Oracle® Advanced Pricing

Implementation Manual

Release 11*i*

Part No. B14385-03

July 2005

Oracle Advanced Pricing Implementation Manual, Release 11*i*

Part No. B14385-03

Primary Author: John Salvini

Contributing Author: Vivian Lee, Maria Viglionese Matheny

Contributor: Rajendra Badadare, Hariram Balasubramanian, Vishwajit Bhave, Rajeshwari Chellam, Amy Cui, Spencer Firestone, T. Geresh, Sripriya Gopal, Vivek Gulati, Dharmender Gupta, Nitin Hase, Leslie Hershey, Natarajan Kannan, Nirmal Kumar, Justin Kuo, Jeff Lee, Justin Lin, Shu-Hui Lin, Elizabeth Looney, Steven Mosura, Tom Myers, Rajmohan Nair, Vinod Narayanan, Sameer Phatarpekar, Sweta Rashmi, David Reitan, Sumeet Rijhsinghani, Nisha Samaga, Renganathan Sreenivasan, Ravi Tata, Boon Tea, Giridhar Tippireddy, Kiran Umakanth, Hock-Shan Wong, Linda Xu, Li Yang

Contents

Send Us Your Comments

Oracle Advanced Pricing Implementation Manual, Release 11*i*

Part No. B14385-03

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

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

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

- Electronic mail: appsdoc_us@oracle.com
- FAX: 650-506-7200 Attn: Oracle Advanced Pricing Documentation Manager
- Postal service:
  Oracle Advanced Pricing Documentation Manager
  Oracle Corporation
  500 Oracle Parkway
  Redwood Shores, CA 94065
  USA

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

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

Preface

Intended Audience

Welcome to Release 11*i* of the *Oracle Advanced Pricing Implementation Manual*.

This manual assumes you have a working knowledge of the following: The principles and customary practices of your business area. Oracle Advanced Pricing. 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. Oracle Self Service Web Applications. To learn more about Oracle Self Service Web Applications, read the Oracle Self-Service Web Applications Implementation Manual. The Oracle Applications graphical user interface To learn more about the Oracle Applications graphical user interface, read the Oracle Applications User's Guide.

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

How to Use this Manual

The *Oracle Advanced Pricing Implementation Manual* contains the information you need to understand and use Oracle Advanced Pricing. This manual contains the following chapters:

Chapter 1	Provides an introduction to Oracle Advanced Pricing implementation.
Chapter 2	Provides an overview of Oracle Advanced Pricing features and implementation.
Chapter 3	Discusses implementation methodology for Oracle Advanced Pricing.
Chapter 4	Describes the profile options and settings for Oracle Advanced Pricing.
Chapter 5	Discusses the implementation of pricing security.
Chapter 6	Describes the pricing data bulk loader feature for uploading pricing data to the pricing tables.
Chapter 7	Discusses the implementation of price lists.
Chapter 8	Discusses the implementation of modifiers.
Chapter 9	Describes archiving and purging pricing entities.
Chapter 10	Discusses the implementation of Multi-Currency price lists and Agreements.
Chapter 11	Discusses the implementation of units of measure (UOM).
Chapter 12	Discusses Oracle Advanced Pricing for multiple organizations.
Chapter 13	Discusses implementation considerations for precedence and best price.
Chapter 14	Discusses attribute management and attribute mapping.
Chapter 15	Discusses the Get Custom Price functionality.
Chapter 16	Explains implementation considerations for events and phases.
Chapter 17	Discusses the functions related to the Pricing Engine Request Viewer window.
Chapter 18	Provides information about integrating with Oracle Advanced Pricing.
Chapter 19	Describes the implementation considerations in pricing for High Volume Order Processing (HVOP).
Chapter 20	Discusses technical considerations for implementing Oracle Advanced Pricing.
Appendix A	Provides a listing of Oracle Advanced Pricing navigation paths.
Appendix B	Lists the seeded (predefined) pricing attributes, product attributes, qualifier attributes, attribute contexts and default pricing attribute mapping rules for Oracle Advanced Pricing.
Appendix C	Describes optimizing performance for Oracle Advanced Pricing.
Appendix D	Provides a case study that examines Oracle Advanced Pricing implementation in a fictional high-tech company.
Appendix E	Provides a case study that examines Oracle Advanced Pricing implementation in a fictional food service company.
Appendix F	Provides a listing of Lookups used in Oracle Advanced Pricing.

Other Information Sources

You can choose from many sources of information, including documentation, training, and support services, to increase your knowledge and understanding of Oracle Advanced Pricing.

When this manual refers to other Oracle Applications documentation, use only the Release 11i versions of those guides.

Online Documentation

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

- PDF Documentation: See the Online Documentation CD for current PDF documentation for your product with each release. This Documentation CD is also available on Oracle*MetaLink* and is updated frequently.
- Online Help: You can refer to Oracle Applications Help for current HTML online help for your product. Oracle provides patchable online help, which you can apply to your system for updated implementation and end user documentation. No system downtime is required to apply online help.
- Release Content Document: See the Release Content Document for descriptions of new features available by release. The Release Content Document is available on Oracle*MetaLink*.
- About document: Refer to the About document for information about your release, including feature updates, installation information, and new documentation or documentation patches that you can download. The About document is available on Oracle*MetaLink*.

See Related Documents on page xiv for more Oracle Applications product information.

TTY Access to Oracle Support Services

Oracle provides dedicated Text Telephone (TTY) access to Oracle Support Services within the United States of America 24 hours a day, seven days a week. For TTY support, call 800.446.2398.

Documentation Accessibility

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

Accessibility of Code Examples in Documentation

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

Accessibility of Links to External Web Sites in Documentation

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

Structure

Related Documents

Oracle Advanced Pricing shares business and setup information with other Oracle Applications products. Therefore, you may want to refer to other guides when you set up and use Oracle Advanced Pricing. You can read the guides online by choosing Library from the expandable menu on your HTML help window, by reading from the Oracle Applications Document Library CD included in your media pack, or by using a Web browser with a URL that your system administrator provides. If you require printed guides, you can purchase them from the Oracle Store at http://oraclestore.oracle.com.

Guides Related to All Products

Oracle Applications User's Guide

Use this guide to learn how to enter data, query, run reports, and navigate using the graphical user interface (GUI). This guide also includes information on setting user profiles, as well as running and reviewing reports and concurrent processes. You can access this user's guide online by choosing "Getting Started with Oracle Applications" from any Oracle Applications help file.

Oracle Applications Developer's Guide

Use this user guide to learn about the coding standards followed by the Oracle Applications development staff. It describes the Oracle Application Object Library components needed to implement the Oracle Applications user interface described in the Oracle Applications User Interface Standards. It also provides information to help you build your custom Oracle Developer forms so that they integrate with Oracle Applications.

Oracle Applications User Interface Standards

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

Guides Related to This Product

Oracle Advanced Pricing User's Guide

Use this user guide to learn how to set up and use Oracle Advanced Pricing features such as price lists, modifiers, qualifiers, formulas, multi-currency lists, agreements, archive and purge tools, reports, and concurrent programs.

Installation and System Administration

Oracle Applications Concepts

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

Installing Oracle Applications

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

Upgrading Oracle Applications

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

"About" Document

For information about implementation and user documentation, instructions for applying patches, new and changed setup steps, and descriptions of software updates, refer to the "About" document for your product. "About" documents are available on OracleMetaLink for most products starting with Release 11.5.8.

Maintaining Oracle Applications

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

Oracle Applications System Administrator's Guide

The guide provides planning and reference information for the Oracle Applications system administrator. The guide contains information on how to define security, customize menus and online help, and manage concurrent processing.

Oracle Alert User's Guide

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

Oracle Applications Developer's Guide

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

Oracle Applications User Interface Standards for Forms-Based Products

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

Other Implementation Documentation

Oracle Applications Product Update Notes

Use this guide as a reference for upgrading an installation of Oracle Applications. It provides a history of the changes to individual Oracle Applications products between Release 11.0 and Release 11i. It includes new features, enhancements, and changes made to database objects, profile options, and seed data for this interval.

Oracle Workflow Administrator's Guide

This guide explains how to complete the setup steps necessary for any Oracle Applications product that includes workflow-enabled processes, as well as how to monitor the progress of runtime workflow processes.

Oracle Workflow Developer's Guide

This guide explains how to define new workflow business processes and customize existing Oracle Applications-embedded workflow processes. It also describes how to define and customize business events and event subscriptions.

Oracle Workflow User's Guide

This guide explains how Oracle Applications users can view and respond to workflow notifications and monitor the progress of their workflow processes.

Oracle Workflow API Reference

This guide describes the APIs provided for developers and administrators to access Oracle Workflow.

Oracle Applications Flexfields Guide

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

Oracle eTechnical Reference Manuals

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

Oracle Applications Message Manual

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

Training and Support

Training

Oracle offers a complete set of training courses to help users and their staff master Oracle Advanced Pricing and reach full productivity quickly. These courses are organized into functional learning paths, so users take only those courses appropriate to their jobs or areas of responsibility. You have a choice of educational environments. You can attend courses offered by Oracle University at any one of our many education centers, you can arrange for our trainers to teach at your facility, or you can use Oracle Learning Network (OLN), Oracle University's online education utility. In addition, Oracle training professionals can tailor standard courses or develop custom courses to meet your needs. For example, you may want to use your organization structure, terminology, and data as examples in a customized training session delivered at your own facility.

Support

From on-site support to central support, our team of experienced professionals provides the help and information needed to keep Oracle Advanced Pricing working for all users. This team includes the technical representative, account manager, and Oracle's large staff of consultants and support specialists with expertise in your business area, managing an Oracle server, and your hardware and software environment.

Do Not Use Database Tools to Modify Oracle Applications Data

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

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

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

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

1 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 sophisticated 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 Management Open Interfaces, API, & Electronic Messaging Guide.*

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 depicts the primary differences between Oracle Advanced Pricing and basic pricing capability included in Oracle Order Management:

Pricing Features	**Basic Pricing in Oracle Order Management**	**Oracle Advanced Pricing**
Archive / Purge Pricing Entities	Basic Pricing supports removal of price list and modifier list lines data no longer required for current operations.	Same capability as Basic Pricing.
Pricing Security	The Pricing Security Administrator responsibility can grant access privileges across Operating Units to the following pricing entity types: • Standard Price List • Modifier • Agreement Price List	Advanced Pricing includes the Basic Pricing Security features plus the Pricing Entity type: Entity Set Using the Entity Set Page, the Pricing Administrator can create 'sets' of pricing objects that can be used as a grant object and grant access roles to any grantee type and grantee.
Pricing Security (Access levels)	The following access levels are supported by site level profile settings: • View Only • Maintain	Access levels and Privileges same as in Basic Pricing.
Pricing Security (Privileges)	You can grant access privileges to the following Grantee Types: • Global • Operating Unit • Responsibility • User Pricing Security supports authorization roles for creating maintaining, and viewing pricing data. Security Privileges are set up and managed in the HTML user interface.	Access levels and Privileges same as in Basic Pricing.

Pricing Features	Basic Pricing in Oracle Order Management	Oracle Advanced Pricing
Price list	Enables you to set up: • Price List Header/Lines • One currency per list • Effective date at list • Rounding factors • Attach payment terms • Attach freight terms· • Attach freight carriers • Percentage price for service items • Service price • Active Flag • Mobile Download • Global Flag to support Security feature • Security enabled rules checking	Advanced Pricing includes all Basic price list features and adds the capability to define point and range price breaks on a price list.
Pricing attributes on Price List	In Basic Pricing, you can use one data source, called a context, per order line. Each context can have 100 attributes.	Advanced Pricing adds 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.
Secondary price lists	One secondary price list is supported.	Advanced Pricing adds capability to define and use unlimited secondary lists. The chaining of secondary lists is NOT supported. A profile option enables a qualifier check for secondary price lists.

Pricing Features	Basic Pricing in Oracle Order Management	Oracle Advanced Pricing
Qualifiers: Price Lists	Qualifiers are supported via defaulting in Basic Pricing. You cannot define qualifiers other than the default values. Seeded defaults include customer, order type and others from Order Management: • Uses current method of customer assigned in RA_Customers where price list is assigned. • Precedence is populated in system and can be updated. • Includes defaulting of TCA (Trading Community Architecture) attributes.	Includes Basic features and adds the following features for the full qualifier feature: • Ability to attach qualifier groups or enter qualifiers for the price list. • All seeded context values for Advanced Pricing as well as user-defined qualifiers (requires Attribute Mapping) are supported. • Precedence is derived from Precedence Number field on the Contexts Setup window and can be configured by the user in Advanced Pricing. • Multiple with and/ or relations between qualifiers is possible. • Qualifiers enable you to define unlimited price lists for an item. • Note: You cannot define Order Amount as a qualifier.
Product hierarchy price list notes	Basic Pricing supports Product Context for Item number, Item Category, and All_Items.	Advanced Pricing includes the product contexts in Basic Pricing, and adds capability to define additional contexts or attributes using the Attribute Mapping feature. Note: You can define additional attributes for the product context, but you cannot create additional product contexts.

Pricing Features	Basic Pricing in Oracle Order Management	Oracle Advanced Pricing
Price breaks on price lists	Not supported in basic pricing.	Advanced Pricing enables price breaks on price lists. The following price break quantity 'drivers' are supported: • Quantity • Amount (excluding Item Amount) • Weight • Other user defined attributes The following break types are supported for Application Method of Unit Price: • Point break • Range break
Price breaks on price lists: Block Pricing	Not supported in Basic Pricing.	Block Pricing (Application Method of Block Price): • Define a price for the entire set of a block. Set up 'lump sum' or flat rate pricing for minimum quantities. • Point break • Range Breaks: Recurring, Value • Set up various price break combinations using Point and Range break types with multiple application methods for the set of the Block. Usage Modification--Prorated Range Breaks for usage calls: • Profile enabled proration of price break ranges from the Break UOM to the Usage UOM whenever they are passed. • One set up supports whatever UOM is used in the pricing call. • Can be used with existing setups (Point Breaks and Range Breaks)

Pricing Features	Basic Pricing in Oracle Order Management	Oracle Advanced Pricing
Multi-Currency Conversion Lists	Basic Pricing does not include multi-currency conversion lists.	Advanced Pricing allows users to maintain prices in a single list with one base currency and define multiple currency conversion rates and currency specific markup/ markdown equations. A Multi-Currency Conversion window enables users to define Currency To 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 define markup criteria per currency definition. • A concurrent program Update Price Lists with Multi-Currency Conversion Criteria is provided.
Formulas	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, prior to that price being available for engine calls. Basic Pricing gives you the capability to attach formulas to price lists. Formulas in Basic Pricing can include mathematical operators, numeric operands, and PL/SQL Functions such as min/max may be imbedded in a formula. Sixteen seeded formulas are included: • Eight Cost-to-Charge formulas • Eight Cost-to-Charge with Markup formulas These formulas can be updated.	Advanced Pricing adds the following features: • You can attach formulas to modifiers as well as price lists. However, static formula generation is available only for price lists in Advanced Pricing. • Dynamic Formula Expansion: Dynamic expansion enables the pricing engine to dynamically calculate the formula price at based on variable values available at run time. Dynamically expanded formulas can be attached to either price lists or modifiers. Note: Price lists that have a dynamic formula with a Modifier Value as a component cannot be attached to a price list line.

Pricing Features	Basic Pricing in Oracle Order Management	Oracle Advanced Pricing
Formula components	Component types include: • numeric values • one product context (item) • factor list • one pricing context throughout formula with up to 100 attributes. Context must match context on price list line that formula is being attached to. Adjustment factors (factor list): basic pricing supports multiple factors. However, there is limitation in basic pricing of one pricing attribute context throughout formula with up to 100 attributes. Seeded context must be used. Context must match context on price list line to which the formula is attached.	Advanced Pricing includes all formula component types available in Basic Pricing, and adds the following: • List price of an item in a specific price list (product and service) pricing attribute. • Modifier Value is entered in the Modifier Value field on the modifier setup window for a modifier line as a component to a formula. • A FUNCTION type which calls Advanced Pricing API Get_ Custom_Price. • Adjustment Factors: Advanced Pricing supports multiple adjustment factors. Both context - seeded and user-defined contexts and attributes can be used as adjustment factors. • Multiple pricing contexts for pricing attributes.

Pricing Features	Basic Pricing in Oracle Order Management	Oracle Advanced Pricing
Price List Maintenance	Not available in Basic Pricing.	Advanced Pricing provides search and maintenance capability for a single price list or across multiple price lists. Use the Bulk Change feature for mass updates or update individual price list lines. Updates can be made to price list lines for: • Value • Static and Dynamic Formula • Price List Line effective dates 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.
Bulk Import of Price List	• Imports large volume of price lists via the interface tables. • Provides increased performance vs. using the Business Object API.	Same capability as Basic Pricing.
Copy price list	Basic Pricing provides this capability. Security-enabled rules checking.	Same as Basic
Adjust price list	Basic Pricing provides this capability. Security enabled rules.	Same as Basic
Add items to price list	Supported. (User must have Maintain access privilege to Add Items.)	Same as Basic. (User must have Maintain access privilege to Add Items.)
GSA pricing	Supported	Same as Basic
Modifier list types	Basic Pricing supports the following Modifier List Types: • Discount • Surcharge • Freight and Special Charges Security-enabled rules checking.	Advanced Pricing provides all Modifier List Types provided in Basic Pricing, and adds the following: • Promotion • Deal

Pricing Features	Basic Pricing in Oracle Order Management	Oracle Advanced Pricing
Modifier application methods	Basic Pricing supports the following modifier application methods: • Manual • Automatic • Overrideable	All Oracle Order Management Basic methods and provides the Ask For promotion application method.
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.
Modifier header qualifiers and attributes	Basic Pricing provides seeded qualifier contexts including: • Customer • Price List • Unlimited price lists (new OM feature) • Seeded attributes within contexts. Customer context includes these attributes: Customer class (defined in RA customers); Site; and Customer Name. Note: Modifier List Type of Freight and Special Charges supports additional modifier header qualifiers and attributes. Security-enabled rules checking.	Includes Basic Pricing features, plus Advanced Pricing adds these capabilities: • User assignment and override of precedence is possible. • Qualifiers can be used with both seeded and user defined contexts possible. • Users can define attributes with user-defined contexts. Up to 100 attributes can be defined for each context, and there is no limit to the number of contexts that users can define. • Seeded qualifier attributes can be redirected to other data sources using attribute mapping feature (see attribute mapping topic). • Users can control qualifier precedence. • Entered Context: Seeded and user-defined is possible. • User control of attribute precedence possible. • Attaching qualifier groups to modifier headers possible.

Pricing Features	Basic Pricing in Oracle Order Management	Oracle Advanced Pricing
Modifiers: line types	Types available include: • Discount: Fixed amount, Percent, New Price • Surcharge: Fixed amount, Percent, New Price • Freight Charges: Fixed amount, Percent, Lumpsum • Price Break	Advanced Pricing includes the Basic Pricing types and adds:· • Coupon Issue • Item Upgrade • Other Item Discount • Terms Substitution • Promotional Goods
Modifiers: line qualifiers and attributes	Basic Pricing provides the following fixed qualifiers including: • Agreement Name • Agreement Type • Order type • Customer PO Usable Product Attributes include: • Item • Item categories • All items Usable pricing attributes:· • Limited to one context when product attribute is ALL Items Note: Modifier Line Type of Freight and Special Charges supports additional modifier line qualifiers and attributes.	Advanced Pricing includes the line level qualifiers provided in Basic Pricing and adds: • Define unlimited number of qualifiers. • Can attach qualifier groups as qualifiers. • Can use seeded contexts and define additional contexts. • User-defined contexts active. • Multiple qualifiers with AND/OR conditions can be created. Product attributes in Advanced Pricing include those defined in Basic Pricing and adds user-defined "alternative" product hierarchy contexts from outside Oracle inventory structure. An unlimited number of pricing attributes can be created in Advanced Pricing. The user can change the precedence.

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 supported for the following modifier types: • Discount • Surcharge • Price Break • Freight and Special Charges
Modifier: Optional Currency	Enables modifier to be used across all transaction currencies: • Optional Currency selected from LOV. • Percent, Amount, New Price, Lumpsum. • No currency conversion applied. • Modifier setup numeric value will always be applied in currency units of the transaction currencies. • No restrictions to use; therefore, user discretion advised for appropriate use.	Advanced Pricing provides the same capabilities available in Basic Pricing.

Pricing Features	Basic Pricing in Oracle Order Management	Oracle Advanced Pricing
Modifiers: Price Breaks	Line level price breaks in basic include: • Percent • Amount • Fixed Price • Equal operator • Break Type Code - Limited to point type only • Volume Type (Item Amount, Item Quantity)	Line level price breaks in Advanced Pricing includes all price break functionality in Basic Pricing, and adds the following: • Equal, between arithmetic operator • Point and Range • Context - Seeded and user defined • Recurring Define automatic price breaks based on Net Amount for line and group of line level. 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 • Attribute Mapping • Runtime Sourced

Pricing Features	Basic Pricing in Oracle Order Management	Oracle Advanced Pricing
Modifiers: Freight and Special Charges	Supported Header Qualifiers • Freight Terms • Order Amount • Line Weight • Order Weight • Line Volume • Order Volume Supported Line Qualifiers • Order Volume • Order Type • Order Category • Line Type • Line Category • Shipment Priority Code • Shipped Flag • Shippable Flag • Freight Cost Type	Advanced Pricing provides the same capabilities available in Basic Pricing.
Effectivity date controls	Effectivity Date: order date only	Order date, plus Advanced Pricing adds: • Ship Date • Order and Ship Date 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.	Advanced Pricing includes this feature. You can define promotional limits by:· • Gross Revenue • Usage • Cumulative Discount Amount • Item Quantity • Accrual Units Limits can be set: • Within current transaction • Across all transactions Can apply limits for: • Customer Hierarchy • Product Hierarchy Types of Limits: • Soft Limit (If limit is exceeded, give the full benefit) • Hard Limit (If limit is exceeded, adjust or deny the benefit) Hard Limit enforcement • Hold • No Hold Security enabled rules checking
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		Advanced Pricing allows the copying of modifier lines, including duplicate modifier lines within a modifier list.

Pricing Features	Basic Pricing in Oracle Order Management	Oracle Advanced Pricing
Pricing Organizer: Modifiers	Basic Pricing does not provide this feature.	Advanced Pricing includes this feature, which allows users to query on the setup of Modifiers. • Modifier List • Modifier Lines • Products (including Excluded Products) • Pricing Attributes • Qualifiers Query criteria can be saved in Personal or Public folders. Can open Modifiers from the Organizer Summary. Security-enabled rules checking.
Pricing Engine Request Viewer	Basic Pricing features include viewing pricing requests information such as: • Pricing Engine Requests • Pricing Engine Request Lines • List Price • Selling Price • Service and Serviceable items • Price List lines and Modifier lines evaluated and deleted by the pricing engine • Pricing Engine Request Line Details • Price List lines and Modifier lines evaluated and deleted by the pricing engine • Attributes sent to the pricing engine by the calling application • Attributes used in pricing by the pricing engine • Debug Log Accessible from the Tools menu of the Sales Order Pad. Security enabled rules checking	Same as Basic Pricing, plus can view the relationship between lines for modifier types promotional goods and other item discounts. Also accessible from the Pricing Manager Responsibility

Pricing Features	Basic Pricing in Oracle Order Management	Oracle Advanced Pricing
Agreements	Basic Pricing agreement features enable you to: • Set payment terms: Invoicing rule, Accounting rule • Set freight terms: Freight carrier • Create Standard and Agreement Price List • Define using customer part numbers. • Make revisions to original terms. • Enter Revision numbers, date, reasons - only at line level. • Set Effective dates. • Create Volume breaks. Security enabled rules checking.	Same as Basic Pricing.
HTML Pricing Setups - Home Page	Basic Pricing does not provide this feature.	With the Oracle Pricing User responsibility, Advanced Pricing users can perform the following tasks in Advanced Pricing HTML Pricing Setup from the Home Page: • Search for price lists and modifier lists • Shortcut links to create price lists, modifier lists, and price list maintenance • View Recently Created Price Lists and Modifier Lists

Pricing Features	Basic Pricing in Oracle Order Management	Oracle Advanced Pricing
HTML Pricing Setups - Price List	Basic Pricing does not provide this feature.	With the Oracle Pricing User responsibility, Advanced Pricing users can perform the following tasks in Advanced Pricing HTML Pricing Setup: • Create a Price List • Create a Price List line with Pricing Attributes • Create a Price Break line with Pricing Attributes • Update Price List and Price List Lines • Delete a price list line • Access Price List Maintenance feature • BSA related fields are not available in the HTML user interface
HTML Pricing Setups - Modifiers	Basic Pricing does not provide this feature.	With the Oracle Pricing User responsibility, Advanced Pricing users can perform the following tasks in Advanced Pricing HTML Pricing Setup for Modifiers: • Create a modifier list (Discount, Surcharge, Deal, or Promotion List) • Create a modifier line for discount, surcharge, price break, or promotional goods (additional buy products not supported) • Update modifier list and modifier lines for types supported • Excluded Products
Attribute Mapping (Extensibility Feature)	Basic Pricing does not provide this capability.	Advanced Pricing adds this feature. You can use Attribute mapping to easily extend Pricing to tap into data from a wide variety of non-standard sources to drive your pricing. These data sources can be within Oracle Applications or from outside Oracle Applications

Pricing Features	Basic Pricing in Oracle Order Management	Oracle Advanced Pricing
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.
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	Order Management integration with Basic Pricing supports the View Adjustments feature and the following features: • Manual price override • Manual discount override • Reason Code • Modifier Dates • Display Qualifier Attributes and display Pricing Attributes buttons on UI • One pricing attribute context • Up to 100 attributes can be used in the single context • Action >Price Order is available. • Action >Price Line is available. • Calculate Price Freeze Flag can be set. • Blanket Order Support: Create a simple price list from the blanket window. 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.	Order Management, when integrated with Advanced Pricing, provides all features supported with Basic Pricing and adds: • View Adjustment: Relationship button, Item Upgrade, Term Substitution • Coupon entry • Ask for promotions • User entered attributes: Multiple attribute contexts can be used • Up to 100 attributes per context • Promotional Limits Hold: P lace holds where violated; No holds are placed; Place order on hold when any violation occurs.
Oracle Order Management integration with Pricing: Blanket Sales Agreement	Create simple in-line price list and modifier list from within the blanket window. Other tab displays additional information: • List Source Document Number (BSA number) and List Source Code automatically populated on Price List and Modifier List. • Automatic creation of BSA as qualifier for Price List and Modifier header and lines • Price List provides fields for: Customer Item, Address, Address Category • Customer Name and Number populated	Create price break range type modifiers based on accumulated BSA fulfillments (across releases). Volume accumulation attributes have been seeded for Blanket Sales Agreement use only. The following accumulation attributes can be selected when setting up the modifier: • Blanket Amount • Blanket Line Quantity • Blanket Line Amount

Pricing Features	Basic Pricing in Oracle Order Management	Oracle Advanced Pricing
Pricing Engine	In Basic pricing, the pricing engine does not return Oracle Advanced Pricing modifiers or features to the calling application. Security rules checking is enabled. Global box and security rules are used.	In Oracle Advanced Pricing, the pricing engine is enabled to return all advanced features.
Reports	Reports with Basic Pricing include: • Order Discount Detail Report • Order Discount Summary Report • Diagnostics List Line Details • Diagnostics Performance Analysis Security-enabled rules checking across all reports.	Advanced Pricing adds the following reports: • Accruals Details Report • Attribute Mapping Rules Error Report • Cross Order Volume Report • Modifier Detail Report • Price List Detail Report (including Multi-currency fields when Multi-currency is installed) • Pricing Formulas Report • Qualifier Grouping Report
Concurrent Programs	Not available in Basic Pricing.	Advanced Pricing adds the following concurrent programs: • Build Attribute Mapping Rules • Build Formula Package • Cross Order Volume Load • Purge Pricing Engine Requests • QP: Bulk Import of Price List • QP: Maintains the denormalized data in QP Qualifiers • Update Price Lists with Multi-Currency Conversion Criteria • Update Promotional Limit Balances

A table of differences between Oracle Order Management basic pricing functionality and the pricing capabilities of Release 10.7 and Release 11.0 of Oracle Applications is documented in the *Oracle Order Management Implementation Manual.*

Terminology

Calling Application

See invoking application.

