Other Technical Considerations

### Pricing Engine

Verify that all the prerequisite (including server technology) patches are applied.

Because the pricing engine uses temporary tables, on-line patching may create a deadlock and the patch may fail.

Temporary tables are created in TEMP table space, hence TEMP table space must be sized based on the size of the largest sales order data. Temporary tables are not sharable.

### Troubleshooting ADPATCH Errors

Log files are written to APPL\_TOP/admin/<db\_name>/log, where <db\_name> is the value of your ORACLE\_SID or TWO\_TASK variable.

For NT, the file is placed in%APPL\_TOP%\admin\<db\_name>\log, where <db\_name> is the value of your local variable.

Review the log file for error messages after you run the utility. There may be one or more worker files if you are running steps that operate in parallel mode.

Review these adwork<number>.log files (adwork01.log, adwork02.log) for more

detailed information about the errors.

### Oracle 8i Temporary Table Locking/Patching Issues

**Note:** While running adpatch, an attempt to alter, create, or drop an index on a temporary table already in use results in an error. If any Oracle Order Management or Oracle iStore user is pricing a line, the temporary tables are in use and adpatch encounters this error. Apply these patches only when users are not in Oracle Advanced Pricing.

Oracle Advanced Pricing patches attempt to drop and create Oracle8i temporary tables. Refer to the following instructions to verify that there are no processes accessing these tables before starting to apply the patch. The following script reveals which temporary tables are in use or locked (although the database session does not exist).

Before starting to apply the patch, the following two SQL statements should return no rows. Run the following statement:

```
select a.sid,a.serial#,c.object_name
from all_objects c , v$lock b, v$session a
where c.object_name in
('QP_PREQ_LINES_TMP','QP_PREQ_LDETS_TMP','QP_PREQ_LINE_ATTRS_TMP',
'QP_PREQ_RLTD_LINES_TMP','QP_PREQ_QUAL_TMP')
and c.object_type = 'TABLE'
and c.object_id = b.id1
and b.sid = a.sid;
```

If the above SQL returns rows, the sessions must be ended. It is possible that the session is ended but reference to that session still exists in v\$lock table. Run the following statement:

```
select a.sid
from v$lock a
where a.id1 in ( select b.object_id
from all_objects b
where b.object_name in
('QP_PREQ_LINES_TMP','QP_PREQ_LDETS_TMP','QP_PREQ_LINE_ATTRS_TMP',
QP_PREQ_RLTD_LINES_TMP','QP_PREQ_QUAL_TMP'))
and not exists (select 'x'
from v$session c
where a.sid = c.sid);
```

If the above statement returns rows, the database must be brought down. Once the database is brought up, run the above SQL statements and verify that no rows are selected before you apply the patch.

## Attribute Management Troubleshooting

The following table lists some potential attribute management troubleshooting problems and their solutions:

Problem Description	Solution
QP_Attr_Mapping_PUB package is invalid due to incorrect attributes mapping setup.	Check dba_errors for this package to determine which attribute mapping API is causing the error. If this is a custom API then correct the API. If this is the seeded API then determine whether a correction patch is available.
The concurrent program Build Attribute Mapping Rules failed with error.	<p>Run the following statement and examine the results:</p> <ol style="list-style-type: none"> <li>1. Select line    '/' position POS, text from dba_errors where name='QP_BUILD_SOURCING_PVT_TMP'.</li> <li>2. Determine which attribute mapping API is causing this error.</li> <li>3. If this is a custom API then correct the API. If this is the seeded API then determine whether a correction patch is available. Please refer to Point 14 for Patches available for upgrade issues in seeded attribute sourcing rules.</li> <li>4. Run the following SQL to get QP_BUILD_SOURCING_PVT_TMP package body and check the attribute mapping rule that might be causing syntax failure: <pre data-bbox="708 1010 1292 1234"> &gt; set head off &gt; set pagesize 100 &gt; spool pkg_body &gt; select text from all_source &gt; where name= 'QP_BUILD_SOURCING_PVT_TMP' &gt; and type = 'PACKAGE BODY' &gt; and owner = 'APPS' &gt; order by line; &gt; spool off </pre> </li> </ol>
The following error occurs while running Build Attribute Mapping Rules concurrent program: ORA-04021: timeout occurred while waiting.	This error occurs when someone makes a pricing call while the concurrent program is running. Do not run the Build Attribute Mapping Rules concurrent program when active users are calling the pricing engine.

Problem Description	Solution
<p>While entering the order line in sales order pad receives an error:  <b>END_AS_UNEXPECTED_ERROR</b>  (PKG_NAME=oe_order_adj_pvt)  (PROCEDURE_NAME=oe_line_adj.calculate_adjustments)  (ERROR_TEXT=calculate_adjustments#130ORA-06508: PL/SQL: could not find program unit being called).</p>	<p>Run the following statement and examine the output:</p> <pre>select line    '/'    position POS, text from dba_errors where name='QP_BUILD_SOURCING_PVT';</pre> <p>Determine whether any custom sourcing API causes the errors. If the seeded sourcing rule causes this error, determine whether a patch is available to correct the seeded rule. If the error is "Encountered the symbol _ when expecting..." then determine which patch should be applied.</p>
<p>After you run Build Attribute Mapping rules, a message confirms that the Build Attribute Mapping program ran successfully; however, the Attribute Mapping status box of the attribute that you just linked is still deselected.</p>	<p>Verify that the following prerequisites were completed:</p> <ol style="list-style-type: none"> <li>1. The Attribute Mapping Enabled check box was not selected.</li> <li>2. Attribute is used in pricing setup. Please verify that the Used in Setup check box was not selected.</li> <li>3. The Attribute Mapping Method was not ATTRIBUTE MAPPING. Attribute Mapping Methods other than ATTRIBUTE MAPPING do not need to be mapped. If the attribute that is linked was a new one, it must be used in at least one valid pricing setup.</li> <li>4. Attribute Sourcing Rules have been defined.</li> </ol>

Problem Description	Solution
<p>Attribute Mapping setup is correct and the flags are set, but still Attribute is not mapped.</p>	<p>Verify that Attribute Mapping Method is set to ATTRIBUTE MAPPING. Please note that USER ENTERED and CUSTOM SOURCED attributes will not be sourced by Build Mapping Rules program and need to be passed by the calling application and custom package, respectively.</p> <p>If Attribute Mapping Method='ATTRIBUTE MAPPING', please verify that:</p> <ol style="list-style-type: none"> <li>1. Used in Setup is set to yes.</li> <li>2. Attribute Mapping Enabled is set to yes.</li> <li>3. Attribute Mapping Status is set to yes.</li> <li>4. Mapping rules have been defined for the request type (such as ONT). If Level is BOTH, mapping rules should be defined for both the Header and Line levels.</li> <li>5. For the request type, verify that: <ul style="list-style-type: none"> <li>• the Enabled flag is selected for attribute mapping rules</li> <li>• the seeded value string and user value string are valid at Header and Line levels (depends on the mapping level defined).</li> </ul> </li> <li>6. Run Attribute Mapping Rules Error Report to check the Errors in mapping rules.</li> </ol> <p>If Attribute Mapping Method is CUSTOM SOURCED, please verify that the Custom Sourcing Package and API call defined in the sourcing rule is valid and that profile QP: Custom Sourced is set to Yes.</p>
<p>While pricing setup, the attribute is mapped dynamically, but still Attribute Sourcing Status check box is not selected.</p>	<p>Dynamic attribute mapping sets the used_in_setup flag for the attribute in attribute mapping, if the attribute is used in active pricing setup. That is, if Active Checkbox is selected for Price list/Modifier/Formula setup. User will need to run the Build Attribute Mapping Rules concurrent program to set the Attribute Mapping Status flag.</p>

Problem Description	Solution
I removed the attribute from the pricing setup, but the Used in Setup flag is still set.	You need to run the Build Attribute Mapping Rules concurrent program. This program will clear used in setup and attribute source status flags for the attributes that are not used in the setup.
Inactivated the pricing setup, but the Used in Setup and Attribute Source status flags are still selected for the attribute.	Run Build Attribute Mapping Rules concurrent program. It will clear Used in Setup and Attribute Source status flags, if attribute is not used in active pricing setup, and if profile option QP: Build Attribute Mapping Options is set to Map attributes used in active pricing setup.
Running Build Attribute Mapping Rules concurrent program only sets attribute sourcing status flag for the attribute used in setup. How do I source all attributes even if they are not used in active setup?	Build Attribute Mapping Rules uses the profile option QP: Build Attributes Mapping Options. If the value is set to "Map all attributes," it will source all the attributes regardless of whether the attribute is used in active pricing setup or not. If value is set to "Map attributes used in active pricing setup," Build Attribute Mapping Rules program will source only attributes that are used in the active setup. This profile can be controlled only at the SITE level.
Attribute is used in pricing setup, but the Used in Setup flag is not selected.	Verify that the profile option QP: Build Attribute Mapping Options value is set to "Map attributes used in active pricing setup." If so, please check Active flag is checked for the Pricing Setup (Price list/Modifier/Formula, etc) and run Build Attribute Mapping Rules concurrent program.
Attribute Mapping Method is set to 'CUSTOM SOURCED' and Build Contexts failed during call to pricing engine.	Please verify that the Custom attribute sourcing procedure QP_CUSTOM_SOURCE.Get_Custom_Attribute_Values is defined and is not throwing any exception/error. Please also check OM debug log with level 5 and check that any exception is thrown while invoking a call to the custom procedure.

---

Problem Description	Solution
---------------------	----------

---

Attribute Mapping and Sourcing Rules setup are correct and the Build Attribute Mapping Rules program ran. However, the attribute is still not sourced.

Please confirm the following:

- Verify QP\_BUILD\_SOURCING\_PVT package body source and confirm the problematic attribute is sourced.
- In calling application, please verify the attributes have been passed to the pricing engine through Pricing Engine Request Viewer and the value passed for the attribute is correct and NOT NULL.
- Please verify if sourcing rule is valid and the API returns the expected value for the input value passed from calling application in SQL Plus\* prompt. For PL/SQL Multi-record API type, please use the anonymous PL/SQL block to check API as follows:

```
//Block starts.
SET serveroutput ON
  DECLARE
    l_bg_tbl  qp_attr_mapping_pub.
t_multirecord;
  BEGIN
    - Please input appropriate values.
    FND_GLOBAL.apps_initialize(l_user_id,
l_resp_id,l_appl_id);
    /*The following is sample API call.
Please replace it with the API call
you want to
    Validate and input appropriate values.
*/
    l_bg_tbl :=ams_qp_qual_pvt.
get_buying_groups
                (&party_id,
&order_line_sold_to_org_id);
    IF l_bg_tbl.count > 0 THEN
      FOR i in l_bg_tbl.first..l_bg_tbl.
last LOOP
        - Check the value(s) returned from
the API.
      END LOOP;
    ELSE
      - API does not return any value for
the input
      Value. If it is custom API, please
check API,
      If it is seeded, please log bug
against appropriate product who owns the API.
    END IF;
  END;
//Block ends.
```

---

---

**Problem Description****Solution**

---

Need to check seeded/custom defined attribute mapping and sourcing rules are valid after upgrade.

Please run the following SQL to get attribute mapping and sourcing rule details of attributes defined in Qualifier/Pricing Attribute contexts and the Product context. Please check Enabled\_flag is set to Y for all the seeded sourcing rules, and seeded\_value\_string and user\_value\_string values.

```
SELECT A.PRC_CONTEXT_TYPE, A.PRC_CONTEXT_CODE,
       A.SEEDED_FLAG SEEDED_CONTEXT,
       A.ENABLED_FLAG CONTEXT_ENABLED,
       B.SEGMENT_ID, B.SEGMENT_CODE CODE,
       B.USER_PRECEDENCE,
          B.SEEDED_FLAG SEEDED_ATTRIBUTE,
       B.SEGMENT_MAPPING_COLUMN COLUMN_MAPPED,
       C.SEGMENT_LEVEL LEVEL_CODE,
       C.SEEDED_SOURCING_METHOD,
       C.USER_SOURCING_METHOD SOURCING_METHOD,
       C.SOURCING_ENABLED SOURCING_ENABLED,
       C.USED_IN_SETUP USED_IN_SETUP,
       C.SOURCING_STATUS SOURCE_STATUS,
       C.LOV_ENABLED LOV_ENABLED,
       C.LIMITS_ENABLED LIMITS_ENABLED,
       B.AVAILABILITY_IN_BASIC AVAIL_IN_BASIC
FROM QP_PRC_CONTEXTS_B A,
     QP_SEGMENTS_B B,
     QP_PTE_SEGMENTS C
WHERE A.PRC_CONTEXT_TYPE IN
      ('PRODUCT', 'QUALIFIER', 'PRICING_ATTRIBUTE')
AND B.PRC_CONTEXT_ID = A.PRC_CONTEXT_ID
AND C.SEGMENT_ID = B.SEGMENT_ID
AND C.PTE_CODE = &pte_code
ORDER BY
  PRC_CONTEXT_TYPE, A.PRC_CONTEXT_CODE;
Attribute Sourcing Rules details :-
SELECT A.PRC_CONTEXT_TYPE, A.PRC_CONTEXT_CODE,
       A.ENABLED_FLAG,
       B.SEGMENT_CODE, B.SEGMENT_MAPPING_COLUMN,
       C.REQUEST_TYPE_CODE, C.ATTRIBUTE_SOURCING_LEVEL,
       C.SEEDED_SOURCING_TYPE, C.SEEDED_VALUE_STRING,
       C.USER_SOURCING_TYPE, C.USER_VALUE_STRING,
       C.ENABLED_FLAG
FROM QP_PRC_CONTEXTS_B A,
     QP_SEGMENTS_B B,
     QP_ATTRIBUTE_SOURCING C
WHERE A.PRC_CONTEXT_TYPE IN
      ('PRODUCT', 'QUALIFIER', 'PRICING_ATTRIBUTE')
AND B.PRC_CONTEXT_ID = A.PRC_CONTEXT_ID
AND C.SEGMENT_ID = B.SEGMENT_ID
ORDER BY
  A.PRC_CONTEXT_TYPE, A.PRC_CONTEXT_CODE,
  B.SEGMENT_CODE, C.REQUEST_TYPE_CODE;
```

---

Problem Description	Solution
<p>The context of an attribute that was successfully mapped did not show up in the Order management Sales Pad Pricing Context list of values.</p>	<p>One of the following solutions may resolve the issue:</p> <ul style="list-style-type: none"> <li>• The only contexts that will show up in the OM Sales Order Pad are the ones that have at least one attribute that is USER ENTERED. If the context has all its attributes as mapping method ATTRIBUTE MAPPING, this context will not show up Pricing Context List of values.</li> <li>• You must create all new attributes using the Attribute Manager Context and Attributes window. Using this method, all attributes of type Pricing Attribute will be created in the OM Flexfield and the flexfield will be registered (if required); its definitions will freeze and then are compiled automatically. Check the Concurrent Manager requests to see if the request was completed as Normal. Remember, the converse is not true. An attribute created using the Flexfield windows will not be created in the Attribute Manager tables. Columns updated in Flexfield window are also not supported in Pricing.</li> </ul>
<p>PL/SQL: could not find program unit error message is recorded in OM Debug log file and no attributes sourced.</p>	<ul style="list-style-type: none"> <li>• Please verify if all the database objects are valid. If any INVALID Objects found, please recompile all INVALID objects</li> <li>• Please check dba_errors table for errors logged in one of the packages used in sourcing rules. Check the error, and recompile the package. If package does not recompile (and it is an Oracle provided package), please check if any patch is available to resolve the package error.</li> <li>• Verify if you are using custom attributes and custom sourcing rules. If QP: Custom Sourced profile is set to Yes and if the error occurs in Custom Procedure, please correct the error.</li> </ul>
<p>Attribute Mapping of an attribute is 'CUSTOM SOURCED', but this attribute is not sourced correctly.</p>	<ul style="list-style-type: none"> <li>• Ensure Profile QP: Customer Sourced is set to Yes for custom sourcing attributes.</li> <li>• Please check Custom Sourcing Procedure is valid and attributes have been sourced correctly. Please refer to Pricing Implementation Guide on how to custom source attributes. See Using Custom Sourced Attributes section for a sample custom sourcing procedure implementation.</li> </ul>

Problem Description	Solution
<p>Is there any report that identifies errors in attribute mapping rules?</p>	<p>Run the Attribute Mapping Rules Error Report and specify request type optionally. Review the report output for errors from attribute mapping rules. This report will show errors in mapping rules only for the attributes that have attribute mapping set to ATTRIBUTE MAPPING. For releases earlier than 11.5.9, see the diagnostic script provided in Attribute Management Diagnostic scripts section.</p>
<p>For troubleshooting Intercompany invoicing with Advanced Pricing issues, where can I find the document details?</p>	<p>Please set QP: Pricing Transaction Entity and QP: Source System code correctly before creating pricelist/modifier so that the contexts/attributes linked to the Intercompany Transaction PTE will be available in LOVs in Pricing setup forms.</p>
<p>Build Attribute Mapping Rules concurrent program does not complete and generates User Defined Exception:</p> <pre data-bbox="451 953 727 1073"> Invalid HVOP Attribute: QP_BULK_PREQ_GRP. G_LINE_REC. INVOICE_TO_PARTY_ID </pre>	<ul style="list-style-type: none"> <li>• Please run Attribute Mapping Rules Error report and verify if there are any errors. This report verifies the packages used in sourcing rules are valid in database and if any file dependencies are missing.</li> <li>• Please verify the custom sourcing rules are valid and the parameters passed to the custom sourcing rules. Parameters should be valid columns and not the entire structure.</li> </ul>
<pre data-bbox="451 1129 727 1249"> Invalid HVOP Attribute: QP_BULK_PREQ_GRP. G_LINE_REC. INVOICE_TO_PARTY_ID </pre>	
<pre data-bbox="451 1306 727 1528"> Call AD_DDL to create BODY of packageQP_BUILD_SOU RCING_PVT_TMP  ERROR in creating PACKAGE BODY: QP_BUILD_SOURCING_P VT_TMP </pre>	
<p>Problem in QP_Attr_Mapping_PUB. Build_Contexts API in Custom Contexts attached to the PTE.</p>	<p>Please check if there are custom contexts attached to the PTE and verify the sourcing rules of the custom attributes are valid. For troubleshooting, please deselect the Attribute Mapping Enabled check box for the custom attributes in the custom contexts attached to the PTE and run Build Attribute Mapping Rules concurrent program. Verify if the issue still exists. See bug# 4709496 and 4887583 for more details.</p>

## Checklist for Building Attribute Mapping Rules

Sometimes the message Attribute Mapping Rule Generation is Successful may display, even though the Attribute Mapping box for that attribute remains deselected when it should be selected. This may occur when you create a new attribute, link it to a Pricing Transaction Entity successfully and then map it using the Build Attribute Mapping Rules from the Tools menu. To ensure a successful mapping, confirm the following:

- The Attribute Mapping Enabled box must be selected.
- The Attribute Mapping Method must be ATTRIBUTE MAPPING. Attributes with mapping method USER ENTERED and CUSTOM SOURCED are never meant to be created by the Build Attribute Mapping Rules program.
- In 11.5.10 and later releases, you can update (enable/disable) the Sourcing Enabled check box for the attribute sourcing rule. Please ensure that this Sourcing Enabled box is selected for the sourcing rule to be active and for the attribute to be sourced by attribute mapping functionality.
- If the attribute is newly created, you must ensure that it is attached to at least one valid Pricing Setup such as a Modifier, Price List or Limit.
- Ensure that the PTE-Attribute link that was created has at least one attribute mapping rule.

### **QP: Build Attributes Mapping Options profile option**

This profile option enables to set attribute mapping rules for attributes in both the active and inactive setups.

#### **Values**

- Map attributes used in active pricing setup: The Build Attribute Mapping Rules program will source the attributes that are used only in the active pricing setup.
- Map all attributes: This program will source the attributes that are used in both the active and inactive setups.

Please set this profile value to *Map attributes used in active pricing setup* to map only those attributes used in the active pricing setup to improve the performance of Build Attribute Mapping Rules program.

## Using Custom Sourced Attributes

Attributes can also be passed to the pricing engine directly, without the need for an attribute mapping rule. In such cases, the Attribute Manager API calls a custom API, QP\_CUSTOM\_SOURCE, where the user has manually defined the attributes being passed and coded the mapping of their values.

The user code is written in the package procedure QP\_CUSTOM\_SOURCE.

Get\_Custom\_Attribute\_Values. The Attribute Manager API program (Build\_Contexts), calls this procedure to pick up custom-sourced attributes if the profile option QP\_CUSTOM\_SOURCED is set to yes. The input parameters to QP\_CUSTOM\_SOURCE are Request Type code and Pricing Type.

Typical values of Request Type Codes that can be passed are ONT, ASO, OKC, IC FTE, or MSD. By using the Request\_Type\_Code, you can control how the attributes are mapped based on the PTE of the calling application. The Pricing Type can be H (Header) or L (Line), which defines the level of the attribute mapping rule. These attributes and their values are passed to the pricing engine in the same manner as the attributes sourced through attribute mapping rules.

### **QP: Custom Sourced profile option**

Profile option QP: Custom Sourced is Yes. Files:

- QPXCSOUS.pls: QP\_CUSTOM\_SOURCE Package Specification.
- QPXCSOUB.pls: QP\_CUSTOM\_SOURCE Package Body. Customer has to implement the Package Body in QPXCSOUB.pls (for example) to source custom mapped qualifier/pricing attributes in Get\_Custom\_Attribute\_Values procedure and need to set QP: Custom Sourced profile option to Yes.

### **Procedure Get\_Custom\_Attribute\_Values**

QPXCSOUS.pls: QP\_CUSTOM\_SOURCE package specification contains the following procedure declaration.

```
PROCEDURE Get_Custom_Attribute_Values
( p_req_type_code           IN VARCHAR2
  , p_pricing_type_code     IN VARCHAR2
  , x_qual_ctxts_result_tbl OUT QP_ATTR_MAPPING_PUB.
  CONTEXTS_RESULT_TBL_TYPE
  , x_price_ctxts_result_tbl OUT QP_ATTR_MAPPING_PUB.
  CONTEXTS_RESULT_TBL_TYPE
);
```

Parameters:

```
p_req_type_code
    Request Type Code. ex, ONT.
p_pricing_type_code
    - 'L' for Line and 'H' for Header Level attribute mapping rule.
x_qual_ctxts_result_tbl
    - Qualifier attributes result table.
x_price_ctxts_result_tbl
    - Pricing attributes result table.
```

If profile option QP: Custom Sourced is set to Yes, Attribute Manager API Build\_Contexts will call QP\_CUSTOM\_SOURCE.Get\_Custom\_Attribute\_Values procedure to source custom mapped attributes (for example, attributes with Attribute Mapping Method=CUSTOM SOURCED).

### **Example code**

The following example explains how you could the body of QP\_CUSTOM\_SOURCE for a particular case. In this case, two segment mapping columns, QUALIFIER\_ATTRIBUTE31 and PRICING\_ATTRIBUTE31 that belong to contexts CUST\_SOURCE\_QUAL\_CON and CUST\_SOURCE\_PRIC\_CON respectively and linked to PTE Order Fulfillment, will have Custom Sourced values as 10 for ORDER as

well as LINE Attribute Mapping levels. You must ensure that the Attribute Mapping method for both these PTE-Attribute links is CUSTOM SOURCED in the Attribute Linking and Mapping setup window.

```

CREATE or REPLACE PACKAGE body QP_CUSTOM_SOURCE AS
/*Customizable Public Procedure*/

PROCEDURE Get_Custom_Attribute_Values
( p_req_type_code          IN   VARCHAR2
 ,p_pricing_type_code     IN   VARCHAR2
 ,x_qual_ctxts_result_tbl OUT QP_ATTR_MAPPING_PUB.
 CONTEXTS_RESULT_TBL_TYPE
 ,x_price_ctxts_result_tbl OUT QP_ATTR_MAPPING_PUB.
 CONTEXTS_RESULT_TBL_TYPE
 ) is
Begin
  /* Note:
    a. Assign Context Code to context_name parameter and not the name
    of the context.
    b. Assign Column Mapped for the attribute to attribute_name
    parameter and not the
       attribute name.
    c. Ensure that attribute_value is assigned to NOT NULL value,
    otherwise the attribute will not
       get sourced and not used by pricing engine for calculation.
  */

  -- The statements below help the user in turning debug on
  -- The user needs to set the oe_debug_pub.G_DIR value.
  -- This value can be found by executing the following statement
  --   select value
  --   from   v$parameter
  --   where name = 'utl_file_dir';
  -- This might return multiple values , and any one of the values can be
  -- taken
  -- Make sure that the value of the directory specified , actually exists

  -- Sample debug message.
  -- oe_debug_pub.add ('In Get_Custom_Attribute_Values');

  If p_req_type_code = 'ONT' and p_pricing_type_code in ('L','H') then

    -- Sourcing qualifier attributes.
    x_qual_ctxts_result_tbl(1).context_name      :=
'CUST_SOURCE_QUAL_CON';
    x_qual_ctxts_result_tbl(1).attribute_name   :=
'QUALIFIER_ATTRIBUTE31';
    x_qual_ctxts_result_tbl(1).attribute_value := '10';

    -- Sourcing pricing attributes.
    x_price_ctxts_result_tbl(1).context_name    :=
'CUST_SOURCE_PRIC_CON';
    x_price_ctxts_result_tbl(1).attribute_name :=
'PRICING_ATTRIBUTE31';
    x_price_ctxts_result_tbl(1).attribute_value:= '10';
  end if;
End Get_Custom_Attribute_Values;
END QP_CUSTOM_SOURCE;
/

```

Please note that, context\_name is actually the context code and not the name of the context. Also, attribute\_name is the column mapped for the attribute and not the attribute name.

### Troubleshooting Custom sourcing attributes

- Please check QP: Custom Sourced profile is set to Yes for the Attribute Manager API Build\_Contexts to call QP\_CUSTOM\_SOURCE.Get\_Custom\_Attribute\_Values procedure to source custom mapped attributes.
- Please set QP: Debug profile to 'Request Viewer On' and reproduce the issue. Look at the Pricing Engine Request Viewer and see the custom sourced attribute is sourced correctly and the value assigned to the attribute is matched with the setup value in the pricelist/modifier/formula setup. Please see 'Pricing Engine Request Viewer – Attributes section' in Advanced Pricing implementation guide on how to use Pricing Engine Request Viewer for attribute sourcing issues.
- Please check the string 'Before Calling Custom Sourcing Package' in the debug log and make sure that there is no exception/error message thrown in custom API call and check the number of custom attributes sourced. Also, check the output of the following SQL to determine the error message thrown in the custom package:  

```
SELECT line || '/' position POS, text from DBA_ERRORS WHERE  
NAME = 'QP_CUSTOM_SOURCE' ;
```
- Please note that the value assigned to the attribute in Get\_Custom\_Attribute\_Values procedure should not be NULL, otherwise the attribute will not get sourced.
- Please put debug messages in Get\_Custom\_Attribute\_Values procedure using oe\_debug\_pub.add API and verify that the debug messages are printed in the debug log to diagnose the issue in the custom procedure implementation.

### Pricing Engine Request Viewer

The Pricing Engine Request Viewer window captures and displays the inputs and outputs of the pricing call. It captures the pricing engine call from any calling application, such as OM, iStore, Order Capture, or Oracle Contracts Core. The information displayed by the Pricing Engine Request Viewer enables you to diagnose which lines were selected or rejected by the pricing engine to determine why certain prices and adjustments were or were not applied. The Pricing Engine Request Viewer window displays the latest pricing request and updates the display information each time the pricing engine captures a new transaction. Previous pricing requests are saved in pricing tables. Using the Pricing Engine Request Viewer window, you can do the following:

- View the controls passed by the calling application to the pricing engine such as Events, Rounding Flag, Search and Calculate Flag, GSA Flag.
- View price request line passed in by the calling application.
- View which modifier lines the pricing engine applied or rejected for benefit adjustments along with the details of the modifier line.