Customer Hierarchy

When installed with Oracle Order Management, the customer hierarchy in Oracle Advanced Pricing is seeded to enable roll up of individual customers according to the following structure, which is based on RA_Customer:

- Customer name
- Customer class
- Site
- Ship to
- Bill to
- Agreement name
- Agreement type
- GSA
- Sales channel
- Account type

You can use elements of the customer hierarchy shown by referencing them as qualifiers for either modifier or price list objects.

Certain attributes from the Trading Community Architecture (TCA) customer tables are also seeded for use as qualifier attributes. These are:

- Party ID
- Customer account ID (sold_to_org_id)
- Ship to party site
- Invoice to party site

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

The previously mentioned hierarchy is built using the Oracle Customer Master and Trading Community Architecture. When Oracle Advanced Pricing is installed without Oracle Order Management or without the standard customer tables, the seeded customer qualifiers listed are not available. In that case, you must define an alternative table structure location where the customer hierarchy exists and list the attributes it contains. It is not necessary for the table structure supporting your alternative structure to exist within Oracle Applications.

Mapping an alternative customer hierarchy can be accomplished using attribute mapping. For more information on attribute mapping, see Overview of Attribute Management, page 14-1.

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.

Pricing Engine
The pricing engine is a program module of Oracle Advanced Pricing called by an calling application as transactions are processed that must be priced.

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 14-1.

Oracle Advanced Pricing Features

The following is a summarized list of Oracle Advanced Pricing features supported by Oracle Applications:

Qualifiers

Qualifiers determine who is eligible for a modifier. 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
- 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 selling price to a product. 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 on setting up and using price lists.

Oracle Advanced Pricing provides you with the capability to instruct the Pricing engine make the selection of the price list based on a qualifier rule.

As an alternative to using price list qualifiers, you may default a 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 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 cannot combine defaults using AND/OR relationships.

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.

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 across multiple price lists, or update individual price list lines.

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** seeded attributes, page B-1.

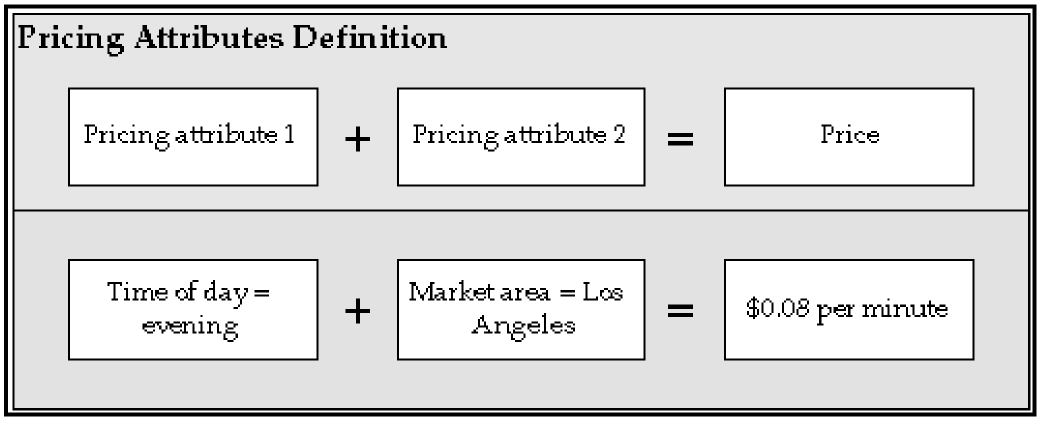
In some situations, an item identifier does not identify the product to a level where a price can be assigned. Pricing attributes enable you to define separate conditions for a product where the product must be priced under different conditions. A pricing attribute can be any condition you use to further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

The following are examples of pricing attributes used to specify conditions of a product or service that affects the price. While these examples are appropriate to price lists, the same concept holds 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 depicts 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.

Pricing Attributes Definition



Attribute Mapping
Oracle Advanced Pricing enables you to refer to data sources and attributes that are not seeded in the delivered product. This capability is called attribute mapping.

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 capabilities are invoked for a window within the Oracle Advanced Pricing Responsibility Menu, 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.
- Make revisions to 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 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.

When processing formulas, the pricing engine begins by locating a price list line which 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, that 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 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 overridable.

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, as an alternative to housing the price in a price list or in a formula. This gives you the capability to obtain a starting or beginning list price from sources that are outside Oracle Advanced Pricing's tables, yet are 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 flows for implementing from a fresh install and implementing from an upgrade both assume that Oracle Applications, including Oracle Order Management has been successfully installed, that Oracle Advanced Pricing has been installed Shared, and that all necessary patches have been applied.

Implementing from Fresh Install

The following table lists the recommended steps for implementation, assuming a fresh install (no prior implementation of Oracle Order Entry/Shipping exists). The recommended implementation steps differ when upgrading from a prior release.

#	Process Step	Step Description
1	Analyze and understand business pricing scenarios.	An high 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".
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.	Prior to 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 properly (the Vision Sample database shipped with the software can be used to for this).
5	Make defaulting decisions.	These decisions must be made prior to 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 1-4 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 involves creating a backup of system as setup.
8	Conduct pre-production system functionality and load tests.	Prior to initial production, a system test should be conducted. This test should exercise all product setups by including examples of all pricing scenarios defined during pre-implementation planning. Verify that testing includes volume to stress system to levels that are experienced during production operations.

Upgrading from Oracle Applications

When upgrading from either Release 10.7 or Release 11, the upgrade of pricing data to Release 11*i* initially establishes tables as basic pricing. The pricing upgrade occurs within the overall flow of upgrading from Order Entry shipping.

While the upgrade process establishes a working 11*i* system, because Oracle Advanced Pricing functionality is new, the upgrade process cannot create an 11*i* database that uses the capabilities of Oracle Advanced Pricing. The database created by the upgrade process has only data setups in it sufficient to drive basic pricing.

Rapid Implementation From an Upgrade

The following table depicts a process flow for a suggested rapid implementation process whereby planning and testing of Oracle Advanced Pricing implementation is performed prior to the conversion. The setup and activation of Oracle Advanced Pricing functionality proceeds rapidly after the upgrade is complete.

Pre- or Post-Conversion?	**Process Step**	**Step Description**
Pre-conversion	Analyze and understand business pricing scenarios.	Establish an exact understanding of pricing business requirements before beginning an Oracle Advanced Pricing implementation.
Pre-conversion	Determine data sources and columns needed for product and customer hierarchy definition.	See Implementation Methodology for more details. Oracle Advanced Pricing ships with seeded values. However, you may need to define additional levels or alternative data sources. For more information, see Chapter 3, " Implementation Methodology".
Pre-conversion	Develop logical pricing model solutions.	For each Pricing scenario, plan how you will use basic pricing. For more information, see Chapter 3, " Implementation Methodology".
Pre-conversion	Install 11*i* Oracle Advanced Pricing and calling applications in a test instance.	Software environment for testing should be available prior to pricing prototype.
Pre-conversion	Set up and test prototype pricing solutions	Prior to implementing a production system, set up prototype Oracle Advanced Pricing solutions for all identified pricing scenarios, and enter test orders against them to determine that they are handled properly. The Vision Sample database shipped with the software can be used to facilitate this process.
Pre-conversion	Plan pricing cut over policies.	Decide how to approach orders at the time Oracle Advanced Pricing setups are activated.
Conversion	Perform upgrade.	For more information, see *Oracle Order Management Implementation Manual*.
Conversion	Perform post upgrade steps.	For more information, see *Oracle Order Management Implementation Manual*.
Post-conversion	Begin product operation using basic pricing.	For more information, see *Oracle Order Management Implementation Manual*.

Pre- or Post-Conversion?	Process Step	Step Description
Post-conversion	Perform Oracle Advanced Pricing product setup tasks.	Examine the results of planning and decision making defined in earlier steps. Create entries in the Pricing system tables that enable Oracle Advanced Pricing to act on your plans. For detail task list, descriptions, sequencing, and instructions, see Chapter 2, "Implementation Overview". Create these setups initially outside the production system.
Post-conversion	Conduct pre-production system functionality and load tests.	Conduct a system test prior to initial production. This test should exercise all product setups by including examples of all Pricing scenarios defined during pre-implementation planning. Testing should stress system to levels that are experienced during production operations. Initially, these tests should be performed outside the production system.
Post-conversion	Cut over to Oracle Advanced Pricing setups.	Set up testing completely, setup data is added to production system. Active flags set.

2
Implementation Overview

This chapter covers the following topics:

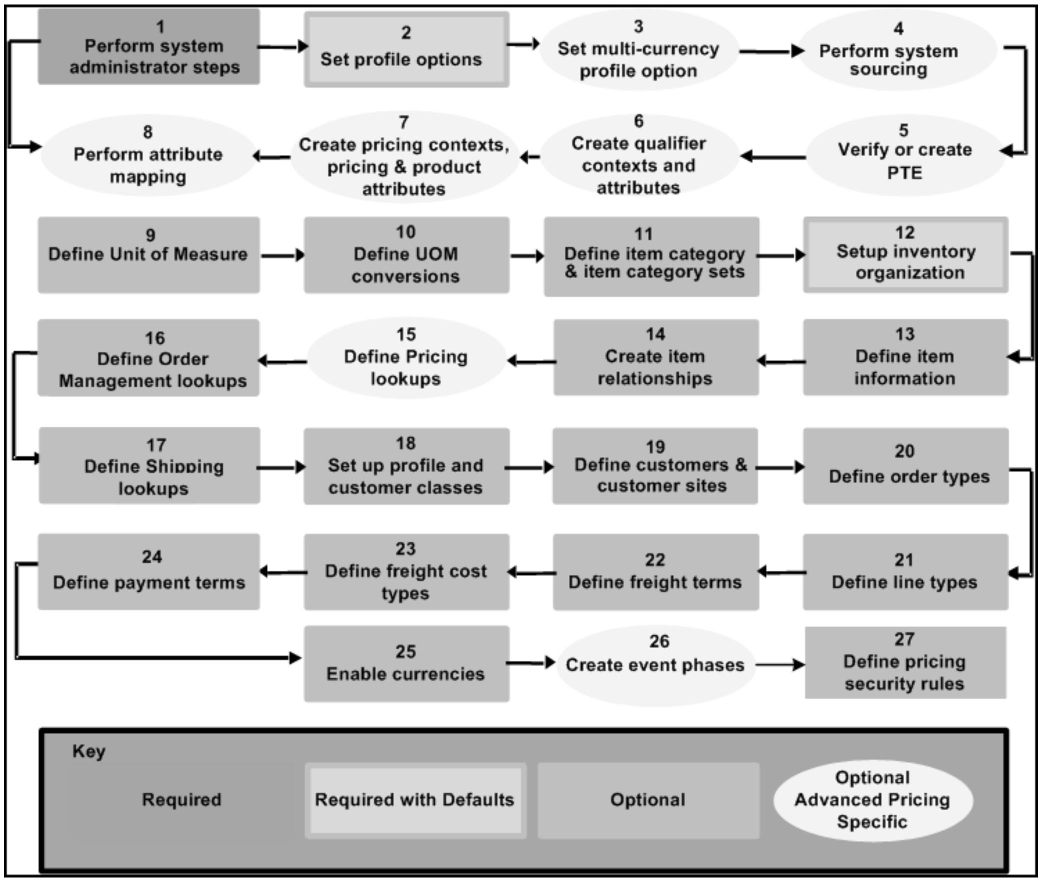
- Setup Flow

Setup Flow

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


1 Perform system administrator steps
2 Set profile options
3 Set multi-currency profile option
4 Perform system sourcing
5 Verify or create PTE
6 Create qualifier contexts and attributes
7 Create pricing contexts, pricing & product attributes
8 Perform attribute mapping
9 Define Unit of Measure
10 Define UOM conversions
11 Define item category & item category sets
12 Setup inventory organization
13 Define item information
14 Create item relationships
15 Define Pricing lookups
16 Define Order Management lookups
17 Define Shipping lookups
18 Set up profile and customer classes
19 Define customers & customer sites
20 Define order types
21 Define line types
22 Define freight terms
23 Define freight cost types
24 Define payment terms
25 Enable currencies
26 Create event phases
27 Define pricing security rules
Key
Required
Required with Defaults
Optional
Optional Advanced Pricing Specific

Setup Steps

1. **Perform System Administration Steps**

   Default: None

   Required

   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 will access the Oracle Pricing Manager responsibility.

2. **Set Profile Options**

   Required

   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 Manual,* Profile Options.

3. **Set Multi-Currency Profile Options**

   Default: None

   Optional

   The profile option QP: Multi-Currency Installed enables the multi-currency price list feature. Multi-Currency Price Lists enables you to maintain a single price list for multiple currencies. Once the profile option is set to Y, 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 this.

4. **Perform System Sourcing**

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

   Use the Pricing Transaction Entities Associations window to control which pricing data is used to price transactions.

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

5. **Verify or Create PTE**

   Default: Order Fulfillment

   Required

   A Pricing Transaction Entity (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. Additional Source System and Request Types can be added to existing PTE's.

   There are very rare instances where it is necessary to create a new 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 that is not already predefined.

6. **Create Qualifier Contexts and Qualifier Attributes**

   Default: Predefined Oracle Advanced Pricing qualifier contexts

   Optional

If you skip this step, users will be able to chose only predefined Oracle qualifier contexts and qualifier attributes for price and modifier eligibility.

Qualifiers provide a highly configurable and flexible method of defining the rules which your business uses to manage pricing. Qualifiers are used by the pricing engine to determine eligibility for price lists and modifiers.

7. **Create Pricing Contexts, Pricing Attributes, and Product Attributes**

   Default: Predefined Oracle Advanced pricing and product attribute contexts.

   Optional

   Pricing and product attributes are a feature of Oracle Advanced Pricing which enables you to define necessary item attributes in order to price or apply a modifier and attributes used in formulas.

   If you do not do this step, users will be able to select only predefined Oracle Advanced Pricing contexts and associated attribute values for benefit options.

8. **Perform Attribute Mapping**

   Default: Predefined Oracle Advanced Pricing attribute mapping rules

   Optional

   Do this step if you have defined any additional qualifiers or pricing attributes in steps 6 and 7, or if you wish 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 can be used 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.

9. **Define Unit of Measure**

   Default: None

   Optional

   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 Manual*, Unit of Measure.

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

10. **Define Unit of Measure Conversions**

    Default: None

    Optional

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, all price adjustments, benefits, and charges need to be defined 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.

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

11. **Define Item Categories**

    Default: Seeded structure name of item categories, category set inventory items, and associated default seeded category code combinations.

    Optional

    Item categories have been seeded as an item attribute in pricing attributes and can be used for 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. Oracle Advanced Pricing uses only one flexfield structure. You can setup multiple categories and multiple category sets.

    These categories and category sets should be defined for the defaulted standard flexfield structure. 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 Inventory User's Guide,* Defining Category Sets and Categories, and *Oracle Application Flexfields User's Guide,* Key Flexfields in Oracle Applications, Item Categories Flexfield.

    Do 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.

12. **Set Up Inventory Organization**

    Default: None

    Required with defaults

    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 in the Price List and Modifier Setup forms.

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

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

13. **Define Item Information**

    Default: None

Optional

Define the items that you wish to price and discount and assign them to the validation organizations defined in step 10. If you want to define qualifier rules which include the seeded qualifiers Line Volume or Line Weight you must set the volume or weight attributes of each item as this is 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.

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

14. **Create Item Relationships**

    Default: None

    Optional

    This step is required if you wish to give item upgrade benefits. You must define a Promotional Upgrade item relationship from the ordered item to the item you wish 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 need to have the same base unit of measure and unit of measure conversions.
    - The modifier unit of measure and the pricing unit of measure on the order line need to be the same.

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

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

15. **Define Pricing Lookups**

    Default: Lookup type dependent

    Optional

    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 it no longer displays in the list of values, or you can use Start and End dates to control when a value displays in a list. For a complete list of Lookup types, see: *Oracle Advanced Pricing Implementation Manual*, Lookups, page G-1.

    The following table depicts lookup types with descriptions:

16. **Define Oracle Order Management Lookups**

    Default: Lookup type dependent

    Optional

    For a list of valid default values for these lookups please refer to *Oracle Order Management User's Guide,* Lookups Appendix.

The following table depicts lookup types with their descriptions:

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

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

17. **Define Shipping Lookups**

    Default: Lookup type dependent

    Optional

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

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

18. **Set Up Customer Class**

    Default: None

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

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

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

    Customer Class and

19. **Set Up Profile Classes**

    Default: None

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

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

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

20. **Define Customers and Customer Sites**

    Default: None

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

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

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

21. **Define Customer Sites**

    Default: None

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

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

    Do 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, give benefits or charge by order type.

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

    Do 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, give benefits, or charge by order line type.

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

    Do 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, give benefits, including upgrading freight terms, or charge by freight terms.

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

    Do 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, give benefits or calculate charges using freight cost types.

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

Do 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, give benefits, or charge by payment terms.

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

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

27. **Enable Currencies**

    Default: All major currencies predefined with Oracle Applications.

    Optional

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

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

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

28. **Perform System Sourcing**

    Default: Predefined Oracle Advanced Pricing record

    > **Note:** 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 type is used to identify the type of transaction being priced. Whenever the pricing engine prices a request, the request must be stamped with the request type so the pricing engine can identify the type of transaction. The source system is recorded on all price and modifier lists and is used to identify which application created this pricing data.

29. **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. If your business manages pricing requires certain types of benefits at a particular point or event in the transaction process flow, do this step.

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

30. **Set up Pricing Security**

    Optional

    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 operating units.
    - 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.

3 Implementation Methodology

This chapter covers the following topics:

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

Overview

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

In Oracle Advanced Pricing (Release 11*i)*, the term "pricing" rules describes the rules-based logic the pricing engine uses to apply a pricing action to a transaction.

Pricing rules are built around the customer and product hierarchies. Pricing policies are often tied 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.

In Oracle Advanced Pricing, you set up pricing rules using the Qualifier window to describe customer and non-product oriented 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. If you wish to specify the same price list as usable only by the VIP customer class, specify 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, discounts that lower the list price down to a net selling price are 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 forms, or objects. These objects are:

- Price lists
- Agreements
- Formulas
- Modifiers

Modifiers are a class of pricing action that modifies the net price either up or down. Modifiers are sometimes called adjustments. Modifiers have several types. Each of these modifier types applies a specific pricing action with specific behavior characteristics.

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.

- **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.

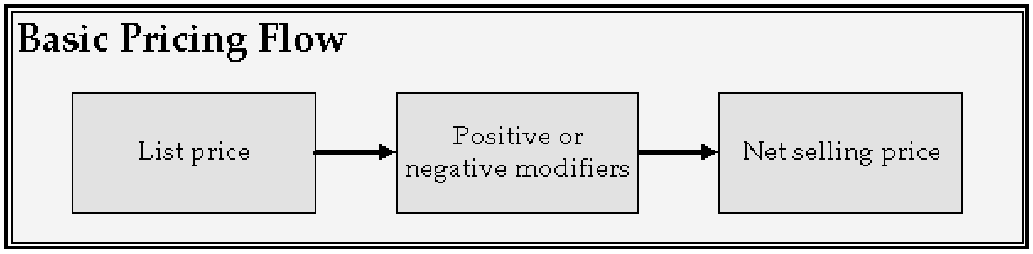
  For example, the Get_Custom_Price API can be linked to a price list using a formula with a term of type Function. You can use this API to make Oracle Advanced Pricing obtain a cost from a purchase order that is used in a calculation that yields a selling price based on actual cost.

  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 Oracle Metalink 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, 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 you can construct a pricing solution.

> **Caution:** 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
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:

Volume	Price Per Case
1-49	55
50 - 199	50
200 - 499	45
500 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
1-49	50
50 - 199	45
200 - 499	40
500 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 prices 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 Hierarchy

Customer hierarchies are 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 context values on Oracle customer tables sufficient to contain customer hierarchy?

When installed with Oracle Order Management, the customer hierarchy in Oracle Advanced Pricing is seeded to roll up individual customers according to the following structure, which is based on RA_Customer:

- Customer name
- Customer class
- Site
- Ship to
- Bill to
- Agreement name
- Agreement type
- GSA
- Sales channel
- Account type

If the seeded qualifier context values from the Oracle customer tables are adequate, then attribute mapping for customer hierarchy are not necessary.

If the seeded qualifier context values from the Oracle customer tables are not sufficient, there are two options for Release 11*i*. Both approaches requires you create attribute mapping to attribute mapping to inform pricing of the location of the customer hierarchy data elements you wish to reference in pricing qualifiers. You must create a default sourcing rule for each qualifier attribute that you define since the pricing engine uses sourcing rules to locate the attribute values. You can use expand the your hierarchies by:

- Use flex fields on customer table to hold flattened hierarchy.
- Use tables outside Oracle to house the customer hierarchy.

When defining your customer hierarchy, the lowest level in your hierarchy has the lowest flexfield segment sequence number. For example, the pricing engine must choose between a price for an item negotiated with a specific customer and a price for the same item given to all customers in a customer class. You want the engine to select the customer-specific price. Set the customer qualifier as more specific than the customer class qualifier, and the engine will select your desired price. This flattens your pricing hierarchy across the Qualifier descriptive flexfield. Each hierarchy may be defined within a context or across contexts depending on how many levels you have in your pricing hierarchies.

Trading Community Architecture (TCA) Attributes

Oracle Advanced Pricing enables you to use certain customer TCA attributes as qualifier attributes. Because modifiers can be set up based on TCA qualifier attributes, TCA qualifier attributes must be derived from Oracle Order Management attributes before calling Oracle Advanced Pricing. These attributes are seeded with the qualifier context as Customer context. The seeded qualifier attributes are:

- Party ID
- Customer account ID (same as sold_to_org_id)
- Ship to party site
- Invoice to party site

Party ID, ship to party site, and invoice to party site are visible to Oracle Order Management Users with Oracle Advanced Pricing even if they do not have Oracle Contracts. These attributes can be used as qualifiers to derive price lists and modifiers. Information about party ID, ship to party site, and invoice to party site are automatically derived by the sourcing rules. Therefore, customers do not see this information in any of the Oracle Order Management windows.

Product Hierarchy

Product hierarchies are single- or multi-level groups to which a product may be associated with. 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. Do this step after you specify your requirements to take best advantage of the flexibility of Oracle Advanced Pricing for managing pricing in your enterprise

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
- Item All
- Pricing attributes to specify a level more detailed than just item

When defining your product pricing hierarchy, the lowest level in your hierarchy has the lowest flexfield segment sequence number. If the pricing engine must choose between an item price and an item category price, set the item segment as more specific than the product group segment. This flattens the pricing hierarchy in the item context of your Pricing Attribute descriptive flexfield. Each level in your hierarchy at which you wish to manage pricing is represented as a segment.

Item categories and category sets

You must design and define any item categories and category sets for pricing and discounting. If you use the seeded context values on the Oracle item master, you can use only the predefined item categories context flexfield structure for the Item Categories Key flexfield within Oracle Advanced Pricing. However, you can change the default category code combination for the seeded inventory category. You are limited to the standard seeded item category structure only. Oracle Advanced Pricing does not consider any new structures that you set up for your Item Categories flexfield.

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. 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: Can I use seeded context values on the Oracle Item Master Tables along with pricing attributes to define my hierarchy?
This decision depends on whether the item/item category and pricing attributes define your hierarchy (for levels above Item). If this is the case, attribute mapping is not necessary.

If the seeded context values are not sufficient, you have two options for Release 11*i*. Both approaches require you create attribute mapping to establish these contexts. This enables you to extend your product hierarchy.