- View the pricing, qualifiers, and product attributes passed to the pricing engine along with the other data generated by the engine.
- View the relationship between order lines for promotional modifiers, price breaks, and service lines (OID, PRG, PBH, and Service Items).
- View the formula step values generated by the pricing engine used in formula calculation.
- View and query fields in the Pricing Debug Log.

### Setting up the user profiles

- **QP: Debug**

To start the Pricing Engine Request viewer, set the profile QP: Debug to Request Viewer On. Alternately, select Request Viewer Off to turn the Request Viewer off. This profile option can be updated at the user level. The pricing engine request viewer is only active for the transactions of the user who set this profile option; other users' transactions are not affected. The default value is set to Request Viewer Off. If the profile is set to Request Viewer On, but the Debug Log is not visible in the Request Viewer, then the Request Viewer captures pricing request details into the pricing debug tables, but the debug log information is not written into the debug log table. The debug log text file will be created.

If the QP: Debug profile value is set to "Request Viewer Off, Generate List Line Details for Debug," then a menu option "Debug: Generate List Line Details" can be selected from the Tools menu in the pricing setup windows (price list, modifier, and pricing agreements windows). To collect list line details, click Debug: Generate List Line Details from the list line record for which the line details needs to be generated. A message displays that the debug information for the list line is being generated through a concurrent request.

- **QP: Set Request Name**

Set the value of the profile option QP: Set Request Name to append the value of the profile with Order ID to be stored in the Request Name field. The default value is Null.

### Attribute Sourcing Issues in Pricing Engine Request Viewer window

- To collect debug information in QP tables which is used by Pricing Engine Request Viewer, only the profile QP: Debug needs to be set to either 'Request Viewer On' or 'Request Viewer On, but Debug Log is not visible in Viewer' option and OM: Debug Level profile option need not be set.
- From the 'Pricing Engine Request Viewer' window, select Attributes button by placing the cursor in the Lines Block, Attributes block shows all the attributes corresponding to that Line. The output qualifier and pricing attribute records

would have the qualifiers and pricing attributes passed to the pricing engine for the line.

- From the Pricing Engine Request Viewer window, select Attributes button by placing the cursor in the Line Details block, Attributes block shows all the attributes corresponding to that Line Detail. The output qualifier and pricing attribute records would have the qualifiers and pricing attributes which qualified the price list line or modifier list line.
- The attributes of Line and Line details are stored in QP\_DEBUG\_REQ\_LINE\_ATTRS table. Please refer to Pricing Engine Request Viewer tables section for more information.
- Ensure that the all the required attributes have been sourced correctly and the values match with the qualifier/product/pricing attributes used in the pricelist/modifier/formula setup.
- Please set QP: Insert Formula Step Values into Temp Table profile to Yes to insert the dynamic formula step values evaluated during runtime. Only if this profile is set to yes, formula step values for the dynamic formula attached to price list/modifier will be inserted into QP\_FORMULA\_STEP\_VALUES\_TMP table which can be used during runtime in custom code and 'Step Values' button in the Pricing Engine Viewer will be enabled. For any formula sourcing issues, please see if any formula processing error is thrown in the debug log and set this profile to Yes to see the formula step values. The data in QP\_FORMULA\_STEP\_VALUES table is copied to debug table QP\_FORMULA\_STEP\_VALUES\_TMP, which is used by Pricing Engine Request Viewer.

See the *Oracle Advanced Pricing Implementation Guide*, Pricing Engine Request Viewer window section for complete details on the columns in each region.

- Please run the Purge Pricing Engine Requests concurrent program on a regular basis to purge the historical data from the pricing debug tables. Periodically purging the historical data from the pricing debug tables will improve the performance of the Pricing Engine Request Viewer window.

**Pricing Engine Request Viewer tables**

Step #	Table Name	Description
1	QP_DEBUG_REQ	This table contains information about the pricing requests.
2	QP_DEBUG_REQ_LINES	This table contains details about the lines being priced.

Step #	Table Name	Description
3	QP_DEBUG_REQ_LDETS	This table contains information about the line details being priced. It has adjustment and discount details for lines, including adjustments and discounts that the pricing engine eliminates.
4	QP_DEBUG_REQ_LINE_ATTRS	This table contains information on the pricing attributes that the attribute mapping functionality passed to the pricing engine.
5	QP_DEBUG_REQ_RLTD_LINES	This table contains information about the related lines.
6	QP_FORMULA_STEP_VALUES	This table contains information about the formula step values that are inserted into the table QP_FORMULA_STEP_VALUES. The step values are inserted into the table during the evaluation of the formula attached to the price list or modifier.

## Attribute Management Diagnostic scripts

### 1. Script: qp\_attribute\_mapping\_detail.sql

Script to get attribute, context, attribute mapping and sourcing rules details to diagnose attribute sourcing related issues.

- Input parameters: Attribute Code
- Details: Attribute Mapping details

```

SELECT A.PRC_CONTEXT_CODE, A.PRC_CONTEXT_TYPE,
       A.SEEDED_FLAG SEEEDED_CONTEXT, A.ENABLED_FLAG CONTEXT_ENABLED,
       B.SEGMENT_ID, B.SEGMENT_CODE CODE, B.USER_PRECEDENCE,
       B.SEEDED_FLAG SEEEDED_ATTRIBUTE, B.SEGMENT_MAPPING_COLUMN
COLUMN_MAPPED,
       C.PTE_CODE, C.SEGMENT_LEVEL LEVEL_CODE, C.SEEDED_SOURCING_METHOD,
       C.USER_SOURCING_METHOD SOURCING_METHOD, C.USED_IN_SETUP
USED_IN_SETUP,
       C.SOURCING_ENABLED SOURCING_ENABLED, C.LOV_ENABLED LOV_ENABLED,
       C.LIMITS_ENABLED LIMITS_ENABLED, C.SOURCING_STATUS SOURCE_STATUS,
       B.AVAILABILITY_IN_BASIC AVAIL_IN_BASIC
FROM QP_PRC_CONTEXTS_B A,
     QP_SEGMENTS_B B,
     QP_PTE_SEGMENTS C
WHERE B.SEGMENT_CODE = &segment_code
AND A.PRC_CONTEXT_ID = B.PRC_CONTEXT_ID
AND C.SEGMENT_ID = B.SEGMENT_ID;

```

b. Attribute Sourcing Rules details:

```

SELECT D.REQUEST_TYPE_CODE, D.ATTRIBUTE_SOURCING_LEVEL,
       D.SEEDED_SOURCING_TYPE, D.USER_SOURCING_TYPE,
       D.SEEDED_VALUE_STRING, D.USER_VALUE_STRING,
       D.SEEDED_FLAG, D.ENABLED_FLAG
FROM QP_SEGMENTS_B B,
     QP_ATTRIBUTE_SOURCING D
WHERE B.SEGMENT_CODE = &segment_code
AND D.SEGMENT_ID = B.SEGMENT_ID
ORDER BY REQUEST_TYPE_CODE, ATTRIBUTE_SOURCING_LEVEL;

```

## 2. Script to diagnose errors in attribute mapping rules setup earlier than release 11.5.9: qp\_attr\_mapping\_error.sql

Script: qp\_attr\_mapping\_error.sql

- Input parameters: Request Type Code (optional)
- Details: Request Type code input parameter is optional; if not specified, this script will report errors in attribute mapping rules regardless of the request type code.

```

REM /* $Header: qp_attr_mapping_error.sql */

REM FILETYPE NOEXEC
REM dbdrv: none

WHenever SQLERROR EXIT FAILURE ROLLBACK;

CLEAR BUFFER;

SET ARRAYSIZE 4;
SET PAGESIZE 58;
SET TERM ON;
SET LINESIZE 145;
SET UNDERLINE =;
SET VERIFY OFF;
SET FEED OFF;
SET SERVEROUTPUT ON SIZE 100000;
SET FEEDBACK OFF;
SET HEADING OFF;

SPOOL qp_attr_mapping_error.lst

DECLARE

TYPE l_cursor_type IS REF CURSOR;

l_attribute_value VARCHAR2(240);
l_attribute_mvalue QP_Attr_Mapping_PUB.t_MultiRecord;
l_context_name qp_prc_contexts_b.prc_context_code%TYPE;
l_attribute_name qp_segments_b.segment_code%TYPE;
l_attribute_map qp_segments_b.segment_mapping_column%TYPE;
l_src_type qp_attribute_sourcing.user_sourcing_type%TYPE;
l_pricing_type qp_attribute_sourcing.attribute_sourcing_level%TYPE;
l_value_string qp_attribute_sourcing.user_value_string%TYPE;
l_request_type qp_attribute_sourcing.request_type_code%TYPE;
l_pte_name qp_pte_request_types_b.pte_code%TYPE;
l_context_type qp_prc_contexts_b.prc_context_type%TYPE;
l_err_msg VARCHAR2(2000);
l_err_count BINARY_INTEGER := 0;
l_sql_stmt VARCHAR2(2500);
l_cursor l_cursor_type;
l_request_type_code qp_attribute_sourcing.request_type_code%TYPE :=
'l_request_type_code';
l_successful BOOLEAN;

BEGIN

IF l_request_type_code IS NULL THEN
OPEN l_cursor FOR
SELECT qpsseg.segment_mapping_column,
qpsour.user_sourcing_type src_type,
qpsour.user_value_string value_string,
qpcon.prc_context_code context_code,
qpsour.attribute_sourcing_level,
qpsour.request_type_code,
qpreq.pte_code,
qpcon.prc_context_type,
qpsseg.segment_code
FROM
qp_segments_b qpsseg,
qp_attribute_sourcing qpsour,
qp_prc_contexts_b qpcon,
qp_pte_request_types_b qpreq,
qp_pte_segments qppseg

```

```

WHERE
    qpsour.segment_id = qpseg.segment_id
    AND qppseg.user_sourcing_method = 'ATTRIBUTE MAPPING'
    AND qpseg.prc_context_id = qpcon.prc_context_id
    AND qpreq.request_type_code = qpsour.request_type_code
    AND qppseg.pte_code = qpreq.pte_code
    AND qppseg.segment_id = qpsour.segment_id
    AND qppseg.sourcing_enabled = 'Y'
    AND qpcon.prc_context_type in ('PRICING_ATTRIBUTE',
'PRODUCT', 'QUALIFIER');
ELSE
    OPEN l_cursor FOR
    SELECT qpseg.segment_mapping_column,
    qpsour.user_sourcing_type src_type,
    qpsour.user_value_string value_string,
    qpcon.prc_context_code context_code,
    qpsour.attribute_sourcing_level,
    qpsour.request_type_code,
    qpreq.pte_code,
    qpcon.prc_context_type,
    qpseg.segment_code
    FROM
    qp_segments_b qpseg,
    qp_attribute_sourcing qpsour,
    qp_prc_contexts_b qpcon,
    qp_pte_request_types_b qpreq,
    qp_pte_segments qppseg
    WHERE
    qpsour.segment_id = qpseg.segment_id
    AND qppseg.user_sourcing_method = 'ATTRIBUTE MAPPING'
    AND qpsour.request_type_code = l_request_type_code
    AND qpseg.prc_context_id = qpcon.prc_context_id
    AND qpreq.request_type_code = qpsour.request_type_code
    AND qppseg.pte_code = qpreq.pte_code
    AND qppseg.segment_id = qpsour.segment_id
    AND qppseg.sourcing_enabled = 'Y'
    AND qpcon.prc_context_type in ('PRICING_ATTRIBUTE',
'PRODUCT', 'QUALIFIER');
    END IF;

LOOP
    FETCH l_cursor INTO
    l_attribute_map,
    l_src_type,
    l_value_string,
    l_context_name,
    l_pricing_type,
    l_request_type,
    l_pte_name,
    l_context_type,
    l_attribute_name;

    EXIT WHEN l_cursor%NOTFOUND;

    l_successful := TRUE;
    IF l_src_type = 'API' THEN
    BEGIN
        l_sql_stmt := 'BEGIN QP_Attr_Mapping_PUB.G_Temp_Value := ' ||
l_value_string || ';' || ' END;';
        EXECUTE IMMEDIATE l_sql_stmt;
        l_attribute_value := QP_Attr_Mapping_PUB.G_Temp_Value;
    EXCEPTION
        WHEN VALUE_ERROR THEN
            NULL;
        WHEN NO_DATA_FOUND THEN
            NULL;
    END IF;

```

```

WHEN OTHERS THEN
    l_err_msg := SQLERRM;
    l_successful := FALSE;
END;
ELSIF l_src_type = 'API_MULTIREC' THEN
    BEGIN
        l_sql_stmt := 'BEGIN QP_Attr_Mapping_PUB.G_Temp_MultiValue :=
' || l_value_string || ';
        || ' END;';
        EXECUTE IMMEDIATE l_sql_stmt;
        l_attribute_mvalue := QP_Attr_Mapping_PUB.G_Temp_MultiValue;
    EXCEPTION
        WHEN NO_DATA_FOUND THEN
            NULL;
        WHEN OTHERS THEN
            l_err_msg := SQLERRM;
            l_successful := FALSE;
    END;
ELSIF l_src_type = 'PROFILE_OPTION' THEN
    BEGIN
        l_attribute_value := fnd_profile.value(l_value_string);
    EXCEPTION
        WHEN NO_DATA_FOUND THEN
            NULL;
        WHEN OTHERS THEN
            l_err_msg := SQLERRM;
            l_successful := FALSE;
    END;
ELSIF l_src_type = 'SYSTEM' THEN
    BEGIN
        l_sql_stmt := 'SELECT ' || l_value_string || ' FROM DUAL';
        EXECUTE IMMEDIATE l_sql_stmt INTO l_attribute_value;
    EXCEPTION
        WHEN NO_DATA_FOUND THEN
            NULL;
        WHEN OTHERS THEN
            l_err_msg := SQLERRM;
            l_successful := FALSE;
    END;
ELSIF l_src_type = 'CONSTANT' THEN
    l_attribute_value := l_value_string;
ELSE
    dbms_output.put_line('Invalid source type: ' || l_src_type);
END IF;

IF l_successful = FALSE THEN
    IF l_err_count = 0 THEN
        DBMS_OUTPUT.PUT_LINE('ATTRIBUTE MAPPING RULES ERRORS FOUND');
        DBMS_OUTPUT.PUT_LINE
('-----');
        END IF;

        DBMS_OUTPUT.PUT_LINE('PTE:          ' || l_pte_name);
        DBMS_OUTPUT.PUT_LINE('Context Type: ' || l_context_type);
        DBMS_OUTPUT.PUT_LINE('Context:      ' || l_context_name);
        DBMS_OUTPUT.PUT_LINE('Attribute:    ' || l_attribute_name || '
(' || l_attribute_map || ')');
        DBMS_OUTPUT.PUT_LINE('Request Type: ' || l_request_type);
        DBMS_OUTPUT.PUT_LINE('Level:        ' || l_pricing_type);
        DBMS_OUTPUT.PUT_LINE('Source Type:  ' || l_src_type);
        DBMS_OUTPUT.PUT_LINE('Sourcing Rule: ' || l_value_string);
        DBMS_OUTPUT.PUT_LINE('Error Message: ' || l_err_msg);
        DBMS_OUTPUT.PUT_LINE
('-----');
        l_err_count := l_err_count + 1;
    END IF;

```

```

END LOOP;
CLOSE l_cursor;

IF l_err_count = 0 THEN
    DBMS_OUTPUT.PUT_LINE('NO ATTRIBUTE MAPPING RULES ERROR(S) FOUND');
ELSE
    DBMS_OUTPUT.PUT_LINE(l_err_count || ' ATTRIBUTE MAPPING RULES
ERROR(S) FOUND');
END IF;
EXCEPTION
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('*** ERROR IN RUNNING THE SCRIPT ***');
        DBMS_OUTPUT.PUT_LINE(SQLERRM);
        DBMS_OUTPUT.PUT_LINE
('*****');
END;
/
SPOOL OFF;

```



---

# Windows and Navigation Paths

## Windows and Navigation Paths

Refer to the following sources for further information about windows and applications:

- *Oracle Order Management User's Guide*
- *Oracle Application User's Guide*

The Window Names and Navigation Paths table lists the navigation path for each window or HTML page. Here are the conventions and short names used in the table:

- HTML user interface (UI) and Self Service Web Application (SSWA): Identifies a navigation path in the HTML user interface.
- []: Square brackets indicate a button name or a link that you must click to display the window or HTML page.
- [T]: Square brackets around the letter T indicate that you must click a tab.

---

<b>Window Name</b>	<b>Navigation Path</b>
Add Items to Price List	Price Lists > Add Items to Price List
Adjust Price List	Price Lists > Adjust Price List
Advanced Pricing - Define Modifier	Modifiers > Modifier Setup
Advanced Pricing - Define Modifier	Modifiers > Modifier Incompatibility Setup > [Modifiers]

---

Window Name	Navigation Path
Advanced Pricing: Home Page	SSWA > Oracle Pricing User responsibility > Home
Advanced Pricing - Price Lists	Price Lists > Price List Setup
Advanced Pricing - Pricing Formulas	Pricing Formulas > Formulas Setup
Advanced Search	SSWA > Oracle Pricing User > Price List Maintenance > [Advanced Search]  <b>Alternate:</b> SSWA > Oracle Pricing Administrator > Price List Maintenance > [Advanced Search]  Alternate:
Archive Pricing Entities	Archive Purge Pricing Entities > Archive Entity
Assign Attributes	Setup > Attribute Management > Attribute Linking and Mapping > [Assign Attributes]
Attribute Mapping	Setup > Attribute Linking and Mapping > [Link Attributes] > Link Attributes window > [Attribute Mapping]
Bulk Change	SSWA > Oracle Pricing User > Price List Maintenance > [Advanced Search] > [Search] > From Advanced Search page > [Bulk Change]  <b>Alternate:</b> SSWA > Oracle Pricing User or Oracle Pricing Administrator > Price List Maintenance > (Do Search) > [Bulk Change]
Bulk Create Privileges	SSWA > Oracle Pricing Administrator Responsibility > Security > Privileges > [Bulk Create Privileges]
Bulk Update Entity Usage	SSWA > Oracle Pricing Administrator Responsibility > Security > Entity Usage > [Bulk Update Entity Usage]

Window Name	Navigation Path
Context Setup	Setup > Attribute Mapping > Context and Attributes
Copy Modifiers	Modifiers > Copy Modifiers
Create Delta Price Book (HTML UI)	SSWA > Oracle Pricing User > Advanced Pricing [Home] > Reports [T] > Search > [Create Delta]  <b>Alternate:</b> SSWA > Oracle Pricing User > Advanced Pricing [Home] > Reports [T] > Search (do search) > [Name] > [Create Delta]
Copy Price List	Price Lists > Copy Price List
Create Entity Set	SSWA > Oracle Pricing Administrator Responsibility > Security > Entity Sets > [Create Entity Set]
Create Line page (modifier) (HTML UI)	SSWA > Oracle Pricing User responsibility > Home page > Shortcuts > Create (Modifier Type) List > (Create modifier) > Next > Update Discount List: Modifier Lines > [Create Line]
Create Price Book (HTML UI)	SSWA > Oracle Pricing User > Advanced Pricing [Home] > Reports [T] > [Create Price Book]
Create Price List: General Information (HTML UI)	SSWA > Oracle Pricing User responsibility > Home > Oracle Advanced Pricing: Home page [Shortcuts: Create Price List]  <b>Alternate:</b> SSWA > Oracle Pricing User responsibility > Home > Oracle Advanced Pricing: Home page [Lists tab] > Create List

Window Name	Navigation Path
Create Price List: Qualifiers (HTML UI)	SSWA > Oracle Pricing User responsibility > Home > Oracle Advanced Pricing: Home page [Shortcuts: Create Price List] > Create Price List: General Information > [Next]  <b>Alternate:</b> SSWA > Oracle Pricing User responsibility > Home > Oracle Advanced Pricing: Home page [Lists tab] > Create List > Create Price List: General Information page [Next]
Debug Log	Pricing Engine Request Viewer > [View Debug Log]
Define Limits	Modifiers > Modifier Setup > Define Modifier (W) > [List Limits] >
Define Modifier	Modifiers > Modifier Setup
Define Modifier - Define GSA price	Price Lists > GSA Pricing Setup
Define Modifier Details	Modifiers > Modifier Setup > [Define Details*]
Descriptive Flexfield Segments	Setup > FlexFields
Detail Change (HTML UI)	SSWA > Oracle Pricing User > Price List Maintenance > [Advanced Search] > From Advanced Search page > [Detail Change]  <b>Alternate:</b> SSWA > Oracle Pricing User or Oracle Pricing Administrator > Price List Maintenance > (Do Search) > [Detail Change]
Entity Set Details (HTML UI)	SSWA > Oracle Pricing Administrator Responsibility > Security > Entity Sets > [Set Id]
Entity Sets	SSWA > Oracle Pricing Administrator Responsibility > Security > Entity Sets
Entity Usage	SSWA > Oracle Pricing Administrator Responsibility > Security > Entity Usage

Window Name	Navigation Path
Event Phases	Setup > Event Phases
Exclude Items	Modifiers > Modifier Setup > [Exclude]
Express Create Privilege	SSWA > Oracle Pricing Administrator Responsibility > Security > Privileges > [Express Create Privilege]
Factors	Pricing Formulas > Formulas Setup > [Factors]
Find Personal Profile Values	Setup > Profiles
Find Modifiers	Pricing Organizer
Find Requests	View Concurrent Requests <b>Alternate:</b> View Concurrent Requests > [Find] > [Find Requests]
Find Formula Factor Lines	Pricing Formulas > Formulas Setup > Advanced Pricing - Pricing Formulas > (Query Formula Name) > Select Formula line > [Factors] > Factors window > Select line which contains the value > [Search icon]
GSA Qualifiers	Price Lists > GSA Pricing Setup > [List Qualifiers]
Incompatibility Groups	Modifiers > Modifier Incompatibility Setup
Link Attributes	Setup > Attribute Linking and Mapping > [Link Attributes]
Log file: request id	View Concurrent Requests > [Find] > [View Log...]
Modifier Lists	SSWA > Oracle Pricing User > Home > Lists {T}, Administrator Responsibility > Modifier Lists

Window Name	Navigation Path
Modifier Lists Search (HTML UI)	SSWA > Oracle Pricing User responsibility > Home > Oracle Advanced Pricing: Home page > [List tab] > Modifier Lists
Modifier Organizer	Pricing Organizer
More Pricing Attributes	Modifiers > Modifier Setup > [Pricing Attributes]
Multi-Currency Conversion List	Price Lists > Multi-Currency Conversion Setup
Parameter Definition (HTML UI)	SSWA > Oracle Pricing Administrator responsibility > Oracle Pricing Administrator Setup: Parameters > Parameter Definition  <b>Alternate:</b> Pricing Manager responsibility > Pricing Transaction Entity - Source System and Request Type
Oracle Pricing Lookups	Setup > Lookups
Personal Profile Values	Setup > Profiles > [Find]
Price Book (HTML UI)	SSWA > Oracle Pricing User > Advanced Pricing [Home] > Reports [T] > Search
Price Breaks	Pricing Agreements > Price Breaks  <b>Alternate:</b> Modifiers > Modifier Setup > [Define Details*] > Price Breaks  <b>Alternate:</b> Price Lists > Price List Setup > [Price Breaks]
Price List Maintenance (HTML UI)	SSWA > Oracle Pricing Administrator Responsibility > Price List Maintenance  <b>Alternate:</b> Oracle Pricing User > Price List > Maintenance
Price Lists Search (HTML UI)	SSWA > Oracle Pricing User responsibility > Home > Oracle Advanced Pricing: Home page > [List tab] > Price Lists

Window Name	Navigation Path
Pricing Agreements	Pricing Agreements
Pricing Attributes	Pricing Agreements > [Pricing Attributes] <b>Alternate:</b> Price Lists > Price List Setup > [Pricing Attributes]
Pricing Engine Request Viewer	Pricing Engine Request Viewer
Pricing Formulas	Pricing Formulas > Formulas Setup
Pricing Organizer	Pricing Organizer
Pricing Transaction Entity-Attribute Linking	Setup > Attribute Management > Attribute Linking and Mapping
Privileges	SSWA > Oracle Pricing Administrator Responsibility > Security > Privileges
Privileges Summary	SSWA > Oracle Pricing Administrator Responsibility > Security > Privileges > [Bulk Create Privileges] > Complete Bulk Create Privileges Step 1 to 3 > [Submit]
Purge Pricing Entities	Archive Purge Pricing Entities > Purge Entity
QUALIFIER - Header Level Qualifiers	Modifiers > Modifier Setup > [List Qualifiers] > [Cancel]
QUALIFIER - Line Level Qualifiers	Modifiers > Modifier Setup > [Line Qualifiers] > [Cancel]
Qualifier Group	Qualifier Setup
Qualifier Groups - List	Modifiers > Modifier Setup > [List Qualifiers]
Qualifier Groups - Line	Modifiers > Modifier Setup > [Line Qualifiers]

Window Name	Navigation Path
Qualifiers (HTML UI)	SSWA > Oracle Pricing User responsibility > Find or create Modifier/Price List > List Qualifiers > Qualifiers page [Append Group]  <b>Alternate:</b> SSWA > Oracle Pricing User responsibility > Find or create modifier line/price list line> List Qualifiers > Qualifiers page [Append Group]
Qualifiers: Append Group (HTML UI)	SSWA > Oracle Pricing User responsibility > Qualifiers page [Append Group]
Redeem Accruals	Modifiers > Accrual Redemption
Report: request id	View Concurrent Requests > [Find] > [View Output]
Request Detail	View Concurrent Requests > [Find] > [View Details...]
Request Diagnostics	View Concurrent Requests > [Find] > [Diagnostics]
Requests	View Concurrent Requests > [Find]
Source Systems	Setup > Source Systems
Submit a New Request	Reports  <b>Alternate:</b> View Concurrent Requests > [Submit a New Request...]
Submit Request	Reports > OK
Update Formula Prices	Pricing Formulas > Update Formula Prices

Window Name	Navigation Path
Update Line (modifier) (HTML UI)	<p>SSWA &gt; Oracle Pricing User responsibility &gt; Lists &gt; Modifier Lists &gt; Modifier Lists Search page (complete search for modifier) &gt; [Update] &gt; Modifier Lines (from Navigation bar) &gt; [Update]</p> <p><b>Alternate:</b> SSWA &gt; Oracle Pricing User responsibility &gt; Home page (Recently Created Modifier Lists) &gt; [Update] &gt; Modifier Lines (from Navigation bar) &gt; [Update]</p>
Update (Modifier Type): Modifier Lines (HTML UI)	<p>SSWA &gt; Oracle Pricing User responsibility &gt; Lists &gt; Modifier Lists &gt; Modifier Lists Search page (complete search for modifier) &gt; [Update] &gt; Modifier Lines (from Navigation bar)</p> <p><b>Alternate:</b> SSWA &gt; Oracle Pricing User responsibility &gt; Home page (Recently Created Modifier Lists) &gt; [Update] &gt; Modifier Lines (from Navigation bar)</p>
Update (Modifier Type): General Information (HTML UI)	<p>SSWA &gt; Oracle Pricing User responsibility &gt; Modifier Lists Search page (after completing a search) &gt; [Update]</p> <p><b>Alternate:</b> SSWA &gt; Oracle Pricing User responsibility &gt; Home page (Recently Created Modifier Lists) &gt; [Update]</p>
Value Sets	Setup > Attribute Management > Context and Attributes > [Value Sets]



---

## Attribute Seed Data

### Overview of Attribute Seed Data

The seeded (predefined) pricing attributes, product attributes, qualifier attributes, attribute contexts and default pricing attribute mapping rules for Oracle Advanced Pricing are provided for each of the following pricing transaction entities (PTE) used with Oracle Advanced Pricing:

- Complex Maintenance Repair and Overhaul PTE Attributes, page B-2
- Demand Planning PTE Attributes, page B-3
- Intercompany Transaction PTE Attributes, page B-4
- Logistics PTE Attributes, page B-7
- Order Fulfillment PTE Attributes, page B-14
- Procurement PTE Attributes, page B-30

To find attributes, refer to the PTE section where the attribute is sourced. For example, to find the Product attribute of *Item Category*, go to the Order Fulfillment PTE Attributes section and look under the Product Attributes heading to find the listed attribute.