- Use flexfields on item table to hold the flattened hierarchy.
- Use tables outside Oracle to house the product hierarchy.

If Oracle Advanced Pricing is installed without Oracle Order Management or the standard item master tables, you must define an alternative table structure location to support the product hierarchy with new contexts and attributes. The table structure does not need to reside within the Oracle Application tables.

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 they 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.

Context	Attribute	Unit Price	Value	Pricing Attribute	Value To	Value From
Product	Item	55	12345	Volume	1	49
Product	Item	50	12345	Volume	50	199
Product	Item	45	12345	Volume	200	499
Product	Item	40	12345	Volume	500	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 structured two lists. The two price lists can be combined. The following is the combined price list:

Context	Attribute	Unit Price	Value	Pricing Attribute	Value To	Value From
Product	Item	55	12345	Volume	1	49
Product	Item	50	12345	Volume	50	199
Product	Item	45	12345	Volume	200	499
Product	Item	40	12345	Volume	500	9999999
Product	Item	$375	65432	Customer class	Hobby	Not applicable
Product	Item	$325	65432	Customer class	Mass Merch	Not applicable

Promotional Discount Modifier

Two adjustments are required for a total solution. First, promotional pricing for edible products that is given to both grocery and mass merch class customers. Second, HugeCo receives special everyday low pricing instead of the standard pricing. A new, additional requirement is 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	Incompatibility 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 Attribute	Value To	Value From	Incomp. Group
Product	Item	Discount	Amount	50	12345	Volume	1	49	GRP01
Product	Item	Discount	Amount	45	12345	Volume	50	199	GRP01
Product	Item	Discount	Amount	40	12345	Volume	200	499	GRP01
Product	Item	Discount	Amount	35	12345	Volume	500	9999999	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.

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 operates as an integration 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 identifier. This enables application invoking the pricing engine to specify the phase identifier, which in turn guides the pricing engine to select only qualified pricing actions that set up in the phase. In Oracle Order Management, the integration with prices ties certain physical events in the order processing cycle to specific phases. This gives order management the capability to 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: *Oracle Advanced Pricing Implementation Manual,* What are Pricing Phases?, page 16-2.

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 Release 10.7 or 11 to Release 11*i*, and are running Oracle Advanced Pricing in Basic Mode with upgraded data, establish a separate test instance of your system and do your testing there.

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.

Related Topics

Diagnostics and Troubleshooting, page 20-22

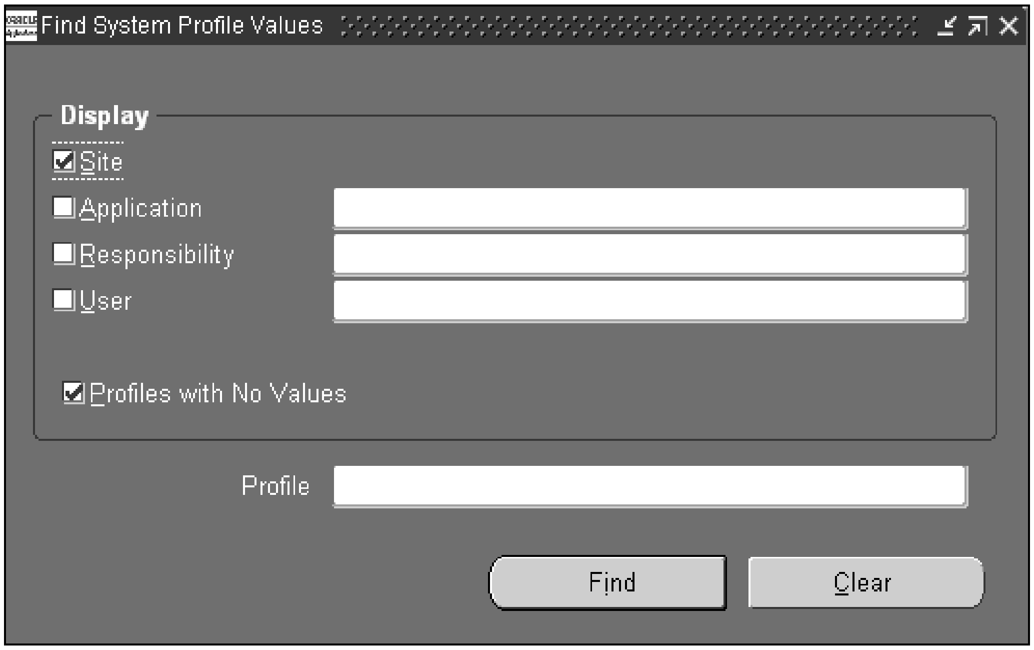
4

Profile Options

Overview

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 Applications System Administrator's Guide*, Setting User Profile Options for more information.

Find System Profile Values


Find System Profile Values

Display

[✓] Site

[ ] Application

[ ] Responsibility

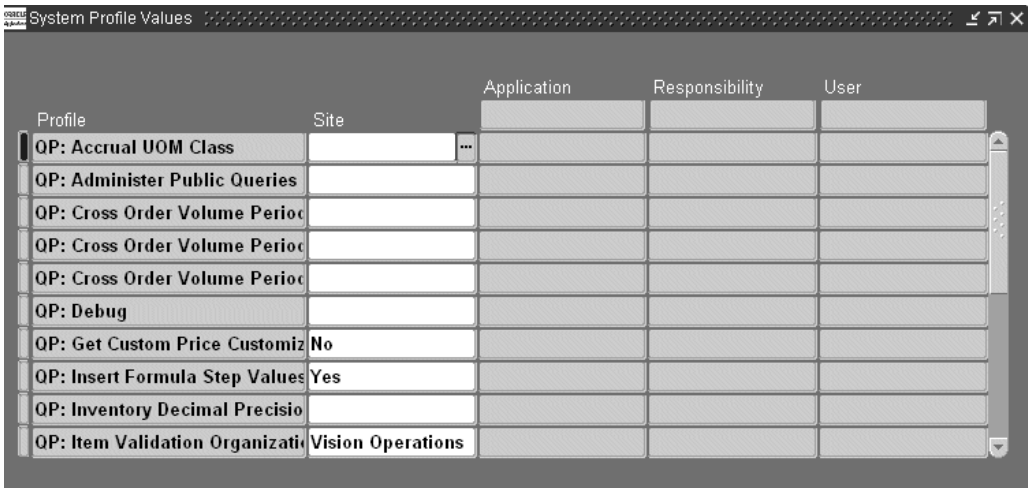
[ ] User

[✓] Profiles with No Values

Profile

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



Profile	Site	Application	Responsibility	User
QP: Accrual UOM Class				
QP: Administer Public Queries				
QP: Cross Order Volume Perioc				
QP: Cross Order Volume Perioc				
QP: Cross Order Volume Perioc				
QP: Debug				
QP: Get Custom Price Customiz	No			
QP: Insert Formula Step Values	Yes			
QP: Inventory Decimal Precisio				
QP: Item Validation Organizati	Vision Operations			

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
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)	• Required (Req.): Requires you to provide a value for the profile option. • Optional: (Opt.) Already provides a default value, so you must change it only if you do not want to accept the default.

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 Application	SA Responsibility	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: Negative Pricing	View Only	View Only	View Only	No	Blank	View Only	Opt
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: 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
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

Profile Option	SA Site	SA Application	SA Responsibility	SA User	Default Value	User	Req?
QP: Build Attributes Mapping Options	Yes	No	No	No	No	View Only	Opt
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 Sourced	Yes	No	No	No	No	View Only	Opt
QP: Debug	No	No	No	Yes	Request Viewer Off	Yes	Opt
QP: Get Custom Price Customized	Yes	No	Yes	No	No	No	Opt
QP: High Volume Order Processing Compliance	No	No	No	No	Blank	No	Opt
QP: Insert Formula Step Values into Temp Table	Yes	No	No	No	No	No	Opt
QP: Inventory Decimal Precision	Yes	Yes	Yes	Yes	10	Yes	Opt
QP: Item Validation Organization	Yes	No	Yes	No	Blank	No	Opt
QP: Licensed for Product	No	Yes	No	No	Blank	--	--

Profile Option	SA Site	SA Application	SA Responsibility	SA User	Default Value	User	Req?
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
QP: Line Weight UOM Code	Yes	Yes	No	No	Blank	No	Opt
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: Price Rounding	Yes	No	No	No	Blank	No	Req
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

Profile Option	SA Site	SA Application	SA Responsibility	SA User	Default Value	User	Req?
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	No	Yes	Individual: = round(listprice) + round(adj)	No	Opt
QP: Set Request Name	Yes	Yes	Yes	Yes	Blank	Yes	Req
QP: Source System Code	Yes	Yes	No	Yes	Oracle Pricing	Yes	Opt
QP: Time UOM Conversion	Yes	Yes	Yes	Yes	Oracle Pricing	Yes	Req
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

Profile Options

OM: Discounting Privilege

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

Values

- Full (default value): 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.

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.

QP: Accrual UOM Class

Default value: Null

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.

QP: Accumulation Attributes Enabled

Default value: No

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.

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.

QP: Administer Public Queries

Default value: Null

Query saved in Public folders can be deleted or renamed only if the 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.

QP: Allow Buckets for Manual Modifiers

Default Value: No

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.

QP: Allow Duplicate Modifiers

Default Value: Yes

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.

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.

Values

- Default Value: 1000

QP: Blind Discount Option

Default value: Yes

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.

QP: Break UOM Proration Allowed

Default value: Blank

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.

QP: Build Attributes Mapping Options

Default value: No

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 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.

QP: Cross Order Volume Period1

Default value: Blank

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.

QP: Cross Order Volume Period2

Default value: Blank

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.

QP: Cross Order Volume Period3

Default value: Blank

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.

QP: Custom Sourced

Default Value: No

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.

QP: Debug

Default value: Request Viewer Off

Values

- Request Viewer On: When set to 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.
- Request Viewer Off: When set to off, nothing is written into pricing debug tables and debug log table. The debug log text file will not be created.
- 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 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.

  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.

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.

QP: Get Custom Price Customized

Default value: No

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.

QP: High Volume Order Processing Compliance

Default value: Blank

This profile option indicates if the pricing setup is compliant with the optimized pricing path when High Volume Order Processing Compliance (HVOP) is used. It can be set only at the site level using the System Administrator responsibility.

Values

- Yes: If no active modifiers of the preceding type are found in the pricing setup, this profile will have a value Yes, which indicates that HVOP takes the optimized path which improves the performance.
- No: If any of the following modifiers or features are active in the pricing setup, HVOP is still supported, but the pricing is not optimized:
  - Coupon Issue modifier
  - Item Upgrade modifier
  - Promotional Goods modifier
  - Promotional limits
  - Terms Substitution modifier

For more information on HVOP, see Overview of High Volume Order Processing (HVOP), page 19-1.

QP: Insert Formula Step Values into Temp Table

Default value: No

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.

QP: Inventory Decimal Precision

Default value: 10

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) = 0.999999999999…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) = 1.33333333333333… which is rounded to 1.333333 DZ.

QP: Item Validation Organization

Default value: None

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.

> **Note:** 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.

QP: Limit Exceed Action

Default value: Hard–Adjust Benefit Amount

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.

QP: Line Volume UOM Code

Default value: Blank

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.

QP: Line Weight UOM Code

Default value: Blank

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.

QP: Licensed for Product

Default value: Blank

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.

QP: Modifier Find Window - Show records

Default value: No

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.

QP: Multi-Currency Installed

Default value: No

If you have global customers or do pricing in different currencies, the multi-currency 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 multi-currency usage.

> **Note:** Once the concurrent program has been run successfully, all existing price list and agreement windows are converted to multi-currency price lists. Users should not return to NON multi-currency 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 multi-currency 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.

QP: Multi Currency Usage

Default value: Blank

This profile option determines if the calling application can use multi-currency price lists.

Values

- Yes: Enables an application to use multi-currency price lists.
- No: Do not use multi currency price lists.

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

QP: Negative Pricing

Default value: No

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.

QP: Pass Qualifiers to Get_Custom_Price API

Default value: Blank

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.

QP: Price Rounding

Default value: Blank

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.

QP: Pricing Transaction Entity

Default value: Order Fulfillment

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 changes 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

QP: Promotional Limits Installed

Default value: No

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!

QP: Qualify Secondary Price Lists

Default value: Yes

This profile option enables secondary price lists to be checked for qualifiers when the primary price list is not validated. It determines if a price list item from a non-validated line can be selected from the related secondary price list.

For example, suppose you have distributors who can only buy certain product families. Without qualifiers being checked on secondary lists, you must create one price list for each possible combination of product family. With this profile change, 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:** The profile option QP: Qualify Secondary Price Lists, which is typically set by the System Administrator,

Values

- Yes: Qualifiers for the secondary price list are evaluated by the pricing engine when the primary price list is not validated.
- No: Secondary price lists will not be checked for qualifiers when the primary price list is not validated.

QP: Return Manual Discounts

Default value: No

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), and always be in the null bucket. Manual adjustments can be overridden.

Manual Discount Flag in the Pricing Engine Request Viewer window
In the Pricing Engine Request Viewer window, the Manual Discount Flag box reflects the setting of the profile QP: Return Manual Discounts. If the Manual Discount Flag box is selected, this indicates that 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.

QP: Satisfied Qualifiers Option

Default value: Yes

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.

QP: Security Control

Default value: Off

This profile option controls the activation of the pricing security feature for your entire installation. It can be set On or Off. Before turning this profile option on, ensure that you have completed all the set up and implementation steps. For more information, see: Pricing Security, *Oracle Advanced Pricing Implementation Manual*.

Values

- Off: After first upgrade to the pricing security feature, the QP: Security Control is set to Off which maintains the pre-upgrade functionality. This means 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: The Security Control profile should not be turned on until all mapping has been completed. All functional users require access privileges to view or maintain their pricing entities.

When the profile QP: Security Control is turned on, each newly created price list and modifier form will be assigned a unique system-generated operating unit identifier.

QP: Security Default Maintain Privilege

Default value: Global

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 operating unit where the price list or modifier was created. 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.

QP: Security Default ViewOnly Privilege

Default value: Global

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.

QP: Selling Price Rounding Options

Default value: Individual: = round(listprice) + round(adj)

This rounding option rounds the selling price after adding unrounded list price and adjustments: selling price = round (list price + adjustments)

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

- NO: = unrounded listprice + unrounded adjustments: No rounding.

- Individual: = round(listprice) + round(adj): Rounds selling price and adjustments.
- Additive: =round(listprice + adj); unrounded Freight: Rounds selling price after adding unrounded list price and adjustments.

Values

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

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 = -round (round(12.60)*0.25) = -3

Selling price = 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 = -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 = -round (round(12.60)*0.25) = -3

Selling price = round (12.60) - 3 = 10

Case 6)

Rounding Flag = NULL

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

Adjustment Amount = -round (round(12.60)*0.25) = -3

Selling price = round(12.60) - 3 = 10

QP: Set Request Name

Default value: Blank

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.

QP: Source System Code

Default value: Oracle Pricing

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:

Application name	Default Value
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)
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.

QP: Time UOM Conversion

Default value: QP: Oracle Pricing

This profile option enforces the unit of measure (UOM) conversion for price lists when the Primary UOM box in a price list is selected. For example, suppose that in a price list, the UOM selected is EA (each) and that the Primary UOM box is selected. (The primary uom in the price list could be different from the primary uom set in Oracle Inventory for the item.)

If an order line is entered with the unit of measure as Dozen, the pricing engine would convert the order quantity to EA and provide a price in EA This occurs only if no eligible price lists are found with Dozen as the unit of measure. To do the conversion, Advanced Pricing uses the inventory UOM conversion APIs.

As for time periods, the inventory UOM conversion uses a static conversion factor of 30 days per month. Since days are set as the base UOM, one year is converted to 12.16666… months by the inventory uom APIs. This result is obtained by dividing 365 days by 30 days rather than one year being converted to 12 months.

For Oracle Contracts, if the price list does not calculate correctly when the Period type = Year, confirm that the UOM conversion for month = 730. Otherwise, when the QP: Time UOM Conversion profile is set to Standard, pricing takes the duration into consideration but converts the hours for 12 months to be 12.1666666666666666666666667 months. When the QP: Time UOM Conversion profile is set to Oracle Contracts, the duration is not considered in the pricing.

To confirm that month = 730, confirm the setup in Oracle Contracts: Setup > Contract > UOM > UOM Classes > Unit of Measure Classes > select Time. (Responsibility: Service Contracts Manager or Corporate Contracts Manager). The following is an example of the Unit of Measure Conversions you may have set up for Time:

- Day = 24 Hours
- Hour = 1 Hour
- Minute = .016667 hours
- Month = 730 Hours
- Quarter = 2190 Hours
- Week = 168 Hours
- Year = 8760 Hours

Values

- Standard: When set to Standard, the existing method of passing the uom quantity to the pricing engine API can be used by the calling applications.
- Oracle Contracts: When set to Oracle Contracts, the pricing engine use the APIs provided by Oracle Contracts to convert the date and time. If the Oracle Contracts API does the uom conversion, the following values from each request line processed need to be passed by the calling applications (such as Order Management, Order Capture or Oracle Contracts) to the pricing engine:
  - Period Start/End Dates
  - Ordered UOM (Time uom in which the price needs to be fetched)
  - Ordered quantity

QP: Unit Price Precision Type

Default value: Standard

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.

QP: Valueset Lookup Filter

Default value: Yes

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 is 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.

QP: Verify GSA Violations

Default value: No

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.

5 Pricing Security

This chapter covers the following topics:

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 functional security and assigns operating unit, responsibility, and system access to users. See the *Oracle Applications System Administrator's Guide* for more information about functional security.

In addition to the existing functional 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 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 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 to pricing entities that control who (the Grantee) 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 only available with license to Advanced Pricing.
- Setting default security access rules for new pricing entities with security profile options.

  Warning: Before setting the profile option QP: Security Control to ON, you must create privileges for existing pricing entities.

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 PTE plus Source System Code security. Functional security is established for each user by responsibility set up. The Oracle Pricing Administrator is a new Responsibility which has complete access to all pricing entities without restriction and is used for global administration of secured access to 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 entity's usage to one or all operating units so 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 entity's usage 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 that 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 within the named operating unit.
- 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.

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, it is recommended that you 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 set the QP: Security Control profile option to ON which turns security on.

> **Warning:** Do not turn the QP: Security Control profile option to ON without completing the implementation steps and guidelines. Otherwise, no price list or modifier list will be visible in the system until you grant a usage to the entity.

You must be assigned the Oracle Pricing Administrator responsibility to set up and maintain the pricing security features for all functional pricing users.

After you have upgraded to pricing security, complete the following steps to set up and use pricing security.

Step 1: Map Complete 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 pre-existing price lists and modifiers to an operating unit. You can also select Global Usage settings that determine if the entity is restricted to that operating unit or available across all operating units.

Step 3: Create Privileges (Privileges page)

The next step is to create all the access privileges for all users in all operating units. You can assign security privileges to grant view or maintain access to a pricing entity.

Step 4: Create Entity Sets (Entity Sets page)

The next step is to create entity sets. (This is an optional step.) You must have a license for Advanced Pricing to use this feature. Entity sets make it convenient to group multiple entities of the same entity type by specified criteria, 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 in the Privileges page.

Step 5: Set up Default Security Profile Options for New Pricing Entities

You can use the following profile options to set the default security privileges for newly-created pricing entities:

- QP: Security Default ViewOnly Privilege
- QP: Security Default Maintain Privilege

These profile options are delivered in default settings that maintain the existing functional security features of Oracle Pricing.

Before changing 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 6: Set the QP: Security Control profile option to ON

The QP: Security Control profile option is the "switch" that turns security on or off for your installation. Before setting the profile option QP: Security Control to ON, it is recommended that you have completed all the preceding implementation steps.

Related Topics

Assigning Ownership of Pricing Entities to Operating Units (Entity Usage page) , page 5-7

Creating Privileges, page 5-12

Creating Pricing Entity Sets, page 5-19

Setting up Default Security Profile Options for New Pricing Entities, page 5-24

Setting the QP: Security Control profile option to ON, page 5-30

Changes to Pricing windows after Upgrading and Turning Security On

This section summarizes the changes that occur to pricing entities after you upgrade to pricing security and turn security on. Some of the changes, such as the new Global box on price lists and modifiers, are only visible to users after pricing security is turned on.

Changes to Existing Pricing windows

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 pricing security is turned on, a Global box - which identifies the global usage status - is dynamically added to the header region of all price lists and modifiers. The Global box is visible to end-users and can be updated (cleared or selected) by users with Maintain access privileges.

Changes to Price Lists

After the upgrade, you can review the operating unit and global usage settings for an entity in the Entity Usage page. An example of the information that displays 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 other changes occur to price lists after the upgrade to pricing security:

- A price list assigned Global Usage can be selected by the pricing engine even if the price list has been assigned to a specific operating unit.

  If the price list is unavailable for global usage (for example, a user clears the Global box in the price list header), then the pricing engine will select this price list only if the current operating unit is the same as the one that created the price list.
- Once security is turned on, all new price lists have their view and update properties determined by the pricing security profile options.
- 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.
- The Public API, QP_PRICE_LIST_PUB.PROCESS_PRICE_LIST will only update price lists as per price list security rule.
- Users selecting using Price Lists > Copy Price Lists can copy price lists. A copied price list is assigned the default privilege from the security profile options. The copied price list will belong to the operating unit of the user who created it, regardless of the operating unit assigned to the original price list.

Changes to Modifier windows

- A modifier assigned Global Usage can be selected by the pricing engine even if the modifier has been assigned to a specific operating unit.

If the modifier is unavailable for global usage (for example, a user clears the Global box in the modifier header), then the pricing engine will select this modifier only if the current operating unit is the same as the one that created the modifier.

- After pricing security is turned on, the default view and maintain properties for all new modifiers are determined by the security profile options.
- Users need at least view-only access privileges to display or query modifiers in the Define Modifier window. A user with view-only access privileges can view all list and line limits for a modifier including attributes and transactions for the limit.
- Users with view-only access privileges cannot modify the header information, lines, list or line qualifiers, pricing attributes, and related modifier information. A message or hint will display to notify the user 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.
- The Public API, QP_MODIFIERS_PUB.PROCESS_MODIFIERS will only update 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 opened by clicking the Modifiers button may be viewed or maintained depending on the privileges defined by the Pricing Security Administrator.
- Users with view-only access to a price list can copy the price list by choosing Price Lists > Copy Price Lists. The copied price list is:
  - Assigned the default privileges from the security profile options.
  - Belongs to the operating unit of the user who copied it, regardless of the operating unit assigned to the original price list.

Changes to Order Management

All price list (LOV) list of values in Oracle Order Management will call the Pricing API, Get_Pricelists(), to return a list of valid price lists. The API returns the price lists owned by the same operating unit as the operating unit of the current user and those price lists where the Global box is selected.

Changes to other Pricing windows

The following table outlines the impact of pricing security and security privileges on various windows in Advanced Pricing:

For the following:	Security Privileges are enforced:
Copy Price Lists	Yes. User needs at least view-only access.
Copy Modifier	Yes. User needs at least view-only access.
Modifier Incompatibility setup	Yes, can be updated if user has maintain access.
Pricing Organizer	Yes. User with view access can view if modifier displayed.
Pricing Mass Maintenance	Yes. User needs maintain access.
Adjust Price List	Yes. User needs maintain access.
Add Items to Price Lists	Yes. User needs 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. User must have at least view-only access.
Modifier Detail report	Yes. User must have at least view-only access.

Assigning Ownership of Pricing Entities to Operating Units (Entity Usage page)

After the upgrade to pricing security, each newly-created price list and modifier is assigned the operating unit of the user who created it. Assigning operating unit ownership to pricing entities restricts usage of that entity to within that operating unit. This prevents the use of an entity across all your operating units.

Since pre-existing price lists and modifiers are not assigned a default operating unit, the Oracle Pricing Administrator can:

- Assign or reassign ownership of pre-existing price lists and modifiers to the appropriate operating unit.
- Grant or revoke Global Usage of pricing entities which enables the pricing entity to be accessed across all operating units.

> **Warning:** It is recommended that the Oracle Pricing Administrator assigns ownership to all price lists and modifiers prior to upgrading or implementing Oracle Pricing Security. This can be done using the Bulk Update Entity Usage feature in the Entity Usage page.

Global Usage and the Global box

In the Entity Usage page, the Global Usage column for a given pricing entity reflects the status of the Global box on the price list and modifier windows.

When global usage is enabled for an entity, that entity can be shared across operating units. Its usage is not restricted to the assigned operating unit.

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 selected, global usage is enabled for the entity.
- If not selected (cleared), global usage is not enabled for the pricing entity, and the entity's usage is restricted to the assigned operating unit.

The Global check box is not visible to users until the profile QP: Security Control is turned on. When 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 box.

If a user creates a new pricing entity (such as a price list) and clears the Global box, then the new entity can be used in pricing only by the operating unit of the creating user. If the Global check box is left selected (the default value), then the entity can be used across all operating units when pricing transactions, even though the creating operating unit owns it.

Alternately, the Pricing Administrator can also update the Global box for one entity at a time or in bulk using the Bulk Update Entity Usage page available from the Entity Usage page.

> **Warning:** Pricing users with Maintain access should be advised of the impact of clearing or selecting the Global box.

Default Ownership for Newly Created Pricing Entities

Each price list and modifier that is *newly* created is assigned a unique system-generated Operating Unit identifier. When querying newly-created pricing entities in the Pricing Security pages, the operating unit ownership is displayed in the Owned By Operating Unit field.

Creating Pricing Entity Usage

Complete the following planning and implementation steps before turning on pricing security. This mapping should be completed by someone with complete knowledge about the various 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.

When mapping your current and proposed security requirements, you may want to prepare separate listings for each of the following entity types:

- Standard price list
- Agreement price list
- Modifier
- Pricing Entity Set

Based on the security policy of your organization, the Oracle Pricing Administrator can grant access privileges to the pricing entities, once entity usage has been set up.

For each entity, decide whether:

- The pricing entity can be used by all operating units in pricing transactions.
- The pricing entity should only be used by a specific operating unit.

Identify which entities are to be used across all operating units:

For each entity, decide whether:

- The pricing entity can be used by all operating units in pricing transactions.

- The pricing entity should only be used by a specific operating unit.

Identify which entities are to be restricted to only one operating unit:

For each entity, decide whether:

- The pricing entity can be used by all operating units in pricing transactions.
- The pricing entity should only be used by a specific operating unit.

Pricing entities used by multiple operating units:

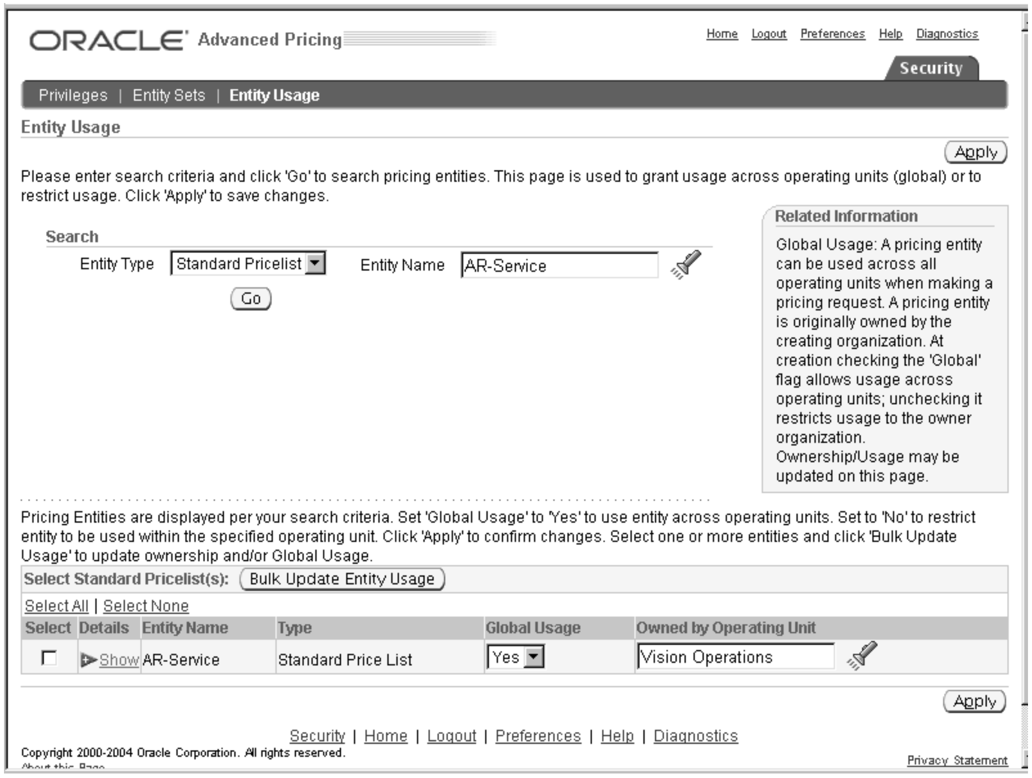
For pricing entities that can be used by *multiple* operating units but not *all* operating units:

- Select Yes for Global Usage.
- In the Pricing Entity page, create qualifiers for the specific operating units.

To create pricing entity usage:

1. Navigate to the Entity Usage page to:
   - Assign operating unit ownership to pricing entities such as price lists and modifier lists.
   - Assign Global Usage values of Yes or No.

Entity Usage page


ORACLE Advanced Pricing
Home Logout Preferences Help Diagnostics
Security
Privileges | Entity Sets | Entity Usage

Entity Usage

Apply

Please enter search criteria and click 'Go' to search pricing entities. This page is used to grant usage across operating units (global) or to restrict usage. Click 'Apply' to save changes.

Search

Entity Type: Standard Pricelist   Entity Name: AR-Service

Go

Related Information

Global Usage: A pricing entity can be used across all operating units when making a pricing request. A pricing entity is originally owned by the creating organization. At creation checking the 'Global' flag allows usage across operating units; unchecking it restricts usage to the owner organization. Ownership/Usage may be updated on this page.

Pricing Entities are displayed per your search criteria. Set 'Global Usage' to 'Yes' to use entity across operating units. Set to 'No' to restrict entity to be used within the specified operating unit. Click 'Apply' to confirm changes. Select one or more entities and click 'Bulk Update Usage' to update ownership and/or Global Usage.

Select Standard Pricelist(s): Bulk Update Entity Usage

Select All | Select None

Select	Details	Entity Name	Type	Global Usage	Owned by Operating Unit
☐	▶Show	AR-Service	Standard Price List	Yes	Vision Operations

Apply

Security | Home | Logout | Preferences | Help | Diagnostics

Copyright 2000-2004 Oracle Corporation. All rights reserved.

Privacy Statement

2. In the Search region, select your search criteria:
   - Entity Type: Select an entity type such as Standard Pricelist or Modifier.

- Entity Name: Optionally, enter an Entity Name to search for a particular price list or modifier.

3. Click Go to display the search results in the Results region of the page. For each listed entity, the following information is displayed:
   - Details: Click the Expand icon to view additional details about the selected Entity such as its Active Status, Start and End Dates, Description, and Currency.
   - Entity Name: Displays the unique name that identifies the selected entity.
   - Type: Describes the Entity Type selected such as Standard Price List or Modifier.
   - Global Usage: Indicates the current usage status of the pricing entity.
   - Owned by Operating Unit: Displays the name of the Operating Unit associated with the Entity.

     Note: For fresh upgrades or new installations, the Global Usage box is Yes (selected) and the Owned by Operating Unit field is blank.

To update Operating Unit and Global Usage for a pricing entity:

For each pricing entity listed in the Results region, you can assign a Global Usage and Owned by Operating Unit value. To make bulk changes to multiple pricing entities, use the Bulk Update Entity Usage feature.

1. Click Go to display the search results in the Results region of the page. For each listed entity, the following information is displayed:
2. To make the entity available across all operating units, select Yes for Global Usage. Alternately, select No to restrict the entity's use to within the specified operating unit.
3. Select the operating unit in the Owned by Operating Unit field.
4. Click OK to save your changes.

Related Topics

Using Bulk Update Entity Usage, page 5-10

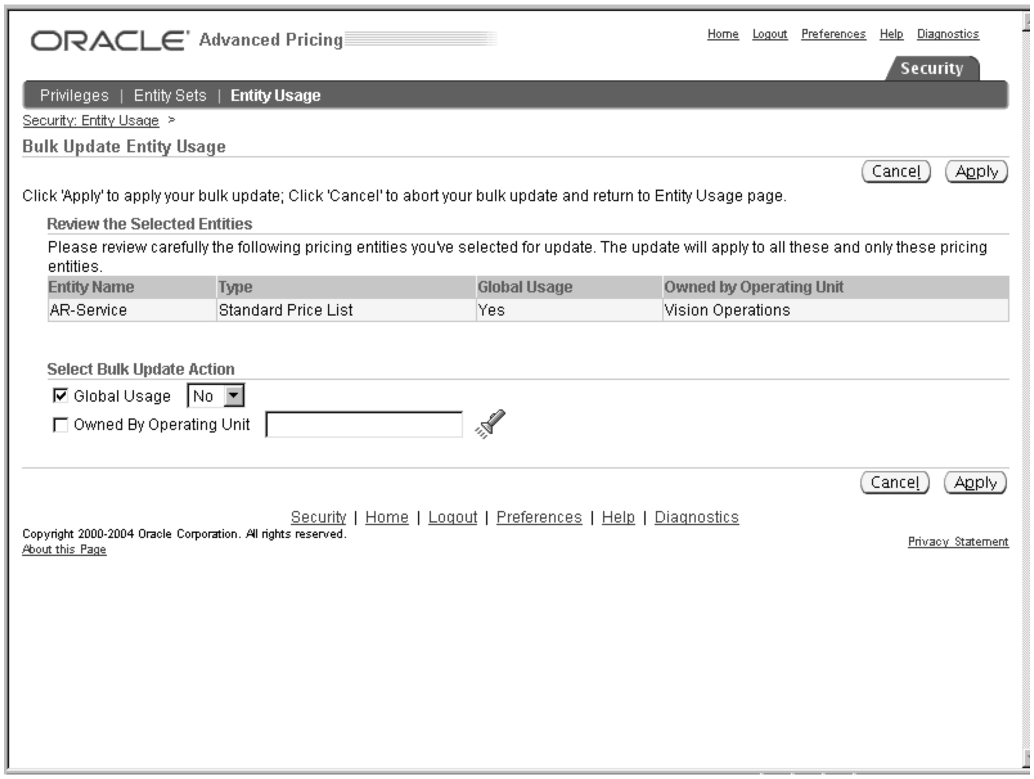
Using Bulk Update Entity Usage

Use the Bulk Update Entity Usage page to quickly apply the same changes across selected pricing entities; for example, to assign the same operating unit across all price lists.

To use bulk update entity usage:

1. Select the pricing entities from the Results region. Alternately, to select all pricing entities on a page, click Select All. If additional entities are listed on subsequent pages, click the Next link, then click Select All. Repeat this process until all the entities to be updated are selected.
2. After the pricing entities are selected, click Bulk Entity Usage to display the selected entity names in the Bulk Update Entity Usage page.

Bulk Update Entity Usage page


ORACLE Advanced Pricing
Home Logout Preferences Help Diagnostics
Security
Privileges | Entity Sets | Entity Usage
Security: Entity Usage >
Cancel
Apply

Bulk Update Entity Usage

Click 'Apply' to apply your bulk update; Click 'Cancel' to abort your bulk update and return to Entity Usage page.

Review the Selected Entities

Please review carefully the following pricing entities you've selected for update. The update will apply to all these and only these pricing entities.

Entity Name	Type	Global Usage	Owned by Operating Unit
AR-Service	Standard Price List	Yes	Vision Operations

Select Bulk Update Action

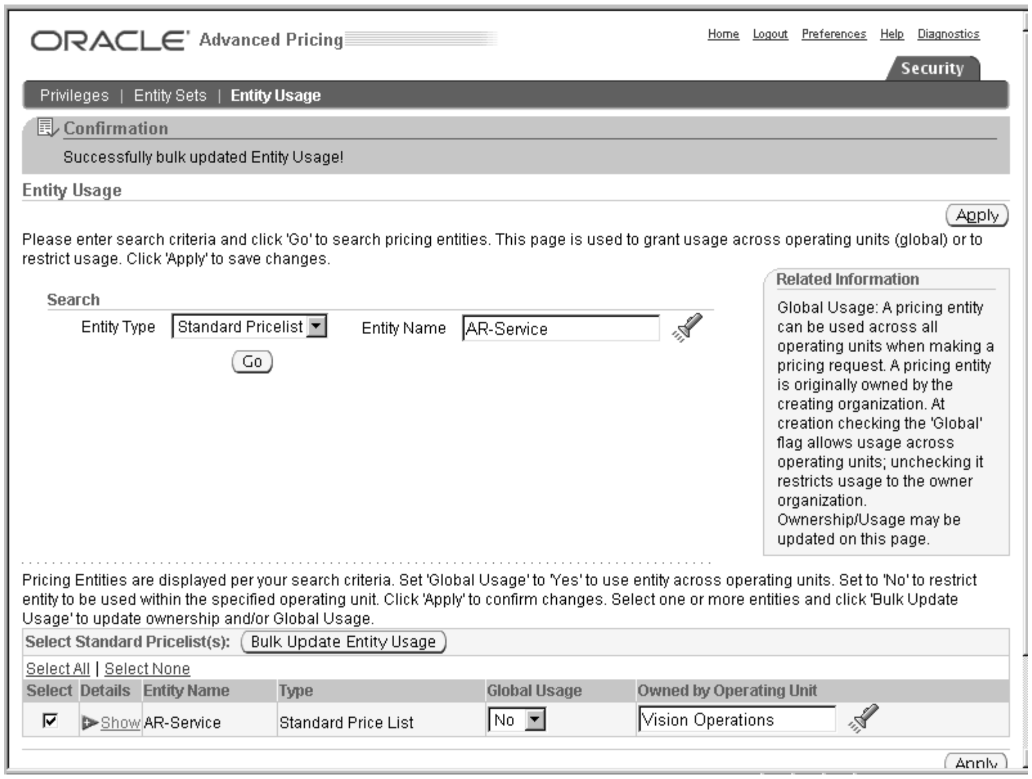
☑ Global Usage No

☐ Owned By Operating Unit

Security | Home | Logout | Preferences | Help | Diagnostics
Copyright 2000-2004 Oracle Corporation. All rights reserved.
About this Page
Privacy Statement

3. Review the pricing entities listed in the Review the Selected Entities region. Any updates will apply to the listed pricing entities.
4. In the Select Bulk Update Action region, select the Global Usage box and select Yes or No to update all the entities global usage to Yes or No.
5. Select the Owned by Operating Unit box to update all the entities with the specified Operating Unit.
6. Click Apply. If successful, a Confirmation message advises that you have successfully bulk updated the entity usage.

Entity Usage: Confirmation page


ORACLE Advanced Pricing
Home Logout Preferences Help Diagnostics
Security
Privileges | Entity Sets | Entity Usage

Confirmation

Successfully bulk updated Entity Usage!

Entity Usage

Apply

Please enter search criteria and click 'Go' to search pricing entities. This page is used to grant usage across operating units (global) or to restrict usage. Click 'Apply' to save changes.

Search

Entity Type: Standard Pricelist   Entity Name: AR-Service

Go

Related Information

Global Usage: A pricing entity can be used across all operating units when making a pricing request. A pricing entity is originally owned by the creating organization. At creation checking the 'Global' flag allows usage across operating units; unchecking it restricts usage to the owner organization. Ownership/Usage may be updated on this page.

Pricing Entities are displayed per your search criteria. Set 'Global Usage' to 'Yes' to use entity across operating units. Set to 'No' to restrict entity to be used within the specified operating unit. Click 'Apply' to confirm changes. Select one or more entities and click 'Bulk Update Usage' to update ownership and/or Global Usage.

Select Standard Pricelist(s): Bulk Update Entity Usage

Select All | Select None

Select	Details	Entity Name	Type	Global Usage	Owned by Operating Unit
✓	Show	AR-Service	Standard Price List	No	Vision Operations

Apply

7. Click Apply to save your changes.

Creating Privileges

Security privileges enable you to define who can access each pricing entity and the level of access permitted: View Only or Maintain.

> **Note:** You must be assigned the Oracle Pricing Administrator responsibility to grant privileges.

The following describes the security privileges that can be assigned:

- 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 within the named operating unit.
  - 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.

- Grant default security privileges for newly created pricing entities. See Setting up Default Security Profile Options for New Pricing Entities, page 5-24 for more information.
- Use Entity Sets to define rules for a pricing entity that will restrict a specific user or group of users from accessing entities restricted to a specific customer, group of customers or a specified customer hierarchy level. See Creating Pricing Entity Sets, page 5-19 for more information.

You can define rules for a pricing entity that will restrict a specific user or group of users to accessing entities for a specific customer, group of customers or a specified customer hierarchy level.

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.

Precedence Levels for Multiple Privileges

A user belonging to a Responsibility classification such as Pricing User typically is assigned the access privileges associated with that Responsibility. However, if a user has View Only access to a pricing entity by virtue of their 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 their user hierarchy (Responsibility, Operating Unit, and Global), they will have Maintain access regardless of any other privileges.

For example: if a user has Maintain access at their operating unit level but a view-only access at their user level, their Maintain access privilege will have precedence.

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 if 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 to 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 which is a higher privilege than that granted to his or her Responsibility.

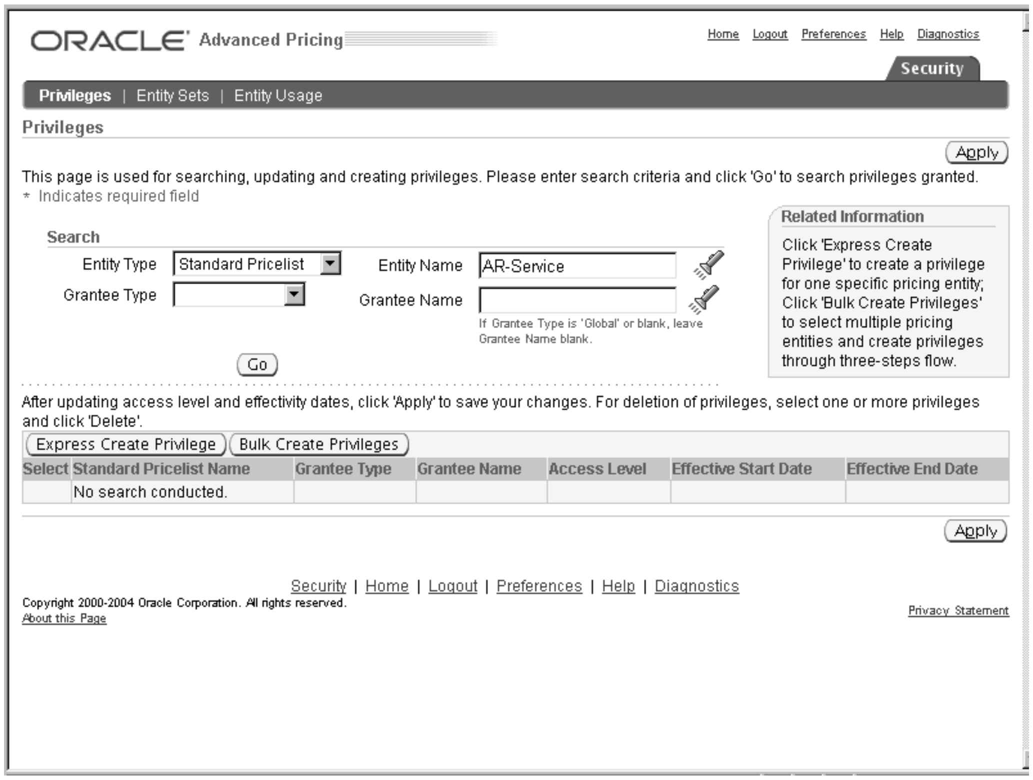
2. **List all users by new access privileges**

   It is recommended that a listing of all users and their access privileges be maintained by the Pricing Administrator. Once mapping has been completed and access privileges granted, you can query the privileges 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.

To create privileges:

1. Navigate to the Privileges page.

Privileges page


ORACLE Advanced Pricing
Home Logout Preferences Help Diagnostics
Security
Privileges | Entity Sets | Entity Usage
Apply
Go
Express Create Privilege
Bulk Create Privileges
Apply
Security | Home | Logout | Preferences | Help | Diagnostics
About this Page
Privacy Statement

Privileges

This page is used for searching, updating and creating privileges. Please enter search criteria and click 'Go' to search privileges granted.
* Indicates required field

Search

Entity Type	Standard Pricelist	Entity Name	AR-Service
Grantee Type		Grantee Name	

If Grantee Type is 'Global' or blank, leave Grantee Name blank.

Related Information
Click 'Express Create Privilege' to create a privilege for one specific pricing entity; Click 'Bulk Create Privileges' to select multiple pricing entities and create privileges through three-steps flow.

After updating access level and effectivity dates, click 'Apply' to save your changes. For deletion of privileges, select one or more privileges and click 'Delete'.

Select	Standard Pricelist Name	Grantee Type	Grantee Name	Access Level	Effective Start Date	Effective End Date
	No search conducted.					

Copyright 2000-2004 Oracle Corporation. All rights reserved.

2. In the Search region, select an Entity Type. Optionally, select additional search criteria such as Entity Name, Grantee Type, or Grantee Name to filter your search results. To view the available values for an Entity Name or Grantee Name, click the Search icon.

3. Click Go to display the search results in the Results region of the Privileges page.

4. If the message *No data exists displays* in the Results: Privilege(s) region then no privileges exist for the entity.

   If search results display, then you can view or update the privileges directly in the Results: Privilege(s) region.

5. To revoke privileges, select the line to delete and click Delete.

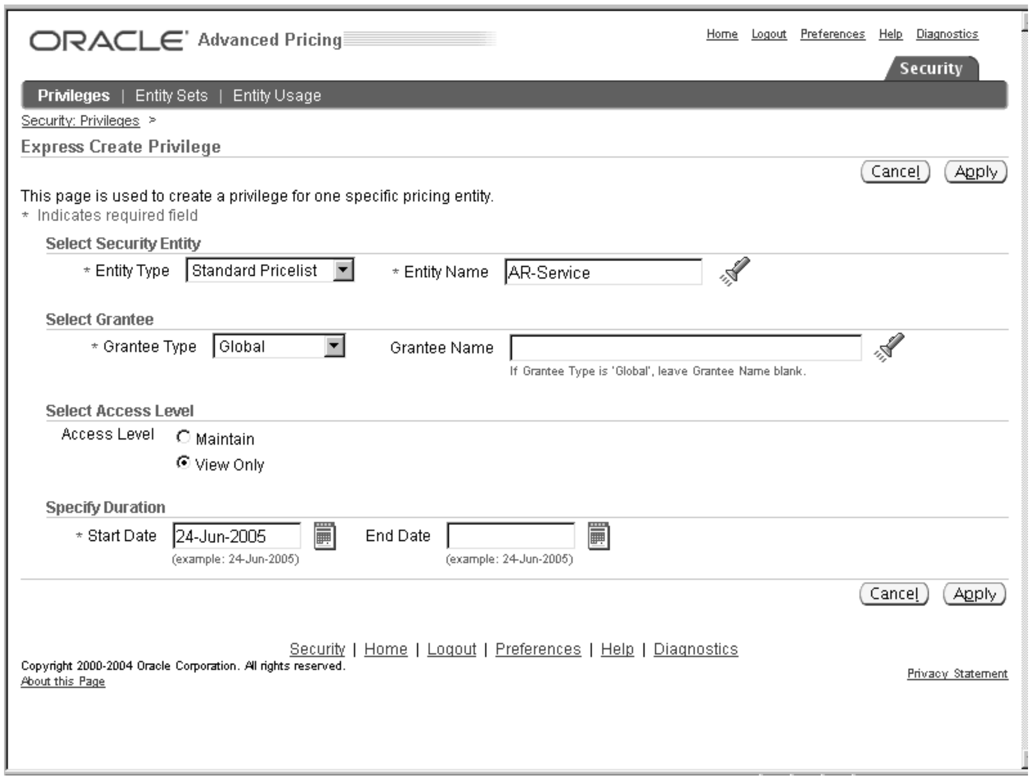
6. To assign or update an Access Level, select Maintain or View Only.

7. Enter or update the Effective Start and End Date and click OK to save your changes.

To use Express Create Privilege:

1. To create a privilege for one specific pricing entity, select the entity and click the Express Create Privilege button to display the Express Create Privilege page.

Express Create Privilege page


ORACLE Advanced Pricing
Home Logout Preferences Help Diagnostics
Security
Privileges | Entity Sets | Entity Usage
Security: Privileges >
Cancel Apply
Cancel Apply
Security | Home | Logout | Preferences | Help | Diagnostics
Privacy Statement
About this Page

Express Create Privilege

This page is used to create a privilege for one specific pricing entity.
* Indicates required field

Select Security Entity
* Entity Type: Standard Pricelist
* Entity Name: AR-Service

Select Grantee
* Grantee Type: Global
Grantee Name:
If Grantee Type is 'Global', leave Grantee Name blank.

Select Access Level
Access Level: Maintain [✓ View Only]

Specify Duration
* Start Date: 24-Jun-2005 (example: 24-Jun-2005)
End Date: (example: 24-Jun-2005)

Copyright 2000-2004 Oracle Corporation. All rights reserved.

2. In the Select Security Entity region, select the Entity Type and Entity Name of the pricing entity to be granted privileges.

3. In the Select Grantee region, select one of the following Grantee Types and 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 belonging to operating unit Vision1.

4. In the Select Access Level region, 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. In the Specify Duration region, select the Start and End Date. For example, to provide temporary access to a temporary employee, you could enter a Start Date of 02-Jul-2004 and an End Date of 31-Aug-2004. Alternately, accept the system dates.
6. Click Apply.

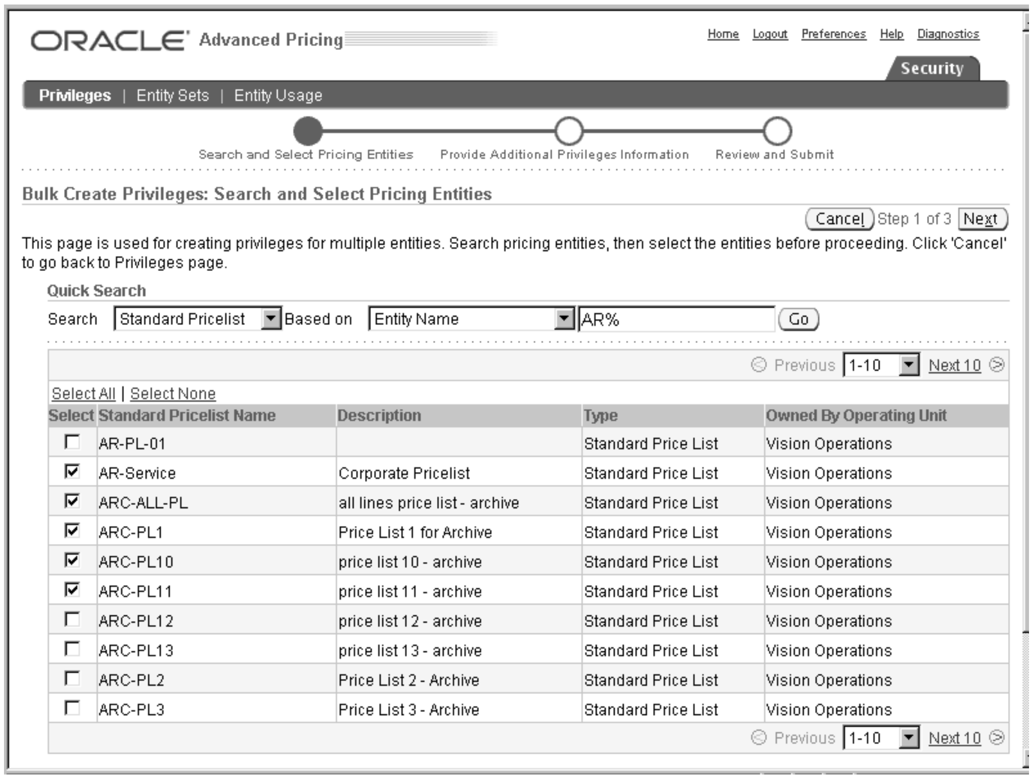
Creating Bulk Privileges

Use the Bulk Create Privileges page to quickly create and assign privileges to multiple entities such as price lists or modifiers for a specific entity type. For example, you could grant access to several price lists to the Operating Unit: Vision France.