The following table lists the short names and definitions used in the attribute tables:

Short name	Definition
AM	Attribute Mapping
AMM	Attribute Mapping Method

Short name	Definition
Attrib	Attribute
ASO	Oracle Order Capture
CMRO	Complex Maintenance Repair and Overhaul
Lmt	Limits
Lvl	Level <ul style="list-style-type: none"> <li>• LN: Line</li> <li>• BT: Both (Means at both the Line and Order levels)</li> <li>• OR: Order</li> </ul>
ONT	Order Management
OKC	Oracle Contracts Core
Prec	Precedence
Req Type	Request Type
UE	User Entered
—	No value/not applicable

## Complex Maintenance Repair and Overhaul PTE Attributes

This section displays the attributes for the Complex Maintenance Repair and Overhaul (CMRO) PTE.

### Pricing Attributes

The following table lists the seeded pricing attributes for the Complex Maintenance Repair and Overhaul PTE:.

Context	Attribute	Attrib Description	Mapped Value	Value Set Name	Lvl	AMM
AHL_PRICING	Resource Duration	Resource Duration	PRICING_ATTRIBUT E1	QP: Number	LIN E	UE

### Product Attributes

The following table lists the seeded product attributes for the Complex Maintenance Repair and Overhaul PTE:

Context	Attribute	Attrib Description	Mapped Value	Value Set Name	Lvl	AMM
ITEM	Item Number	Item Number	PRICING_ATTRIBUT TE1	--	LINE	UE

### Qualifier Attributes

This PTE has no seeded qualifier attributes.

## Demand Planning PTE Attributes

This section displays the attributes for the Demand Planning PTE.

### Pricing Attributes

This PTE has no seeded pricing attributes.

### Product Attributes

This PTE has no seeded product attributes.

### Qualifier Attributes

The following table lists the seeded qualifier attributes for the Demand Planning PTE:

Context	Attribute	Attrib Description	Mapped Value	Value Set Name	Lvl	AMM
MODLIST	Coupon Number	Not Used	QUALIFIER_ATTRIBUTES3	QP_SRS_COUPON_NO	BN	UE
MODLIST	List Line Number	Not Used	QUALIFIER_ATTRIBUTES2	QP_SRS_LIST_LINE_NUMBER	LN	UE
MODLIST	Promotion No	Not Used	QUALIFIER_ATTRIBUTES1	QP_SRS_PROMOTION_NO	BT	UE

Context	Attribute	Attribute Description	Mapped Value	Value Set Name	Lvl	AMM
ORDER	Sales Agreement Number	Not Used	QUALIFIER_ATTRIBUTES3	QP_BLANKET_NUMBER	LN	UE
PARTY	Buyer	Customer Party, displayed as level value in Geography dimension in Demand Planning.	QUALIFIER_ATTRIBUTES2	QP_PARTY	LN	UE
PARTY	Sales Organization	Selling Operating Unit, displayed as level value in Ship From dimension in Demand Planning.	QUALIFIER_ATTRIBUTES3	QP_SALES_ORGANIZATION	LN	UE
PARTY	Supplier	Supplier Party, displayed as level value in Geography dimension in Demand Planning.	QUALIFIER_ATTRIBUTES1	QP_PARTY	LN	UE

## Intercompany Transaction PTE Attributes

The following section displays the attributes for the Intercompany Transaction PTE.

### Pricing Attributes

This PTE has no seeded pricing attributes.

### Product Attributes

The following table lists the seeded product attributes for the Intercompany Transaction PTE:

Context	Attribute	Attrib Description	Mapped Value	Value Set Name	Lvl	AM M
ITEM	Item Number	Item Number	PRICING_ATTRIBUT1	--	LINE	AM

### Qualifier Attributes

The following table lists the seeded qualifier attributes for the Intercompany Transaction PTE:

Context	Attribute	Attrib Description	Mapped Value	Value Set Name	Lvl	AM M
GLOBAL_PROCUREMENT	Procuring Operating Unit	Procuring Operating Unit	QUALIFIER_ATTRIBUT1	HR_OPERATING_UNITS	LN	AM
GLOBAL_PROCUREMENT	Receiving Operating Unit	Receiving Operating Unit	QUALIFIER_ATTRIBUT2	HR_OPERATING_UNITS	LN	AM
GLOBAL_PROCUREMENT	Vendor Site	Vendor Site	QUALIFIER_ATTRIBUT4	INV_IC_VENDOR_SITE	LN	AM
GLOBAL_PROCUREMENT	Vendor	Vendor	QUALIFIER_ATTRIBUT3	INV_IC_VENDOR	LN	AM
INTERCOMPANY_INVOICING	Customer Site	Customer Site	QUALIFIER_ATTRIBUT4	INV_IC_CUSTOMER_SITE	LN	AM
INTERCOMPANY_INVOICING	Customer	Customer	QUALIFIER_ATTRIBUT3	INV_IC_CUSTOMER	LN	AM
INTERCOMPANY_INVOICING	Selling Operating Unit	Selling Operating Unit	QUALIFIER_ATTRIBUT2	HR_OPERATING_UNITS	LN	AM
INTERCOMPANY_INVOICING	Shipping Operating Unit	Shipping Operating Unit	QUALIFIER_ATTRIBUT1	HR_OPERATING_UNITS	LN	AM

Context	Attribute	Attribute Description	Mapped Value	Value Set Name	L	A
					v	M
					I	M
INTERORG_TRANSFER	From Organization Code	From Organization Code	QUALIFIER_A TTRIBUTE1	GMF_ORGANIZATIONS_ALL	L N	A M
INTERORG_TRANSFER	To Organization Code	To Organization Code	QUALIFIER_A TTRIBUTE2	GMF_ORGANIZATIONS_ALL	L N	A M
ITEM	Item Number	Item Number	PRICING_ATTRIBUTE1		L N	A M
MODLIST	Coupon Number	Coupon Number for applying Coupon Modifier	QUALIFIER_A TTRIBUTE3	QP_SRS_COUPON_NO	B T	UE
MODLIST	List LN Number	Used in conjunction with Promotion No attribute. Identifies Modifier LN Number for applying Ask For Promotion.	QUALIFIER_A TTRIBUTE2	QP_SRS_LIST_LN_NUMBER	L N	UE
MODLIST	Promotion No	Promotion Number for applying Ask for Promotion	QUALIFIER_A TTRIBUTE1	QP_SRS_PROMOTION_NO	B T	UE
ORDER	Order Date	Order Date	QUALIFIER_A TTRIBUTE1	QP: Date	B T	A M

Context	Attribute	Attrib Description	Mapped Value	Value Set Name	Lvl	AMM
ORDER	Sales Agreement Number	Sales Agreement Number	QUALIFIER_A TTRIBUTE3	QP_BLANKET_NUMBER	LN	UE
PARTY	Buyer	Buyer Party	QUALIFIER_A TTRIBUTE2	QP_PARTY	LN	UE
PARTY	SALES ORGANIZATION	Selling Operating Unit	QUALIFIER_A TTRIBUTE3	QP_SALES_ORGANIZATION	LN	UE
PARTY	Supplier	Supplier Party	QUALIFIER_A TTRIBUTE1	QP_PARTY	LN	UE

## Logistics PTE Attributes

The following section displays the attributes for the Logistics PTE.

### Pricing Attributes

The following table lists the seeded pricing attributes for the Logistics PTE:

Context	Attribute	Attrib Description	Mapped Value	Value Set Name	Lvl	AMM
LOGISTICS	Additional Service	Additional Service	PRICING_ATTRIBUTEBUTE4	QP_ADDITIONAL_SERVICE	LN	UE
LOGISTICS	Commodity	Commodity	PRICING_ATTRIBUTEBUTE1	QP: Number	LN	UE
LOGISTICS	Container Type	Container Type	PRICING_ATTRIBUTEBUTE2	QP_CONTAINER_TYPE	LN	UE
LOGISTICS	Destination Zone	Destination Zone	PRICING_ATTRIBUTEBUTE8	QP: Number	LN	UE

Context	Attribute	Attrib Description	Mapped Value	Value Set Name	Lvl	AMM
LOGISTICS	FREIGHT_CLASSES	Freight Class	PRICING_ATTRIBUTE6	QP_FREIGHT_CLASS	LN	UE
LOGISTICS	HAZARD_CODE	Hazard Code	PRICING_ATTRIBUTE5	-	LN	UE
LOGISTICS	Loading Protocol	Loading Protocol	PRICING_ATTRIBUTE17	QP: Text	BT	UE
LOGISTICS	Origin Zone	Origin Zone	PRICING_ATTRIBUTE7	QP: Number	LN	UE
LOGISTICS	Service Type	Service Type	PRICING_ATTRIBUTE3	QP_SERVICE_TYPE	LN	UE
LOGISTICS	TL Continuous Move Discount Flag	TL Continuous Move Discount Flag	PRICING_ATTRIBUTE18	QP: Yes/No	BT	UE
LOGISTICS	TL Deadhead Rate Variant Flag	TL Deadhead Rate Variant Flag	PRICING_ATTRIBUTE19	QP: Yes/No	BT	UE
LOGISTICS	TL Distance Type	TL Distance Type	PRICING_ATTRIBUTE12	QP: Text	BT	UE
LOGISTICS	TL Handling Activity	TL Handling Activity	PRICING_ATTRIBUTE22	QP: Yes/No	BT	UE
LOGISTICS	TL Number of Weekend Layovers	TL Number of Weekend Layovers	PRICING_ATTRIBUTE16	QP: Number	BT	UE
LOGISTICS	TL Rate Basis	TL Rate Basis	PRICING_ATTRIBUTE11	QP: Text	BT	UE

Context	Attribute	Attrib Description	Mapped Value	Value Set Name	Lvl	AMM
LOGISTICS	TL Rate Type	TL Rate Type	PRICING_ATTRI BUTE13	QP: Text	BT	UE
LOGISTICS	TL Service Type	TL Service Type	PRICING_ATTRI BUTE14	QP: Text	BT	UE
LOGISTICS	TL Stop Loading Activity	TL Stop Loading Activity	PRICING_ATTRI BUTE20	QP: Yes/No	BT	UE
LOGISTICS	TL Stop Unloading Activity	TL Stop Unloading Activity	PRICING_ATTRI BUTE21	QP: Yes/No	BT	UE
LOGISTICS	Total Shipment Quantity	Total Shipment Quantity	PRICING_ATTRI BUTE9	QP: Number	LN	UE
LOGISTICS	Vehicle	Vehicle	PRICING_ATTRI BUTE15	QP: Number	BT	UE
LOGISTICS	Facility Dropoff Containers	Facility Dropoff Containers	PRICING_ATTRI BUTE41	QP: Number	BT	UE
LOGISTICS	Facility Dropoff Pallets	Facility Dropoff Pallets	PRICING_ATTRI BUTE42	QP: Number	BT	UE
LOGISTICS	Facility Dropoff Volume	Facility Dropoff Volume	PRICING_ATTRI BUTE40	QP: Number	BT	UE
LOGISTICS	Facility Dropoff Weight	Facility Dropoff Weight	PRICING_ATTRI BUTE39	QP: Number	BT	UE
LOGISTICS	Facility Handling Containers	Facility Handling Containers	PRICING_ATTRI BUTE45	QP: Number	BT	UE

Context	Attribute	Attrib Description	Mapped Value	Value Set Name	Lvl	AM M
LOGISTICS	Facility Handling Pallets	Facility Handling Pallets	PRICING_ATTRI BUTE46	QP: Number	BT	UE
LOGISTICS	Facility Handling Volume	Facility Handling Volume	PRICING_ATTRI BUTE44	QP: Number	BT	UE
LOGISTICS	Facility Handling Weight	Facility Handling Weight	PRICING_ATTRI BUTE43	QP: Number	BT	UE
LOGISTICS	Facility Pickup Container	Facility Pickup Container	PRICING_ATTRI BUTE37	QP: Number	BT	UE
LOGISTICS	Facility Pickup Pallets	Facility Pickup Pallets	PRICING_ATTRI BUTE38	QP: Number	BT	UE
LOGISTICS	Facility Pickup Volume	Facility Pickup Volume	PRICING_ATTRI BUTE36	QP: Number	BT	UE
LOGISTICS	Facility Pickup Weight	Facility Pickup Weight	PRICING_ATTRI BUTE35	QP: Number	BT	UE
LOGISTICS	Group Amount	Group Amount	PRICING_ATTRI BUTE5	QP: Number	BT	UE
LOGISTICS	Group Quantity	Group Quantity	PRICING_ATTRI BUTE4	QP: Number	BT	UE
LOGISTICS	Item Amount	Item Amount	PRICING_ATTRI BUTE12	QP: Number	LN	UE
LOGISTICS	Item Quantity	Item Quantity	PRICING_ATTRI BUTE10	QP: Number	LN	UE

Context	Attribute	Attrib Description	Mapped Value	Value Set Name	Lvl	AMM
LOGISTICS	TL Charged Out of Route Distance	TL Charged Out of Route Distance	PRICING_ATTRI BUTE22	QP: Number	BT	UE
LOGISTICS	TL Dropoff Containers	TL Dropoff Containers	PRICING_ATTRI BUTE31	QP: Number	BT	UE
LOGISTICS	TL Dropoff Pallets	TL Dropoff Pallets	PRICING_ATTRI BUTE32	QP: Number	BT	UE
LOGISTICS	TL Dropoff Volume	TL Dropoff Volume	PRICING_ATTRI BUTE30	QP: Number	BT	UE
LOGISTICS	TL Dropoff Weight	TL Dropoff Weight	PRICING_ATTRI BUTE29	QP: Number	BT	UE
LOGISTICS	TL Handling Volume	TL Handling Volume	PRICING_ATTRI BUTE34	QP: Number	BT	UE
LOGISTICS	TL Handling Weight	TL Handling Weight	PRICING_ATTRI BUTE33	QP: Number	BT	UE
LOGISTICS	TL Number Of Stops	TL Number Of Stops	PRICING_ATTRI BUTE21	QP: Number	BT	UE
LOGISTICS	TL Number of Weekday Layovers	TL Number of Weekday Layovers	PRICING_ATTRI BUTE23	QP: Number	BT	UE
LOGISTICS	TL Pickup Containers	TL Pickup Containers	PRICING_ATTRI BUTE27	QP: Number	BT	UE
LOGISTICS	TL Pickup Pallets	TL Pickup Pallets	PRICING_ATTRI BUTE28	QP: Number	BT	UE
LOGISTICS	TL Pickup Volume	TL Pickup Volume	PRICING_ATTRI BUTE26	QP: Number	BT	UE

Context	Attribute	Attrib Description	Mapped Value	Value Set Name	Lvl	AMM
LOGISTICS	TL Pickup Weight	TL Pickup Weight	PRICING_ATTRIBUTUTE25	QP: Number	BT	UE
LOGISTICS	TL Weekend Layover Mileage	TL Weekend Layover Mileage	PRICING_ATTRIBUTUTE24	QP: Number	BT	UE
LOGISTICS	Total Item Quantity	Total Item Quantity	PRICING_ATTRIBUTUTE20	QP: Number	LN	UE

### Product Attributes

This table displays the seeded product attributes for the Logistics PTE:

Context	Attribute	Attrib Description	Mapped Value	Value Set Name	Lvl	AMM
ITEM	All Items	All Items	PRICING_ATTRIBUTE3	QP:ITEM_ALL	LN	UE

### Qualifier Attributes

This table displays the seeded qualifier attributes for the Logistics PTE:

Context	Attribute	Attrib Description	Mapped Value	Value Set Name	Lvl	AMM
LOGISTICS	Additional Service	Additional Service	QUALIFIER_ATTRIBUTE8	QP_ADDITIONAL_SERVICE	LN	UE
LOGISTICS	Contract Reference	Contract Reference	QUALIFIER_ATTRIBUTE4	QP_LANE_CONTRACT_REFERENCE	LN	UE
LOGISTICS	Contracted Lane	Contracted Lane	QUALIFIER_ATTRIBUTE9	QP: Yes/No	LN	UE
LOGISTICS	Destination	Destination	QUALIFIER_ATTRIBUTE6	QP: Number	LN	UE

Context	Attribute	Attrib Description	Mapped Value	Value Set Name	Lv I	AM M
LOGISTICS	Mode of Transportation	Mode of Transportation	QUALIFIER_AT TRIBUTE7	QP: Text	L N	UE
LOGISTICS	Origin	Origin	QUALIFIER_AT TRIBUTE5	QP: Number	L N	UE
LOGISTICS	Service Type	Service Type	QUALIFIER_AT TRIBUTE10	QP: Text	L N	UE
MODLIST	List Line Number	Used in conjunction with Promotion No attribute, identifies Modifier Line Number for applying Ask For Promotion	QUALIFIER_AT TRIBUTE2	QP_SRS_LIST_LN _NUMBER	L N	UE
MODLIST	Price List	Price List	QUALIFIER_AT TRIBUTE4	QP_SRS_PRICE_L IST_NAME	L N	UE
MODLIST	Promotion No	Promotion Number for applying Ask for Promotions	QUALIFIER_AT TRIBUTE1	QP_SRS_PROMO TION_NO	BT	UE
ORDER	Sales Agreement Number	Sales Agreement Number	QUALIFIER_AT TRIBUTE3	QP_BLANKET_N UMBER	L N	UE
PARTY	Buyer	Buyer Party	QUALIFIER_AT TRIBUTE2	QP_PARTY	L N	UE
PARTY	SALES ORGANIZATION	Selling Operating Unit	QUALIFIER_AT TRIBUTE3	QP_SALES_ORG ANIZATION	L N	UE

Context	Attribute	Attrib Description	Mapped Value	Value Set Name	Lv I	AM M
PARTY	Supplier	Supplier Party	QUALIFIER_ATTRIBUTE1	QP_PARTY	L N	UE

## Order Fulfillment PTE Attributes

The following section displays the attributes for the Order Fulfillment PTE.

### Pricing Attributes

The following table lists the seeded pricing attributes for the Order Fulfillment PTE:

Context	Attribute	Attrib Description	Mapped Value	Value Set Name	Lv I	AM M
BREAK_UOM	Usage UOM	Unit of Measure for usage-based pricing. This attribute is used to prorate price breaks. Currently, only supported for Oracle Service Contracts integration. <sup>1</sup>	PRICING_ATTRIBUTE1	--	L N	A M
OTA	Number of students	Number of students	PRICING_ATTRIBUTE1	QP: Integer	L N	UE
PERIODICITY	Charge Periodicity	Charge periodicity for a recurring charge line such as Yearly, Monthly, Weekly. <sup>2</sup>	PRICING_ATTRIBUTE1	QP_CHARGE_PERIODICITY	L N	A M
PRICING_ATTRIBUTE	Administration Cost	Used in cost to charge conversion formulas.	PRICING_ATTRIBUTE1	QP: Number	L N	A M
PRICING_ATTRIBUTE	Customer Item	Customer Item	PRICING_ATTRIBUTE1	--	L N	A M

Context	Attribute	Attrib Description	Mapped Value	Value Set Name	Lvl	AMM
PRICING ATTRIBUTE	Duty Cost	Used in cost to charge conversion formulas.	PRICING_ATTRIBUTE15	QP: Number	L N	A M
PRICING ATTRIBUTE	Estimated Transportation Charge	Used in cost to charge conversion formulas.	PRICING_ATTRIBUTE24	QP: Number	L N	A M
PRICING ATTRIBUTE	Estimated Transportation Price	Used in cost to charge conversion formulas.	PRICING_ATTRIBUTE23	QP: Number	L N	A M
PRICING ATTRIBUTE	Export Cost	Used in cost to charge conversion formulas.	PRICING_ATTRIBUTE14	QP: Number	L N	A M
PRICING ATTRIBUTE	Freight Cost	Can be used in cost to charge conversion formulas.	PRICING_ATTRIBUTE16	QP: Number	L N	A M
PRICING ATTRIBUTE	Grade	Grade for Process Items	PRICING_ATTRIBUTE19	OPM_QC_GRADE	L N	A M
PRICING ATTRIBUTE	Handling Cost	Can be used in cost to charge conversion formulas.	PRICING_ATTRIBUTE13	QP: Number	L N	A M
PRICING ATTRIBUTE	Insurance Cost	Can be used in cost to charge conversion formulas.	PRICING_ATTRIBUTE12	QP: Number	L N	A M
PRICING ATTRIBUTE	Model Id	For a line in a configuration, this attribute identifies the item on the top model line	PRICING_ATTRIBUTE1	QP_MODEL_ITEM_ID	L N	A M

Context	Attribute	Attribute Description	Mapped Value	Value Set Name	Level	Access Mode
PRICING ATTRIBUTE	Parent List Price	Attribute is applicable only to service items to determine the list price of product parent line that the service is for.	PRICING_ATTRIBUTE TE10	QP: Number	L N	A M
PRICING ATTRIBUTE	Transportation Charge	Transportation Charge	PRICING_ATTRIBUTE TE21	QP: Number	L N	A M
PRICING ATTRIBUTE	Transportation Price	Transportation Price	PRICING_ATTRIBUTE TE20	QP: Number	L N	A M
QP_INTER NAL	List Line Id	Used for price locking to ensure that the price specified on Sales Agreement or Service Contract is applied and only to transactions referencing that agreement/contract.	PRICING_ATTRIBUTE TE1	QP: Integer	L N	A M
VOLUME	Sales Agreement Amount	Released Amount from Sales Agreement Header	PRICING_ATTRIBUTE TE6	QP: Number	L N	A M
VOLUME	Sales Agreement Line Amount	Released Amount from Sales Agreement Line	PRICING_ATTRIBUTE TE7	QP: Number	L N	A M
VOLUME	Sales Agreement Line Quantity	Released Quantity from Sales Agreement Line	PRICING_ATTRIBUTE TE8	QP: Number	L N	A M

Context	Attribute	Attrib Description	Mapped Value	Value Set Name	Lv	AMM
VOLUME	Item Amount	Item Amount is calculated as Quantity Unit multiplied by List Price. This value is sourced by pricing engine at run-time.	PRICING_ATTRIBU TE12	QP: Number	L N	UE
VOLUME	Item Quantity	This value is sourced by pricing engine at run-time.	PRICING_ATTRIBU TE10	QP: Number	L N	UE
VOLUME	Period1 Item Amount	Item amount ordered by the customer in Period 1. <sup>3</sup>	PRICING_ATTRIBU TE13	QP: Number	L N	AM
VOLUME	Period1 Item Quantity	Item quantity ordered by the customer in Period 1. <sup>3</sup>	PRICING_ATTRIBU TE3	QP: Number	L N	AM
VOLUME	Period2 Item Amount	Item amount ordered by the customer in Period 2. <sup>3</sup>	PRICING_ATTRIBU TE14	QP: Number	L N	AM
VOLUME	Period2 Item Quantity	Item quantity ordered by the customer in Period 2. <sup>3</sup>	PRICING_ATTRIBU TE1	QP: Number	L N	AM
VOLUME	Period3 Item Amount	Item amount ordered by the customer in Period 3. <sup>3</sup>	PRICING_ATTRIBU TE15	QP: Number	L N	AM
VOLUME	Period3 Item Quantity	Item quantity ordered by the customer in Period 3. <sup>3</sup>	PRICING_ATTRIBU TE11	QP: Number	L N	AM

1. For more information, see *Oracle Advanced Pricing Implementation Guide*.
2. For more information, see *Oracle Advanced Pricing Implementation Guide*, Recurring Charges.
3. For more information, *Oracle Advanced Pricing Implementation Guide* and *Oracle Advanced Pricing User's Guide*, Cross Order Volume sections.