To create bulk privileges:

1. From the Privileges page, click Bulk Create Privileges to display the **Bulk Create Privileges: Search and Select Pricing Entities** page.

Bulk Create Privileges page


ORACLE Advanced Pricing
Home Logout Preferences Help Diagnostics
Security
Privileges | Entity Sets | Entity Usage
Search and Select Pricing Entities
Provide Additional Privileges Information
Review and Submit
Cancel
Step 1 of 3
Next
Go
Previous
1-10
Next 10
Select All | Select None

Bulk Create Privileges: Search and Select Pricing Entities

This page is used for creating privileges for multiple entities. Search pricing entities, then select the entities before proceeding. Click 'Cancel' to go back to Privileges page.

Quick Search

Search: Standard Pricelist Based on: Entity Name AR%

Select	Standard Pricelist Name	Description	Type	Owned By Operating Unit
☐	AR-PL-01		Standard Price List	Vision Operations
☑	AR-Service	Corporate Pricelist	Standard Price List	Vision Operations
☑	ARC-ALL-PL	all lines price list - archive	Standard Price List	Vision Operations
☑	ARC-PL1	Price List 1 for Archive	Standard Price List	Vision Operations
☑	ARC-PL10	price list 10 - archive	Standard Price List	Vision Operations
☑	ARC-PL11	price list 11 - archive	Standard Price List	Vision Operations
☐	ARC-PL12	price list 12 - archive	Standard Price List	Vision Operations
☐	ARC-PL13	price list 13 - archive	Standard Price List	Vision Operations
☐	ARC-PL2	Price List 2 - Archive	Standard Price List	Vision Operations
☐	ARC-PL3	Price List 3 - Archive	Standard Price List	Vision Operations

2. In the Quick Search region, do a search by Entity Type to find the pricing entity or entities to be granted privileges. For example, select Standard Pricelist to search for standard price lists.

3. Optionally, select additional search criteria to refine your search. In the Based on field, select Owned by Operating Unit or Entity Name then enter related details.

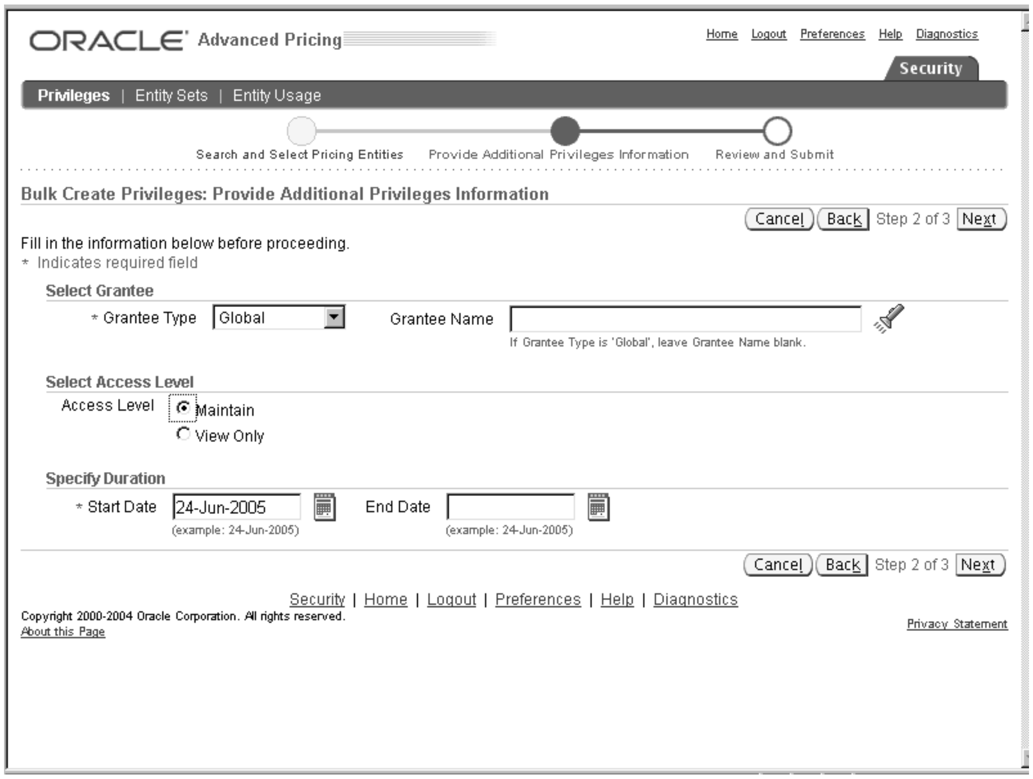
   For example, to find the Summer Pricelist, select Standard Pricelist as the Entity Type, then select Entity Name and enter Summer Pricelist to specify your search criteria.

4. Click Go to display the search results in the Results region.

5. From the search results, select the entities to be assigned privileges.

6. Click Next to display the Bulk Create Privileges: Provide Additional Privileges Information page.

Bulk Create Privileges: Provide Additional Privileges Information page


ORACLE Advanced Pricing
Home Logout Preferences Help Diagnostics
Security
Privileges | Entity Sets | Entity Usage
Search and Select Pricing Entities
Provide Additional Privileges Information
Review and Submit

Bulk Create Privileges: Provide Additional Privileges Information

Cancel Back Step 2 of 3 Next

Fill in the information below before proceeding.
* Indicates required field

Select Grantee

* Grantee Type: Global
Grantee Name:
If Grantee Type is 'Global', leave Grantee Name blank.

Select Access Level

Access Level: [✓ Maintain] View Only

Specify Duration

* Start Date: 24-Jun-2005 (example: 24-Jun-2005)
End Date: (example: 24-Jun-2005)

Cancel Back Step 2 of 3 Next
Security | Home | Logout | Preferences | Help | Diagnostics
Copyright 2000-2004 Oracle Corporation. All rights reserved.
About this Page
Privacy Statement

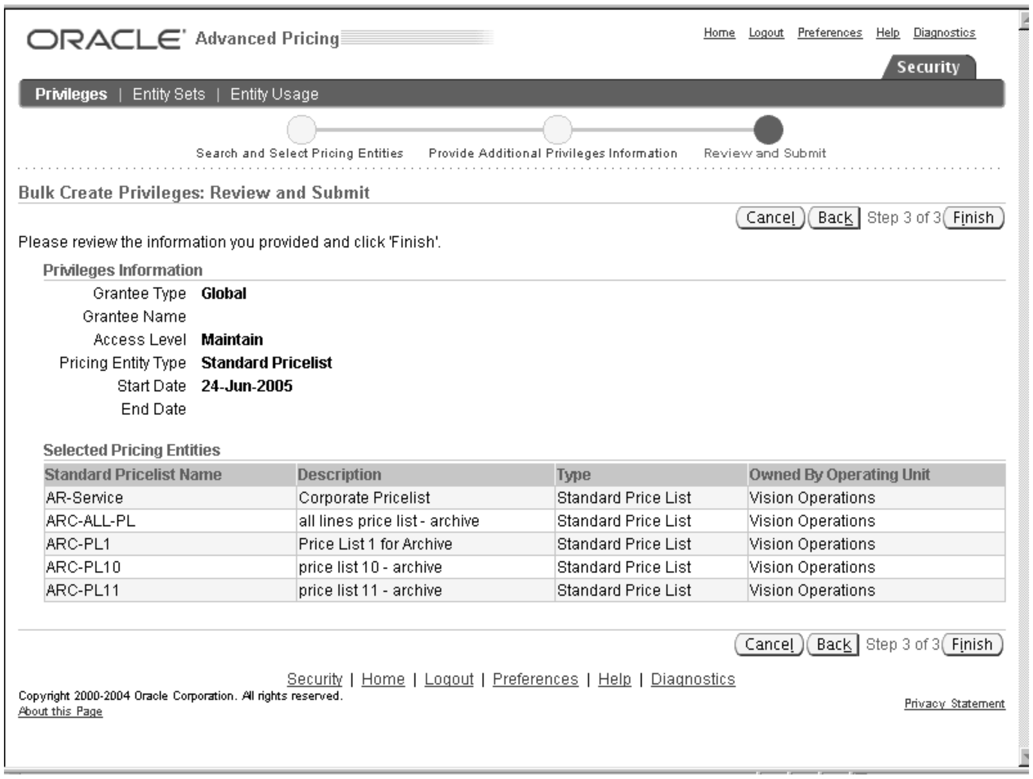
7. Select a Grantee Type and an associated Grantee Name. To display the available values for a Grantee Name, click the Search icon:

   - 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 across operating units.

- Operating Unit: Grants the privilege to a specific operating unit. For example, select Vision1 to assign the pricing entity to operating unit Vision1. Users who are not from operating unit Vision1 are unable to access this pricing entity.

8. Select the Access Level to be granted to the Grantee:
   - View Only: Enables users to view but not update the pricing entity.
   - Maintain: Enables users to delete, view, and update the pricing entity.
9. Select the Start and End Date in the Specify Duration region. For example, to grant temporary access to a summer employee, you could enter a Start Date of 02-Jul-2005 and an End Date of 31-Aug-2005. Alternately, accept the default system dates.
10. Click Next to display the Bulk Create Privileges: Review and Submit page.

Bulk Create Privileges: Review and Submit page


ORACLE Advanced Pricing
Home Logout Preferences Help Diagnostics
Security
Privileges | Entity Sets | Entity Usage
Search and Select Pricing Entities
Provide Additional Privileges Information
Review and Submit
Cancel Back Step 3 of 3 Finish

Bulk Create Privileges: Review and Submit

Please review the information you provided and click 'Finish'.

Privileges Information

Grantee Type	Global
Grantee Name	
Access Level	Maintain
Pricing Entity Type	Standard Pricelist
Start Date	24-Jun-2005
End Date	

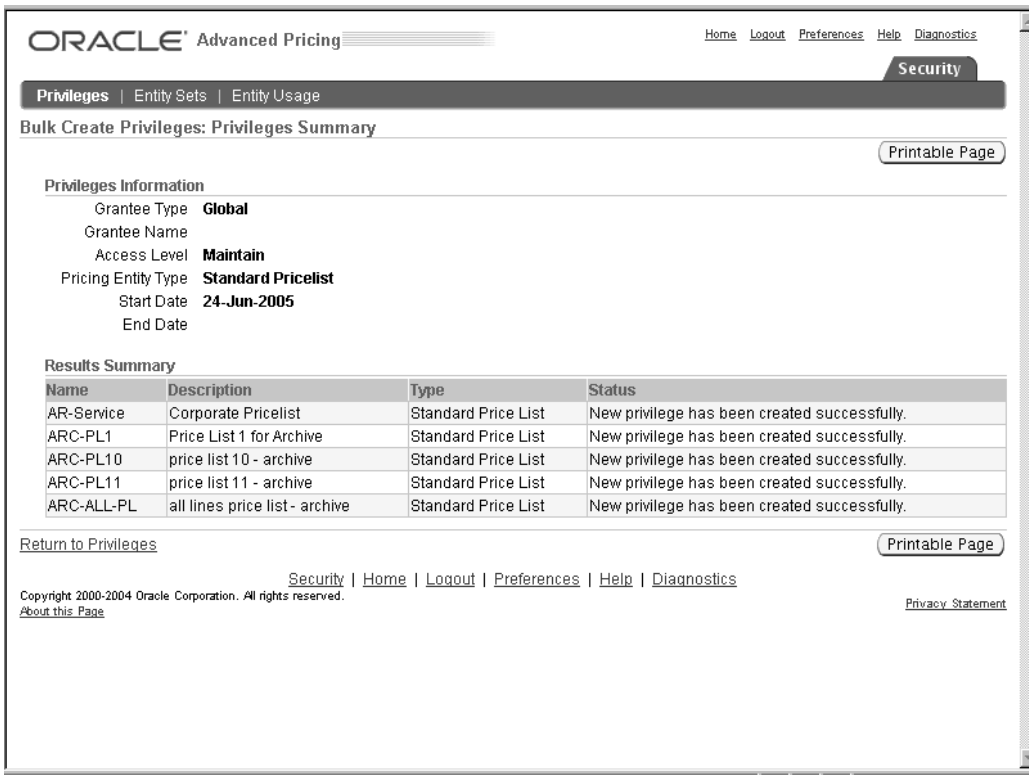
Selected Pricing Entities

Standard Pricelist Name	Description	Type	Owned By Operating Unit
AR-Service	Corporate Pricelist	Standard Price List	Vision Operations
ARC-ALL-PL	all lines price list - archive	Standard Price List	Vision Operations
ARC-PL1	Price List 1 for Archive	Standard Price List	Vision Operations
ARC-PL10	price list 10 - archive	Standard Price List	Vision Operations
ARC-PL11	price list 11 - archive	Standard Price List	Vision Operations

Cancel Back Step 3 of 3 Finish
Security | Home | Logout | Preferences | Help | Diagnostics
Copyright 2000-2004 Oracle Corporation. All rights reserved.
About this Page
Privacy Statement

11. Review the information in the following regions before submitting your changes to ensure it is correct:
    - Privileges Information region: Displays the privilege information.
    - Selected Pricing Entities region: Displays the following information about the pricing entities to be granted the privileges listed in the Privileges Information region: Entity Name, Description, Type, Owned By Operating Unit.
12. If changes are required, click Back, or click Cancel to stop the process completely.
13. Click Finish to view the Privileges Summary page.

Privileges Summary page


ORACLE Advanced Pricing
Home Logout Preferences Help Diagnostics
Security
Privileges | Entity Sets | Entity Usage

Bulk Create Privileges: Privileges Summary

Printable Page

Privileges Information

Grantee Type	Global
Grantee Name	
Access Level	Maintain
Pricing Entity Type	Standard Pricelist
Start Date	24-Jun-2005
End Date	

Results Summary

Name	Description	Type	Status
AR-Service	Corporate Pricelist	Standard Price List	New privilege has been created successfully.
ARC-PL1	Price List 1 for Archive	Standard Price List	New privilege has been created successfully.
ARC-PL10	price list 10 - archive	Standard Price List	New privilege has been created successfully.
ARC-PL11	price list 11 - archive	Standard Price List	New privilege has been created successfully.
ARC-ALL-PL	all lines price list - archive	Standard Price List	New privilege has been created successfully.

Return to Privileges

Printable Page

Copyright 2000-2004 Oracle Corporation. All rights reserved.
About this Page
Privacy Statement

Creating Pricing Entity Sets

You can create "sets" of pricing entities that can be used as a single entity when creating a privilege(s) for a given grantee. Each entity set can contain multiple pricing entities of the same entity type such as a set for price lists and another for modifiers.

For example, you could create an entity set consisting of all price lists based on a specific customer name, then grant maintain access (in the Privileges pages) to a specific user for the entity set.

The following outlines the steps for creating and using an Entity Set:

- Create an Entity Set using the Create Entity Set page. In this page, you can select the criteria for your set. Only header level criteria can be used in creating the set.
- Use the Entity Set as the grant object (with object type as ENTITY SET) and grant access roles to any grantee type and grantee.

It is important to identify the selected criteria in the description of the entity set. Once an entity set is defined, you cannot copy the entity set or change its criteria. If changes are required, a new entity set must be created. The Entity Set feature is only available to licensed users of Oracle Advanced Pricing.

> **Note:** Entity sets cannot be copied or updated. You can revoke or add privileges as needed. However, the entity set cannot be deleted if there are any existing privileges on that entity set.

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 in 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 inherits view only access.

For entity sets:

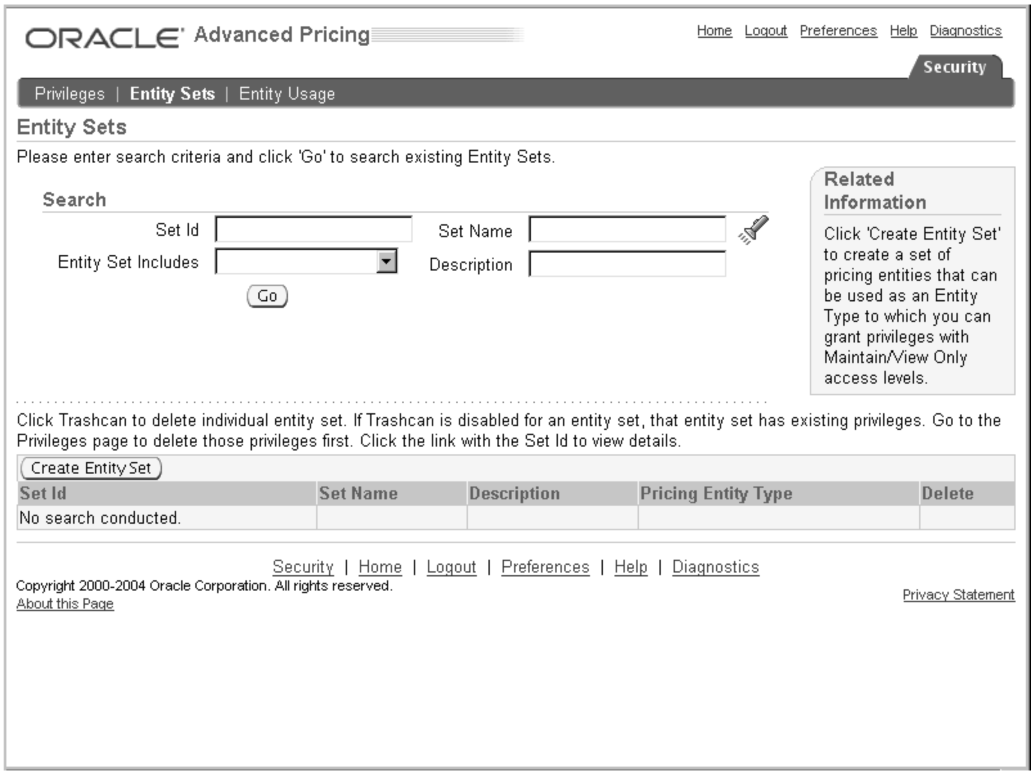
- It is possible to create an Entity Set for a specified set of criteria that does not currently exist in the system.
- It is possible to create access privileges for this entity set even when no records currently exist in the system.
- Any new records created that meet the set criteria are automatically assigned to the set and inherit the privileges 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.

To create an entity set:

1. Navigate to the Entity Sets page.

Entity Sets page



ORACLE Advanced Pricing
Home Logout Preferences Help Diagnostics
Security
Privileges | Entity Sets | Entity Usage

Entity Sets

Please enter search criteria and click 'Go' to search existing Entity Sets.

Search

Set Id		Set Name	
Entity Set Includes		Description	

Go

Related Information

Click 'Create Entity Set' to create a set of pricing entities that can be used as an Entity Type to which you can grant privileges with Maintain/View Only access levels.

Click Trashcan to delete individual entity set. If Trashcan is disabled for an entity set, that entity set has existing privileges. Go to the Privileges page to delete those privileges first. Click the link with the Set Id to view details.

Create Entity Set

Set Id	Set Name	Description	Pricing Entity Type	Delete
No search conducted.				

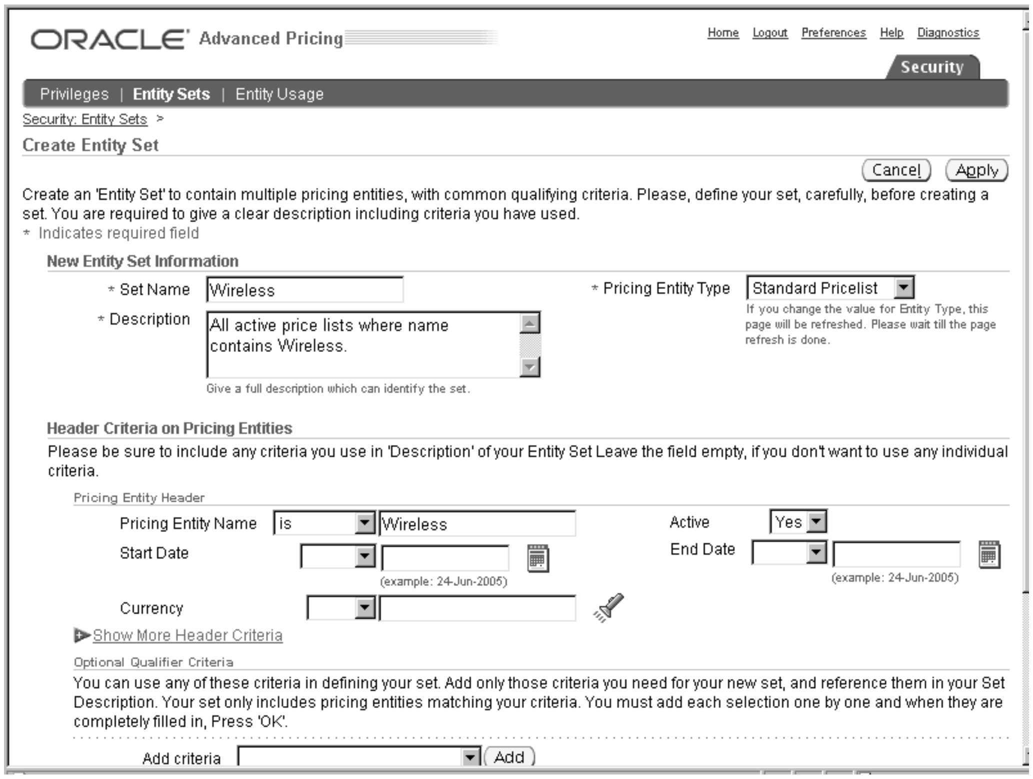
Copyright 2000-2004 Oracle Corporation. All rights reserved.

About this Page

Privacy Statement

2. Click Create Entity Set to display the Create Entity Set page.

Create Entity Set page


ORACLE Advanced Pricing
Home Logout Preferences Help Diagnostics
Security
Privileges | Entity Sets | Entity Usage
Security: Entity Sets >
Cancel Apply

Create Entity Set

Create an 'Entity Set' to contain multiple pricing entities, with common qualifying criteria. Please, define your set, carefully, before creating a set. You are required to give a clear description including criteria you have used.

* Indicates required field

New Entity Set Information

* Set Name	Wireless	* Pricing Entity Type	Standard Pricelist
* Description	All active price lists where name contains Wireless.		

Give a full description which can identify the set.

If you change the value for Entity Type, this page will be refreshed. Please wait till the page refresh is done.

Header Criteria on Pricing Entities

Please be sure to include any criteria you use in 'Description' of your Entity Set Leave the field empty, if you don't want to use any individual criteria.

Pricing Entity Header

Pricing Entity Name	is	Wireless	Active	Yes
Start Date		(example: 24-Jun-2005)	End Date	(example: 24-Jun-2005)
Currency				

Show More Header Criteria

Optional Qualifier Criteria

You can use any of these criteria in defining your set. Add only those criteria you need for your new set, and reference them in your Set Description. Your set only includes pricing entities matching your criteria. You must add each selection one by one and when they are completely filled in, Press 'OK'.

Add criteria [ ] Add

3. In the New Entity Set Information region, enter a Set Name that uniquely identifies the entity set you are creating.

4. Enter a Description that is simple, meaningful, and includes all the criteria selected for this entity set.

   Note: An Entity Set can only contain one unique pricing entity type. For example, Entity Set1 cannot contain both entity type Standard Price List and Modifier. To delete an entity set, you must first revoke all privileges on this set and then delete it.

5. Select one of the following Pricing Entity Types to be included in the entity set: Agreement Pricelist, Modifier, Standard Pricelist. Only one Pricing Entity Type can be included in an entity set.

6. In the Pricing Entity Header region, select the criteria for the pricing entities to be included in the set. The criteria to define the set should be included in the Description for the set.

7. For Pricing Entity Name, select an operator - is, is not, contains, starts with, ends with - 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 named Summer Price List will be included in the entity set. (Assuming Standard Price List was selected as the Pricing Entity Type.)

8. Optionally, select a Start and End Date and Currency as additional criteria.

9. Additionally, in the Optional Qualifier Criteria region, select criteria from Add Criteria field to add additional criteria and click the Add button.

   Add only the criteria needed 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.