### Product Attributes

The following section displays the product attributes for the Order Fulfillment PTE:

Con text	Attribute	Attrib Description	Mapped Value	Value Set Name	Lvl	AM M
ITEM	All Items	Use this attribute if price list or price modifier should be available to All Items.	PRICING_ATTRIBUTEBUTE3	QP: ITEM_ALL	LN	AM
ITEM	Item Category	Item Category, derived from Item setup in Oracle Inventory. Use this attribute if price list or price modifier should be available to items within the Item Category.	PRICING_ATTRIBUTEBUTE2	QP_ITEM_CATEGORIES	LN	AM
ITEM	Item Number	Item Number	PRICING_ATTRIBUTEBUTE1	--	LN	AM
ITEM	SEGMENT_1 - SEGMENT_20	Segments 1-20 from the Inventory Item key flexfield.	PRICING_ATTRIBUTEBUTE4	--	LN	AM
ITEM	iStore Section	Sections in Oracle Istore that contain the order item.	PRICING_ATTRIBUTEBUTE24	AMS_ISTORE_SECTION	LN	AM

### Qualifier Attributes

The following section displays the seeded qualifier attributes for the Order Fulfillment

PTE:

Context	Attribute	Attribute Description	Mapped Value	Value Set Name	Lvl	AMM
ASOPARTY INFO	Bill to Party Site	Bill To Party Site	QUALIFIER_AT TRIBUTE11	QP_SHIP_TO_PA RTY_SITE	B T	A M
ASOPARTY INFO	Customer Party	Customer Party	QUALIFIER_AT TRIBUTE1	QP_CUSTOMER_ PARTY	B T	A M
ASOPARTY INFO	Ship to Party Site	Ship To Party Site	QUALIFIER_AT TRIBUTE10	QP_SHIP_TO_PA RTY_SITE	B T	A M
CUSTOMER	Account Type	Customer Profile Class - derived from Customer Account setup in Oracle TCA	QUALIFIER_AT TRIBUTE12	QP_ACCOUNT_T YPE	B T	A M
CUSTOMER	Agreement Name	Pricing Agreement Name	QUALIFIER_AT TRIBUTE7	QP_AGREEMENT _NAME	B T	A M
CUSTOMER	Agreement Type	Pricing Agreement Type - derived from Pricing Agreement setup	QUALIFIER_AT TRIBUTE8	QP_AGREEMENT _TYPE	B T	A M
CUSTOMER	Bill To	Bill To Location	QUALIFIER_AT TRIBUTE14	QP_INVOICE_TO _ORGS	B T	A M
CUSTOMER	Customer Class	Customer Class - derived from Customer Account set up in Oracle TCA.	QUALIFIER_AT TRIBUTE1	QP_CUSTOMER_ CLASS	B T	A M
CUSTOMER	Customer Name	Customer Name	QUALIFIER_AT TRIBUTE2	QP_CUSTOMERS	B T	A M

Context	Attribute	Attribute Description	Mapped Value	Value Set Name	L vl	A M M
CUSTOMER	End Customer Name	End Customer Name	QUALIFIER_ATTRIBUTES20	QP_CUSTOMERS	L N	A M
CUSTOMER	GSA	Identifies if this is a GSA customer - derived from Customer Account setup in Oracle TCA. <sup>1</sup>	QUALIFIER_ATTRIBUTES15	QP: Yes/No	B T	A M
CUSTOMER	Invoice To Party Site	Same as Bill To Party Site attribute under Party qualifier context. This attribute (unlike Party qualifier context) is also mapped for Order Management (OM) request type. Use this attribute to have the qualifier apply to orders for OM request type.	QUALIFIER_ATTRIBUTES18	QP_INVOICE_TO_PARTY_SITE_ID	B T	A M

Context	Attribute	Attribute Description	Mapped Value	Value Set Name	Lvl	AMM
CUSTOMER	Party ID	Same as Customer Party attribute under Party qualifier context. This attribute (unlike Party qualifier context) is also mapped for OM request type. Use this attribute to have the qualifier apply to orders for OM request type.	QUALIFIER_ATTRIBUTE16	QP_PARTY	BT	AM
CUSTOMER	Sales Channel	Sales Channel	QUALIFIER_ATTRIBUTE13	QP_SALES_CHANNEL_CODE	BT	AM
CUSTOMER	Ship To Party Site	Same as Ship To Party Site attribute under Party qualifier context. This attribute (unlike Party qualifier context) is also mapped for OM request type. Use this attribute to have the qualifier apply to orders for OM request type.	QUALIFIER_ATTRIBUTE17	QP_SHIP_TO_PARTY_SITE_ID	BT	AM
CUSTOMER	Ship To	Ship To Location	QUALIFIER_ATTRIBUTE11	QP_SHIP_TO_ORGS	BT	AM

Context	Attribute	Attribute Description	Mapped Value	Value Set Name	L vl	A M M
CUSTOMER	Customer Account Site	Sources both ship to location and bill to location. Use this attribute if a pricing qualifier should be applicable when the account site is either ship to location or bill to location on the order. Use this method rather than create two qualifiers, one each for the ship to location attribute and bill to location attribute.	QUALIFIER_AT TRIBUTE5	QP_CUSTOMER_SITES	B T	A M
CUSTOMER_GROUP	Buying Groups	Buying Groups that Customer or Customer Party belongs to. This is based on setups in Oracle Marketing.	QUALIFIER_AT TRIBUTE3	AMS_BUYING_GROUP	B T	A M
CUSTOMER_GROUP	Lists	Marketing Lists that Customer or Customer Party belongs to. This is based on setups in Oracle Marketing.	QUALIFIER_AT TRIBUTE1	AMS_LISTS	B T	A M

Context	Attribute	Attribute Description	Mapped Value	Value Set Name	L vl	A M M
CUSTOMER_GROUP	Segments	Marketing Segments that Customer or Customer Party belongs to. This is based on setups in Oracle Marketing.	QUALIFIER_ATTRIBUTES2	AMS_SEGMENTS	B T	A M
MODLIST	Coupon Number	Coupon Number for applying Coupon Modifier	QUALIFIER_ATTRIBUTES3	QP_SRS_COUPON_NO	B T	U E
MODLIST	List Line Number	Used in conjunction with Promotion No (number) attribute, identifies Modifier Line Number for applying Ask For Promotion.	QUALIFIER_ATTRIBUTES2	QP_SRS_LIST_LINE_NUMBER	L N	U E
MODLIST	Price List	This qualifier is used when a modifier should apply only if a specific price list is used.	QUALIFIER_ATTRIBUTES4	QP_SRS_PRICE_LIST_NAME	B T	A M
MODLIST	Promotion No	Promotion Number for applying Ask for Promotion	QUALIFIER_ATTRIBUTES1	QP_SRS_PROMOTION_NO	B T	U E
ORDER	Customer PO	Customer PO Number	QUALIFIER_ATTRIBUTES12	QP_CUSTOMER_PO	B T	A M

Context	Attribute	Attribute Description	Mapped Value	Value Set Name	L vl	A M M
ORDER	Freight Cost Type Code	Identifies cost types with associated costs for the order line. Oracle Order Management gets cost/charge information at time of shipping and stores this in price adjustment tables with list line type code of COST.	QUALIFIER_AT TRIBUTE20	QP_FREIGHT_COST_TYPE	L N	A M
ORDER	Line Category	Line Category - Order or Return	QUALIFIER_AT TRIBUTE19	QP_LN_CATEGORY	L N	A M
ORDER	Line Type	Transaction type for the order line	QUALIFIER_AT TRIBUTE2	QP_LN_TYPE	L N	A M
ORDER	Order Category	Order Category - Order, Return or Mixed	QUALIFIER_AT TRIBUTE13	QP_ORDER_CATEGORY	B T	A M
ORDER	Order Date	Order Date	QUALIFIER_AT TRIBUTE1	QP: Date	B T	A M
ORDER	Order Type	Transaction type for the order header	QUALIFIER_AT TRIBUTE9	QP_ORDER_TYPES_ALL	B T	A M
ORDER	Pricing Date	Pricing Date	QUALIFIER_AT TRIBUTE14	QP: Date	B T	A M
ORDER	Quote Source Code	Quote Source Code	QUALIFIER_AT TRIBUTE4	--	O R	A M
ORDER	Request Date	Request Date	QUALIFIER_AT TRIBUTE17	QP: Date	B T	A M

Context	Attribute	Attribute Description	Mapped Value	Value Set Name	Lvl	AMM
ORDER	Sales Agreement ID	Sales Agreement	QUALIFIER_AT TRIBUTE5	QP_BLANKET_H EADER_ID	B T	A M
ORDER	Sales Agreement Line Number	Sales Agreement Line Number	QUALIFIER_AT TRIBUTE6	QP_BLANKET_L N_ID	L N	A M
ORDER	Sales Order Number	Sales Order Number	QUALIFIER_AT TRIBUTE21	QP_SALES_ORDE R_NUMBER	B T	A M
ORDER	Ship From	Ship From Location (also referred to as Warehouse)	QUALIFIER_AT TRIBUTE18	QP_SHIP_FROM	B T	A M
ORDER	Shipment Priority Code	Shipment Priority Code	QUALIFIER_AT TRIBUTE16	QP_SHIPMENT_P RORITY	B T	A M
ORDER	Shippable Flag	Identifies if the item is shippable.	QUALIFIER_AT TRIBUTE10	QP: Yes/No	B T	A M
ORDER	Shipped Date	Shipped Date	QUALIFIER_AT TRIBUTE8	QP: Date	B T	A M
ORDER	Shipped Flag	Identifies if the item has been shipped	QUALIFIER_AT TRIBUTE11	QP: Yes/No	L N	A M
ORDER	Source Type	External (for drop ship order lines) or Internal	QUALIFIER_AT TRIBUTE15	QP_SOURCE_TYP E	L N	A M
PARTY	Buyer	Buyer Party	QUALIFIER_AT TRIBUTE2	QP_PARTY	L N	U E

Context	Attribute	Attribute Description	Mapped Value	Value Set Name	Lvl	AMM
PARTY	SALES ORGANIZATION	Selling Operating Unit	QUALIFIER_AT TRIBUTE3	QP_SALES_ORGANIZATION	B T	A M
PARTY	Supplier	Supplier Party	QUALIFIER_AT TRIBUTE1	QP_PARTY	L N	U E
SEGMENT	Target Segment	Target Segment	QUALIFIER_AT TRIBUTE2	--	B T	A M
SOLD_BY	Distributor List	Oracle Trade Management tracks sales from distributors to retailers, also called indirect sales. Distributor List allows for offers to be targeted to indirect sales from certain distributors. <sup>2</sup>	QUALIFIER_AT TRIBUTE3	AMS_LISTS	B T	A M
SOLD_BY	Distributor Name	Enables offers to be targeted to indirect sales from a specific distributor. <sup>2</sup>	QUALIFIER_AT TRIBUTE2	QP_CUSTOMERS	B T	A M
SOLD_BY	Distributor Segment	Segments that the distributor belongs to. <sup>2</sup>	QUALIFIER_AT TRIBUTE4	AMS_SEGMENTS	B T	A M
SOLD_BY	Distributor Territory	Territories that the distributor belongs to. <sup>2</sup>	QUALIFIER_AT TRIBUTE5	AMS_TM_OFFER_TERRITORIES	B T	A M

Context	Attribute	Attribute Description	Mapped Value	Value Set Name	Lvl	AMM
SOLD_BY	Sales Method	Use this attribute if offer is to be extended only to direct or only to indirect sales. <sup>2</sup>	QUALIFIER_AT TRIBUTE1	OZF_SALES_MET HOD	B T	A M
STORE	MINISITE_ID	Minisite ID for orders placed from Oracle Istore	QUALIFIER_AT TRIBUTE1	QP_IBE_MINISIT ES	B T	A M
TERMS	Freight Terms	Freight Terms	QUALIFIER_AT TRIBUTE10	QP_FREIGHT_TE RMS	B T	A M
TERMS	Payment Terms	Payment Terms	QUALIFIER_AT TRIBUTE1	QP_PAYMENT_T ERMS	B T	A M
TERMS	Shipping Method	Shipping Method	QUALIFIER_AT TRIBUTE11	QP_SHIP_METH ODS	B T	A M
TERRITORY	Sales Territories	Sales territories in Oracle Territory Manager that the customer belongs to. <sup>3</sup>	QUALIFIER_AT TRIBUTE2	AMS_SALES_AC CT_TERRITORIES	B T	A M
TERRITORY	Trade Management Territories	Trade Management Territories	QUALIFIER_AT TRIBUTE1	AMS_TM_OFFER _TERRITORIES	B T	A M
VOLUME	Line Volume	The volume is derived by converting line item quantity from order line UOM to volume UOM specified in profile QP: Line Volume UOM Code. <sup>4</sup>	QUALIFIER_AT TRIBUTE15	QP: Number	L N	A M

Context	Attribute	Attribute Description	Mapped Value	Value Set Name	L vl	A M M
VOLUME	Line Weight	The line weight is derived by converting line item quantity from order line UOM to weight UOM specified in profile QP: Line Weight UOM Code. <sup>4</sup>	QUALIFIER_AT TRIBUTE14	QP: Number	B T	A M
VOLUME	Order Amount	Sum of unit list price times quantity on all order lines for the current order being priced. This calculation excludes return lines and recurring charge lines. <sup>5</sup>	QUALIFIER_AT TRIBUTE10	QP: Number	B T	A M
VOLUME	Order Volume	Total volume of all order lines on the order, derived by converting each line quantity from order line UOM to volume UOM specified in profile QP: Line Volume UOM Code and then adding up line quantities in this volume UOM. <sup>4</sup>	QUALIFIER_AT TRIBUTE17	QP: Number	O R	A M

Context	Attribute	Attribute Description	Mapped Value	Value Set Name	L vl	A M M
VOLUME	Order Weight	Total weight of all order lines on the order, derived by converting each line quantity from order line UOM to weight UOM specified in profile QP: Line Weight UOM Code and then adding up line quantities in this weight UOM. <sup>4</sup>	QUALIFIER_AT TRIBUTE16	QP: Number	O R	A M
VOLUME	Period1 Order Amount	Amount ordered (across all items) by a customer in Period 1. <sup>4</sup>	QUALIFIER_AT TRIBUTE12	QP: Number	B T	A M
VOLUME	Period2 Order Amount	Amount ordered (across all items) by a customer in Period 2. <sup>69</sup>	QUALIFIER_AT TRIBUTE13	QP: Number	B T	A M
VOLUME	Period3 Order Amount	Amount ordered (across all items) by a customer in Period 3. <sup>6</sup>	QUALIFIER_AT TRIBUTE11	QP: Number	B T	A M

1. For more information on GSA Pricing, see *Oracle Advanced Pricing Implementation Guide* and *Oracle Advanced Pricing User's Guide*.
2. For more information, see Oracle Trade Management documentation.
3. For more information, see Oracle Territory Manager documentation.

4. For more information, see *Oracle Advanced Pricing Implementation Guide*, Profile Options.
5. For more information, see *Oracle Advanced Pricing Implementation Guide*, Recurring Charges.
6. For more information, see *Oracle Advanced Pricing Implementation Guide* and *Oracle Advanced Pricing User's Guide*, Cross Order Volume sections.

## Procurement PTE Attributes

The following section displays the attributes for the Procurement (PO) PTE.

### Pricing Attributes

The following table lists the seeded pricing attributes for the Procurement PTE:

Context	Context Type	Attribute	Attrib Description	Mapped Value	Value Set Name	L v I	AM M
PO_PRICING_ATTRIBUTES	PRICING_ATTRIBUTES	Destination Organization	Destination organization for the shipment line	PRICING_ATTRIBUTES13	-	L N	A M
PO_PRICING_ATTRIBUTES	PRICING_ATTRIBUTES	Hazardous Material Class	Hazardous Material Class	PRICING_ATTRIBUTES9	-	L N	A M
PO_PRICING_ATTRIBUTES	PRICING_ATTRIBUTES	Hazardous Material Code	Hazardous Material Code	PRICING_ATTRIBUTES10	-	L N	A M
PO_PRICING_ATTRIBUTES	PRICING_ATTRIBUTES	Item Revision	Item Revision	PRICING_ATTRIBUTES2	QP: Text	L N	A M

Context	Context Type	Attribute	Attrib Description	Mapped Value	Value Set Name	L v I	AM M
PO_PRICING _ATTRIBUTE S	PRICING_A TTRIBUTE	PO Agreement Line Number	Source Document Line Reference on Requisition or PO Line. Value is dependent on PO Agreement Number.	PRICING_AT TRIBUTE6	QP_PO _AGRE EMEN T_LN_ NUM	L N	A M
PO_PRICING _ATTRIBUTE S	PRICING_A TTRIBUTE	PO Vendor Item Number	Supplier Item Number on Requisition or PO Line	PRICING_AT TRIBUTE1	QP: Text	L N	A M
PO_PRICING _ATTRIBUTE S	PRICING_A TTRIBUTE	Primary Unit of Measure	Primary unit of measure for the shipment line	PRICING_AT TRIBUTE12	--	L N	A M
PO_PRICING _ATTRIBUTE S	PRICING_A TTRIBUTE	Purchasing Org	Operating Unit in which Requisition or Purchase Order was created.	PRICING_AT TRIBUTE3	QP_PO _ORG	O R	A M

Context	Context Type	Attribute	Attrib Description	Mapped Value	Value Set Name	L v I	AM M
PO_PRICING_ATTRIBUTE	PRICING_ATTRIBUTES	Ship To Location	Deliver-To Location on the Requisition OR Default Ship-To Location on PO Header OR Ship-To Location on PO Shipment.	PRICING_ATTRIBUTES5	QP_PO_SHIP_TO_LO_CATIO	N	A M

Context	Context Type	Attribute	Attrib Description	Mapped Value	Value Set Name	L v	AM M
PO_PRICING_ATTRIBUTE_S	PRICING_ATTRIBUTES	Ship To Org	Can be one of the following: <ul style="list-style-type: none"> <li>Ship-To Organization on PO Shipment</li> <li>Organization associated with the Ship-To Location on PO Header</li> <li>Organization associated with the Ship-To Location in the Financial options</li> </ul>	PRICING_ATTRIBUTES4	QP_PO_SHIP_TO_ORG	L N	A M
PO_PRICING_ATTRIBUTE_S	PRICING_ATTRIBUTES	Shipment Header Identifier	Shipment Header ID	PRICING_ATTRIBUTES7	--	L N	A M
PO_PRICING_ATTRIBUTE_S	PRICING_ATTRIBUTES	Shipment Line Identifier	Shipment Line ID	PRICING_ATTRIBUTES8	--	L N	A M

Context	Context Type	Attribute	Attrib Description	Mapped Value	Value Set Name	Lvl	AM
VOLUME	PRICING_ATTRIBUTES	Item Amount	Item Amount is calculated as Quantity times Unit List Price. This value is sourced by pricing engine at run-time.	PRICING_ATTRIBUTES12	QP: Number	LN	UE
VOLUME	PRICING_ATTRIBUTES	Item Quantity	Item Quantity. This value is sourced by pricing engine at run-time.	PRICING_ATTRIBUTES10	QP: Number	LN	UE

### Product Attributes

The following table lists the seeded product attributes for the Procurement PTE:

Context	Attribute	Attrib Description	Mapped Value	Value Set Name	Lvl	AM
ITEM	All Items	Use this attribute if price list or price modifier should be available to All Items.	PRICING_ATTRIBUTES3	QP: ITEM_ALL	LN	AM
ITEM	Item Category	Item Category, derived from Item setup in Oracle Inventory. Use this attribute if price list or price modifier should be available to items within the Item Category.	PRICING_ATTRIBUTES2	QP_ITEM_CATEGORIES	LN	AM
ITEM	Item Number	Item Number	PRICING_ATTRIBUTES1		LN	AM

### Qualifier Attributes

The following table lists the seeded qualifier attributes for the Procurement PTE:

Context	Attribute	Attrib Description	Mapped Value	Value Set Name	Lvl	AM M
PO_BUY ER	PO Agreement Number	Contract or Blanket Agreement Number	QUALIFIER_A TTRIBUTE2	QP_PO_AGREE MENT_NUMBER	LN	AM
PO_BUY ER	PO Agreement Type	Contract or Blanket Agreement	QUALIFIER_A TTRIBUTE1	QP_PO_AGREE MENT_TYPE	LN	AM
PO_BUY ER	PO Ship To Location	Deliver-To Location on the Requisition OR Default Ship-To Location on PO Header OR Ship- To Location on PO Shipment	QUALIFIER_A TTRIBUTE4	QP_PO_SHIP_TO _LOCATION	BT	AM
PO_BUY ER	PO Ship To Org	Can be one of the following: <ul style="list-style-type: none"> <li>Ship-To Organization on PO Shipment</li> <li>Organizatio n associated with the Ship-To Location on PO Header</li> <li>Organizatio n associated with the Ship-To Location in the Financial options</li> </ul>	QUALIFIER_A TTRIBUTE3	QP_PO_SHIP_TO _ORG	BT	AM
PO_OR	PO Creation Date	Requisition or PO Creation Date	QUALIFIER_A TTRIBUTE2	QP: Date	OR	AM

Context	Attribute	Attrib Description	Mapped Value	Value Set Name	Lvl	AM M
PO_OR	PO Need By Date	Need-By Date on the Requisition Line OR Need-By Date on the PO Line OR Need-By Date on the PO Shipment	QUALIFIER_A TTRIBUTE4	QP: Date	LN	AM
PO_OR	PO Order Type	Requisition or Standard Purchase Order	QUALIFIER_A TTRIBUTE3	QP_PO_OR_TYP E	OR	AM
PO_OR	Shipment number	Shipment Number	QUALIFIER_A TTRIBUTE6	--	OR	AM
PO_OR	Source Document Code	Source Document Code for the Shipment	QUALIFIER_A TTRIBUTE5	--	LN	AM
PO_OR	Source Organization	Source organization associated with the shipment line	QUALIFIER_A TTRIBUTE1	--	OR	AM
PO_SUPPLIER	Carrier Name	Carrier Name for the shipment line	QUALIFIER_A TTRIBUTE3	--	OR	AM
PO_SUPPLIER	PO Vendor Site	Supplier Site	QUALIFIER_A TTRIBUTE2	QP_PO_VENDOR_SITE	BT	AM
PO_SUPPLIER	PO Vendor	Supplier	QUALIFIER_A TTRIBUTE1	QP_PO_VENDOR	BT	AM
RCV_RECEIPT	Carrier transportation method code	Transportation Method Code for the shipment line	QUALIFIER_A TTRIBUTE1	--	OR	AM

Context	Attribute	Attrib Description	Mapped Value	Value Set Name	Lvl	AM M
RCV_RE CEIPT	Expected arrival date	Expected arrival date	QUALIFIER_A TTRIBUTE2	--	OR	AM
RCV_RE CEIPT	Shipment Header Identifier	Shipment Header ID	QUALIFIER_A TTRIBUTE5	--	OR	AM
RCV_RE CEIPT	Shipment packaging code	Shipment packaging code	QUALIFIER_A TTRIBUTE3	--	OR	AM
RCV_RE CEIPT	Shipping Date	Shipping date	QUALIFIER_A TTRIBUTE4	--	OR	AM
TERMS	Freight Terms	Freight terms for the shipment line	QUALIFIER_A TTRIBUTE10	QP_FREIGHT_TERMS	OR	AM



---

## Seeded Formulas

This appendix covers the following topics:

- Overview of Seeded Formulas
- Seeded Cost-to-Charge Conversion Formulas
- Seeded Markup Formulas

### Overview of Seeded Formulas

Oracle Advanced Pricing provides the following seeded formulas that you can use when setting up freight charges:

- Cost-to-charge conversion formulas (simple pass-through formulas)
- Cost-to-charge markup formulas (simple markup formulas)

Each seeded formula consists of a formula expression, which contains the steps that define how to calculate the freight charges. Rather than create a new formula and expression, you can select an existing seeded formula when setting up freight charges. For example, you could select the QP: Cost to charge conversion of Administration Cost formula to convert the Administration Cost pricing attribute to a charge.

Alternatively, you can update the formula header or formula lines for an existing seeded formula.

You can review the available seeded and non-seeded formulas in the Advanced Pricing - Pricing Formulas window. The Seeded check box indicates if the formula is seeded or not.

**Note:** If the name of a seeded formula is updated then the formula will no longer be identified as seeded.

## Seeded Cost-to-Charge Conversion Formulas

The following tables list details about the seeded cost to charge conversion (pass-through) formulas:

### 1) QP: Cost to charge conversion of Administration Cost

Description: Formula to convert Administration Cost to charge.

Field Name	Value	Field Level
Formula	1	Header
Formula Type	Pricing Attribute	Line
Pricing Attribute Context	Pricing Attribute	Line
Pricing Attribute	Administration Cost	Line
Step	1	Line

### 2) QP: Cost to charge conversion of Duty Cost

Description: Formula to convert Duty Cost to charge.

Field Name	Value	Field Level
Formula	1	Header
Formula Type	Pricing Attribute	Line
Pricing Attribute Context	Pricing Attribute	Line
Pricing Attribute	Duty Cost	Line
Step	1	Line

### 3) QP: Cost to charge conversion of Export Cost

Description: Formula to convert Export Cost to charge.

Field Name	Value	Field Level
Formula	1	Header
Formula Type	Pricing Attribute	Line
Pricing Attribute Context	Pricing Attribute	Line
Pricing Attribute	Export Cost	Line
Step	1	Line

#### 4) QP: Cost to charge conversion of Freight Cost

Description: Formula to convert Freight Cost to charge.

Field Name	Value	Field Level
Formula	1	Header
Formula Type	Pricing Attribute	Line
Pricing Attribute Context	Pricing Attribute	Line
Pricing Attribute	Freight Cost	Line
Step	1	Line

#### 5) QP: Cost to charge conversion of Handling Cost

Description: Formula to convert Handling Cost to charge.

Field Name	Value	Field Level
Formula	1	Header
Formula Type	Pricing Attribute	Line
Pricing Attribute Context	Pricing Attribute	Line
Pricing Attribute	Handling Cost	Line

Field Name	Value	Field Level
Step	1	Line

#### 6) QP: Cost to charge conversion of Insurance Cost

Description: Formula to convert Insurance Cost to charge.

Field Name	Value	Field Level
Formula	1	Header
Formula Type	Pricing Attribute	Line
Pricing Attribute Context	Pricing Attribute	Line
Pricing Attribute	Insurance Cost	Line
Step	1	Line

#### 7) QP: Cost to charge conversion of Transportation Price

Description: Formula to convert Transportation Price to charge.

Field Name	Value	Field Level
Formula	1	Header
Formula Type	Pricing Attribute	Line
Pricing Attribute Context	Pricing Attribute	Line
Pricing Attribute	Transportation Price	Line
Step	1	Line

#### 8) QP: Cost to charge conversion of Transportation Charge

Description: Formula to convert Transportation Charge to charge.

Field Name	Value	Field Level
Formula	1	Header
Formula Type	Pricing Attribute	Line
Pricing Attribute Context	Pricing Attribute	Line
Pricing Attribute	Transportation Charge	Line
Step	1	Line

## Seeded Markup Formulas

The following tables describe the names and setup details for the seeded cost-to-charge with markup formulas:

### 1) QP: Cost to charge markup of Administration Cost

Description: Formula to convert Administration Cost to charge.

Field Name	Value	Field Level
Formula	1*2	Header
Formula Type	Pricing Attribute	Line 1
Pricing Attribute Context	Pricing Attribute	Line 1
Pricing Attribute	Administration Cost	Line 1
Step	1	Line 1
Formula Type	Numeric Constant	Line 2
Component	1	Line 2
Step	2	Line 2

### 2) QP: Cost to charge markup of Duty Cost

Description: Formula to convert Duty Cost to charge.

Field Name	Value	Field Level
Formula	1*2	Header
Formula Type	Pricing Attribute	Line 1
Pricing Attribute Context	Pricing Attribute	Line 1
Pricing Attribute	Duty Cost	Line 1
Step	1	Line 1
Formula Type	Numeric Constant	Line 2
Component	1	Line 2
Step	2	Line 2

### 3) QP: Cost to charge markup of Export Cost

Description: Formula to convert Export Cost to charge.

Field Name	Value	Field Level
Formula	1*2	Header
Formula Type	Pricing Attribute	Line 1
Pricing Attribute Context	Pricing Attribute	Line 1
Pricing Attribute	Export Cost	Line 1
Step	1	Line 1
Formula Type	Numeric Constant	Line 2
Component	1	Line 2
Step	2	Line 2

### 4) QP: Cost to charge markup of Freight Cost

Description: Formula to convert Freight Cost to charge.

Field Name	Value	Field Level
Formula	1*2	Header
Formula Type	Pricing Attribute	Line 1
Pricing Attribute Context	Pricing Attribute	Line 1
Pricing Attribute	Freight Cost	Line 1
Step	1	Line 1
Formula Type	Numeric Constant	Line 2
Component	1	Line 2
Step	2	Line 2

#### 5) QP: Cost to charge markup of Handling Cost

Description: Formula to convert Handling Cost to charge.

Field Name	Value	Field Level
Formula	1*2	Header
Formula Type	Pricing Attribute	Line 1
Pricing Attribute Context	Pricing Attribute	Line 1
Pricing Attribute	Handling Cost	Line 1
Step	1	Line 1
Formula Type	Numeric Constant	Line 2
Component	1	Line 2
Step	2	Line 2

#### 6) QP: Cost to charge markup of Insurance Cost

Description: Formula to convert Insurance Cost to charge.

Field Name	Value	Field Level
Formula	1*2	Header
Formula Type	Pricing Attribute	Line 1
Pricing Attribute Context	Pricing Attribute	Line 1
Pricing Attribute	Insurance Cost	Line 1
Step	1	Line 1
Formula Type	Numeric Constant	Line 2
Component	1	Line 2
Step	2	Line 2

#### 7) QP: Cost to charge markup of Transportation Price

Description: Formula to convert Transportation Price to charge.

Field Name	Value	Field Level
Formula	1*2	Header
Formula Type	Pricing Attribute	Line 1
Pricing Attribute Context	Pricing Attribute	Line 1
Pricing Attribute	Transportation Price	Line 1
Step	1	Line 1
Formula Type	Numeric Constant	Line 2
Component	1	Line 2
Step	2	Line 2

#### 8) QP: Cost to charge markup of Transportation Charge

Description: Formula to convert Transportation Charge to charge.

<b>Field Name</b>	<b>Value</b>	<b>Field Level</b>
Formula	1*2	Header
Formula Type	Pricing Attribute	Line 1
Pricing Attribute Context	Pricing Attribute	Line 1
Pricing Attribute	Transportation Charge	Line 1
Step	1	Line 1
Formula Type	Numeric Constant	Line 2
Component	1	Line 2
Step	2	Line 2



---

## Optimal Performance

This appendix covers the following topics:

- Overview of Optimal Performance
- Oracle Advanced Pricing Setup Considerations
- Qualifier Selectivity
- Qualifier Selectivity Examples
- Pattern Search
- Ignore Pricing
- Additional Tips for Better Performance
- Analyzing your Data Distributions Using a Script
- Technical Improvements

### Overview of Optimal Performance

This appendix addresses implementation and setup considerations, and provides specific recommendations that improve performance from Oracle Advanced Pricing.

Oracle Advanced Pricing is designed to deliver maximum flexibility when pricing customer transactions. In e-business, transactions may be entered:

- By consumers using Oracle iStore.
- By telephone sales personnel using Oracle Telesales.
- By EDI orders received electronically into Oracle Order Management
- From other sources.

These Oracle applications integrate with Advanced Pricing in Oracle Release 11i, and use the pricing engine to price these transactions.

Each time a customer transaction is entered, the pricing engine is called to search through the applicable pricing rules, called qualifiers, to apply to this transaction. Then these rules are used to select the correct set of pricing actions including price lists, formulas, discounts, and promotions required to correctly price the transaction.

Potentially, thousands of available actions may have been set up in the internal tables of Advanced Pricing, so the task of the pricing engine to provide maximum flexibility while delivering fast performance for e-businesses is very challenging.

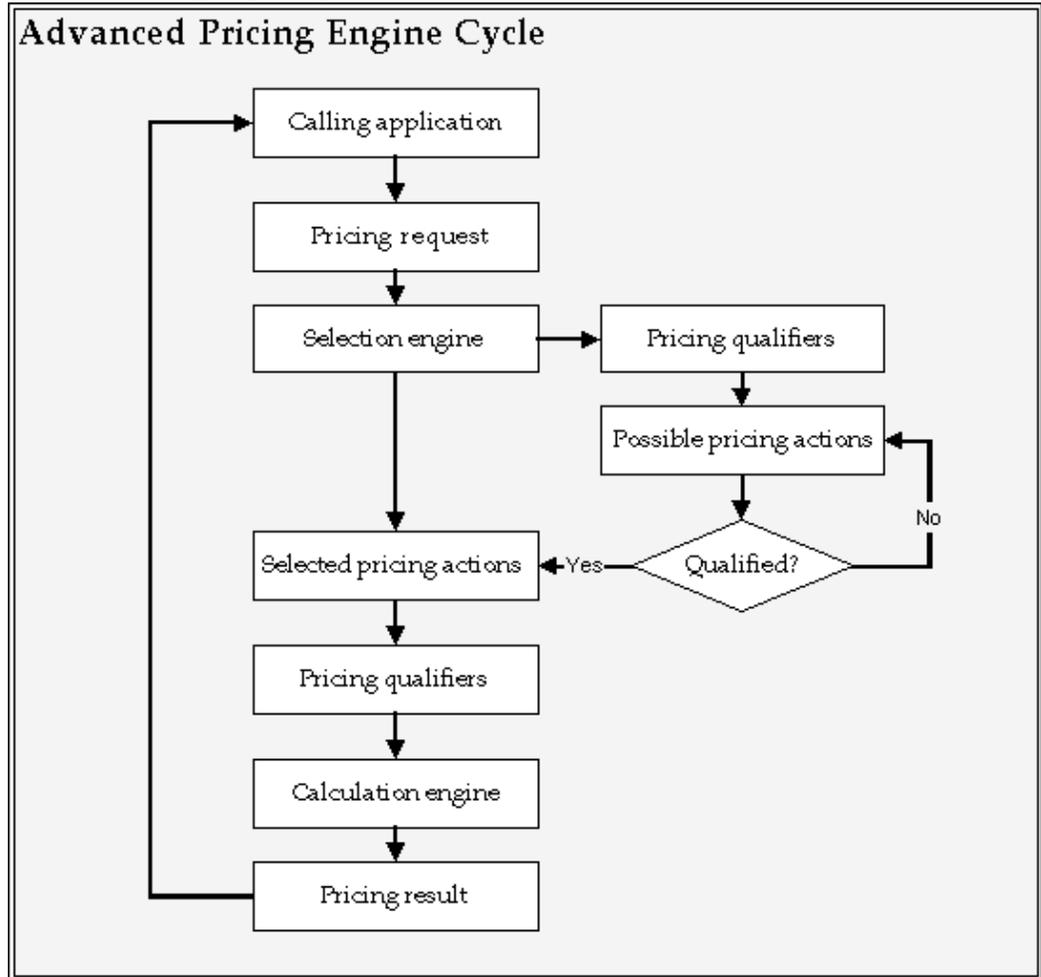
The pricing engine cycles each time a calling application makes a pricing request to the engine. Calling applications in Release 11*i* include Oracle iStore, Oracle Order Management, Oracle Contracts, Oracle TeleSales, Oracle Inventory, and Oracle Quoting.

Pricing engine performance is the amount of cycle time that elapses between when the pricing request is submitted to the engine and when the pricing result is returned.

The pricing engine runs through two types of processing activities each time it executes. First, the engine evaluates user-established qualifier rules. Based on these rules, the engine selects qualified pricing actions the calling application may need to apply to the transaction being processed. Second, the calculation engine performs the calculations necessary to compute selling price.

The following image depicts the Oracle Advanced Pricing engine cycle:

## Advanced Pricing Engine Cycle



Analysis of pricing engine performance characteristics reveals that the selection process is subject to more execution time variability, due to the number of records that may be selected. This chapter addresses the issue of optimizing selection engine performance.

Meeting the demanding performance requirements of e-businesses has been a major design goal. To achieve this goal, the Pricing Development team strives to optimize product performance particularly of the pricing engine.

Choices you make when selecting settings or setting up your pricing data can substantially improve the performance of the Advanced Pricing Engine. For this purpose, the implementation recommendations and considerations in the *Oracle Advanced Pricing Implementation Guide* are designed to assist you in optimizing the performance you receive from Oracle Advanced Pricing.

## Oracle Advanced Pricing Setup Considerations

Analyzing your pricing data setups is the best way to improve Oracle Advanced Pricing engine performance. Removing any unnecessary qualifiers and inactivating any unnecessary, price lists, and modifiers can substantially improve pricing engine performance.

There are distributions of data in the pricing data setup, however, that can slow the pricing engine execution. The first of these is qualifier selectivity.

### Qualifier Selectivity

As the pricing engine finds qualifiers that apply to a transaction, it selects all active price lists or modifier actions that the qualifier pertains to. When qualifiers identify a narrow range of pricing actions, the majority of which should be applied to the transaction, then that qualifier has high selectivity. When a single qualifier is linked to many pricing actions, the majority of which cannot be simultaneously applied to a transaction, then that qualifier is said to have low selectivity.

The new search optimizer contains the latest performance capabilities. Search optimizer introduces the ability of the pricing engine to tag the most selective qualifier within a group of qualifiers attached to the same modifier. For example, a 2% discount modifier has two qualifiers: Price list = Corporate and Customer = XYZ. Price List = Corporate is a non-selective qualifier (it is attached to many modifiers). Customer = XYZ is a selective qualifier (occurrence is low). The pricing engine first matches the most selective qualifier within the qualifier group and then matches the less selective qualifiers for the selected modifier. Search Optimizer uses the qualifier number of occurrences as criteria to tag selectivity. The pricing engine distinguishes selectivity of the qualifiers Price List = Common Price List and Price List = Customer Specific Price List.

If a modifier has qualifier as well as the product attached to it, then the pricing engine is qualifier driven rather than product driven. For example, 2% discount modifier has a qualifier attached: Price list = Corporate. It also has a product attached: Item=ABC. The pricing engine matches the qualifier first and then matches the product. At least one qualifier should be selective within the qualifier group.

#### Low Qualifier Selectivity Impact

Low selectivity has an adverse impact on pricing engine performance. The effect of low selectivity depends on the distribution of pricing data. The larger the data volume when qualifier selectivity is low, the greater the negative impact on performance.

## Qualifier Selectivity Examples

### High Qualifier Selectivity

To illustrate the effects of selectivity, use the following business example consider how different pricing setups with high and low qualifier selectivity can be structured. Pricing engine performance implications are examined.

Qualifier Group has four qualifiers namely customer name, order type, price list, region. Customer Name is occurring five times. Therefore, it is tagged as most selective qualifier in the group. Although the other qualifiers are not highly selective, pricing engine will perform better because most selective qualifier is searched first. The following illustrates the setup with high qualifier selectivity:

Qualifier Group	Qualifier	Qualifier Count	Selectivity (Search Indicator)	Exclusivity Group	Precedence
1	Customer Name = 'ABC'	5	1	1	100
1	Order Type= Standard	300	2	1	100
1	Price List = Corporate	200	2	1	100
1	Region=Western	50	2	1	200

### Low Selectivity Due to Historical Records

A company has low setup selectivity because of a large number of modifier and price list records with effectivity dates in the past. This company has been in business for several years. It has several price lists and discount records in its system, many of which are outside their effectivity dates but still in the system for historical purposes. Using the same base as in the previous example, the setup data for this example could be as follows:

<b>Qualifier (Customer Class)</b>	<b>Qualifier Effective Dates</b>	<b>Price List</b>	<b>Effective Dates</b>	<b>Modifier</b>	<b>Exclusivity Group</b>	<b>Precedence</b>
Wholesale	2/15/1991 - present	First Quarter Wholesale	2/15/2001- 3/31/2001	5% discount	1	100
Wholesale	2/15/1991 - present	First Quarter Wholesale	2/15/2000 - 3/31/2000	4% discount	1	100
Wholesale	2/15/1991 - present	First Quarter Wholesale	2/15/1999 - 3/31/1999	6% discount	1	100
Wholesale	2/15/1991 - present	First Quarter Wholesale	2/15/1998- 3/31/1998	5% discount	1	100
Wholesale	2/15/1991 - present	First Quarter Wholesale	2/15/1997 - 3/31/1997	4% discount	1	100
Wholesale	2/15/1991 - present	First Quarter Wholesale	2/15/1996 - 3/31/1996	6% discount	1	100
Wholesale	2/15/1991 - present	First Quarter Wholesale	2/15/1995- 3/31/1995	5% discount	1	100
Wholesale	2/15/1991 - present	First Quarter Wholesale	2/15/1994 - 3/31/1994	4% discount	1	100
Wholesale	2/15/1991 - present	First Quarter Wholesale	2/15/1993 - 3/31/1993	6% discount	1	100
Wholesale	2/15/1991 - present	First Quarter Wholesale	2/15/1992 - 3/31/1992	4% discount	1	100

Qualifier (Customer Class)	Qualifier Effective Dates	Price List	Effective Dates	Modifier	Exclusivity Group	Precedence
Wholesale	2/15/1991 - present	First Quarter Wholesale	2/15/1991 - 3/31/1991	5% discount	1	100

Assume that the other classes, Retail and Other, have similar data. Now examine customer All, where management has experimented with many different discount structures over time.

Qualifier (Customer Class)	Qualifier Effective Dates	Price List	Effective Dates	Modifier	Exclusivity Group	Precedence
All	1/01/1991 - present	Corporate	1/01/2001 - 12/31/2001	2% discount	1	200
All	1/01/1991 - present	Corporate	1/01/2000 - 12/31/2000	1.9% discount	1	200
All	1/01/1991 - present	Corporate	1/01/1999 - 12/31/1999	1.8% discount	1	200
All	1/01/1991 - present	Corporate	1/01/1998 - 12/31/1998	1.7% discount	1	200
All	1/01/1991 - present	Corporate	1/01/1997 - 12/31/1997	1.6% discount	1	200
All	1/01/1991 - present	Corporate	1/01/1996 - 12/31/1996	1.5% discount	1	200
All	1/01/1991 - present	Corporate	1/01/1995 - 12/31/1995	1.4% discount	1	200
All	1/01/1991 - present	Corporate	1/01/1994 - 12/31/1994	1.3% discount	1	200
All	1/01/1991 - present	Corporate	1/01/1993 - 12/31/1993	1.2% discount	1	200

Qualifier (Customer Class)	Qualifier Effective Dates	Price List	Effective Dates	Modifier	Exclusivity Group	Precedence
All	1/01/1991 - present	Corporate	1/01/1992 - 12/31/1992	1.1% discount	1	200
All	1/01/1991 - present	Corporate	1/01/1991 - 12/31/1991	1% discount	1	200

In the previous example, the qualifier selectivity is very low, negatively impacting pricing engine performance. When the pricing engine executes against this data, it will find that all the historical records (those with effectivity dates that are already past) will be preliminarily qualified and selected by the Pricing Engine for further processing, even though only one price list and modifier will be finally selected to apply to the transaction. Specifically, for the historical records, the pricing engine will be forced to compare the exterior pricing date passed to it from the calling application to the effectivity date range of each record to determine if whether the selection process should proceed to the next step of considering the precedence. Since the effectivity date evaluation is a record-by-record process, large number of historical records will have an adverse impact on Pricing engine performance.

## Correcting Low Qualifier Selectivity Due to Historical Records

Pricing engine performance can be improved by properly handling historical records. While the most direct route to improving engine performance is to eliminate historical records from the system, many companies must retain historical records.

Oracle Advanced Pricing provides a flag on both price list and modifier records that informs the engine whether the record is active. Because the pricing engine determines record status during the qualifier scan process, records set to inactive are automatically excluded from further consideration.

## Low Qualifier Selectivity Due to Release 10.7/11 Pricing

Oracle Advanced Pricing provides more flexibility in setting up your pricing data than Release 10.7 or Release 11.

In Release 10.7 and in Release 11.0, discounts had to be associated to a price list because a price list was a mandatory qualifier for a discount. Users had to create different discounts because only one discount could be linked to one price list. In turn, price lists could be linked to either the customer, the Order Entry order type, or manually overridden on the order.

Therefore, in releases 10.7 and 11.0, many customers could potentially have large numbers of discount records, even when the number of different discrete discounts

used was low.

## **Correcting Low Qualifier Selectivity due to Release 10.7/11 Upgrade**

Reducing the number of discount records and making the qualifiers specific will help you optimize 11i pricing engine performance. Consider merging these discount records into as few 11i modifier records as possible, and then use 11i Pricing to tie them to customer groups.

If you examine your qualifiers and your business pricing requirement, you may find that you can qualify pricing by using either price lists or modifiers at the higher level of the product hierarchy. For example, suppose you are selling greeting cards and have only 15 distinct prices but 100,000 items. By grouping the items into item categories and using item category as the qualifier, you only have to create 15 price list lines rather than 100,000 lines. This grouping results in the pricing engine searching through price list lines, which substantially improves pricing engine performance. If your business needs do not require a price list to act as a qualifier to a discount, you should delete any records that have been created as part of your upgrade process.

## **Low Selectivity When Qualifiers are Used as Constraints**

If, for example, you have modifier lists, out of which 200 lists have Price List = Corporate as the only qualifier, this results in low selectivity of the qualifier because the pricing engine must process every list that satisfies this qualifier.

## **Correcting Low Selectivity When Qualifiers are Used as Constraints**

If your business requires non-selective qualifiers as the only qualifiers, consider combining lists so that there are fewer lists for the engine to scan.

## **Redundant Qualifiers**

Pricing engine performance can be boosted by avoiding redundant qualifiers. Here is an example of a redundant qualifier:

Customer = XYZ AND Customer Site = ABC

Because it is more specific than Customer, Customer Site is sufficient for selection of the appropriate price list or modifiers. Adding the customer qualifier causes the unnecessary evaluation of the customer condition. By eliminating such redundant qualifiers, pricing engine performance is optimized.

## **Blind Modifiers**

Modifiers without any qualifier or any product attached are blind modifiers. These modifiers are processed by the engine for every request line. Engine performance is negatively affected as more blind modifiers are defined in the system.

## Use of ALL\_ITEMS as a Product

Modifiers defined for ALL\_ITEMS are processed by the engine for every request line. Engine performance is negatively affected as more ALL\_ITEMS are defined in the system.

## Use of Exclusions and the NOT= Operator

The pricing engine needs additional processing time to evaluate NOT = Operator in qualifiers, as well as EXCLUDE in the product hierarchy. If you have a high volume of setup data, use caution when implementing these operators.

## Pattern Search

Flexibility of pricing setup with attribute management provides you with ample opportunity to fit your business needs. As a result of this flexibility, the pricing engine needs to look for all combinations and permutations to get the appropriate price lists and modifier lines eligible for the incoming line. Since all the attributes (pricing, product and qualifier) for price lists and modifier lines are optional, it is challenging to decide the best search selectivity path to suit customer setup.

The Pattern Search feature solves the indefinite search path to a fixed best possible search path. You can enable the search engine part of the pricing engine to use pattern search to see improved performance. Pattern search ensures better pricing performance in all pricing related flows such as, order entry, save, quote entry, save, reprice, opening configurator, and adding top model item.

To make all pricing calls to use the pattern search engine, you must set up the following:

1. Set an applicable value for the QP: Pattern Search and QP: Pattern Search Path profile options. See Profile Options Setup Summary in *Oracle Advanced Pricing Implementation Guide*.
2. As a one-time action, run the QP: Pattern Upgrade concurrent program with empty criteria to upgrade the existing pricing setup to patterns. See Reports and Concurrent Programs in the *Oracle Advanced Pricing User Guide*.

For further information, see 'Pattern Based Searching Logic in Pricing Engine For Performance Improvement', My Oracle Support Note 948900.1.

## Ignore Pricing

You can inform the pricing engine to ignore lines that have no significance from pricing perspective using the QP: Custom Ignore Pricing profile option and implementing the QP\_CUSTOM\_IGNORE.IGNORE\_ITEMLINE\_FOR\_PRICING custom hook API. See

Profile Options Setup Summary and 'Ignore Pricing for Order Lines – Custom Hook' white paper available in My Oracle Support Note 113492.1.

## Additional Tips for Better Performance

Here are additional tips to improve processing:

1. Always pass the price list that needs to be used for pricing, unless your business demands otherwise.
2. Try to avoid multiple price list lines from triggering incompatibility processing which eliminates ineligible list lines - this prevents unnecessary post selection processing.
3. Avoid using the same attribute for both the header and line level qualifiers. For example, if Customer = Joe's Diner is a Header level qualifier, it is not necessary to define it again as a line level qualifier.

## Analyzing your Data Distributions Using a Script

Oracle Advanced Pricing provides a script to analyze data distribution. The script is delivered as part of the product at `$QP_TOP/patch/115/sql/qpperf.sql`. If you encounter performance problems, then run this script at the SQL\*Plus prompt using APPS login. If a customer performance bug has been logged, then the script results should be provided to the pricing team for their review. The script also provides hints that may assist you identify performance issues.

Alternately, you can run this script as a concurrent program and view the output in the concurrent request output file. Using the Oracle Pricing Manager responsibility, navigate to Reports, and submit a request for the concurrent program Diagnostics: Performance Analysis.

**Note:** Scripts may change, so it is important to verify with Oracle Support that you have latest version of the script.

## Technical Improvements

When you implement Oracle Advanced Pricing, several technical measures can be taken to ensure the best response time from the pricing engine. These measures are related to implementation-time activities.

### Attribute Mapping

Oracle Advanced Pricing enables you to perform attribute mapping. Use caution when writing your own attribute sourcing. The code you write is executed for every pricing

engine call. If the code is not tightly written, this can negatively impact pricing engine performance.

Pricing provides you with an option to source all the setup attributes or source attributes of only active price list, modifiers. Performance will be improved if the profile QP: Build Attributes Mapping Options is set to active only.

## Phases and Events

Oracle Advanced Pricing enables pricing engine execution to be divided into phases with each phase associated with an event in the calling application.

You do not need to fetch or view the discounts when you enter the order lines - you can modify the event phases records to run the discounting phases when you save the order lines. This approach causes the engine to perform the price list line selections as each line is entered, while preventing the engine from doing the selection or calculation of modifiers until the save event causes the modifier selection to cycle. For users who must not view the discounts line by line as the order is entered, this technique can enhance pricing engine performance.

If you have unused pricing event phases, set the end date to a past date. This prevents pricing engine from attempting to select when the event that triggers the pricing phase occurs.

## Temporary Tablespace

Oracle Advanced Pricing uses the temporary tablespace feature of the Oracle database extensively. Proper sizing of temporary tables is important to obtaining optimal performance. Depending on the size of the transaction to be priced, the initial extent for temporary tables should be sized at between 64K and 256K. The temporary tablespace should be defined as locally managed.

## Memory Considerations

Because the pricing engine is frequently called during the order entry process, pricing packages must always be loaded into memory. Keep the following Pricing packages in memory:

- QP\_BUILD\_SOURCING\_PVT
- QP\_CALCULATE\_PRICE\_PUB
- QP\_CLEANUP\_ADJUSTMENTS\_PVT
- QP\_CUSTOM
- QP\_FORMULA\_PRICE\_CALC\_PVT

- QP\_PREQ\_GRP
- QP\_PREQ\_PUB
- QP\_RESOLVE\_INCOMPATIBILITY\_PVT

**Warning:** Ensure that your shared pool size is calculated based on system use!

## Performance in the Pricing Setup window

The following performance-related improvements in the setup window have been made:

- Price List window: To query a list price by product, a new find window is provided.
- Agreements window: A new find window is provided.
- Modifiers window: A list of values for Customer Name, site\_use, and ship\_to is provided.

**Note:** The user is responsible for the performance of the list of values, validation of the user-extended qualifiers, and pricing attributes. Therefore, ensure that the where clause in the user-defined value set is properly tuned.

## Performance in applying Pricing Denormalization Patches

If you have a high volume of price list and modifier data, it may take from 5 minutes to an hour to apply pricing patches. A denormalization script is provided for the existing data. The denormalized columns are important for the pricing engine to select the list price or modifiers, and large denormalization patches have been provided with parallel threading to improve the performance.

## Pricing Engine Redo Log

Users have experienced significant increase in the redo log while pricing the transactions. In the latest release, this redo is minimized by avoiding the delete operation on the temporary tables. Please check with support to ensure that you have the redo reduction patch applied.

## Performance of Asked for Promotions

Users have experienced significant performance whenever asked for promotions were defined in the system. In the latest release, these issues are resolved by pre-matching the asked for Promotions. Please check with support to ensure that you have applied the latest performance patches.

## Summary of Recommendations

The pricing engine is a resource intensive process that uses Oracle 8i temporary tables. The performance of pricing engine statements is significantly affected by inaccurate CBO settings, memory settings, or any other significant high-load process. Many of the pricing performance bugs were resolved by tuning the database parameters or correcting other high-load statements. Following table lists common causes of performance issue:

Symptom	Probable Cause	How to fix it
1) Performance is slow and OM debug file is getting created in the debug directory	OM debug or QP debug is causing very high pl/sql processing	Make sure that Profile ONT_DEBUG_LEVEL is set to 0 and profile QP: QP_DEBUG is set to No.
2) Trace File is showing a big difference between CPU and Elapsed time	Pricing Engine Temporary tables are being thrown out of memory.	1) Some other non-pricing statement may be taking a huge amount of Logical Reads (These high-load statements would deteriorate Pricing engine performance). Execute the following statement to find high-load sql. Select sql_text, buffer_gets from v\$sql where buffer_gets in (select max (buffer_gets) from v\$sql);  2) Change db_block_buffers to a higher number.3--There may be CPU contention. Make sure that there are no runaway processes. Check your hardware resources.

Symptom	Probable Cause	How to fix it
3) Significant amount of time is spent during Pricing	Pricing Performance patches may not have been applied	Contact your support analyst to obtain any known performance fixes applicable to the specific release.
3a) Huge amount of redo is seen while pricing the transaction	Delete operations on Pricing temporary tables are generating redo	Redo is reduced by eliminating deletes. Please contact support to get the appropriate patch.
4) Trace File is showing Misses in library cache during parse	Not enough Shared Pool	Increase the value of Shared Pool parameter in the frequently used Pricing Engine packages.
5) Trace file shows huge volume in the pricing temporary table qp_preq_qual_tmp	Qualifiers may be unselected which causes large number of records being selected by the engine	Run qpperf.sql to see if any unselected qualifiers exist. Restructure the setup either by moving these qualifiers to Pricing attribute or by moving the qualifier from the line to the header. Use qpperf.sql Stmt # 10,16.
6) Trace File is showing obj\$ and other sys statements	SYS/SYSTEM schema is analyzed.	Do not analyze SYS/SYSTEM schema.
7) CBO is causing full table scans	CBO parameters in Init.ora may not be set properly.	Use/fnddev/fnd/11.5/admin/sql/AFCHKCBO.sql to check this).
8) Excessive calls made to the pricing engine at booking and scheduling	Pricing may be turned on at scheduling.	OM profile "OM: Deactivate Pricing at Scheduling" should be set to yes.
9) Form stops responding	ST patches for temporary table may not have been applied.	Check My Oracle Support Note 119542.1.
10) Unnecessary Pricing call is made at Booking	Pricing Event Phase is active for Book Event	Use Pricing Manager responsibility, invoke Event-Phase screen, Query Book event, set the end date to some prior date.

Symptom	Probable Cause	How to fix it
11) Unnecessary Modifiers are being fetched with every line	Automatic or Manual Modifiers are created without any qualification.	Analyze the price adjustments data to check if certain modifiers are selected for every line and are unnecessary. Inactivate these modifiers. Use qpperf.sql Stmt # 17, 27
12) Sales Order processing slow at Booking	OM performance patches may not have been applied	Check My Oracle Support for recommended performance patches.
13) Many obsolete modifier headers are being queried by Pricing Engine	Upgraded obsolete Pricing data is still active.	Inactivate the obsolete List headers by clearing the active flag. Use qpperf.sql Stmt# 5.
14) Trace file is showing custom code being executed many times as well as taking a huge amount of time.	Inefficient Extension /custom code is being called by Attributes mapping or by get custom price. Caching is not implemented in the custom code	Implement caching and tune the custom code. If you are not able to implement caching for order attributes at the line level, which changes for every call, then please look at the Order Amount sourcing as an example of caching such attributes.
15) Pricing debug screen is showing unnecessary qualifiers/ attributes being passed to the pricing engine.	Somebody has setup prototype modifies using the qualifiers and attributes which are not used by the production application	Pricing engine will get all the qualifiers, pricing attributes even if those are setup for prototype purpose. If you have setup such modifiers then inactivate these modifiers in the production instance. Then change the value of the profile QP_BUILD_ATTRIBUTES_MAPPING_OPTIONS to Y and run the Build Context concurrent program.

Symptom	Probable Cause	How to fix it
16) Many modifiers and price list lines coming into the pricing engine.	<p>1) Multiple "Not=" qualifiers without any selective qualifier in the qualifier group.</p> <p>2) Excessive use of "ALL_ITEMS" product element.</p>	<p>Too many lines of the same product element or same qualifier will reduce the selectivity of the engine. Evaluate alternate setup.</p> <p>If you have many "Not = " qualifiers without having any other selective qualifier in the group then performance will be affected. You can change your sourcing rule to convert the "Not = " qualifier to Equal to.</p> <p>For example instead of saying Customer Not= cust1 or Customer = cust2, you can evaluate the condition in Attributes mapping if customer not in (cust1, cust2) then cust_code = 1 by using cust_code as a qualifier.</p>
17) Temporary Table space growing significantly. Causing huge overheads.	Initial extent of the Temp table space may be high. Pricing engine temp tables are created with big initial extents.	Change the storage clause of Temporary table space with a minimal setting of initial extent. Change the temporary table space to be locally managed.
18) Performance is slow when profile is enabled to save requests in Pricing Engine Request Viewer.	Huge quantities of data are being written into the request viewer tables.	Change the profile QP_Debug to Not store debug Log into request viewer tables. Also, purge data from Request viewer tables by using the Purge Concurrent program.

## Pricing Setup Window

The following performance-related improvements were made:

- Price List Setup window, query list price by product: A new find window is provided.

- Agreements Setup window: A new find window is provided.
- Modifiers window: A list of values customer name, site\_use, ship\_to is provided.
- Calling Modifier Organizer from Modifier windows is provided for faster search.