10. When you have completed your entries, click Apply.

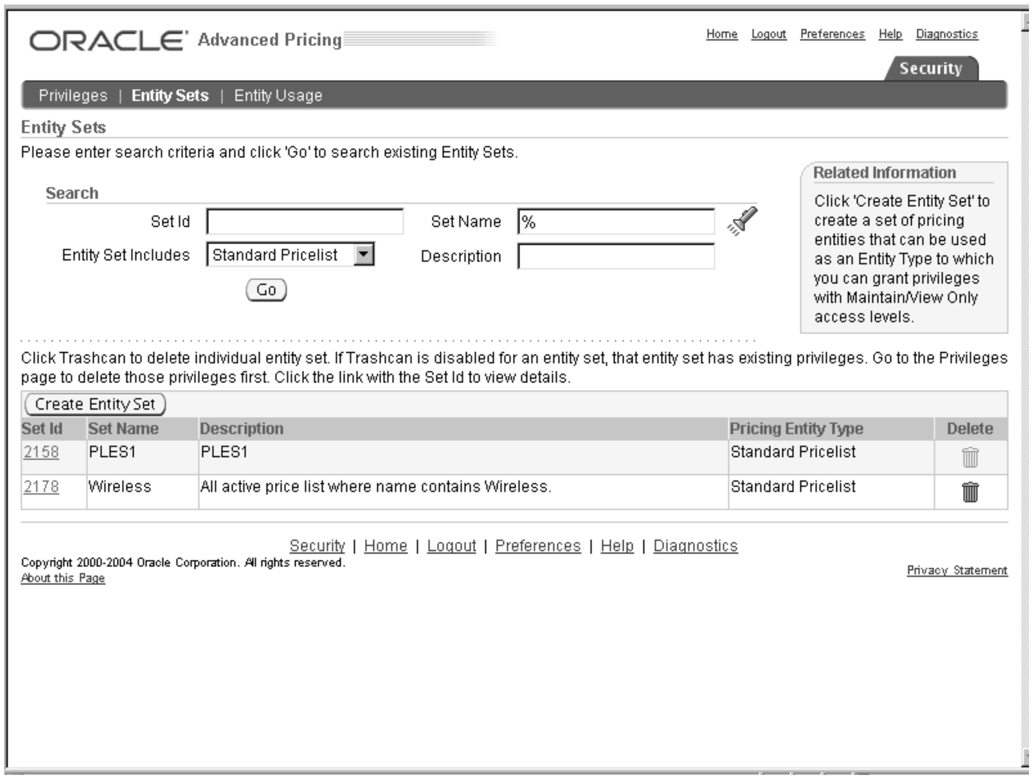
    You can use an Entity Set to restrict maintain access to a few individual users. You can then use the same set to give view-only access privilege to all other users. The set inherits the access level of the privilege for all users who are given access to the set.

    You can experiment with this function and find many ways to make use of it.

To view existing entity sets:

1. Navigate to the Entity Sets page to find an existing entity set.
2. In the Search region, enter the Set Id of the entity set and select the Set Name. Click the Search icon to view the available Set Names. Optionally, to reduce the search results, select or enter additional search criteria:
   - In the Entity Set Includes field, select Standard Price List, Agreement Price List, or Modifier.
   - Enter a Description or partial description of an entity set.
3. To view details about a specific entity set, click the Set Id of an entity set.
4. Click Go to display the search results in the Results: Entity Sets region. The following details display for each entity set retrieved:
   - Set Id: A system assigned value that uniquely identifies the entity set.
   - Set Name: A name for the entity set entered by the user when the entity set is created.
   - Description: Describes the entity set entered by the user when the entity set is created.
   - Pricing Entity Type: Sets contain all the same entity types meeting the selected criteria.
5. The Set Entity Details page displays additional details about the selected entity set including the Set Name, the Description, and the Total of Pricing Entities Included.

Entity Set page


ORACLE Advanced Pricing
Home Logout Preferences Help Diagnostics
Security
Privileges | Entity Sets | Entity Usage

Entity Sets

Please enter search criteria and click 'Go' to search existing Entity Sets.

Search

Set Id		Set Name	%
Entity Set Includes	Standard Pricelist	Description	

Go

Related Information

Click 'Create Entity Set' to create a set of pricing entities that can be used as an Entity Type to which you can grant privileges with Maintain/View Only access levels.

Click Trashcan to delete individual entity set. If Trashcan is disabled for an entity set, that entity set has existing privileges. Go to the Privileges page to delete those privileges first. Click the link with the Set Id to view details.

Create Entity Set

Set Id	Set Name	Description	Pricing Entity Type	Delete
2158	PLES1	PLES1	Standard Pricelist	🗑
2178	Wireless	All active price list where name contains Wireless.	Standard Pricelist	🗑

Security | Home | Logout | Preferences | Help | Diagnostics

Copyright 2000-2004 Oracle Corporation. All rights reserved.

About this Page

Privacy Statement

6. Click the Show all pricing entities included link to view the entities included in the entity set. The listing displays a table with the following information about each entity: Name, Description, Type (Standard Pricelist, Modifier, or Standard Agreement), and Owned By Operating Unit.

   You can either click OK or click Printable Page to display the information in a convenient format that can be printed.

To delete an entity set:

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 up Default Security Profile Options for New Pricing Entities

Security profile options are used to define the default security privileges for 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. The following discussion will assist you in choosing the combination of settings to meet your security policy.

The following profile options are used to assign the default view-only or maintain access privileges to newly created price lists or modifiers:

- 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. 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 pricing users of the operating unit where the price list or modifier was created. This profile option enables you to set maintain privileges at one of the following levels: Global (Default), Operating Unit, Responsibility, User, or None.

Before setting the security profile options and changing the defaulting privilege profiles, complete all security setup requirements.

> **Note:** To change the access privileges for *pre-existing* price lists and modifiers, use the Security Privileges window.

> **Warning:** The profile option QP: Security Control turns pricing on and off. Before turning the profile option QP: Security Control to ON, ensure that you have completed all the set up and implementation steps first, **otherwise, users will be unable to query any price lists or modifiers in the pricing windows**
>
> If you are upgrading or freshly installing the security feature for the first time, ensure you have completed the following steps before you set the default security profile options and turn on security for your installation:
>
> - You have assessed and mapped out the behavior your business requires when a new price list or modifier is created. See Assigning Ownership of Pricing Entities to Operating Units (Entity Usage page), page 5-7 for more information. .
> - Assigned an operating unit owner for existing pricing entities.
> - Granted privileges at all levels based on your security policy and needs.

Security profile options and existing pricing entities

The two security profile options, QP: Security Default Maintain Privilege and QP: Security Default ViewOnly Privilege, do not change the behavior of existing pricing entities. Access to existing pricing entities depends on the privileges already granted by the Oracle Pricing Administrator using the Security Privileges and related pages.

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 will have 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 creating a new Pricing Entity

The following shows behavior by combinations of profile settings when 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
None	None	Entity can be viewed/ updated while being created.	1. The new entity cannot be viewed or updated by anyone.
None	User	Entity can be viewed/ updated while being created.	2. The new entity can be viewed and updated by the user who created it only.
None	Responsibility	Entity can be viewed/ updated while being created.	3. The new entity can be viewed and updated 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 within the same OU as the user who created the entity only.
None	Global	Entity can be viewed/ updated while being created.	5. The new entity can be viewed and updated by all users.

Security Profile ON: Behavior when creating a new Pricing Entity for Combination: Values for User

The following show behavior by combinations of profile settings when 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
User	None	Entity can be viewed/ 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 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/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/she is exempted from the 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 creating a new Pricing Entity for Combination: Values for Responsibility

The following show behavior by combinations of profile settings when 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 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 within the same OU as the user who create it can also update it.
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 creating a new Pricing Entity for Combination: Values for Operating Unit

The following show behavior by combinations of profile settings when 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
Operating Unit	None	Entity can be viewed and maintained by user who created it.	All the users within the same OU 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 within the same OU as the user who created it can view the new entity. And, only the user who created it can update it.
Operating Unit	Responsibility	Entity can be viewed and maintained by user who created it.	All the users within the same OU as the user who created it can view the new entity. And, 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 by all users within the same OU as the user who created the entity only.
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 creating a new Pricing Entity for Combination: Values for Global

The following show behavior by combinations of profile settings when 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. And, 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. And, all the users within the same 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.

The pricing administrator can make changes to Global Usage of price lists and modifiers one by one, or use the Bulk Update Entity Usage function in the Entity Usage page to make changes more quickly.

Warning: It is important that the Oracle Pricing Administrator assigns ownership to all price lists and modifiers prior to upgrading or implementing pricing security. This can be done using the Bulk Update Entity Usage feature in the Entity Usage page. Otherwise, users will be unable to query any price lists or modifiers.

Setting the QP: Security Control profile option to ON

When the profile QP: Security Control is first set to On, a Global check box is dynamically added to the header region of all price lists and modifiers. The Global box indicates if global usage is enabled for the entity. When the Global box is selected, the entity is available across all operating units in your organization.

The Global box is visible to end-users and can be updated (cleared or selected) by users with Maintain access privileges.

You can update the Global box for each price list and modifier window singly, or do bulk updates in the Bulk Update Entity Usage page. See Assigning Ownership of Pricing Entities to Operating Units (Entity Usage page), page 5-7 for more information.

Prior to setting profile QP: Security Control to 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 turning pricing security on, all newly created pricing entities are assigned a unique default operating unit identification that makes the creating operating unit the owner of the pricing entity.

The following table shows the behavior of existing pricing entities when QP: Security Control is set to ON and no pre-security is assigned:

QP: Default View Only Privilege	QP: Default Maintain Privilege	Privileges from Pricing Security Administrator	Behavior
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 selected from the Oracle HTML user interface.
Not applicable	Not applicable	Maintain	Entity can be viewed and updated by the user with Maintain access privileges.

6
Pricing Data Bulk Loader

This chapter covers the following topics:

- Overview of Pricing Data Bulk Loader
- Populating the Interface Table
- Using the Concurrent Program QP: Bulk Import of Price List

Overview of Pricing Data Bulk Loader

The Pricing Data Bulk Loader 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.

Profile Options used in Pricing Data Bulk Loader

The following profile options are used with the 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 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 "hang." The default value is 1000.

Note: For a Basic Pricing installation, only the Basic Pricing data is imported into the system.

List Source Code BSO Not Imported. The price lists with list source code BSO (Blanket Sales Order) are not imported by Pricing Data Bulk Loader.

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 (list_type_code ,name ,description, currency_code, active_flag , process_flag , orig_sys_header_ref, interface_action_code, language, source_lang, rounding_ factor, process_id, CURRENCY_HEADER_ID, process_type, START_DATE_ACT IVE, END_DATE_ACTIVE).

  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).

  When you define the following data types for pricing attribute values and qualifier attribute values, use these formats:

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 associated with the price lists to be imported.
- QP_INTERFACE_PRICING_ATTRIBS: The product and pricing attributes data is captured in this table.

Attributes in the Interface Tables

The following 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.

ORG_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.

ORG_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 updating and deleting, the value the attribute value identifies the product/pricing attribute. Also the uniqueness of this field is checked 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.

Using the Concurrent Program QP: Bulk Import of Price List

The Pricing Data Bulk Loader API is implemented as the concurrent program QP: Bulk Import of Price List. Define the criteria for the price lists to be imported, then run the concurrent program to import the data. You can select from the following criteria:

- Entity: The entity imported using the concurrent program. The allowed value is PRL for Price List.
- Entity Name: The Name of the price list to be imported. This is an optional parameter and if NULL is selected, all the price lists eligible to be processed: for example, PROCESS_STATUS_FLAG = 'P', PROCESS_FLAG = 'Y' and REQUEST_ID = NULL are considered for the import.

When the name is specified, the 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: If specified, the import considers only those records with the PROCESS_ID that match the value you enter here.
- Process Type: If specified, the import will consider only those records with PROCESS_TYPE matching the value entered here.
- Process Parent: Valid values are Yes or No. Indicates if the import program should process a parent record if a child record has errors. This is applicable only to the List Line and Pricing Attributes parent-child relation.
- Number of Threads: Indicates the total number of child processes or threads to start in order to achieve multi-threading for list lines and their child records. An appropriate value must be specified for this parameter based on the hardware configuration at the installation (number of processors).
- Turn Debug On: Valid values are Yes or No. If Yes, detailed debug messages are written to the concurrent program log file. As debug messages add to the performance overhead, specifying No results in quicker completion of the concurrent request.

Validation of Pricing Data Bulk Loader API

Before importing price list data, ensure that the following values are populated:

- Currency
- Name and ORIG_SYS_HEADER_REF should be populated with unique values for price list headers
- Arithmetic Operator and List Line Type code

For a price list line, the following fields cannot be NULL:

- Operand (Price Value)
- price by formula id
- generate using formula id (Static and dynamic formula)

ORIG_SYS_LINE_REF should be populated with a unique value.

Since price lists have no line level qualifiers, line level qualifiers are not populated in the interface table for process. ORIG_SYS_QUALIFIER_REF should be populated with a unique value.

For each line, there should be a record in the Pricing Attribute table with one record with the product attribute data populated and pricing attribute data null. ORIG_ SYS_PRICING_ATTR_REF should be populated with a unique value.

Post Processing

When the upload process is complete, you can review a summary of the process status in the concurrent program output file including:

- Number of records processed
- Number of successful records

- Error messages (if applicable).

If the Turn Debug On parameter was selected, then you can review detailed log information about the errors in the concurrent program log.

The records successfully processed are deleted from the interface tables. Those with errors remain in the table with PROCESS_STATUS_FLAG = NULL with the appropriate REQUEST_ID populated. The header record can be resubmitted for processing by specifying the name in the concurrent program parameter. For other records, after the errors are corrected, the PROCESS_STATUS_FLAG should be updated to P and the REQUEST_ID should be updated to NULL before resubmitting.

You need to periodically purge the QP_INTERFACE_ERRORS table to remove error messages from prior requests, otherwise performance may be impacted.

Sometimes you may need to correct and resubmit a child record after the parent record (and grandparent if applicable) was successfully interfaced and deleted by the previous run of the program. In these cases, you need to repopulate the parent (and grandparent if applicable) record with a PROCESS_STATUS_FLAG= P, and INTERFACE_ACTION_CODE = UPDATE in the interface table.

7 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 item and/or item category prices. 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.

> **Note:** You can only view or update a price list for your pricing transaction entity.
>
> - The profile QP: Pricing Transaction Entity must match the pricing transaction entity of the price list.
> - You can only view or update a price list in your source system. The profile QP: Source System Code must match the source system of the price list. Otherwise, the price list is view-only.
> - Price rounding considerations: The profile option QP: Selling Price Rounding Options affects the rounding of adjustments. For more information on this and other profile options, see: *Oracle Advanced Pricing Implementation Manual*, Profile Options.

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 only using the Oracle Forms-based UI.

Additional 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

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 complete the following tasks:

- Import new price lists.
- Update and delete the price list data.

This API, which consists of a set of interface tables and a concurrent program, can be used as an alternative to the Price List UI and Price List Business Object API to import large volumes of price list data, for example, from a legacy system.

Related Topics

Pricing Data Bulk Loader API, page 6-1

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 only be associated to 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 need to be billed, the usage breaks need to 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.

How break 'From' and 'To' values are converted

You must use whole numbers (not decimal values) for prorated breaks. The following should also be considered when using prorated breaks:

- The Price Breaks should always have a gap of 1 among the ranges.
- The Price Break setup should be in whole numbers.
- The usage passed to the pricing engine should be a whole number.
- The conversion factor between the usage UOM and break UOM could be in decimals.
- After multiplying the setup break range with proration factor, it is truncated to a whole number. After that Gap = 1 rule will apply.

Engine Proration Logic

The first break line is prorated as follows:

- `Truncate (From * Conv.  Factor) Truncate(To * Conv.  Factor)`

```
The remaining break lines are defined as follows:
Gap = BreakLine (n) "From" - BreakLine (n-1) "To"
If Gap = 1
     Prorated BreakLine (n-1) To + 1 Truncate(To * Conv. Factor
Else
     Truncate(BreakLine (n) From * Conv. Factor) Truncate(To * Con
v. Factor )
End If
```

Example 1

The following table displays a break defined by Quarter:

From	To
0	5
6	10
11	20
21	99

Requested Break UOM = Month

Conversion Factor = 1/3

From	To	Calculations
0	1	Applying 1st break line rule 5*1/3 =
2	3	Applying Gap = 1 Rule 10*1/3 = 3.333333333333
4	6	Applying Gap = 1 Rule 20*1/3 = 6.666666666667
7	33	Applying Gap = 1 Rule 99*1/3 = 33

Example 2

The following table displays a break defined as Month:

From	To
0	5
6	10
11	20
21	99

Requested Break UOM = Quarter

Conversion Factor = 3

From	To	Calculations
0	15	Applying 1st break line rule 5*3 = 15
16	30	Applying Gap = 1 Rule 10*3 = 30
31	60	Applying Gap = 1 Rule 20*3 = 60
61	297	Applying Gap = 1 Rule 99*1/3 = 33

8 Modifiers

This chapter covers the following topics:

Overview

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, 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 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

Implementation Decisions

As you analyze and understand a client's business pricing scenario, there are some key implementation decisions to address in order to develop a logical pricing solution. These decisions will be discussed throughout the chapter.

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 a pricing request based meeting the condition of a 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

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 in the product hierarchy.

Key Implementation Decision: How do I assign modifier application methods?

You can select from the available application methods to create different pricing action(s):

- Amount: Creates a fixed price adjustment on each unit for the amount specified in the value field.
- Percentage: Creates a percentage price adjustment on each unit for the percentage 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 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.

Modifier Application Methods: Line Level

Modifier Line type (column below)	Percent Value	Percent Formula	Amount Value	Amount Formula	New Price Value	New Price Formula	Lumpsum Value	Lumpsum Formula
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

Modifier Application Methods: Group of lines Level

Modifier Line type (column below)	Percent Value	Percent Formula	Amount Value	Amount Formula	New Price Value	New Price Formula	Lumpsum Value	Lumpsum Formula
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

Modifier Line type (column below)	Percent Value	Percent Formula	Amount Value	Amount Formula	New Price Value	New Price Formula	Lumpsum Value	Lumpsum Formula
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/Promotional Goods.

For modifier types at the Group of Lines level, be aware of the following considerations:

- You can only select the Volume Type of Item Quantity and Item Amount.
- If no excluded items or item categories are set up, then order lines that qualify for this modifier are considered for group of lines calculation.
- If some excluded items or item categories are set up, order lines 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, 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 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 if 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 as follows:

- 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.

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 on 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 an non-excluded item category, the excluded item category is honored.

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 assess that Item Z is in IC1, the modifier is not applied, although the modifier is in IC2 and IC3.

Unit of Measure (UOM)

There are no UOM conversions for modifiers. The pricing engine evaluates only modifier lines that have matching units-of-measure (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 only consider 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, 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 in the qualifier context. For more information on precedence, see Implementation Methodology, page 3-1 and Precedence and Best Price, page 13-1.

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 regarding incompatibility levels, refer to Precedence and Best Price, page 13-1.

Buckets

Pricing buckets (pricing group sequences) control how your calculation of discounts and other benefits is sequenced. Grouping determines net selling price. Buckets are applied across phases.

Key Implementation Decision: How many bucket levels do I need?

Determine the number of buckets 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. Manual and order level modifiers are always in the null bucket.

Oracle Advanced Pricing does not restrict the number of buckets you can define.

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 not selected then the modifier adjusted price cannot be overridden. The Override box displays 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: 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, price may be completely frozen, or additional modifiers may 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. In order to override the selling price directly on the order line, you must define a manual discount or a manual surcharge. Remember the following 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 defined in the profile option QP: Return Manual Discounts. If this is set to Y, then all manual discounts are returned and all automatic discounts 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) 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.
- In the manual adjustments LOV for the unit selling price on the sales order line ONLY line and line group manual adjustments are displayed.
- 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 there are no applicable manual adjustments for a line, a note displays that indicates there are no applicable discounts 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

In order 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 only be defined 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 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.

> **Warning:** Do not change the Calculate Price Flag for a promotional "free good" item to Calculate when viewing the item in the Adjustments window (Sales Order window > Actions > View Adjustments).

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 line level term substitution.

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 only recognizes the original item for additional modifiers; it never recognizes the upgraded item.

Price Breaks

Price breaks can be defined for any pricing attribute in the pricing context volume. The modifier level line or group of lines determines how the pricing engine handles these price breaks.

- For manual price breaks only point price breaks are allowed.
- When you split a line the selling price changes if volume falls into a different price bracket. At that point, the outcome will depend 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.

Point Price Break

Consider the following example discount rules:

- Item quantity ordered 1-10: 1% discount
- Item quantity ordered 11-50: 2% discount
- Item quantity ordered 51-999: 5% discount

If this is a line level price break and the ordered quantity is 55, then a 5% 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 via accumulation attributes. See Using Accumulated Range Breaks, page 8-14 for more information on setting up and using accumulated range breaks.

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 1PC 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 is 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 stand-alone 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 set up in the Get region can only be defined 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 there are order lines where the Calculate Price flag is set to No, then additional order level modifiers are not applied.

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.

Coupon Issue

Two modifier lines must be set up when defining a coupon issue in the modifiers window. The price adjustment or benefit is linked to the coupon issue. When 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 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. If the order is not eligible for the promotion, the engine will return an error message.

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
- Price break
- Promotional good (additional buy items do not recur)
- Coupon issue

An example of a recurring coupon issue is: for every 5 items ordered, get a coupon for 10% off a future order. If you order 15 items, then you receive 3 coupons.

Accruals

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. Reference to accruals may be made by selecting the Accrual flag.

Remember the following when setting up accrual discounts:

- Select the Accrual box in the Discounts/Charges tab.
- Benefit quantity and benefit UOM are of the benefit you wish to accrue.
- Expiration date specifies when the accrued transactions expire (optional).
- For the fields expiration period and expiration type, 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. An example of a conversion rate is: 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 order line selling price 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 may not be passed to Oracle Accounts Receivable without a work-around 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.
- There is no interface 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 may 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, 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 evaluating a break.
- "Regular" (non-accumulated) price breaks: Regular breaks are evaluated starting from zero.

Calling applications, such as Oracle Order Management, pass accumulation values to the pricing engine via 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 on accumulated range break features.

The following methods are used to pass the accumulation value to the pricing engine:

- Attribute Mapping: 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 only used for accumulated range breaks.

Related Topics

Oracle Advanced Pricing Implementation Manual, Setting up Runtime Sourcing for Accumulated Range Breaks

Setting up Runtime Sourcing for Accumulated Range Breaks

Unlike attribute mapping, runtime sourcing can only be used for accumulated range breaks. 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).

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 outlines the general steps for implementing and using runtime sourcing:

Pricing Steps:

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 package QP_RUNTIME_SOURCE is located in the file QPXRSRCB.pls. Although the specification for the function Get_numeric_attribute_value is provided there, you must write the corresponding body which takes 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 assigned to the order, if supported by the calling application. This value may be null.
- p_order_line_id: the ID assigned to the request line of the order.
- p_price_effective_date: price effective date 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 found in package QP_FORMULA_PRICE_CALC_PVT.

The p_accum_rec record contains three fields, but may be expanded for any future requirements:

- p_request_type_code: the request type code 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 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 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, since those operations should be done within the calling application itself.

Sample Runtime Sourcing Code

The following is an example of what could be written 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 'PRICING_ATTRIBUTE31' and order type as context 'ORDER', attribute '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 t
hat is
     -- passed on the request line.  We only use the p_req_line_at
trs_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 v
alue
     -- 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 there are, this function (utilizing the way the data are stored in the table) will return a different and correct accumulation value for each pair 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.

9 Archiving and Purging Pricing Entities

This chapter covers the following topics:

- Overview of Archiving and Purging 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.

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.

Related Topics

Oracle Advanced Pricing User's Guide, Archiving and Purging Pricing Entities

10

Multi-Currency Price Lists and Agreements

This chapter covers the following topics:

Overview

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, there is one base currency price list and an attached Currency Conversion list defining conversion factors and rules for converting prices.

It is very important 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 Multi-Currency 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 there is no relationship between prices defined.

Multiple Currency Price Lists

Multiple Currency price lists enable 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 pricing 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 price is converted from the base currency and calculates the ordering currency based upon the established conversion rules.

Multi-currency 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, multi-currency 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

It is possible to define multiple conversion criteria based upon certain attributes for the same base currency. For further information on attributes, see: Attribute Management chapters in the *Advanced Pricing Implementation Guide* 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**: The 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.

Fresh Install using Multiple Currency Price List

Profile: QP Multi-Currency Installed

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 defined in its own base currency (non-multi-currency enabled).
- **Yes**: Enables you to attach a multi-currency 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 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 Lists > Multi-Currency Conversion setup > Create a base currency multi-currency conversion list
3. 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 it is necessary to run the concurrent program *Update Price Lists with Multi-Currency Conversion Criteria*. This will convert all existing price list and agreement forms as Multi-Currency and activate 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: Multi-Currency Installed
The profile QP: Multi-Currency Installed controls which type of price lists you will use. Set this profile to Yes. This will allow 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 Multi-Currency Conversion Criteria
To start using the Multi-Currency 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 since without this the pricing engine will not be able to use the current price lists as Multi-Currency 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 running the concurrent program, all price lists and agreement will be multi-currency enabled. You can now add additional To Currencies and their conversions.

Implementation Decisions for Creating Multi-Currency Conversion Lists

Prior to using Multi-Currency price lists, the following needs to be considered:

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 multi-currency price list, to which you attach the conversion list. Otherwise you will need to define multiple conversion criteria based upon attribute 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	Conversion Type	Fixed Value	Attribute Code	Attribute Value	Start Date	End Date	Precedence
To-Currency	CAD	Fixed	2	--	--	01/01/02	112/31/02	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	--
To-Currency	Euro	Spot	--	--	--	01/01/02	12/31/02	--

Deactivating Price lists

You should deactivate a price list once you have merged it successfully with a multi-currency conversion list.

Rounding

There are three rounding profiles that affect rounding in Multi Currency price lists. These profiles are discussed in the Profile Options section of the Advanced Pricing Implementation Guide.

- QP: Unit Price Precision Type: 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 may 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*.

There are some limitations in using formulas in multiple currency price lists.

- **Base Markup Formula**: The value 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 returned by the Formula selected, is the conversion rate.

  A formula with a component type as 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 as 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, 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, end date and having different Attribute Value. This occurs because when more than one attribute is passed from the calling application, the one 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	Precedence
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 defined for Item Number= AS54888 is used by the pricing engine 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 14-4

Precedence and Best Price, page 13-1

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 Multi-Currency 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 whose 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.

11
Unit of Measure

This chapter covers the following topics:

- Overview

Overview

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 units of measure if you use seeded qualifiers for line volume or line weight.

Defining Unit of Measure

You must define unit of measure in order to use Oracle Advanced Pricing. UOM is used 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 unit of measure is not necessary if you have already installed and setup Oracle Inventory, or if you performed this common applications setup for another Oracle product. Refer to *Oracle Inventory User's Guide, Defining Unit of Measure* for more information.