**Note:** The user is responsible for the performance of the list of values/validation of user-extended qualifiers/pricing attributes. Verify that the where clause in the user-defined value set is properly tuned.

## Global Engine Flag for Caching Attribute Mapping

On repricing, pricing engine is called once for each order. For multiple lines order, the build\_sourcing is executed for every line to source all the attributes such as order amount and customer information. These attributes are therefore repeatedly sourced multiple times.

A pricing engine global flag, G\_NEW\_PRICING\_FLAG is introduced to indicate that the attributes has been sourced for the first line so that sourcing is not required for all the following line(s). The flag has an initial value of 'Y' and build sourcing is only executed when this value is 'Y'. At the end of initial build sourcing call, this value is set to 'N' so that subsequent build sourcing is not necessary. The flag is reset back to 'N' at the end of the pricing engine call. This implementation helps to avoid unnecessary processing and hence improve the performance.

Customers who have any customized attributes can use the flag in a similar way to improve the pricing engine response time. This flag is set internally as described above. Developer can use the value of this flag to decide if re-sourcing is necessary.

Flag: QP\_PREQ\_GRP.G\_NEW\_PRICNG\_CALL

Value: 'Y' or 'N'

### Example of caching the order amount

```
PROCEDURE Get_Order_AMT_and_QTY (p_header_id IN NUMBER) IS
    orders_total_amt      NUMBER;
    orders_total_qty      NUMBER;
    returns_total_amt     NUMBER;
    returns_total_qty     NUMBER;
BEGIN

    SELECT SUM(nvl(pricing_quantity,0)*(unit_list_price)),
           SUM(nvl(pricing_quantity,0))
           INTO orders_total_amt, orders_total_qty
    FROM oe_order_lines
    WHERE header_id = p_header_id
           AND (cancelled_flag = 'N' OR cancelled_flag IS NULL)
           AND (line_category_code <> 'RETURN' OR line_category_code IS
    NULL)
           GROUP BY header_id;
    G_Order_Info.header_id := p_header_id;
    G_Order_Info.order_amount := FND_NUMBER.NUMBER_TO_CANONICAL(NVL
    (orders_total_amt,0)-NVL(returns_total_amt,0));
    G_Order_Info.order_quantity := FND_NUMBER.NUMBER_TO_CANONICAL(NVL
    (orders_total_qty,0)-NVL(returns_total_qty,0));
EXCEPTION
    WHEN no_data_found THEN
        OE_DEBUG_PUB.ADD('NO Data Found');
END;

FUNCTION Get_Order_Amount(p_header_id IN NUMBER) RETURN VARCHAR2 IS
BEGIN
    IF qp_preq_grp.g_new_pricing_call = qp_preq_grp.g_no THEN
        RETURN G_Order_Info.order_amount;
    ELSE
        Get_Order_AMT_and_QTY(p_header_id);
        RETURN G_Order_Info.order_amount;
    END IF;
END Get_Order_Amount;
```

## Custom Applications Integrating with Oracle Applications

All the custom applications integrating with Oracle Applications need to integrate with the QP\_PREQ\_PUB.PRICE\_REQUEST API instead of QP\_PREQ\_GRP.PRICE\_REQUEST for the latest features and improved performance. For more information, see: Integrating with Oracle Advanced Pricing, page 21-1.

## Performance Patches

Patches help improve the performance of the pricing engine. You should check with support for information on new and updated patches.



---

## Case Study: Pricing Scenarios in the High-Tech Industry

This appendix covers the following topics:

- Overview of Pricing Scenario

### Overview of Pricing Scenario

Tech Emporium is a fictitious manufacturer of high-tech gadgets for the networking industry. The company's two most popular products are Brainglo and Infratimers. Both Brainglo and Infratimers are sold in a product grouping for tools items. Brainglo also belongs to the infrastructure product grouping.

Tech Emporium sells its products to two classes of customers: OEM companies and emerging growth companies. In this case study, we examine orders placed from National OEM (an OEM company) and an emerging growth company called HTG (Hoping to Grow).

The following tables illustrate Tech Emporium's pricing situation. The first table depicts the price lists, discounts by customer are depicted in the next table, and discounts by products are depicted in the third table. These tables are for reference throughout this case study.

Products	Corporate Price (default)	National OEM Price	HTG Price
Brainglo	\$200	\$175	Not applicable
Infratimers	\$160	Not applicable	\$150

**Note:** The Corporate Price List contains all items. If a product is not on

a price list (marked not applicable), then the price defaults to Corporate.

Customer	Customer Class	Price List	Discount	Discount Name	Exceptions
National OEM	OEM	National OEM	3%	VIP discount	Customer class is OEM
HTG Ltd.	Emerging growth	HTG	Null	Null	Null

Product	Product Grouping	Discount	Exception
Brainglo	Infrastructure	5%	Customer specific price list
Brainglo	Tools	10%	Null
Infratimers	Tools	10%	Null

## Problem Definition

The following section introduces several pricing scenarios.

### Price List Scenario

Tech Emporium wants certain customers to receive special pricing treatment for some products and standard pricing treatment for other products.

National OEM and HTG want to place orders for Brainglo and Infratimers.

### Pricing Actions and Rules

To receive the special and standard prices, pricing actions, and rules are defined. Break down the pricing action into two parts.

<b>Pricing Requirement</b>	<b>Pricing Rules</b>
Receive special price for product base on an applicable price list.	Price list is not specified on the order. Customer is eligible for a specific price list. Ordered item must appear on the specific price list.
Receive standard treatment price for other products.	Ordered item must not appear on the customer's specific price list.

### **Discount Scenario**

Tech Emporium offers discounts based on customers and product groupings.

National OEM is entitled to receive a special VIP discount of 3 percent for all of its orders. HTG belongs is in the emerging growth customer class; it is not eligible for the VIP customer discount.

There is a 5 percent discount on products in the infrastructure product group and a 10 percent discount on the tools product group (Brainglo is in both the Infrastructure and Tools product group and Infratimers is only in the Tools product group). If a customer is eligible for multiple discounts based on product groupings, then that customer will receive the lesser of two discounts. For example, a customer can not receive both a 5 percent and 10 discount for Brainglo.

- **Pricing Requirement 2**

In certain cases, multiple discounts cannot be applied at the same time to the order. If a customer is eligible for both a 5 percent and 10 percent discount, then that customer should get only the 5 percent discount.

- **Pricing Requirement 3**

Applying discounts depends on if an item has received special pricing promotions. Products in the infrastructure category get a 5 percent discount if the price is derived from the customer specific price list. Products in the tools category are entitled to a 10 percent discount.

- **Pricing Requirement 4**

Discounts can be given based on a the structure of the customer hierarchy. Customers in the OEM customer class are entitled to 3 percent discount on all orders, whereas customers in HTG customer class are not entitled to any discounts.

- **Pricing Requirement 5**

There are discounts that look at individual lines on the order. Other discounts may look at the entire order. For example, Customers in the OEM customer class are entitled to an order level VIP discount. Therefore, the 5 percent discount and 10

percent discount can be applied only to the order and not to individual lines.

<b>Pricing Action</b>	<b>Pricing Rules</b>
Give 5% discount	Only for the order line. Applies when user leaves the line. Applies when ordered item is from Infrastructure product group. Only when customer specific price list is selected. If there are conflicting discount, this one is selected.
Give 10% discount	Only for the order line. Applies when user leaves the line. Only when ordered item is from the Tools product group.
Give 3% discount	Only the entire order. Applies when customers are in the OEM customer class (in this case, only National OEM will receive the discount).

## Applying Oracle Advanced Pricing

Now that each requirement has been divided into individual pricing actions and pricing rules, we will show you a method of implementing these rules and actions using Oracle Advanced Pricing.

### Price List Setup

Setup requires two customer specific price lists and one corporate price list. Attaching a qualifier of customer name makes the price list customer specific. Customers do not specify a price list when ordering. This enables the pricing engine to search for a price list. In Oracle Advanced Pricing qualifiers direct the pricing engine towards a specific price list.

A secondary price list must be set up for the products that do not have special pricing. For this example, the Corporate Price List, which includes all of Tech Emporium's products, is the secondary (or default) price list. When the pricing engine does not find an ordered item on the primary price list, the pricing engine searches the secondary price list.

### Discount Setup

- **Modifiers and Qualifiers**

Other pricing actions, such as discounts, are implemented through the use of modifiers and qualifiers. You can map each pricing rule to a section on the modifier form. You can also attach qualifiers to modifiers to specify the eligibility conditions for a discount.

- **Incompatibility Groups**

A customer may be eligible for a 5 percent and a 10 percent discount. A rule is set that the customer cannot receive both discounts; the customer should receive only the 5 percent discount. By grouping discounts into the same incompatibility group, discounts no longer conflict. The pricing engine examines modifiers within the same incompatibility grouping level and selects only one modifier. Implementers select whether precedence or best price is used for incompatibility resolution for each pricing phase. Precedence has been selected for this case study. Both the 5 percent and 10 percent discounts are in the same incompatibility group; the pricing engine will select the modifier line with the lowest precedence value.

- **Pricing Phase**

Incompatibility works only within a pricing phase. Discounts can be applied at different times in the order cycle. Tech Emporium applies three discounts at different times within the order cycle. The first two discounts apply when you leave the order line; this pricing phase is called list line adjustment. The third discount is applied when the order is saved: this pricing phase is called all lines adjustment. For all lines adjustment, the pricing engine must evaluate all the lines on the order.

### **Discount Example**

One of Tech Emporium's pricing rules is that customers associated with the OEM customer class are entitled to a 3 percent order level discount. National OEM is eligible for this discount. There is no need to define an incompatibility group for this during setup because it applies in addition to other discounts. Some discounts are at the line or group of lines level, while others are order level discounts. The 3 percent VIP discount is an order level discount, while the 5 percent and 10 percent product discounts are line level discounts.

### **5 Percent Discount**

To ensure that the discount applies when there are conflicting discounts, the precedence value must be smaller than that of the other discount. Incompatibility is set to Level 1 Incompatibility Group.

### **10 Percent Discount**

This discount is applied after the user leaves the order line. The incompatibility is Level 1 Incompatibility Group. The precedence value is a higher number (lower precedence).

### **3 Percent Discount**

You must define the order level discount and attach a customer class qualifier. The discount always applies when the customer meets the requirement. Do not define an incompatibility group. Customer class is the qualifier.

## **Results of Pricing Scenario**

Based on the price list and modifier setups discussed in this case study, the following tables depict how orders from National OEM and HTG appear.

The following table depicts a National OEM order:

<b>Item</b>	<b>Quantity</b>	<b>Price List</b>	<b>List Price</b>	<b>Selling Price</b>	<b>Price Adjustment</b>
Brainglo	1	National OEM	175.00	161.00	3% OEM discount  5% infrastructure products discount
Infratimers	1	Corporate	160.00	139.20	3% OEM discount  10% tools discount

The following table depicts an HTG order for the same items:

<b>Item</b>	<b>Quantity</b>	<b>Price List</b>	<b>List Price</b>	<b>Selling Price</b>	<b>Price Adjustment</b>
Brainglo	1	Corporate	200.00	180.00	10% tools discount
Infratimers	1	HTG	150.00	135.00	10% tools discount

---

# Case Study: Using Oracle Advanced Pricing Formulas for Healthy Fast Food

This appendix covers the following topics:

- Introduction
- Problem Definition
- Pricing the Burger

## Introduction

Healthy Fast Food (HFF) is a chain of fast food restaurants that offers healthful fast food alternatives to traditional fast food. More than 2,000 HFF restaurants operate nationally. HFF provides eat in, take out, or home delivery of fresh vegetarian burgers and side dishes. HFF is well known for its Fresh-from-the-Garden Burger: a vegetarian burger with an unlimited number of toppings. HFF is dedicated to freshness and quality. Every day, fresh ingredients are shipped across the country to six distribution centers and then rushed to local stores.

## Problem Definition

HFF sells primarily one item, the Fresh-from-the-garden Burger (the Burger). HFF wants to maintain one price list and one item on their price list for the Burger, which has no fixed price. The selling price is based upon several factors:

- Size of the burger
- Type of bun
- Type of cheese
- Number of toppings

- Distance of the store from the distribution center

The cost of toppings varies according to season and availability. The price of the Burger must change to reflect new topping prices. To minimize price fluctuations, HFF only adjusts cost components monthly. The final price a customer is charged for the Burger depends on the size of the burger, type of bun, and type of cheese. An extra charge is added for lettuce, tomato, onion, pickles, mustard, or other toppings. Each topping selected will add one cent to the final price.

The Burger comes in three different sizes: single, double, and triple. The final price charged varies depending on the size ordered. A single adds 99 cents, a double adds \$1.49, and a triple adds \$1.99 to the final price. Buns come in several types, including white, wheat, whole grain, honey wheat, and organic grain. Each of these has a different add-on value, ranging from two cents for white to as high as ten cents for organic grain.

Cheese options include cheddar, jack, and Gouda. These add two cents, five cents, and ten cents, respectively, to the final price. The customer is charged an add-on adjustment factor depending on the geographic region of the store serving the customer. For purposes of computing the store add-on, six different geographic areas exist. This add-on adjustment can contribute 17 to 27 cents of the final price. For example, the price of Double Fresh-from-the-Garden Burger on a whole grain bun with lettuce, tomato, and Gouda cheese in store A (which is in New York City) would be calculated as follows:

- Price = \$1.49 (burger size) + \$0.02 (two toppings at one cent each) + \$0.05 (bun) + \$0.10 (cheese) + \$0.25 (store location charge) = \$1.91

## Pricing the Burger

HFF can use Oracle Advanced Pricing to price their Fresh-from-the-Garden Burger.

The first step in this pricing process is to identify the pricing actions that will be used for pricing. HFF wants to create a single item for the burger and have the price changed based upon the components of the burger. In Oracle Advanced Pricing, HFF can use the price list functionality and create a single item for the Burger. The item can be priced using a dynamic formula with user entered and sourced pricing attributes for input to determine the various pricing components. HFF is using two pricing actions: price lists and dynamic formula. Each one of these actions needs to be set up.

### 1. Set Up Price List for the Burger

1. Navigate to the price list setup form using the Oracle Pricing Manager responsibility.
2. Create a price list with the following information:
  - Price list name: HFF Corporate Price List

- Currency: USD
3. Navigate to the list lines and enter item information for the Burger:
    - Product context: Item
    - Product Attribute: Item Number
    - Product value: Burger
    - Unit of measure: Each
    - Line type: Price list line
    - Application method: Unit price
    - Value: null
    - Dynamic formula: Fresh-from-the-garden Burger Calculation

## 2. Identify and Set Up Pricing Components

HFF wants to maintain one item for the Burger, yet the price of the Burger is dependant on many input variables. HFF will use the formula feature to dynamically price the Burger at the time that the burger is ordered. All of these variables need to be identified and the source of their input needs to be determined.

### Pricing Attributes

1. Identify all of the attributes that are used in the formula calculation:
  - Size of the Burger
  - Type of bun
  - Type of cheese
  - Number of toppings
  - Store/distribution center matrix
2. Determine which attributes will be entered at the time of order and which attributes can be sourced from the order request. Pricing attributes are setup by navigating to the Pricing Context window under the Oracle Pricing Manager responsibility.

For information on the setup for entered and sourced pricing attributes, see Overview of Attribute Management, page 17-1.

- Setup entered attributes:

- Size of the Burger
- Type of bun
- Type of cheese
- Setup sourced attributes:
- Number of toppings
- Distribution center location
- Store location

**Note:** The sourced pricing attributes in this case study are for illustrative purposes only and do not suggest features that are available in Oracle Advanced Pricing or other Oracle Applications.

### 3. Set Up Formula for the Fresh-from-the-Garden Burger

The Fresh-from-the-Garden Burger formula calculation is HFF's most difficult pricing structure. At order entry, the customer chooses the size of the burger, type of bun, type of cheese, and toppings. The store location where the burger is purchased and the distribution center from where the Burger was shipped are also part of the Burger's final price. HFF uses the following formula to determine the price of the Burger:

Formula: Burger size + type of bun + type of cheese + number of toppings + distribution center/store location = price

1. Navigate to the pricing setup form from the Oracle Pricing Manager responsibility to create a formula for the Burger:
  - Formula Name: Fresh-from-the-Garden Burger Calculation
  - Step: 1 + 2 + (NVL 3, 7) + (NVL 4, 75) + 6
2. Navigate to the formula lines and enter these formula details:

Formula Type	Pricing Attribute Context	Pricing Attribute	Component	Step
Factor	Null	Null	Burger Size	1

Formula Type	Pricing Attribute Context	Pricing Attribute	Component	Step
Factor	Null	Null	Type of Bun	2
Factor	Null	Null	Cheese	3
Pricing Attribute	Garden	Number of Toppings	Null	4
Numeric Constant	Null	Null	\$0.01	5
Factor	Null	Null	Distribution Store Matrix	6
Numeric Constant	Null	Null	0	7

3. Navigate to Factor and fill in this information about the burger size:

Base Pricing Attribute Context	Base Pricing Attribute	Operator	Value From	Value To	Adjustment Factor
Garden	Size	Equals	Single	Null	\$0.99
Garden	Size	Equals	Double	Null	\$1.49
Garden	Size	Equals	Triple	Null	\$1.99

4. Navigate to Factor and fill in this information about the type of bun:

Base Pricing Attribute Context	Base Pricing Attribute	Operator	Value From	Value To	Adjustment Factor
Garden	Bun	Equals	White	Null	.02

<b>Base Pricing Attribute Context</b>	<b>Base Pricing Attribute</b>	<b>Operator</b>	<b>Value From</b>	<b>Value To</b>	<b>Adjustment Factor</b>
Garden	Bun	Equals	Wheat	Null	.02
Garden	Bun	Equals	Whole Grain	Null	.05
Garden	Bun	Equals	Honey Wheat	Null	.05
Garden	Bun	Equals	Organic Grain	Null	.10

5. Navigate to Factor and fill in this information about the type of cheese:

<b>Base Pricing Attribute Context</b>	<b>Base Pricing Attribute</b>	<b>Operator</b>	<b>Value From</b>	<b>Value To</b>	<b>Adjustment Factor</b>
Garden	Cheese	Equals	Cheddar	Null	.02
Garden	Cheese	Equals	Jack	Null	.05
Garden	Cheese	Equals	Gouda	Null	.10

6. Navigate to Factor and fill in this information about the distribution costs:

<b>Line<sup>1</sup></b>	<b>Attribute Context</b>	<b>Base Pricing Attribute</b>	<b>Value From</b>	<b>Value To</b>	<b>Adjustment Factor</b>
1	Distribution Costs	Region	Northeast	-	.21
2	Distribution Costs	Region	Northeast	-	.25
3	Distribution Costs	Region	Northeast	-	.27
4	Distribution Costs	Region	Northeast	-	.30

Line <sup>1</sup>	Attribute Context	Base Pricing Attribute	Value From	Value To	Adjustment Factor
5	Distribution Costs	Region	Southeast	-	.11
6	Distribution Costs	Region	Southeast	-	.15
7	Distribution Costs	Region	Southeast	-	.17

**Additional Information:** <sup>1</sup>The following fields and values apply to each line listed in the previous table:

- Start Date: 01-Jan- 2001
- End Date: 31-Jan- 2001
- Operator: Equals

Associate these pricing attributes for line 1:

Associated Pricing Attribute Context	Associated Pricing Attribute	Operator	Value From	Value To
Store	Location	Equals	Baltimore	Null

Associate these pricing attributes for line 2:

Associated Pricing Attribute Context	Associated Pricing Attribute	Operator	Value From	Value To
Store	Location	Equals	New York City	Null

### Identify Pricing Rules

This pricing model applies to all Fresh-from-the-Garden Burgers sold to any customer in any store.

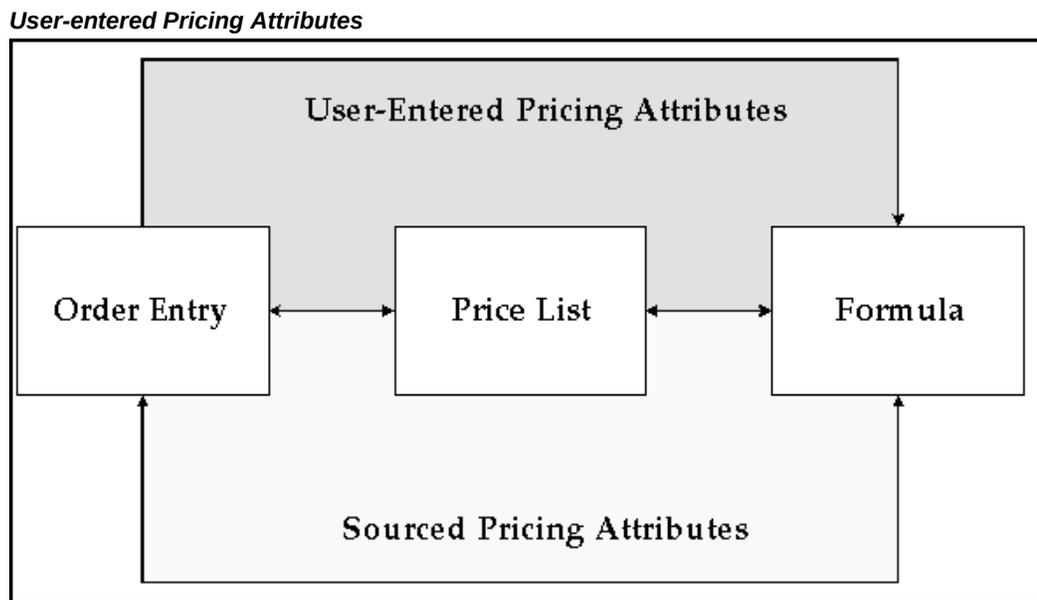
### Identify Controls

HFF requires that the selling price of the Burger can change if the costs of the

ingredients change. These price changes can only take effect on a monthly basis. HFF can use effectivity dates on the formula factors to control when these new costs go into effect.

#### 4. Calculate the Burger Price

Once HFF has set up the price list and formula, it can use the formula to calculate the Burger price. The following image illustrates the flow of information from the order entry system to price the Burger.



For example, a counter clerk in the New York City store enters the following information:

Burger Details	Values
Size of burger	Double
Type of bun	Whole grain
Cheese	Gouda
Toppings	Lettuce, tomato

The Fresh-from-the-Garden Burger is sent to the Oracle Advanced Pricing engine. The engine finds the price list for this item, HFF Corporate Price List, and determines that it is attached to the Fresh-from-the-Garden Burger calculation. Additional information

from the order entry system is necessary.

The following information is sent to the Oracle Advanced Pricing engine:

- Size of burger: Double
- Type of bun: Wheat
- Cheese: None
- Number of toppings: 4

To finish completing the formula calculation, the engine needs to source the following information:

- Store location: New York City
- Region: Northeast

The engine now calculates the price of the Burger using the formula:

- Burger size (double = \$1.49) + type of bun (wheat = \$0.05) + type of cheese (gouda = \$0.10) + number of toppings (2 = \$0.02) + distribution center/store location (northeast/New York City = \$0.25) = \$1.91

The price of \$1.91 is sent back to the order entry system.



---

# Lookups

## Overview of Lookups

### Accessorial Charges (ACCESSORIAL\_SURCHARGE)

Access Level: Extensible

Defines the seeded accessorial charges.

---

Code	Meaning
BONDED STORAGE	Bonded Storage
BREAK-BULK	Break-Bulk
CONSOLIDATIONS	Consolidations
DOCUMENTATION	Documentation
ESD HANDLING	ESD Handling
HANDLING	Handling
IMPORT/EXPORT	Import/Export Compliance
INSPECTION	Inspection
LABELING	Labeling
LOT COMBINING	Lot Combining

---

<b>Code</b>	<b>Meaning</b>
MERGE	Merge
PUT-AWAY	Put-Away
RECEIVING	Receiving
SCHEDULING	Scheduling
STAGING	Staging
STOP-OFF	Stop-Off
STORAGE	Storage
TRANSLOADING	Transloading

#### **Access Level (ACCESS\_LEVEL)**

Access Level: System

The Access Level lookups are used in pricing security to define the level of access granted to a price list or modifier (pricing entities).

<b>Code</b>	<b>Meaning</b>
MAINTAIN	Maintain
VIEWONLY	View Only

#### **Agreement Source Code (AGREEMENT\_SOURCE\_CODE)**

Access Level: System

The Agreement Source code, which displays in the Pricing Agreements window, identifies whether the agreement displayed is a Pricing or Contract type of agreement.

- Contract type agreements (agreement\_source\_code = MCTR) are created through the Agreements public API (Business Object API or BOI) only. They can be viewed but not maintained in the Pricing Agreements window.
- Pricing type agreements (agreement\_source\_code = PAGR) are created and maintained in the Pricing Agreements window.

<b>Code</b>	<b>Meaning</b>	<b>Function</b>
MCTR	Contract	Identifies Contract type agreements (agreement_source_code = MCTR) created through the Agreements public API. They can be viewed but not updated in the Pricing Agreements window.
PAGR	Pricing Agreement	Identifies Pricing type agreements (agreement_source_code = PAGR), which can be created, viewed, and maintained in the Pricing Agreements window.

#### **Additional Service Charge (SERVICE\_SURCHARGE)**

Access Level: Extensible

Defines seeded service charges that can be applied.

<b>Code</b>	<b>Meaning</b>
INSURANCE	Insurance
PACKAGING	Packaging
PALLETIZING	Palletizing
SHRINK WRAP	Shrink Wrapping

#### **Agreement Type (QP\_AGREEMENT\_TYPE)**

Access Level: Extensible

Enables the user to optionally categorize agreements by defining unique agreement types. For example, the user could set up an agreement type per contract type, or use the categorization for reporting purposes. An agreement type is optional on a pricing agreement.

The user can choose to use the seeded agreement types or add new types.

The following table lists the default (seeded) values for this lookup type:

<b>Code</b>	<b>Meaning</b>	<b>Function</b>
GSA	Government Services Agreement	Used to categorize pricing agreements.
STANDARD	Standard Terms and Conditions	Used to categorize pricing agreements.
VPA	Volume Purchase Agreement	Used to categorize pricing agreements.

### **Arithmetic Operator (ARITHMETIC\_OPERATOR)**

Access Level: System

The method by which a price or modifier is calculated. Used in the Price List and Modifier Setup UIs.

The following table lists the default (seeded) values for this lookup type:

<b>Code</b>	<b>Meaning</b>	<b>Function</b>
%	Percent	Modifier value is calculated as a per unit percentage of the List Price.
AMT	Amount	Modifier value is calculated as per unit amount +/- the List Price.
BLOCKPRICE	Block Price	Modifier value is calculated using a block price.
BREAKUNIT	Break Unit	Modifier value is calculated using a break unit, which can be either a Point or Range break.
LUMPSUM	Lump Sum	Modifier value is a fixed amount, not per unit.
NEWPRICE	New Price	Modifier value overrides the selling price.

<b>Code</b>	<b>Meaning</b>	<b>Function</b>
PERCENT_PRICE	Percent Price	List Price is derived as a percentage of an associated item.
UNIT_PRICE	Unit Price	List Price is a per unit price.

### **Attribute Mapping Options (QP\_ATTRIBUTE\_MAPPING\_OPTIONS)**

Access Level: User

Describes the attribute mapping options.

The following table lists the default (seeded) values for this lookup type:

<b>Code</b>	<b>Meaning</b>
N	Map all attributes
Y	Map all attributes used

### **Calculate Price Flag (QP\_CALCULATE\_PRICE\_FLAG)**

Access Level: User

Indicates the degree to which the price is frozen.

The following table lists the default (seeded) values for this lookup type:

<b>Code</b>	<b>Meaning</b>
N	Freeze price
P	Partial price
Y	Calculate price

### **column\_type\_code (COLUMN\_TYPE\_CODE)**

Access Level: System

This lookup describes the allowed column type code.

The following table lists the default (seeded) values for this lookup type:

<b>Code</b>	<b>Meaning</b>
DESC	Descriptive Flex Segment
KEY	Item Flex Segment
KEYFLEXFIELD	Descriptive Flex Segment
OFORM	Field in OFORM

### **Comparison Operator (COMPARISON\_OPERATOR)**

Access Level: System

Used when setting up Qualifiers and Pricing Attributes to define the rule as to how the search engine should evaluate the attribute on the request line.

The following table lists the default (seeded) values for this lookup type:

<b>Code</b>	<b>Meaning</b>	<b>Function</b>
=	Is	Qualifier/Pricing Attribute value on the incoming request should match the Qualifier/Pricing Attribute value.
BETWEEN	Between	Qualifier/Pricing Attribute value on the incoming request should be in the range defined by the Qualifier / Pricing Attributes.
Not =	Is Not	Qualifier Attribute value on the incoming request should NOT match the Qualifier Attribute value.

### **Comparison Operator - Framework (COMPARISON\_OPERATOR\_FWK)**

Access Level: System

Used when setting up qualifiers and pricing attributes to define how the search engine should evaluate the attribute on the request line.

The following table lists the default (seeded) values for this lookup type:

<b>Code</b>	<b>Meaning</b>	<b>Function</b>
=	Is	Value on the incoming request should match the Qualifier/Pricing Attribute value.
BETWEEN	Between	Value on the incoming request should be in the range defined by the Qualifier / Pricing Attributes.
Not =	Is Not	Value on the incoming request should NOT match the Qualifier Attribute value.

### **Conversion Date Type (CONVERSION\_DATE\_TYPE)**

Access Level: System

The following table lists the default (seeded) values for this lookup type:

<b>Code</b>	<b>Meaning</b>
FIXED	Fixed Date Type
PRICING_EFFECTIVITY_DATE	Pricing Effectively Date Type

### **Currency Conversion Method (CONVERSION\_METHOD)**

Access Level: System

Defines the currency conversion method.

The following table lists the default (seeded) values for this lookup type:

<b>Code</b>	<b>Meaning</b>
FIXED	Fixed Conversion Method
FORMULA	Formula Conversion Method
TRANSACTION	Transaction Conversion Method

### **Currency Precision Type (CURRENCY\_PRECISION\_TYPE)**

Access Level: System

Valid values for the profile option QP: Unit Price Precision Type. Indicates whether the currencies standard or extended precision should be used.

The following table lists the default (seeded) values for this lookup type:

Precision Type	Rounding Factor
Extended	Rounding Factor is defaulted to the currencies extended precision
Standard	Rounding Factor is defaulted to the currencies standard precision

### Effective Date Types (EFFECTIVE\_DATE\_TYPES)

Access Level: System

Effective date ranges of these types can optionally be defined on some types of modifier lists. The Search Engine will use these dates, if passed by the calling application, in addition to the pricing effective date to determine which Modifier Lists are eligible.

The following table lists the default (seeded) values for this lookup type:

Code	Meaning	Function
ORD	Order Date	Order Date must be within the date range.
SHIP	Requested Ship Date	Customer requested Ship Date must be within the date range.

### Entity Quick Search Criteria (ENTITY\_QUICK\_SEARCH\_CRITERIA)

Defines the entity search criteria used to query entities in pricing security.

The following table lists the default (seeded) values for this lookup type:

Code	Meaning
Name	Entity Name
OU	Owned By Operating Unit

### Formula Type (QP\_FORMULA\_TYPE)

These lookups determine how a formula calculates the price:

Code	Meaning	Definition
DYNAMIC	Dynamic	The list price resulting from the formula calculation is not calculated or stored anywhere until the sales order is entered with that Price List line item. When the sales order is entered, the pricing engine evaluates the formula and displays the final list price on the order.
STATIC	Static	If the formula is attached to a price list line for static calculation of the final list price, you can run a concurrent program at any time to calculate the final list price using the formula up front (not wait until order entry time) and also store it in the price list.

### Freight Charges Type (FREIGHT\_CHARGES\_TYPE)

Access Level: User

The following table lists the default (seeded) values for this lookup type:

Code	Meaning
MISCELLANEOUS	Miscellaneous Charges

### Grantee Type (GRANTEE\_TYPE)

Access Level: User

Grantee Type refers to the hierarchy of users to which privileges can be granted:

- Global: Includes all users with access to pricing menus.
- Operating Unit: Includes users of the named operating unit.
- Responsibility: Includes users with the specified or named responsibility.
- User: Specifies a specific named user.

The following table lists the default (seeded) values for this lookup type:

<b>Code</b>	<b>Meaning</b>
GLOBAL	Global
NONE	None
OU	Operating Unit
RESP	Responsibility
USER	User

### **Home Page Modifier List Type (HOMEPG\_MODIFIER\_LIST\_TYPE)**

Access Level: User

Defines the types of modifier lists available from the Home page in the HTML user interface.

The following table lists the default (seeded) values for this lookup type:

<b>Code</b>	<b>Meaning</b>
DEL	Deal
DLT	Discount List
PRO	Promotion
SLT	Surcharge List

### **Home Page Modifier SubList Type (HOMEPG\_MODIFIER\_SUBLIST\_TYPE)**

Access Level: User

Defines the modifier search criteria that can be selected when searching for modifier lists from the HTML UI.

The following table lists the default (seeded) values for this lookup type:

<b>Code</b>	<b>Meaning</b>
CURRENCY	Currency

<b>Code</b>	<b>Meaning</b>
LIST_NO	Number
LIST_TYPE	Type
MODIFIER_LINE_NO	Modifier Line Number
MODIFIER_LIST_NAME	Name

### **Home Page Pricelist Sublist Type (HOMEPG\_PRICELIST\_SUBLIST\_TYPE)**

Access Level: User

Defines the price list search criteria that can be selected from the HTML UI.

The following table lists the default (seeded) values for this lookup type:

<b>Code</b>	<b>Meaning</b>
CURRENCY	Currency
PRICE_LIST_NAME	Name

### **Home Page Search List Type (HOMEPG\_SEARCH\_LIST\_TYPE)**

Access Level: User

Defines the list types that can be searched for from the HTML UI Home page.

The following table lists the default (seeded) values for this lookup type:

<b>Code</b>	<b>Meaning</b>
MODLIST	Modifier List
PRL	Price List

### **Home Page View List Type (HOMEPG\_VIEW\_LIST\_TYPE)**

Access Level: User

You can view recently-created modifier lists or price lists in the HTML UI Home page.

The following lookups define the entities that can be viewed:

<b>Code</b>	<b>Meaning</b>
MODLIST	Recently Created Modifier Lists
PRL	Recently Created Price Lists

### **HTML Framework Page Titles (QP\_FWK\_MODIFIER\_LIST\_TITLES)**

Access Level: Extensible

These lookups define the HTML page names available in the Oracle Advanced Pricing HTML UI.

The following table lists the default (seeded) values for this lookup type:

<b>Code</b>	<b>Meaning</b>
CRDEL	Create Deal List
CRDLT	Create Discount List
CRPRO	Create Promotion List
CRSLT	Create Surcharge List
DETDEL	View Deal List
DEDTLT	View Discount List
DETPRO	View Promotion List
DETSLT	View Surcharge List
UPDDEL	Update Deal List
UPDDLTL	Update Discount List
UPDLINES	Update
UPDPRO	Update Promotion List
UPDSLTL	Update Surcharge List

### **Incompatibility Groups (INCOMPATIBILITY\_GROUPS)**

Access Level: Extensible

Incompatibility Groups enable the user to define which modifiers cannot be applied to a request line with other modifiers (incompatible) and which modifiers cannot be applied to a request line with any other modifier (are exclusive).

All modifiers in a phase that are incompatible should be assigned to the same Incompatibility Groups, LVL1 - LVL3, and any modifier in a phase that is exclusive should be placed in the EXCL - Exclusive Group.

Users may define additional incompatibility groups, but only the seeded EXCL - Exclusive group is treated as "incompatible with ALL."

The following table lists the default (seeded) values for this lookup type:

<b>Code</b>	<b>Meaning</b>	<b>Function</b>
EXCL	Exclusive group	Incompatible with all other Modifiers in a Phase.
LVL1	Level 1 Incompatibility	Incompatible with other Modifiers in this incompatibility group in a Phase.
LVL2	Level 2 Incompatibility	Incompatible with other Modifiers in this incompatibility group in a Phase.
LVL3	Level 3 Incompatibility	Incompatible with other Modifiers in this incompatibility group in a Phase.

### **Incompatibility Resolution Code (INCOMPAT\_RESOLVE\_CODE)**

Access Level: System

Methods of deciding which modifier should be selected when multiple modifiers in the same incompatibility group are eligible to be applied to a request line in the same pricing phase. The method for resolving incompatibility is specified by pricing phase when maintaining pricing phases in the Event to Phase Mapping Setup Up.

The following table lists the default (seeded) values for this lookup type:

<b>Code</b>	<b>Meaning</b>	<b>Function</b>
BEST PRICE	Best Price	Search Engine selects the modifier that gives the lowest price to the customer.
PRECEDENCE	PRECEDENCE	Search Engine selects the Modifier with the lowest precedence, i.e. the highest specificity.

### **Levels of Sourcing Rule for an Attribute (QP\_ATTRIBUTE\_MAPPING\_LEVEL)**

Access Level: Extensible

Define levels of the sourcing rule for an attribute.

The following table lists the default (seeded) values for this lookup type:

<b>Code</b>	<b>Meaning</b>
LINE	Order Line Level Sourcing Rule
ORDER	Order Header Level Sourcing Rule

### **Limit Attribute Type (LIMIT\_ATTRIBUTE\_TYPE)**

Access Level: System

Indicates the entity of a limit dimension attribute.

The following table lists the default (seeded) values for this lookup type:

<b>Code</b>	<b>Meaning</b>
QUALIFIER	Qualifier Attribute
PRICING	Pricing Attribute
PRODUCT	Product Attribute

### **Limit Basis (QP\_LIMIT\_BASIS)**

Access Level: System

Indicates the basis on which a promotional cap limit is calculated. The following table lists the default (seeded) values for this lookup type:

<b>Code</b>	<b>Meaning</b>
ACCRUAL	Accrual Units
CHARGE	Charge Amount
COST	Cumulative Discount
GROSS_REVENUE	Gross Revenue
QUANTITY	Item Quantity
USAGE	Usage

### **Limit Exceed Action (LIMIT\_EXCEED\_ACTION)**

Access Level: System

Indicates the action to take if a promotion or modifier applied to a sales order exceeds the promotional cap (available balance) of a promotional limit.

The following table lists the default (seeded) values for this lookup type:

<b>Code</b>	<b>Meaning</b>
HARD	Adjust the order benefit amount so that the order meets but does not exceed the promotional limit. Apply that adjusted amount to the order. Inactivate the modifier or modifier list.
SOFT	Apply the full benefit to the order and then place a promotional hold on the order.

### **Limit Level (LIMIT\_LEVEL)**

Access Level: System

Indicates how the pricing engine should maintain the promotional limit balance transactions.

The following table lists the default (seeded) values for this lookup type:

Code	Meaning
TRANSACTION	The pricing engine maintains a consumption record for each order that consumes the limit.
ACCROSS_TRANSACTION	The pricing engine maintains one consumption record for all orders that consume the limit.

### Line Type (QP\_LINE\_TYPE)

Access Level: User

Indicates the type of line within the pricing request.

Code	Meaning
LINE	Line
ORDER	Order

### List Type Code (LIST\_TYPE\_CODE)

Access Level: System

Categorizes the type of list that groups price list lines or modifiers. Used for validation, including which types of lines can be included on the list, and reporting purposes.

The following table lists the default (seeded) values for this lookup type:

Code	Meaning
AGR	Agreement Price List
CHARGES	Freight and Special charge List
DEL	Deal
DLT	Discount List
PML	Price Modifier List
PRL	Standard Price List

Code	Meaning
PRO	Promotion
SLT	Surcharge List

### Markup Operator (MARKUP\_OPERATOR)

Access Level: System

These values are used with multi-currency conversion lists to determine how the Markup Value (for example, 10) is applied against the base currency, either percent or amount.

Code	Meaning
%	Percent
AMT	Amount

### Miscellaneous Charges (MISCELLANEOUS)

Access Level: Extensible

Defines user-defined miscellaneous charges.

Code	Meaning
MISC	Miscellaneous Charges
PENALTY	Charge for late payment
RESTOCKING	Restocking Fee
RETURN	Return Fee

### Modifier Level Code (MODIFIER\_LEVEL\_CODE)

Access Level: System

Determines what qualifiers and pricing attributes are considered by the search engine when deciding if a request line qualifies for a modifier. This code also determines at what level, i.e. individual line or summary, a modifier should be applied to the request.

The following table lists the default (seeded) values for this lookup type:

<b>Code</b>	<b>Meaning</b>	<b>Function</b>
Line	Line	Line Group
Line Group	Group of lines	The quantity, in the pricing UOM, and amount spent on an item is summed across all request lines. Hence the total item quantity and amount, on the request, or total quantity and amount at a level in the product hierarchy, is considered by the search engine when deciding if a modifier is qualified or not. Modifier application is at the request line level.
Order	Order	Only qualifiers or pricing attributes of the summary request line, or header, are considered by the search engine when deciding if a modifier is qualified. Note: it is not possible for a header level modifier to be qualified by a request line. Modifier application is at the summary request line, or header level.

**Modifier List Framework Search Options (QP\_MLH\_SEARCH\_OPTIONS\_TYPE)**

Access Level: Extensible

Determines the available search options for modifier lists in the HTML UI.

**Modifier List Line Type Code (LIST\_LINE\_TYPE\_CODE)**

Access Level: System

Defines the behavior of a List Line. A List Line maybe a Price List Line or a type of Modifier: for example, a price adjustment, benefit or charge.

The following table lists the default (seeded) values for this lookup type:

<b>Code</b>	<b>Meaning</b>	<b>Function</b>
CIE	Coupon Issue	Creation of a coupon that qualifies for a discount or promotional goods on a future request.
DIS	Discount	Reduction of the list price, or selling of the previous pricing bucket, according to the calculation rules of the arithmetic operator.
FREIGHT_CHARGE	Freight and Special Charges	Calculation of monetary charges based on attributes of a request line. These do not affect the selling price on the request line.
IUE	Item Upgrade	Substitution of one item for another on a request line according to the predefined promotional upgrade relationship between the two items.
OID	Other item Discount	A discount for which eligibility can be qualified by one or more request lines, but is applied to the same or different request lines that are on the request.
PBH	Price Break Header	A series of base price or price adjustments that are eligible for application to the pricing request according to a delimited break unit range and the rules of the break type.
PLL	Price List Line	Setting of the base price of an item or level in product hierarchy.

<b>Code</b>	<b>Meaning</b>	<b>Function</b>
PMR	Price Modifier	One or more pricing attributes, whose value or range of values is used to derive a factor on a formula line.
PRG	Promotional Goods	A discount for which eligibility can be qualified by one or more request lines, but for which a new request line is created for the discounted item.
SUR	Surcharge	List price increase or selling of the previous pricing bucket, according to the calculation rules of the arithmetic operator.
TSN	Term Substitution	Changing value of qualifier attribute in terms context on request line. Seeded qualifier attributes in terms context are Freight, Shipping and Payment Terms.

### **Multi Currency Attribute Type (MULTI\_CURR\_ATTRIBUTE\_TYPE)**

Access Level: System

<b>Code</b>	<b>Meaning</b>
PRICING	Pricing Attribute
PRODUCT	Product Attribute
QUALIFIER	Qualifier Attribute ORGANIZER_FORMULA_TYPE

### **Optional Currency (OPTCUR)**

Access Level: User

When Optional Currency is selected for a modifier, the modifier can be used with any price list regardless of its currency. Such modifiers could be used with both single price

lists that are set up in a single currency, or with multi-currency enabled price lists.

---

<b>Code</b>	<b>Meaning</b>
OPTCUR	Optional Currency

---

**Organizer Formula Type (ORGANIZER\_FORMULA\_TYPE)**

Access Level: System

Defines the formula types to be used in the Pricing Organizer.

---

<b>Code</b>	<b>Meaning</b>
ANY	<ANY FORMULA>
NO	<NO FORMULA> ORGANIZER_PRIC_ATTR_OPTION

---

**Organizer Pricing Attributes Option (ORGANIZER\_PRIC\_ATTR\_OPTION)**

Access Level: System

Defines the pricing attributes option to be used in the Pricing Organizer.

---

<b>Code</b>	<b>Meaning</b>
N	No Pricing Attributes
P	Pricing Attributes
S	Not Specified

---

**Organizer Product Attributes Option (ORGANIZER\_PROD\_ATTR\_OPTION)**

Access Level: System

Defines the product attributes option to be used in the Pricing Organizer.

---

<b>Code</b>	<b>Meaning</b>
N	No Products
P	Products

---

Code	Meaning
S	Not Specified

#### Organizer Qualifiers Option (ORGANIZER\_QUAL\_OPTION)

Access Level: System

Defines the qualifiers option to be used in the Pricing Organizer.

Code	Meaning
N	No Qualifiers
Q	Qualifiers
S	Not Specified

#### Price Break Type Code (PRICE\_BREAK\_TYPE\_CODE)

Access Level: System

Rules that determine which delimited break unit range or ranges the qualifying break unit quantity falls into.

The following table lists the default (seeded) values for this lookup type:

Code	Meaning	Function
POINT	Point	Volume break in which each volume of break unit gets price/discount in the break range into which it falls.
RANGE	Range	Volume break in which each volume of break unit gets base price/modifier in the break range within which the <i>total</i> volume falls.
RECURRING	Recurring	Volume break in which the modifier is given <i>for each</i> volume of break unit that falls into the break range.  Used for modifiers only.

## Price Formula Line Type Code (PRICE\_FORMULA\_LINE\_TYPE\_CODE)

Access Level: System

Defines the behavior of a formula line. The first table lists the lookups for basic pricing in Order Management, and the second table lists the lookups defined for Oracle Pricing. The following tables list the default (seeded) values for this lookup type:

- Basic Pricing in OM
- Oracle Pricing Only

Code	Function	Meaning
ML	Factor List	Formula uses a price modifier list to derive the value for the formula line.  A price modifier list is a grouping of price modifier lines, each line having one or more pricing attributes, whose value or range of values is used to derive a factor.
NUM	Numeric Constant	Fixed value
PRA	Pricing Attributes	Formula takes as input the pricing attribute for the item referenced by the formula line.

The following table lists the default (seeded) values for the Price Formula Line Type Code in Oracle Pricing Only:

Code	Function	Meaning
FUNC	Function	Formula uses a function to derive the value for the formula line
LP	Price List Line	Formula takes as input the list price of the price list line to which it is attached

<b>Code</b>	<b>Function</b>	<b>Meaning</b>
ML	Factor List	Formula uses a price modifier list to derive the value for the formula line.  A price modifier list is a grouping of price modifier lines, each line having one or more pricing attributes, whose value or range of values is used to derive a factor.
MV	Modifier Value	A modifier value
NUM	Numeric Constant	Fixed value
PLL	Price List Line	Formula takes as input the list price from the price list line (any price list line) referenced by the formula line.
PRA	Pricing Attribute	Formula takes as input the pricing attribute for the item referenced by the formula line.

### **Price Rounding (PRICE\_ROUNDING)**

Access Level: System

Defines the price rounding option selected. The following tables list the default values for this lookup type:

<b>Code</b>	<b>Meaning</b>
Factor	Enforce Price List Rounding Factor
Precision	Enforce Currency Precision

These values are defined as follows:

- **Enforce Price List Rounding Factor:** Use the value of the Round To field on the price list for rounding the display of the value on the price list line. Use the value of the

Round To field on the price list for rounding the engine result of the static formula calculation and displaying this value on the price list line

- Enforce Currency Precision: Use the currency precision for controlling the number of decimal place to display on the Price List and for rounding the engine result of the static formula calculation and displaying this value on the Price List Line.

If neither box is selected, then no limit on the number of decimal places entered on the price list and no rounding of the engine result of the static formula calculation and displaying this value on the Price List Line. Both boxes cannot be selected at the same time.

### **Pricing Control Flag (QP\_PRICING\_CONTROL\_FLAG)**

Access Level: User

Used by engine to identify whether the engine should just recalculate the selling price without retrieving new adjustments or calculate the selling price by retrieving new adjustments.

The following table lists the default (seeded) value for this lookup type:

<b>Code</b>	<b>Meaning</b>
C	Calculate Engine
N	Search Engine
Y	Calculate & Search Engine

### **Pricing Events (PRICING\_EVENTS)**

Access Level: System

A pricing event is a "point" in the process flow of the transaction system/calling application when a call is made to the Pricing Engine (analogous to a Workflow Event). Each event represents a stage in the order cycle at which pricing is performed.

The following seeded lookups are for Oracle Order Management integration with pricing. The information returned by pricing such as base prices, price adjustments, promotions, freight charges and so on, depends on the pricing phases that are processed for this event.

**Note:** In this release, you cannot create new pricing events.

The following table lists the default (seeded) values for this lookup type:

<b>Code</b>	<b>Meaning</b>	<b>Function</b>
BATCH	Batch Processing	Calls pricing engine when orders are processed in batch, replaces 'Line' and 'Order' events.
BOOK	Book Order	Calls pricing engine as order is booked.
FTE_APPLY_MOD	FTE: Apply Modifiers to Price	Calls pricing engine when a modifier is priced in Oracle Transportation Execution.
FTE_PRICE_LINE	FTE: Price a Transportation Line	Calls pricing engine when a transportation line in Oracle Transportation Execution is priced.
ICBATCH	INV: Batch Processing for Intercompany Transfer Pricing	Calls pricing engine when batch processing transaction is initiated in Oracle Inventory.
LINE	Enter Order Line	Calls pricing engine to get line level modifiers as user navigates out of a line or saves the order.
ORDER	Save Order Event	Calls pricing engine, as user saves order, to get order level modifiers and other benefits, which depend on multiple order lines.
PRICE	Fetch List Price	Calls pricing engine to get base price as user enters item, quantity and unit of measure on the order line.
PRICE_LOAD	Price a Logistics Load	
REPRICE_LINE	Reprice Line	Pricing event that can be used to reprice an order line at any point during the order flow.

Code	Meaning	Function
SHIP	Enter Shipments	Calls pricing engine as order is shipped.

### **Pricing Group Sequence (PRICING\_GROUP\_SEQUENCE)**

Access Level: Extensible

A Pricing Group Sequence controls the application order of price adjustments and retrospective discounts such as accruals. The sequence of application of these modifiers becomes important when the adjustment or accrual value is derived from the selling price (the price resulting from applying prior price adjustments) rather than the list price. This is known as discounts on discounts or cascading discounts.

The sequence number of the group determines which order the calculation engine will apply the modifiers. The pricing group sequence allows the user to place all price adjustments and retrospective discounts in a pricing bucket. All modifiers in a bucket are additive, meaning that the adjustment amount for all modifiers in a bucket is calculated from the final selling price, or subtotal, of the previous bucket. The user can add additional pricing group sequences or buckets if they require further subtotals or cascading of modifiers. Pricing Group Sequence 0 is reserved for base price calculation.

The following table lists the default (seeded) values for this lookup type:

Code	Meaning	Function
0	Base Price	Base Price calculation
1	Price Adjustments Bucket 1	First modifier subtotal
2	Price Adjustments Bucket 2	Second modifier subtotal
3	Price Adjustments Bucket 3	Third modifier subtotal

### **Pricing Engine Request Viewer Options (QP\_REQUEST\_VIEWER\_OPTIONS)**

Access Level: User

Used to control the options for Pricing Engine Request Viewer.

The following table lists the default (seeded) value for this lookup type:

<b>Code</b>	<b>Meaning</b>
N	Request Viewer Off
V	Request Viewer On, but Debug Log is not visible in Viewer
Y	Request Viewer On

### **Pricing Security Bulk Create Privileges Results Status (BULK\_CREATE\_STATUS)**

Access Level: User

These values indicate the status of transaction when the Pricing Administrator creates a bulk grant in the Bulk Create Privileges page (pricing security).

The following table lists the default (seeded) value for this lookup type:

<b>Code</b>	<b>Meaning</b>
C	Privilege has existed and been updated correctly.
F	Failed to create the wanted privilege.
N	New privilege has been created successfully.
U	Privilege has existed and remains unchanged.

### **Pricing Status Code (QP\_PRICING\_STATUS)**

Access Level: User

Indicates the returned status of the pricing engine.

The following table lists the default (seeded) value for this lookup type:

<b>Code</b>	<b>Meaning</b>
CALC	Error in calculation engine
DUPLICATE_PRIC E_LIST	Duplicate price list
D_PBH	Delete in Price Break Processing

<b>Code</b>	<b>Meaning</b>
FER	Error processing formula
GSA	GSA violation
INVALID_BEST_P RICE	Could not resolve best price
INVALID_INCOM P	Could not resolve incompatibility
INVALID_UOM	Invalid unit of measure
INVALID_UOM_C ONV	Unit of measure conversion not found
IPL	Invalid price list
N	New record created
OER	Other error
OTHER_ITEM_BE NEFITS	Other Item Benefits
UOM	Failed to price unit of measure
UPDATED	Updated
X	Unchanged

**Print on Invoice Flag (PRINT\_ON\_INVOICE\_FLAG)**

Access Level: System

This code tells whether to print a discount on an invoice or not. The Print on Invoice flag is no longer available on the Modifier window.

The following table lists the default (seeded) value for this lookup type:

<b>Code</b>	<b>Meaning</b>
M	Print Message

<b>Code</b>	<b>Meaning</b>
N	Don't Print Discount
Y	Print Discount

#### **Process Code (QP\_PROCESS\_CODE)**

Access Level: User

Used by the engine for selecting lines for calculation.

The following table lists the default (seeded) value for this lookup type:

<b>Code</b>	<b>Meaning</b>
D	Deleted
N	New
X	Unchanged

#### **Proration Type (PRORATION\_TYPE)**

Access Level: System

Defines methods used to prorate discounts (none, category, all lines).

The following table lists the default (seeded) value for this lookup type:

<b>Code</b>	<b>Meaning</b>
C	Category
N	None
Y	All Lines

#### **QP\_INCREMENT\_DECREMENT (QP\_INCREMENT\_DECREMENT)**

Access Level: User

Defines the price list mass maintenance value change types.

The following table lists the default (seeded) value for this lookup type:

<b>Code</b>	<b>Meaning</b>
CV	Clear Value
IP	Percent
IV	Amount
NV	Replace Value
XX	No Change

#### **QP\_MM\_ACTION\_TYPE (QP\_MM\_ACTION\_TYPE)**

Access Level: User

Defines the pricing mass maintenance action type.

The following table lists the default (seeded) value for this lookup type:

<b>Code</b>	<b>Meaning</b>
NV	End Date and Create New
OV	Override

#### **QP\_MM\_DATE\_CHANGE (QP\_MM\_DATE\_CHANGE)**

Access Level: User

Defines the pricing mass maintenance date change type.

The following table lists the default (seeded) value for this lookup type:

<b>Code</b>	<b>Meaning</b>
NO	No Change
YES	Change Date

#### **QP\_MM\_FORMULA\_CHANGE (QP\_MM\_FORMULA\_CHANGE)**

Access Level: User

Defines the pricing mass maintenance formula change criteria.

The following table lists the default (seeded) value for this lookup type:

<b>Code</b>	<b>Meaning</b>
NO	No Change
REM_FOR	Remove Static and Dynamic Formula
REP_DYN	Replace All Formulas with Dynamic Formula
REP_STA	Replace All Formulas with Static Formula

### **Query Operator (QUERY\_OPERATOR)**

Access Level: System

Indicates the available values for a query.