Defining Unit of Measure 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 need to be defined in the same unit of measure as on the price list.

> **Note:** This step is not necessary if you have already installed and setup Oracle Inventory or performed this common applications setup for another Oracle product. Refer to *Oracle Inventory User's Guide,* Defining Unit of Measure Classes, for more information.

Pricing Actions: Primary UOM/Pricing UOM

The primary UOM feature is used to price an item in different units of measure without having to explicitly define prices for each UOM. The pricing UOM is the unit of measure in which the pricing engine prices the order line.The pricing quantity is the order quantity expressed in the pricing unit of measure. 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 1 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 will default 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 critical to UOM (unit of measure) in Oracle Advanced Pricing:

- **QP: Accrual UOM Class**

  Specifies the UOM class used to define accrual units of measure. 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 unit of measure 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 units of measure currently defined to Oracle
- Visible and can be updated at the site and application levels

- **QP: Line Weight UOM Code**

  Specifies the unit of measure 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 which 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, 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

12
Multiple Organizations

This chapter covers the following topics:

- Overview

Overview

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.

Multiple Organizations, Pricing Security, and Pricing Actions

Oracle Advanced Pricing does not use the multiple organization structure of Oracle Applications. However, pricing security features are provided that enable you to:

- Assign or reassign Operating Unit ownership to price lists and modifiers.
- Control which operating units can use pricing entities when pricing transactions.

Using Qualifiers to Create Multiple Organizations

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 sourcing 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, only those price lists and modifiers for the specified operating unit are eligible.

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 indicates the Oracle Manufacturing organizations that items are validated and viewed against when entering price lists or modifiers. This profile value should be set to the master organization.

Modifier Type: Item Upgrade

When you define item relationships for item upgrade type of modifiers, the item relationship must belong to the same item organization as is specified in QP: Item Validation Organization.

Cross Order Volume Loader

The concurrent program accumulates all cross order totals for an operating unit, but it does not summarize across operating units. The engine only considers orders that a customer places with this sales organization, and 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, it 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.

Organization 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 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

Price List LOV in Oracle Order Management

Because price lists are not operating unit specific, price lists displayed in the price list LOV (list of values) at the sales order header and line level are not specific to the operating unit to which OM is set. All operating units' price lists are visible. This also applies when a qualifier attribute is attached to a price list. The pricing engine validates a qualifier of an operating unit only once the sales order or line has been sent to pricing.

13

Precedence and Best Price

This chapter covers the following topics:

- Overview
- Default Precedence Numbers
- Matched Qualifiers for Modifiers/Price Lists
- Price List Incompatibility Resolution
- Modifier Incompatibility Resolution
- Incompatibility Resolution Examples

Overview

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 on phases.

Default Precedence Numbers

When resolving incompatibilities by precedence, the pricing engine evaluates the precedence numbers 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 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 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 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 only uses matched qualifiers when ordering the qualifier attributes to determine which has the highest priority.

The following table lists several of the seeded qualifier contexts and qualifier attributes with the attribute number for each attribute that are on a specific 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. Only the value for order type is used when evaluating 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 described in the precedence resolution section of this chapter.

If two or more price list lines are matched and have the same precedence, the engine selects the price list line with the higher pricing attribute count. If the engine cannot resolve this incompatibility between price lists, it will return an error message to the calling application that it cannot apply a price and will return the names of the price lists that this incompatibility resides.

Modifier Incompatibility Resolution

The incompatibility resolution codes for phases 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 attached chart shows which column the engine uses to evaluate 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 of modifier such as Comparison Value. If not provided then best price is set to zero.
Item upgrade	Estimated discount value of modifier such as Comparison Value. If not provided then best price is set to zero.
Coupon issue	Estimated discount value of modifier such as Comparison Value on the coupon issue line. If not provided then best price is set to zero.
Other Item discount	Not included - always valued at zero
Promotional goods	Not included - always valued at zero
Freight Charges	Same as discount and surcharges.

The pricing engine does not consider modifiers with formulas for best price calculation.

Best Price Calculation Ignores Buckets

The 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	Incompatibility	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.

Calculating Common Benefit Percent

The 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, there is no stated value that the engine can use to evaluate best price, thus the engine uses the value the user inputs 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. It is the value that the engine uses to calculate the common benefit percent.

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 as follows (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 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.

Modifier Incompatibility Setup window

The Modifier Incompatibility Setup window enables you to query modifiers by phase sequence, phase name and incompatibility group name. Because modifiers are incompatible within a phase, the query is phase-specific; this makes is simple to identify which resolve method the engine uses to process incompatibilities. You can make changes to the incompatibility level of a modifier directly on this window. The window displays various modifier line details that provide a cross modifier view of modifiers that are incompatible with each other.

The product precedence field displays for each modifier line that can be used in identifying discounts that have a higher precedence. However, this view is incomplete; it does not display qualifier attribute precedence. These attributes are evaluated by the engine during resolution. This window contains a direct link to display the modifier detail where you can review and revise modifier details.

Incompatibility Resolution Examples

The following provides examples of 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, 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 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	Incompatibility 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
Header level adjustments	Order amount discount	Level 1	Customer name Item number	260 220	No
Header level adjustments	Independence Day promotion	Level 1	Sales channel All items	320 315	No
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.

14
Attribute Management

This chapter covers the following topics:

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/attribute properties
- Disable existing attributes.

Creating new contexts and attributes extends Oracle Advanced Pricing's ability 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 product attributes are used to define customer or product hierarchies:

- A *product* hierarchy element defines product pricing characteristics such as a product item, item number, category, or brand.
- A *customer* hierarchy element 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 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 could be set up to 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 from Release 11*i*

In the R11*i* initial release, the qualifiers and pricing attributes were created through the Flexfields and Value Set windows using the Oracle Flexfield Setup features. Now, contexts and attributes are created and maintained in the Attribute Management windows in Oracle Advanced Pricing, and attribute mapping windows for provided for mapping attributes.

You must create all new attributes using the new Attribute Manager setup windows. Oracle Pricing Manager Responsibility > Setup > Attribute Management > Context and Attributes.

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:

Step 1. Creating Context and Attributes for Pricing Setup windows

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.

Step 2. Linking Attributes to a Pricing Transaction Entity

Once the context-attribute grouping is created, you can link it to a specific Pricing Transaction Entity (PTE). A Pricing Transaction Entity consists of a group of applications that point to the same setup data and attributes, and includes Request Types and Source Systems:

- Source System: the application(s) that captures the pricing setup data.
- Request Type: identifies the type of transaction that is being priced.

For a given PTE, the context/attribute combination can be used within a pricing setup. Each PTE has its own unique combination of attributes.

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.

> **Note:** The profile QP: Pricing Transaction Entity indicates the current Pricing Transaction Entity in use. It can be updated at the user level. Only those contexts and attributes assigned to the current Pricing Transaction Entity are available from the list fields in the setup windows. Querying setup data 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.

Step 3. Running the Build Attribute Mapping Rules program

This applies to attributes with the Attribute Mapping Method of Attribute Mapping only: whenever you create/update these attributes or change attribute mapping rules, you should run the Build Attribute Mapping Rules program.

Related Topics

Creating Context and Attributes for Pricing Setup windows, page 14-4

Linking Attributes to a Pricing Transaction Entity, page 14-9

Creating a New Pricing Transaction Entity, page 14-20

Running the Build Attribute Mapping Rules program (Attribute Mapping only), page 14-17

Terminology for Attribute Management

Pricing Transaction Entity

A new entity has been created called a Pricing Transaction Entity. A Pricing Transaction Entity (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.

Creating Context and Attributes to be used for Pricing Setup windows

The following sections describe how to set up contexts and attributes. Contexts are a group of related pricing elements that can have attributes linked to them. Once the context is defined, you can create its attributes to define specific values which define pricing rules. For example, a Customer context can include pricing attributes such as Customer Name or Customer Class.

Implementation Flow for Creating a New Attribute

1. Verify the request types and source systems for a given PTE (Pricing Transaction Entity). Otherwise create a new pricing Transaction Entity in QP Lookups.
2. Create a new context.
3. Define the attribute for that context to be used for Pricing Setup.
4. Link the attribute to a Pricing Transaction Entity (PTE).
5. Define properties of the attribute for the given PTE.
6. For mapped attributes, define Attribute Mapping Rules.
7. Use the attribute in a valid pricing setup.
8. Running the Build Attribute Mapping Rules Program.
9. Enter an order.
10. Verify the mapped and user enter attributes have been correctly passed to the pricing engine from the Pricing Engine Request Viewer.

Creating Contexts:

In Attribute Management, you can set up the following contexts:

- **Qualifier Contexts** and its attributes: 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** and its attributes: The Product Context refers to how the 'items' used in price lists and modifiers are defined. They are defined using the Product Context: Item. Oracle Advanced Pricing supports price and modifier definitions at the following item levels (these levels are considered "attributes" of the context):

  - All Items
  - Item Number
  - Item Category

  For example, if the attribute is All Items, then the pricing engine would look at all items. Oracle supports only **one** Product Context which is ITEM. If you wanted to create an item hierarchy not seeded from Oracle, you could set up additional attributes in the Product Context: Item. For example, Product Context: Items / Product Attribute: 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** (associated with pricing attributes): Define eligibility for a price list line or modifier. They can be used for a price list line, as a formula component, or in modifiers.

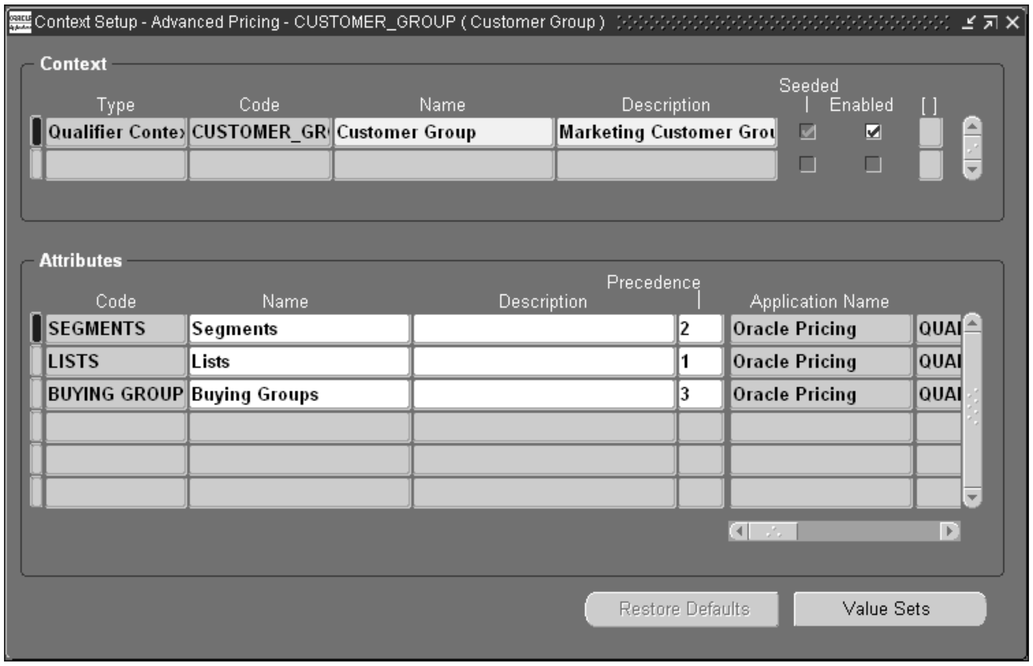
For example, a product context such as Item can be further categorized by attributes such as Item Name and Item Number while a qualifier context such as Customer can consist of attributes such Customer Name and Customer ID.

Context Type	Context Name	Attribute
Qualifier Context	Customer	Customer Name
Product Context	Item	Joe's Items
Pricing Context	Color	Blue

To create new contexts

1. Navigate to the Context Setup window.

Context Setup window


Context Setup - Advanced Pricing - CUSTOMER_GROUP (Customer Group)

Context

Type	Code	Name	Description	Seeded	Enabled	[ ]
Qualifier Conte›	CUSTOMER_GR	Customer Group	Marketing Customer Grou	✓	✓	

Attributes

Code	Name	Description	Precedence	Application Name	
SEGMENTS	Segments		2	Oracle Pricing	QUAI
LISTS	Lists		1	Oracle Pricing	QUAI
BUYING GROUP	Buying Groups		3	Oracle Pricing	QUAI

2. Select the Context Type you want to create such as a Qualifier, Pricing, or Product context.
3. Enter a Code which is a short name for the context. Once created, it cannot be updated.
4. 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 pricing context fields in Price List and Modifier windows. Since new contexts can be introduced by different applications other than Oracle Pricing, entering a context Description is informative.

   The Seeded box is selected automatically if the context is a seeded value. It remains selected even if you overwrite a seeded context.
5. Select the Enabled box to make this context available for pricing setup windows. If not selected, the context is disabled and does not display in the context field on the pricing setup windows and all the attributes defined under this context will also be unavailable for setup.
6. When you have completed your entries, enter the attributes for this context in the Attributes region.

Creating Attributes:

Once the context is defined, you can create its attributes. Attributes determine the specific values in which you define pricing rules. For example, customer hierarchy can include pricing characteristics such as Customer Name = Customer ABC.

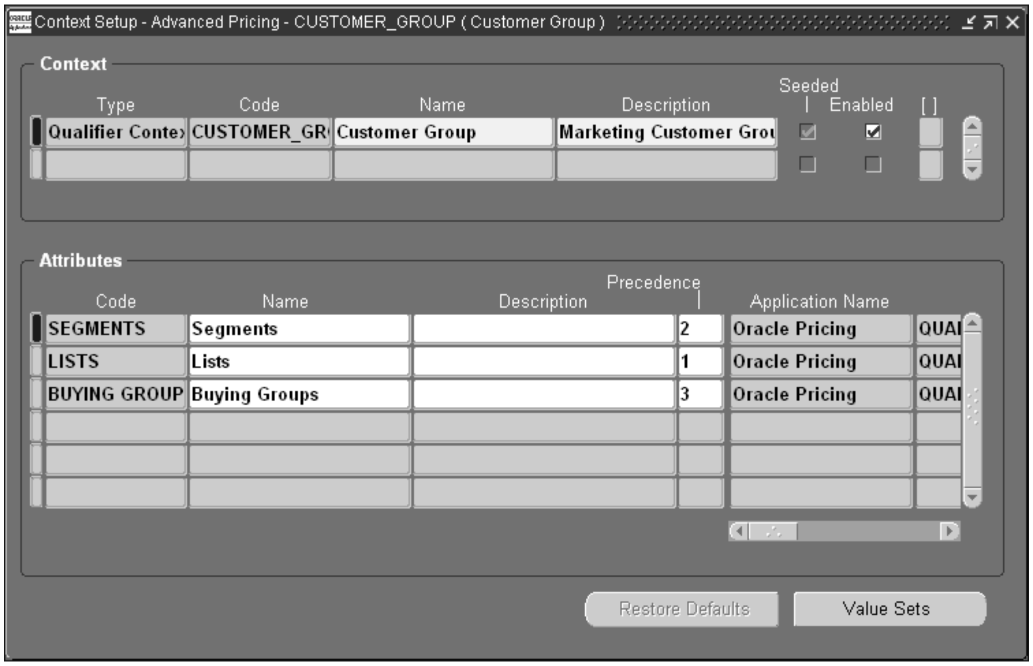
The pricing engine evaluates attributes during engine run time to determine which price lists and modifiers are eligible for the transaction. Attribute values can be derived at the order level, line level, or for both order and line levels.

With the Attribute Management feature, you can define attributes for a given context, determine how they are sourced, decide which attributes can be selected from a list in the Pricing Setup windows, and determine which ones are used in promotional limits.

To create new attributes

1. Navigate to the Context Setup window and select the context to which you want to link attributes.

Context Setup window



Context Setup - Advanced Pricing - CUSTOMER_GROUP (Customer Group)

Context

Type	Code	Name	Description	Seeded	Enabled
Qualifier Conte›	CUSTOMER_GR‹	Customer Group	Marketing Customer Grou	✓	✓

Attributes

Code	Name	Description	Precedence	Application Name	
SEGMENTS	Segments		2	Oracle Pricing	QUAI
LISTS	Lists		1	Oracle Pricing	QUAI
BUYING GROUP	Buying Groups		3	Oracle Pricing	QUAI

2. In the Attributes region, enter the Code. A Code is a unique short name for a given attribute and is used internally. Once created, it cannot be updated.
3. 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.
4. You can update seeded names and descriptions.

   > **Note:** If the original seeded values are changed, they can be restored using the Restore Defaults button. For more information, see Restoring Seeded Data Using the Restore Defaults button, page 14-24.

5. Enter a numeric Precedence value which decides the processing sequence of qualifier/pricing attributes. The precedence is restricted to a maximum of 3 digits (any number between 1 and 999).

   For example, if two incompatible discounts qualify for the same item, then the discount with the higher precedence is given. Precedence numbers in the series of 5's and 10's are reserved for seeded qualifiers/pricing attributes and should not be used.
6. Enter the Application Name that created this attribute. If an Application Name is not entered, the system defaults Oracle Pricing as the creator of the attribute.

7. Select a Column Mapped value to which an attribute will be mapped. The list displays the names of the unused columns only.

   Columns QUALIFIER_ATTRIBUTE1 through QUALIFIER_ATTRIBUTE30 are reserved for seeded qualifier attributes. Columns 1-30 are available only for user Datamerge. The Pricing Manager user's list has unused columns between 31-100.

   Columns1 to 100 are available and it is recommended to use pricing attributes 1 to 30 as user-entered pricing attributes.

8. Select a value from the Value Set field to define a domain of valid values for an attribute. The Datatype value indicates if the Value Set is numeric (Number) or alphabetic (Char).

   Alternately, 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:** If you want to replace a Value Set of an attribute, the new Value Set must be the same datatype as the old one.

   The Seeded box is selected automatically if the context is a seeded value. It remains selected even if you overwrite a seeded context.

9. Select Required to make the attribute a required value in pricing windows. If the attribute is made Required, then 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.

   Once you have created the context and its attributes, you can link an attribute grouping to a specific Pricing Transaction Entity (PTE).

Related Topics

Creating Attributes, page 14-7

Linking Attributes to a Pricing Transaction Entity, page 14-9

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 already used in pricing setup windows cannot be deleted.

Linking Attributes to a Pricing Transaction Entity

A Pricing Transaction Entity 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. The Request Type identifies the type of transaction that is being priced.

Once you have created a context and its attributes, you link the context/attributes grouping to a specific Pricing Transaction Entity (PTE). For a given PTE, this combination becomes available in the pricing setup windows. Each PTE has its own unique combination of attributes.

Oracle Advanced Pricing provides the following seeded PTEs:

Pricing Transaction Entity	**Code**
Complex MRO	**CMRO**
Demand Planning	DEMAND
Intercompany Transaction	INTCOM
Logistics	LOGSTX
Order Fulfillment	ORDFUL
Procurement	PO

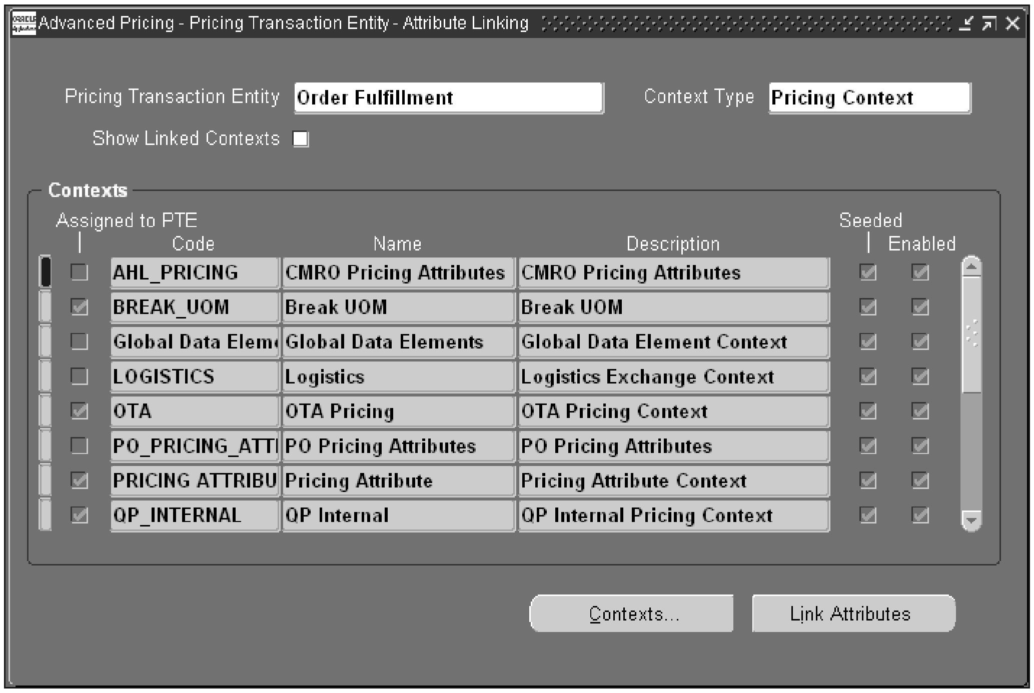
The following tables display the seeded Request Types and Source Systems for each PTE:

Pricing Transaction Entity	**Source Systems**	**Request Types**
Complex MRO	AHL	AHL
Demand Planning	QP	MSD
Intercompany Transaction	INV, QP	IC
Logistics	FTE	FTE
Order Fulfillment	AMS, ASO, OKC, OKS, QP	ASO, OKC, OKS, ONT
Procurement	PO	PO

To link attributes to a Pricing Transaction Entity:

1. Navigate to the Advanced Pricing - Pricing Transaction Entity-Attribute Linking window.

Pricing Transaction Entity-Attribute Linking window



Advanced Pricing - Pricing Transaction Entity - Attribute Linking

Pricing Transaction Entity: Order Fulfillment
Context Type: Pricing Context
Show Linked Contexts: ☐

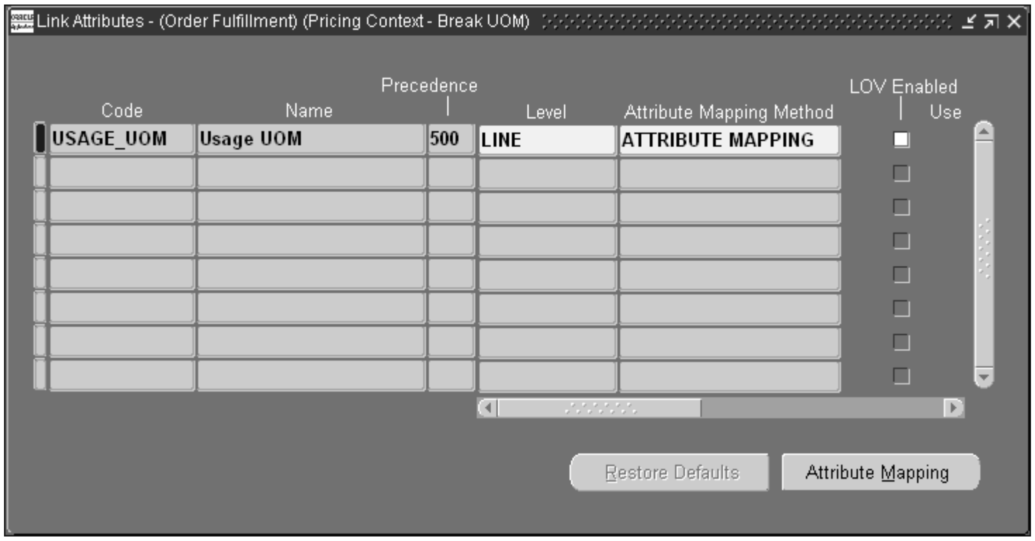
Contexts

Assigned to PTE	Code	Name	Description	Seeded	Enabled
☐	AHL_PRICING	CMRO Pricing Attributes	CMRO Pricing Attributes	✓	✓
✓	BREAK_UOM	Break UOM	Break UOM	✓	✓
☐	Global Data Elem	Global Data Elements	Global Data Element Context	✓	✓
☐	LOGISTICS	Logistics	Logistics Exchange Context	✓	✓
✓	OTA	OTA Pricing	OTA Pricing Context	✓	✓
☐	PO_PRICING_ATTI	PO Pricing Attributes	PO Pricing Attributes	✓	✓
✓	PRICING ATTRIBU	Pricing Attribute	Pricing Attribute Context	✓	✓
✓	QP_INTERNAL	QP Internal	QP Internal Pricing Context	✓	✓

2. Select a Pricing Transaction Entity. Alternately, you can search for a Pricing Transaction Entity by clicking the Find icon.
3. Select the Context Type such as Pricing Context, Product Context, or Qualifier Context.
4. Select the Show Linked Contexts box to display only those contexts assigned to the selected Pricing Transaction Entity.

   The context(s) matching the criteria for the selected PTE and Context Type display in the Contexts region.
5. 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/selected for that context in the given PTE.
6. Click Link Attributes to display the Link Attributes window.

Link Attributes window



Link Attributes - (Order Fulfillment) (Pricing Context - Break UOM)
Restore Defaults
Attribute Mapping

Code	Name	Precedence	Level	Attribute Mapping Method	LOV Enabled	Use
USAGE_UOM	Usage UOM	500	LINE	ATTRIBUTE MAPPING	☐	
					☐	
					☐	
					☐	
					☐	
					☐	
					☐	
					☐	

7. Select a Code (short name) for the attribute.

8. Select an attribute Level: ORDER, LINE, or BOTH (Order and Line levels).

   For example, for an attribute Agreement Id, if the level is ORDER it means that 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/qualifiers in the Pricing Setup list of values.

9. Select an Attribute Mapping Method to define how you want the attribute mapped:

   - ATTRIBUTE MAPPING: Attribute mapping is required. After you complete the remaining entries, 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. See Using Custom Sourced Attributes, page 14-23 for more information.

   - RUNTIME SOURCED: Sourced in the pricing engine by a Custom API. This 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 Attribute Mapping setup). See Setting up Runtime Sourcing for Accumulated Range Breaks, page 8-15 for more information on using runtime sourcing for accumulation attributes.