The following table lists the default (seeded) value for this lookup type:

<b>Code</b>	<b>Meaning</b>
=	Equal
BETWEEN	Between
LIKE	Like

### **Reason Code (QP\_CHANGE\_REASON\_CODE)**

Access Level: Extensible

Indicates the reason for an adjustment to a promotional limit balance. You adjust a balance by creating a consumption record against it.

The following table lists the default (seeded) values for this lookup type:

<b>Code</b>	<b>Meaning</b>
MISC	Miscellaneous

### **Rebate Payment Transaction Type Code (REBATE\_TRANSACTION\_TYPE\_CODE)**

Access Level: System

Defines the Rebate Payment Transaction Type Code.

The following table lists the default (seeded) value for this lookup type:

<b>Code</b>	<b>Meaning</b>
CREDIT_MEMO	Credit Memo

### **Related Modifier Group Type (RLTD\_MODIFIER\_GRP\_TYPE)**

Access Level: System

Used by Oracle Pricing internally to identify relationships between, and functional groupings, of modifiers.

The following table lists the default (seeded) values for this lookup type:

<b>Code</b>	<b>Meaning</b>	<b>Function</b>
BENEFIT	Benefit	Identifies those modifiers that are given as a benefit once the qualification criteria has been met.
COUPON	Coupon	Identifies the benefit that is given for a Coupon Issue.
PRICE BREAK	Price Break	Records which modifiers are price break lines for a price break.
QUALIFIER	Qualifier	Identifies those modifiers that the request must qualify for in order to get a benefit.

### **Relationship Type Code (QP\_RELATIONSHIP\_TYPE\_CODE)**

Access Level: User

Used in identifying the relation between the lines.

The following table lists the default (seeded) value for this lookup type:

<b>Code</b>	<b>Meaning</b>
BUY	Buy
DETAIL_TO_DETAIL	Detail to Detail

<b>Code</b>	<b>Meaning</b>
GENERATED_LINE	Generated Line
GET	Get
LINE_TO_DETAIL	Line to Detail
LINE_TO_LINE	Line to Line
ORDER_TO_LINE	Order to Line
PBH_LINE	Price Break Header Line
RELATED_ITEM_PRICE	Related Item Price
SERVICE_LINE	Service Line

### **Request Type (REQUEST\_TYPE)**

Access Level: Extensible

A Request Type indicates to the pricing engine the type of transaction being priced. This information is important to pricing, as the engine will use this information to only consider data created specifically to price this particular type of transaction.

The following seeded lookup codes are for Oracle Order Management integration with pricing. Any application that wants to use Oracle Pricing should create a request type lookup code to identify its transaction.

The following table lists the default (seeded) value for this lookup type:

<b>Code</b>	<b>Meaning</b>	<b>Function</b>
ASO	Order Capture	Used to price an Order Capture transaction.
FTE	Oracle Transportation Execution Shipment	Used to price an Oracle Transportation Execution Shipment.
IC	Inter Company Invoicing	Used to price Inter Company Invoicing.

<b>Code</b>	<b>Meaning</b>	<b>Function</b>
MSD	Demand Planning	Used to price a Demand Planning transaction.
OKC	Oracle Contracts	Oracle Contracts Core
ONT	Order Management Order	Used to price an Order Management Order.

### **Revision Reason Code (QP\_REVISION\_REASON\_CODE)**

Access Level: User

Defines the reason for revising the agreement header and lines.

The following table lists the default (seeded) value for this lookup type:

<b>Code</b>	<b>Meaning</b>
NOREV	No Revision for this Agreement

### **Rounding Type (QP\_ROUNDING\_TYPE)**

Access Level: User

Used by the engine for rounding the price.

The following table lists the default (seeded) value for this lookup type:

<b>Code</b>	<b>Meaning</b>
N	No Rounding
Q	Look at the QP Profile QP: SELLING PRICE ROUNDING OPTIONS
U	Round Price After Adding Unrounded List Price and Adjustments
Y	Round Selling Price and Adjustments

### **Security Control on/off (QP\_SECURITY\_CONTROL)**

Access Level: User

Defines the available values for the profile option to turn security on or off.

The following table lists the default (seeded) value for this lookup type:

Code	Meaning
OFF	Off
ON	On

### Security Entity Type (SECURITY\_OBJECT\_TYPE)

Access Level: User

Defines the available entity types to which security privileges can be assigned.

The following table lists the default (seeded) value for this lookup type:

Code	Meaning
AGR	Agreement Pricelist
MOD	Modifier
PRL	Standard Price list
SET	Pricing Entity Set

### Selling Price Rounding Options (QP\_ROUNDING\_OPTIONS)

Access Level: User

Defines the available rounding options for the selling price.

The following table lists the default (seeded) value for this lookup type:

Code	Meaning	Description
NO_ROUND	No: unrounded listprice + unrounded adj	No Rounding: List Price and adjustments are not rounded. The selling price also is not rounded.
NO_ROUND_ADJ	Additive: round(listprice + adj); unrounded Freight	Round Selling Price after adding unrounded list price and unrounded adjustments. Freight charges are unrounded.

Code	Meaning	Description
ROUND_ADJ	Individual: round(listprice) + round(adj)	Round Selling Price and adjustments

### Source System (SOURCE\_SYSTEM)

Access Level: Extensible

Defines the seeded source systems used when setting up Pricing Transaction Entities.

The following table lists the default (seeded) value for this lookup type:

Code	Meaning
AMS	Oracle Marketing
ASO	Oracle Capture
FTE	Oracle Transportation Execution
INV	Oracle Inventory
OKC	Oracle Contracts
QP	Oracle Pricing

### Surcharges Type (SURCHARGES\_TYPE)

Access Level: Extensible

Defines the seeded types of surcharges.

The following table lists the default (seeded) value for this lookup type:

Code	Meaning
ACCESSORIAL	Accessorial Charge
SERVICE_SURCHARGE	Additional Service Charge

### Types of Context (QP\_CONTEXT\_TYPE)

Access Level: Extensible

Defines the type of available contexts.

The following table lists the default (seeded) value for this lookup type:

<b>Code</b>	<b>Meaning</b>
PRICING_ATTRIBUTE	Pricing Context
PRODUCT	Product Context
QUALIFIER	Qualifier Context

### **Types of Pricing Transaction Entities (QP\_PTE\_TYPE)**

Access Level: Extensible

Indicates the type of pricing transaction entity.

The following table lists the default (seeded) value for this lookup type:

<b>Code</b>	<b>Meaning</b>
DEMAND	Demand Planning
INTCOM	Intercompany Transaction
LOGSTX	Logistics
ORDFUL	Order Fulfillment

### **Types of Segment Levels (QP\_SEGMENT\_LEVEL)**

Access Level: Extensible

Indicates the level at which an attribute can be sourced.

The following table lists the default (seeded) value for this lookup type:

<b>Code</b>	<b>Meaning</b>	<b>Description</b>
BOTH	Order Header As Well As Order Line Sourcing	Attribute Sourced at Order Header as well as Order Line
LINE	Order Line Sourcing Only	Attribute Sourced at Order Line only

Code	Meaning	Description
ORDER	Order Header Sourcing Only	Attribute Sourced at Order Header only

### Usage Pricing Type (QP\_USAGE\_PRICING\_TYPE)

Access Level: User

Indicates the usage pricing type.

The following table lists the default (seeded) value for this lookup type:

Code	Meaning
REGULAR	Regular
BILLING	Billing
AUTHORING	Authoring

### Ways of Sourcing an Attribute (QP\_SOURCING\_METHOD)

Access Level: User

The sourcing method indicates different ways an attribute can be sourced.

The following table lists the default (seeded) value for this lookup type:

Code	Meaning
ATTRIBUTE MAPPING	Sourcing Rule Needs To Be Defined.
CUSTOM SOURCED	Actual Sourcing Code Provided: No Sourcing Rule Needed
USER ENTERED	Attribute Sourced While Pricing An Order

### Yes No (YES\_NO)

Access Level: System

Defines the seeded Yes and No values.

The following table lists the default (seeded) value for this lookup type:

---

<b>Code</b>	<b>Meaning</b>
No	No
Yes	Yes

---

---

# Index

## A

---

- access privileges. See pricing security, privileges, 6-14
- accruals, 10-21
  - accounting limitations, 10-22
  - buckets with monetary accruals, 10-21
  - fields reserved for future use, 10-21
  - redemption, 10-22
- accumulated range breaks, 10-23
  - runtime sourcing, 10-23
- accumulated range breaks in modifiers, 10-20
- Across All Phases, 19-3
- active/inactive lists, 23-4
- ADPATCH errors, troubleshooting, 24-27
- Advanced Pricing
  - architectural overview, 23-2
  - engine processing, 23-3
  - features, 1-34
  - methodology, 3-5
  - overview, 3-1
  - process flow for implementation, 1-41
  - profile options, 5-3
  - search engine, 23-3
  - terminology, 1-33
- Advanced Pricing Command Center
  - configure flexfields, 4-4
  - load pricing data, 4-4
  - profile options, 4-3
  - review security setup, 4-2
  - set up, 4-2
  - setup and configuration, 4-2
- Advanced Pricing vs. Basic Pricing
  - technical considerations, 23-1
- agreements, 1-38
- All Lines Adjustment, 19-3
- API, 11-2
- architectural changes, Advanced Pricing, 23-3
- archiving pricing entities
  - overview, 11-1
- ask for promotions, 10-20
- assigning PTE to existing Modifiers, 17-41
- attribute management
  - contexts, deleting, 17-11
  - creating
    - attributes, 17-3
    - contexts, 17-3
    - PTEs, 17-11
  - deleting attributes, 17-11
  - mapping
    - building attribute mapping rules, 17-31
  - overview, 17-1
  - pricing transaction entity (PTE), 17-25
  - terms, 17-1
  - troubleshooting, 17-31
  - upgrade considerations, 17-1
- attribute management, troubleshooting, 17-33
- attribute mapping, 1-37
  - checklist for building attribute mapping rules, 17-21
  - of type ATTRIBUTE MAPPING, 17-18
  - upgrading attribute mapping rules, 17-40
- attribute rounding errors, 5-42
- attributes, 2-4
  - creating, 2-4, 17-3

- deleting, 17-11
- precedence, 16-2
- pricing, 3-12
- Attributes window, 20-20
- auditing
  - auditing modifiers, 12-1
  - audit setup, 12-2
- auditing modifiers, 12-1
- auditing price lists, 12-1
- Audit Setup
  - auditing price lists, 12-1

## **B**

---

- basic pricing definition, 1-2
- best price, 16-1
- Build Attribute Mapping Rules program, 17-21
- Bulk Create Privileges page, 6-14
- bulk loader, 7-1
  - interface tables, 7-2
  - profile options, 7-1
  - QP\_Batch Size for Bulk Import, 7-1
  - QP: Bulk Import of Price List program, 7-12
  - sample scripts, 7-11
  - uploading pricing data, 8-4

## **C**

---

- Calculate Price Flag, 10-15
- calculation engine, 23-21
  - GSA pricing, 23-21
- calling application, 1-33
- case study: Using Oracle Advanced Pricing Formulas for Healthy Fast Food, F-1, F-1, F-2
- categories, hierarchal, 17-25
- category set, 17-25
- changed lines API, 21-28
- charge periodicity, 21-42
- charges
  - header/all lines, 19-3
- code
  - group of lines level, 23-11
  - line level, 23-11
  - modifier level, 23-10
  - order level, 23-11
- Complex Maintenance Repair and Overhaul (CMRO) PTE Attributes, B-2
- Concurrent program

- QP: Enterprise Command Center Data Load, 4-4
- contexts
  - creating, 17-3
  - deleting, 17-11
- contexts and attributes
  - creating
    - pricing, 2-4
    - qualifier, 2-4
  - restoring seeded values, 17-31
- controls, pricing, 3-2
- cost to charge conversion formulas, seeded, C-1
- coupon issue, 10-20
- creating PTE and Attribute links, 17-39
- cross order volume-based modifiers, 21-21
- currencies
  - enabling, 2-10
  - single vs. multi-currency price lists, 3-10
- customer class, defining, 2-8
- customer hierarchy, 1-33
- customers, defining, 2-9
- customer sites, defining, 2-9

## **D**

---

- Debug: Generate List Line Details, 5-20
- Debug Log window, 20-30
  - analyzing error messages, 20-31
- deleting
  - attributes, 17-11
  - contexts, 17-11
- Demand Planning PTE Attributes, B-3
- diagnostics and troubleshooting, 24-1
  - pricing engine, 24-1
- duration, calculating, 21-36
- duration and partial period pricing, 21-38

## **E**

---

- effectivity dates, 3-17
- eligibility, 23-3
- entity sets. See pricing security, 6-17
- entity usage. See pricing security, 6-11
- Entity Usage page, 6-11
- error messages
  - price lists, 24-4
- evaluation of passed in modifier, 21-20
- events. See pricing events, 19-1

events and phases, 19-1  
examples of Usage Proration with Oracle Service Contracts (OKS), 21-39  
excluding item categories, 10-11  
Express Create Privilege page, 6-15  
extendibility features, 23-23  
extensibility, 3-2

## F

---

flexfields and pricing attributes, 17-3  
formulas, 1-38  
    GET\_CUSTOM\_PRICE, 18-1  
    seeded, C-1  
Formula Step Values window, 20-28  
freight and special charges, 1-40  
freight charges integration, error message, 24-12  
freight cost type, defining, 2-10  
freight terms, defining, 2-10  
functional security. See pricing security, 6-1

## G

---

GET\_CATALOG\_002 (price books), 9-6  
GET\_CUSTOM\_PRICE, 18-1  
get custom price, 1-41, 18-1  
    customizing, 18-7  
    implementation overview, 18-1  
    implementation steps, 18-4  
get region and benefit items, 10-15  
Global box, 6-7, 6-11  
global usage in pricing security, 6-11  
group of lines discounts, 10-8  
group of lines level code, 23-11  
GSA (General Services Administration) pricing, 1-38, 23-21

## H

---

Header Level Adjustments, 19-3  
hierarchical categories, support for , 17-25  
hierarchical item categories, 8-2  
High Volume Order Processing (HVOP), 22-1  
    Attribute Mapping Rules Restrictions, 22-2  
    QP\_High Volume Order Processing Compliance, 22-1  
HTML UI (user interface) in pricing  
    accessing, 3-5

    modifier lists and modifiers, 10-3  
    overview, 3-4  
    price lists and price list lines, 8-1  
HVOP. See High Volume Order Processing (HVOP), 22-1

## I

---

ignore pricing, D-10  
implementation  
    attribute mapping, 2-5  
    currencies, 2-10  
    customer class and profile classes, 2-8  
    customers and customer sites, 2-9  
    events and phases, 2-11  
    freight cost types, 2-10  
    freight terms, 2-10  
    line types, 2-10  
    order types, 2-9  
    payment terms, 2-10  
    pricing lookups, 2-7  
    pricing security, 2-11  
    setup flow, 2-1  
    setup steps  
        item relationships, 2-7  
        unit of measure, 2-5  
implementation planning, 10-4  
implementing from fresh install, 1-41  
incompatibility groups, 16-7  
incompatibility resolution  
    examples, 16-8  
    modifiers, 16-3  
    price list, 16-3  
integrating with Oracle Advanced Pricing, 21-1  
    Oracle Service Contracts (OKS) Integration, 21-30  
    overview, 21-1  
    steps, 21-1  
integration and attributes mapping messages and errors, 24-11  
integration flow for proration, 21-36  
Intercompany Transaction PTE Attributes , B-4  
inventory organization setup, 2-6  
item categories, 2-6, 3-12  
item categories, hierarchical, 8-2  
item category sets, 2-6, 3-12  
item information, defining, 2-7

item relationships, creating, 2-7  
item upgrade, 10-17

## L

---

Line Charges, 19-3  
Line Charges - Manual, 19-3  
line level code, 23-11  
line type, defining, 2-10  
List Line Adjustment, 19-3  
List Line Base Price, 19-3  
Logistics PTE Attributes, B-7  
lookups, G-1  
    for Order Management, 2-7  
    Shipping, 2-8

## M

---

maintaining price lists, 1-37  
manual modifiers, 10-14  
mapping of seeded request types and source systems, 17-38  
messages  
    price lists, 24-4  
messages and errors  
    freight charges integration, 24-12  
    modifiers, 24-9  
MOAC  
    *See* multi-org access control  
Modifier Incompatibility Setup window, 16-7  
modifier level code, 23-10  
modifier lists  
    selecting based on qualifiers, 23-7  
modifier messages and errors, 24-9  
modifiers, 1-39  
    accruals, 10-21  
    accumulated range breaks, 10-20  
    application methods, 10-6  
    ask for promotions, 10-20  
    blind, D-9  
    considerations, 10-10  
    copy modifiers, 10-20  
    coupon issue, 10-20  
    cross order volume-based, 21-21  
    excluding item categories, 10-11  
    incompatibility level, 10-11  
    incompatibility resolution, 16-3  
    incompatibility setup, 16-7

level code, 23-10  
levels, 10-6  
manual, 10-14  
matched qualifiers, 16-2  
messages and errors, 24-9  
overview, 10-1  
precedence, 10-11  
price breaks, 10-18  
pricing controls, 10-13  
pricing phase, 10-11  
recurring, 10-21  
type setup, 10-14  
modifiers (HTML UI)  
    types, 10-3, 10-3  
Modifiers for BOOK Event, 19-3  
modifier type setup  
    get region and benefit items, 10-15  
    item upgrade, 10-17  
    other item discount, 10-15  
    point price break, 10-20  
    promotional goods, 10-15  
    range price break, 10-20  
    term substitution, 10-17  
multicurrency conversion lists  
    implementation decisions, 13-9  
multicurrency price lists, 13-1, 13-3  
    fresh install, 13-7  
    upgrading, 13-8  
multi-currency scenarios, 24-21  
multi-org access control (MOAC), impact on pricing, 6-4  
multiple organizations, 15-1  
    cross order volume loader, 15-1  
    item upgrade, 15-3  
    organization specific seeded pricing attributes, 15-1  
    organization specific seeded qualifiers, 15-1  
    price list LOV, 15-1  
    using qualifiers to create, 15-1

## N

---

navigation paths and windows, A-1

## O

---

optimal performance, D-1  
    blind modifiers, D-9

- correcting low selectivity, D-8
- data distributions analysis script, D-11
- qualifier selectivity, D-4, D-5
- redundant qualifiers, D-9
- setup considerations, D-4
- technical improvements, D-11
- Oracle 8i temporary table locking, 24-27
- Oracle Pricing Administrator responsibility, in pricing security, 6-1
- Oracle Service Contracts
  - duration and partial period pricing, 21-38
  - partial period pricing, 21-36
- Oracle Telecommunications Service Ordering (TSO), 21-42
- Order Fulfillment PTE Attributes, B-14
- order level code, 23-11
- Order Management lookups, defining, 2-7
- order type, defining, 2-9
- organizations. See multiple organizations, 15-1
- other item discount, 10-15
- overview
  - attribute management, 17-1
  - bulk loader, 7-1
  - Oracle Advanced Pricing, 1-1

## **P**

---

- partial period pricing, 21-36
- pattern search, D-10
- payment terms, defining, 2-10
- performance, improving, D-11
- phases. See pricing phases, 19-3
- point price break, 10-20
- precedence, 16-1
  - attributes, 16-2
  - default numbers, 16-2
  - matched qualifiers, 16-2
  - modifier incompatibility resolution, 16-3
  - modifier incompatibility setup, 16-7
  - price list, 16-3
- price book
  - implementation steps, 9-4
  - message maps
    - GET\_CATALOG\_002, 9-6
    - SYNC\_CATALOG\_003, 9-6
  - overview, 9-1
  - setting up

- default printer, 9-5
- E-mail server, 9-5
- Oracle XML Gateway, 9-4
- Oracle XML Gateway Message Maps, 9-6
- Oracle XML Publisher, 9-4, 9-5
- price book UI, 9-8
- pricing parameters, 9-8
- profile options, 9-4
  - XML Gateway Trading Partners, 9-6
- price breaks, 10-18
  - accumulated range breaks, 10-23
  - calculating, 23-22
  - point, 10-20
  - range, 10-20
  - usage proration, 8-4
- price list locking, integration flow, 21-32
- price list locking and proration, 21-30
- price list locking feature, 21-35
- price list maintenance feature, 1-36
- price lists, 1-35
  - currencies
    - multiple, 3-10
    - single, 3-10
  - importing using bulk loader, 8-4
  - matched qualifiers, 16-2
  - messages and errors, 24-4
  - overview, 8-1
  - price list maintenance feature, 1-11, 1-36
  - resolving incompatibility, 16-3
  - selecting based on qualifiers, 23-7
  - single versus multiple currency, 3-10
  - using in the HTML UI, 8-1
- price lists (HTML UI)
  - introduction, 8-1
- price rounding limits, 5-42
- pricing
  - actions, 3-2
  - advanced versus basic, 1-2
  - attributes, 3-12
  - buckets, 3-17
  - controls, 3-2
  - extensibility, 3-2
  - methodology, 3-5
- pricing attributes. See attributes, 1-36
- pricing attributes and flexfields, 17-3
- pricing components
  - service items, 21-15

- pricing contexts. See contexts, 2-4
- pricing data bulk loader
  - See bulk loader
- Pricing Data Bulk Loader API, 7-1
- pricing engine, 1-33
  - calculation entity, 23-21
  - diagnostics and troubleshooting, 24-1
  - messages and diagnosis, 24-4
  - search engine, 23-3
- pricing engine interaction details, 21-7
- Pricing Engine Request Line Details region, 20-13
- Pricing Engine Request Lines region, 20-8
- Pricing Engine Requests region, 20-4
- Pricing Engine Request Viewer
  - user profile options, 20-2
    - QP\_Set Request Name, 20-2
  - window regions explained, 20-3
- Pricing Engine Request Viewer window, 20-1, 20-2
- pricing events, 19-1
  - creating, 2-11
- pricing formula problems, 24-17
- pricing lookups, defining, 2-7
- pricing organizer problem, 24-17
- pricing parameters
  - overview, 5-43
  - seeded, 5-46
  - viewing and updating, 5-45
  - viewing parameter definitions, 5-44
- pricing phases, 19-3
  - assigning, 19-4
  - creating, 2-11
  - seeded, 19-3
- pricing request, 1-33
- pricing request lines, freezing, 23-11
- pricing rules and actions, 3-1
- pricing scenarios, E-5
- pricing security
  - bulk create privileges, 6-14
  - bulk update entity usage, 6-11
  - changes to pricing windows, 6-7
  - entity sets
    - creating, 6-17
    - deleting, 6-19
  - entity usage, 6-11
    - bulk update, 6-11
    - express create privileges, 6-15
  - global usage and Global box, 6-11
  - Oracle Pricing Administrator, 6-1
  - overview, 6-1
  - privileges
    - bulk create, 6-14
    - creating, 6-14
    - express create, 6-15
    - precedence for multiple access
    - privileges, 6-14
    - setup suggestions, 6-13
  - profile options, 6-19
    - compared, 6-20
    - QP\_Security Default Maintain Privilege, 6-19
    - QP\_Security Default ViewOnly Privilege, 6-19
    - turning security on with QP\_Security Control, 6-26
  - setting up, 2-11
  - turning security on, 6-26
- pricing security terms, 6-3
- pricing solutions
  - developing, 3-9
  - testing, 3-18
- pricing transaction entity (PTE)
  - defining, 17-25
  - seeded values, 17-25
  - source systems, 17-25
- pricing transactions, phasing, 23-5
- privileges. See pricing security, 6-14
- process flow for implementation, 1-41
- Procurement PTE Attributes, B-30
- product hierarchy, 1-34
- profile class, defining, 2-8
- profile options
  - overview, 5-1
  - pricing security setup, 6-19
  - QP\_Security Control, 6-26
  - QP: Item Validation Organization, 15-3
  - setup, 5-3
  - setup summary, 5-3
- profile options setup summary, 5-3
- promotional goods, 10-15
- promotional limits, 21-22
- promotional limits
  - troubleshooting, 24-15
- proration, usage, 8-4

proration and price list locking, 21-30  
proration feature, 21-38  
Purge Pricing Entities, 11-4  
purging pricing data, 11-1

## Q

---

QP\_ARCHIVE\_ENTITY\_PUB, 11-2  
QP\_Batch Size for Bulk Import, 7-1  
QP\_Bulk Import of Price List program, 8-4  
QP\_High Volume Order Processing Compliance, 22-1  
QP\_INTERFACE\_LIST\_HEADERS (bulk loader), 7-2  
QP\_INTERFACE\_LIST\_LINES (bulk loader), 7-2  
QP\_INTERFACE\_PRICING\_ATTRIBS (bulk loader), 7-2  
QP\_INTERFACE\_QUALIFIERS (bulk loader), 7-2  
QP\_PURGE\_ENTITY\_PUB API, 11-4  
QP\_RUNTIME\_SOURCE, 10-23  
QP\_Security Default ViewOnly Privilege profile option, 6-19  
QP\_Source System Code, 7-1  
QP: Enterprise Command Center Data Load, 4-4  
qualifier groups, 1-35  
qualifiers, 1-34

- attributes, 2-4
- creating contexts and attributes, 2-4
- groups, 1-35

## R

---

range price break, 10-20  
recurring charges, 21-42  
recurring modifiers, 10-21  
Related Lines window, 20-27  
reports and concurrent programs

- QP\_Bulk Import of Price List program, 7-12

Restore Defaults button, 17-31  
rounding errors, 5-42  
rounding options, 5-38

## S

---

sample code using Order Management Structure, 21-26  
scenarios in the high-tech industry, E-1

search engine

- determining eligibility, 23-3

security. See pricing security, 6-1  
seeded cost to charge conversion formulas , C-2  
seeded data

- restoring, 17-31

seeded formulas, C-1  
seeded markup formulas, C-5  
seeded pricing phase/Order Management event relationships, 3-17  
service item pricing, 21-15  
service items, calculating duration, 21-36  
service items, pricing, 21-15  
setting up auditing, 12-1  
setup considerations, D-4  
setup steps for implementing pricing security, 6-5  
Shipping lookups, defining, 2-8  
source systems, creating, 17-25  
SYNC\_CATALOG\_003 (price books), 9-6  
system administration steps, 2-2

## T

---

TCA (Trading Community Architecture)  
attributes, 3-11  
technical considerations, 23-1, 24-26

- Advanced Pricing Engine Processing, 23-3
- architectural overview, 23-2
- Basic versus Oracle Advanced Pricing, 23-1
- engine processing, 23-3
- search engine, 23-3

term substitution, 10-17  
Trading Community Architecture (TCA)  
attributes, 3-11  
troubleshooting

- attribute management, 24-27
- during pricing setup, 17-34
- entering orders, 17-34
- in Pricing Setup windows, 24-13
- integration or runtime, 17-34
- setting up attributes , 17-33

troubleshooting

- promotional limits, 24-15

troubleshooting and diagnostics, 24-1

- summary of pricing engine messages and diagnosis, 24-4

## **U**

---

- unit of measure
  - conversion logic, 23-10
  - conversions, 14-2
  - defining, 2-5, 14-1
  - implementation decisions, 14-1
  - pricing, 14-2
  - primary, 14-2
  - profile options, 14-3
  - required for setup, 3-18
- upgrade considerations, 17-36
- upgrading context and attributes, 17-38
- usage price break proration, 8-4
- usage proration, 8-4
- Used In Setup box, 17-17
- using custom sourced attributes, 17-30
- using multiple currency price list with other Oracle Products, 13-13

## **W**

---

- windows, navigating, A-1
- windows and navigator paths, A-1

## **X**

---

- XML Gateway Trading Partners for price books, 9-6