   - 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:

     a. Navigate to the Flexfields window.

b. Select the Freeze Flexfield Definition box.

c. 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/re-login 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 display as a mandatory field (in yellow) in the Pricing Attributes flexfield window in the Sales Order pad, until a responsibility switch/re-login 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.

10. Select LOV Enabled to display the attribute in the list of values of modifier, qualifier, price list, and formula windows. If not selected, the attributes will not be available in these windows. However, existing modifiers defined using that attribute will behave 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 will be raised.
    - 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 untouched whenever data upgrade scripts are run.

11. Select Use in Limits to enable this attribute to be selected from the Attribute list of values in the Limits setup windows (including the Other Attributes window).

12. If you selected ATTRIBUTE MAPPING as the Attribute Mapping Method, select the Attribute Mapping Enabled box to enable the attribute to be mapped successfully. This enables the concurrent program Build_context API to generate code for this attribute.

    If the box is cleared, the attribute will not be mapped. This box is cleared for all seeded attributes to avoid the mapping of unwanted attributes.

    The Column Mapped value is a default value.

13. The Used in Setup box indicates if 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:

If the setting for QP: Build Attributes Mapping Options is:	Then...
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.
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.

The Attribute Mapping Status box is selected (or cleared) automatically by the concurrent program which generates the Build_Contexts API for mapped attributes. The Attribute Mapping Status box is cleared in the following cases:

- For new attributes until the concurrent program is run after defining the attribute mapping for it.
- 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 non-navigable.

14. 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. See Linking Attributes to a Pricing Transaction Entity, page 14-9.

    > **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 you created can be viewed in the pricing setup windows.

15. Optionally, click Contexts to add a new context, update, enable, or view an existing context.

To link attributes to a Pricing Transaction Entity (Attribute Mapping only):

If you selected ATTRIBUTE MAPPING as the Attribute Mapping Method for an attribute, you need to map the attribute before it can be used by the pricing engine. Otherwise, the pricing engine will not know where to find/derive attribute values when you create modifiers and price lists, or attach qualifiers or product hierarchy elements.

> **Note:** If you selected USER ENTERED or CUSTOM SOURCED as the Attribute Mapping Method, then no further attribute mapping is required. The attribute is linked to a Pricing Transaction Entity. See Linking Attributes to a Pricing Transaction Entity (PTE), page 14-9for more information.

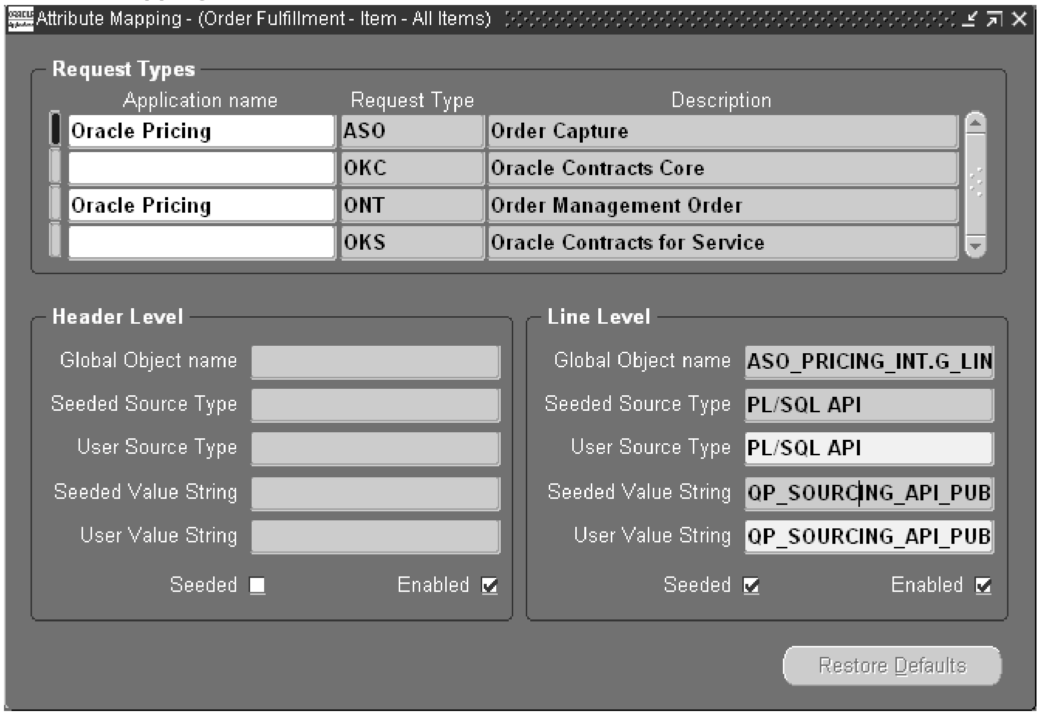
Attribute mapping provides additional flexibility for the pricing engine. For example, Customer Region can be derived from a OE_ORDER_PUB.G_HDR.sold_to_org_id for Order Management; however this same element may have to be derived from ASO_PRICING_INT.G_HEADER_REC.cust_account_id for Order Capture.

This section describes how to complete attribute mapping for attributes with Attribute Mapping selected as the Attribute Mapping Method.

Complete the initial setup steps for the attribute. See Linking Attributes to a Pricing Transaction Entity (PTE), page 14-9 for more information.

1. In the Link Attributes window, click the Attribute Mapping button to display the Attribute Mapping window.

Attribute Mapping window



Attribute Mapping - (Order Fulfillment - Item - All Items)

Request Types

Application name	Request Type	Description
Oracle Pricing	ASO	Order Capture
	OKC	Oracle Contracts Core
Oracle Pricing	ONT	Order Management Order
	OKS	Oracle Contracts for Service

Header Level

Global Object name	
Seeded Source Type	
User Source Type	
Seeded Value String	
User Value String	
Seeded	☐
Enabled	✓

Line Level

Global Object name	ASO_PRICING_INT.G_LIN
Seeded Source Type	PL/SQL API
User Source Type	PL/SQL API
Seeded Value String	QP_SOURCING_API_PUB
User Value String	QP_SOURCING_API_PUB
Seeded	✓
Enabled	✓

2. In the Request types region, select the Application 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.

   Complete the following information in the Header Level region:

   > **Note:** The fields in the Header Level 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.

   - The 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.
   - The Seeded Source Type displays 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.

3. Select a User Source Type:

   - PL/SQL API: Attribute can be sourced directly from a global structure defined for the given source system. The following flexfield window will pop up. For Oracle Order Management, all the record structures are defined in the package OE_ORDER_PUB. For example, if you want to use 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. 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 only be a table of VARCHAR2s: for example, to get the inventory categories for an item. The flexfield window above displays. 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 corresponding User Value String.

4. Enter a User Value String.

   The Seeded box indicates if the mapping rule is seeded or not.

5. Select the Enabled box to enable attribute mapping rules for various Request types for qualifier, product, or pricing attributes. Alternately, 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.

6. In the Line Level region, you can complete your entries if the attribute level (defined in 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.

   Once 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, page 14-17

Attribute Levels and Restrictions in Pricing Setup windows

The following describes the attributes (and associated restrictions) that display in each of the pricing component setup list of values based on the attribute level 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 display 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 in 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 applies 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 No. All the qualifiers within a Qualifier Grouping No must have the attributes with the level either as Order or Both or as Line or Both.

Formula Lines and Factors List

Since a formula can be applied either at order level or at line level, it is not possible at the definition time to restrict the attributes appearing in the lists based on the attribute level. There will be no level-based restrictions in the list of values.

Pricing Attributes and Flexfields tables

A context of type Pricing Attribute will also be created as Flexfield Pricing Contexts in the Flexfield, once it is created using the Contexts and Attributes window. All the above definitions of contexts and attributes are stored in QP tables. However, the same data will be stored in flexfield tables as well, if the context type is Pricing Attribute. This occurs because Order Management (OM) uses flexfield structure for pricing attributes. Hence for pricing attributes to display on OM's windows, these definitions need to be in the flexfield tables as well.

However, Pricing Attribute setup must always be done in the Context and Attributes setup menu. If user does a pricing attribute setup in flexfield tables, the setup data will not be available in the QP tables as data is not copied from flexfield tables to QP tables. So even though the pricing attribute displays in OM windows, it will never be picked up by the engine.

A context and attributes of type Pricing Attribute will get updated and deleted in the flexfield, if updated and deleted in the Context and Attributes window if it meets the deletion criteria.

Contexts or Attributes and the Pricing Attribute type in Flexfield tables:

A context of the type "Pricing Attribute" will also be created as Flexfield Pricing Contexts in the Flexfield, once it is created using the Contexts and Attributes window.

1. A context of type Pricing Attribute will also get 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 his 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 Qualifier Context.

Running the Build Attribute Mapping Rules program (Attribute Mapping only)

This section this applies ONLY to attributes with the following setup criteria:

- Attribute Mapping Method is Attribute Mapping.
- Attribute Mapping Enabled box is selected.

Note: You do not need to run the Build Attribute Mapping Rules program for attributes where 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 box is selected) every time you:

- Create or update a new attribute (one that has not been used by any other modifier, price list, 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 a pricing setups, it is recommended that you run the Build Attribute Mapping Rules program for better performance.

> **Warning:** Since 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/Pricing Attributes 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 displays a status message to advise if the program completed successfully or not. 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. This starts the static generated 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 Pricing Transaction Entity helps narrow the data that the search engine evaluates. The search engine evaluates only the setup data generated by all the source systems defined for that Pricing Transaction Entity. It also makes contexts of one Pricing Transaction Entity unavailable to other pricing applications families.

Checklist for Building Attribute Mapping Rules

Sometimes users believe they have successfully mapped an attribute when they get the message Attribute Mapping Rule Generation is Successful, but find that the new Attribute Mapping box for that attribute remains cleared when it should be selected. This may occur when they 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 and to avoid the situation described above, users must ensure 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/pricing attribute (for which attribute mapping is defined) and generates a source code for attributes for which attribute mapping rules are modified:

> **Note:** Since this program changes the status of database objects, it is recommended this program be run 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 completed 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 Pricing Transaction Entity.
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 Attribute Mapping Enabled box selected.

7. Check if the program has completed 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 that the attribute has been correctly sent to the pricing engine. You can also verify if the pricing engine used the correct attributes while pricing. For more information, see: *Pricing Engine Request Viewer*.

   Note: Before running Build Attribute Mapping Rules program, select the profile QP: Build Attributes Mapping Options. When set to Yes, mapping rules can be generated for attributes used in active pricing setup.

   When set to No, mapping rules are generated for all the attributes. The Build Attribute Mapping Rules program will map the attributes that are used only in the active pricing setup if the profile value QP: Check For Active Flag is set to Yes. If the profile value QP: Check For Active Flag is set to No, this program will map the attributes that are used in both the active and inactive setup.

To run the Attribute Mapping Rules Error Report:

If you run the Build Attribute Mapping Rules program and it generates attribute mapping errors, 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 a New Pricing Transaction Entity

A Pricing Transaction Entity (PTE) consists of a group of applications that share the same setup data and attributes. For example, Istore and Order Management can be part of one Pricing Transaction Entity because they are both 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.

The Pricing Transaction Entity narrows the data the search engine reviews. The search engine examines only the setup data generated by the source systems defined for that Pricing Transaction Entity. All applications belonging to the same pricing transaction entity have the same set of attributes available to them. A same PTE ensures that the applications sharing the same data always give the same price for an item irrespective of the request type.

You can set up a Pricing Transaction Entity (PTE) that groups the source systems and request types. Attributes can be linked to one or more Pricing Transaction Entities. The Pricing Transaction Entity does the following functions:

- Narrows the data the search engine examines. The search engine examines only the setup data generated by the source systems defined for that Pricing Transaction Entity.
- All applications belonging to the same pricing transaction entity have the same set of attributes available to them.

> **Note:** The profile QP: Pricing Transaction Entity indicates the current Pricing Transaction Entity in use. It can be updated at the user level. Only those contexts and attributes assigned to the current Pricing Transaction Entity are available in the list fields on the setup windows. Querying setup data 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 right value before creating or querying any setup data in the Pricing Application.

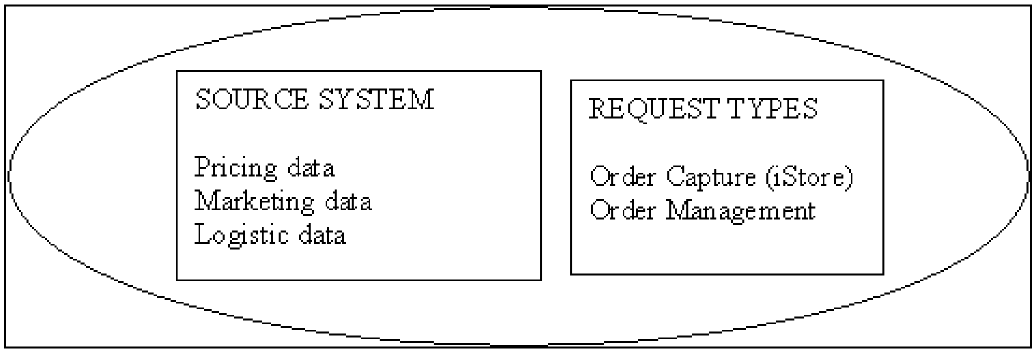
When to Define a New Pricing Transaction Entity

When a new request type is created - for example, a new ordering structure is added - the 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 Pricing Transaction Entity

The following graphic displays the Order Fulfillment Pricing Transaction Entity. Within this PTE, the request types such as iStore and Order Management evaluate pricing data.

Pricing Transaction Entity - Order Fulfillment



Global Structures

The table below shows examples of the global structures at the order level and line level for the different request types of the Order Fulfillment Pricing Transaction Entity.

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_INFO	OKC_PRICE_PUB.G_ CONTRACT_INFO
Order Fulfillment	Order Management Order	OE_ORDER_PUB.G_ HDR	OE_ORDER_PUB.G_L INE
Intercompany Transaction	Intercompany Invoicing	INV_IC_ORDER_PUB. G_HDR	INV_IC_ORDER_PUB. G_LINE

To create a new Pricing Transaction Entity:

1. Navigate to the Pricing Lookups window.
2. Query on QP_PTE_TYPE as lookup type
3. Enter a code (short name) for a Pricing Transaction Entity.
4. Enter a Meaning and Description.
5. Enter optional fields.
6. Save the record.
7. Navigate to the Pricing Transaction Entity - Source System and Request Types window. In this window, you can define the request types and source system for a given Pricing Transaction Entity.

   A source system defines the application generating setup data. For example iStore, Oracle Pricing and Oracle Marketing generate modifiers. Hence these applications could be source systems.
8. Click the Source Systems tab.
9. Enter a Code which is the short name for the source system application.
10. Enter a Description for the source system.
11. Select the Enabled box to activate the Source System.
12. Click the Request Types tab.
13. Enter a Code which is a short name for the request type.
14. Enter a Description of the request type.
15. Enter a Header or Line Structure and/or a Header and Line View.

    A request type may have a global record structure or a view defined to map the data. You can either enter global structures for header and line or view names for header and line. All of these four fields are optional.

    If data is entered in any of these fields, based on this data a list of values (LOV) will be provided on ValueString field of the Attribute Mapping setup window. If no data is entered in any of these fields, no LOV will be provided on Value String field of the Attribute Mapping setup window.

16. Enter a Header Structure: The global Structure for the Header.
17. Enter a Line Structure: The global Structure for the Line.
18. Enter a Header View name: View name for the header
19. Enter Line View name: View name for the line.
20. Select Enabled to activate the request type.
21. Save the record.

Note: A request type cannot be created for more than one Pricing Transaction Entity.

A request type cannot be deleted if a mapping rule is defined for it. A source system cannot be deleted if it is used in a setup for a given Pricing Transaction Entity.

Using Custom Sourced Attributes

A code must be provided by the user to source an attribute. Attribute mapping is not required as the attribute will be sourced by Get custom attributes.

Confirm that the profile option QP: Custom Sourced 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.

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 sourcing 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 Y. 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, the user can control how the attributes are sourced 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.

Example using QP_CUSTOM_SOURCE

The follow example explains how a customer may code 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. The user 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_RE
SULT_TBL_TYPE
     ,x_price_ctxts_result_tbl  OUT QP_ATTR_MAPPING_PUB.CONTEXTS_R
ESULT_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_ATTR
IBUTE31';
      x_qual_ctxts_result_tbl(1).attribute_value := '10';
      x_price_ctxts_result_tbl(1).context_name := 'CUST_SOURCE_PRI
C_CON';
      x_price_ctxts_result_tbl(1).attribute_name := 'PRICING_ATTRI
BUTE31';
      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 not created by the user. A Seeded check box next to a context indicates if the context is seeded or not (when 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 replaces values in all the fields with seeded values. The Restore Defaults button is grayed out for non-seeded attributes and for seeded attributes whose 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 display in the Name and Description fields; however, if you change the seeded values, the new values display 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, 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 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 field will always be hidden.

The Restore Default button restores the original seeded attribute mapping method of an attribute. If the user overwrites a seeded attribute and wishes to go back to seeded values again later, user 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 same as seeded Attribute Mapping method.

> **Note:** The attribute Total Item Quantity is only seeded for Oracle Transportation Execution (FTE). However, you can manually link an attribute to the Order Fulfillment Pricing Transaction Entity, 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 same as 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 be displayed on setup windows when user tries to use in the setup, any newly added attribute having Attribute Mapping Method = ATTRIBUTE MAPPING whose 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 be displayed on setup windows when user tries to use in the setup, any newly added attribute having Attribute Mapping Method =ATTRIBUTE MAPPING whose 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 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

User tries to attach a combination of qualifier with the attributes that have Attribute Level of LINE and ORDER within the same Qualifier Grouping No 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 be displayed 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 No.

Message: There is a mix of LINE and ORDER qualifier attribute level for list line id &LIST_LINE_ID and qualifier grouping no &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 no.

Troubleshooting during Pricing Setup

While defining Qualifier or Pricing Attributes in the Pricing Setup windows, only those Attributes with LOV Enabled box selected display in the LOV's on these windows. If basic pricing is installed, only those attributes that have been set up with the AVAILABLE_IN_BASIC box selected display in the LOV's on the various Pricing Components setup windows. However, all attributes that are setup are available to Advanced Pricing users, subject to meeting the other attribute level conditions.

It is important to set the profile QP: Pricing Transaction Entity to the right value before creating or querying any setup data in the pricing application. It is not recommended to change this profile often as 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 provides answers to common troubleshooting questions you may have during integration or runtime.

Viewing Attributes

Question 1
Why do I not see the Pricing Contexts and Attributes defined by me in the Pricing setup windows?

Answer: The following conditions must be met:

- The contexts and attributes defined by you using the Attributes Manager windows are assigned to the right Pricing Transaction Entity Code.
- 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 Pricing Transaction Entity in use. Only those contexts and attributes assigned to the current Pricing Transaction Entity will be available in the LOV's 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 Pricing Transaction Entity.

Entering Orders

Question 1
When you enter an order line in the Sales Order Pad, the following error displays:

```
FND_AS_UNEXPECTED_ERROR (PKG_NAME=oe_order_adj_pvt) (PROCEDURE_NA
ME=oe_line_adj.calculate_adjustments) (ERROR_TEXT=calculate_adjust
ments#130 ORA-06508: PL/SQL: could not find program unit being cal
led).
```

Answer: Check 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 Order management Sales Pad Pricing Context list of values.

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 created using the Flexfield windows will not get created in the Attribute Manager tables. Columns updated in Flexfield window is 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 see if 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, it is advised that you confirm that this has occurred.

Additional Attribute Management Considerations

The following table outlines some problems and solutions 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 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.	Execute 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 sales order PAD receives an 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).	Execute 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 there is a patch 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 Pricing Transaction Entities and attribute links
4. Upgrade attribute mapping rules
5. Assign Pricing Transaction Entities 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 Pricing Transaction Entity
Context and Attributes created and maintained in Flexfields	Context and Attributes created and maintained in QP tables
All attributes available to all applications for Pricing Setup	Attributes available to applications based on PTE. Attributes are now LOV Enabled and Limits Enabled
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 i.e. Order Fulfillment.	At least 4 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
Attribute Mapping Rules	oOe_def_attr_def_rules	qp_attribute_sourcing

Upgrading Context and Attributes

This section consists of upgrading Context and Attributes from the current Flexfield Structure to a set of new normalized QP tables.

Upgrading Contexts

In the current system, all the contexts are stored in qualifier contexts and pricing contexts flexfields. The upgrade program will copy all the contexts from these flexfields to the new QP tables. All the Qualifier Contexts flexfield qualifiers will be copied as contexts 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 created in the flexfields by user 1 will be treated as seeded data. All other information, like name and description in various languages, 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, as some attributes use Key flexfields for their valid values. All other information like names, mapping columns, 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 has a new entity called Pricing Transaction Entity which can be described as an ordering structure with associated Request Types and Source Systems. When upgrading, a new set of Pricing Transaction Entities are created in the target system. More Pricing Transaction Entities can be created depending on the Customer's own Request Types and Source Systems, different from the seeded data.

Mapping of Seeded Request Types and Source Systems

Secondly, there is a pre-determined process to associate Request Types and Source Systems to these Pricing Transaction Entities. These Request Types with their associations will act as seeded data. Please refer to the earlier section 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 associated with them will be as shown below:

Request Types	Global Structure Name: Header Level	Global Structure Name: Line Level
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 or otherwise. 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 find out 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.

As an example, say Segment-1 is associated with QP Source System, as per the Defaulting Condition Templates. The upgrade program will create 3 links for this attribute, represented as 3 rows in the new PTE-Attribute mapping table as shown below.

Pricing Transaction Entity	Attribute
1 Unassigned-2	Segment-1
2 Order Fulfillment	Segment-1
3 Intercompany Transaction	Segment-1

The three rows get created since 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 a Attribute Mapping rule. These are the attributes for which there are no transactions in Price Lists, Modifiers, and Formulas. For such attributes, they will be linked to all the PTEs that we created so far (including the seeded ones) in the PTE Attribute Mapping Table.

Upgrade Attribute Mapping Rules

When 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 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 skip the upload/upgrade of that attribute:

- At customer location that uses the same segment mapping column as used by a seeded attribute (in case of Attribute manager Loader program) or
- A user attribute created in the flexfield (in case of Attribute Manager Upgrade program).

However, unlike the current system, every attribute that will have a 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 may 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 Pricing Transaction Entity 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 QP_LIST_HEADERS_B table which stores the Pricing Transaction Entity. The Upgrade program updates the PTE_CODE for every modifier list/price list/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 finds out the PTE(s), for every modifier from its Source System. If the Source System is associated to 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 REPLAC
E
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 11i's custom sourcing routines used for
-- retrieving additional information from the apps tables into pri
cing
-- 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
```

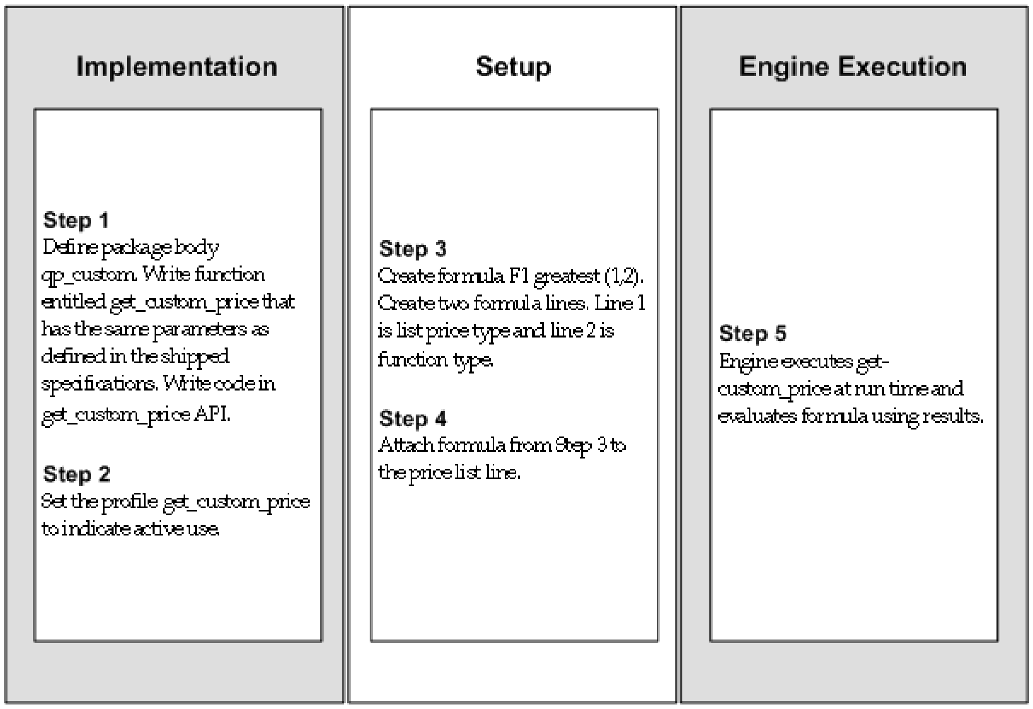
15
Get Custom Price

Overview of Get Custom Price Implementation

The actual value of a modifier or price is accomplished using the Get_Custom_Price function. 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 involved in the implementation of Get_Custom_Price is depicted in the following image.

Overview of Get_Custom_Price implementation steps


Implementation
Step 1
Define package body qp_custom. Write function entitled get_custom_price that has the same parameters as defined in the shipped specifications. Write code in get_custom_price API.
Step 2
Set the profile get_custom_price to indicate active use.
Setup
Step 3
Create formula F1 greatest (1,2). Create two formula lines. Line 1 is list price type and line 2 is function type.
Step 4
Attach formula from Step 3 to the price list line.
Engine Execution
Step 5
Engine executes get-custom_price at run time and evaluates formula using results.

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 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 like 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 in 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;
```

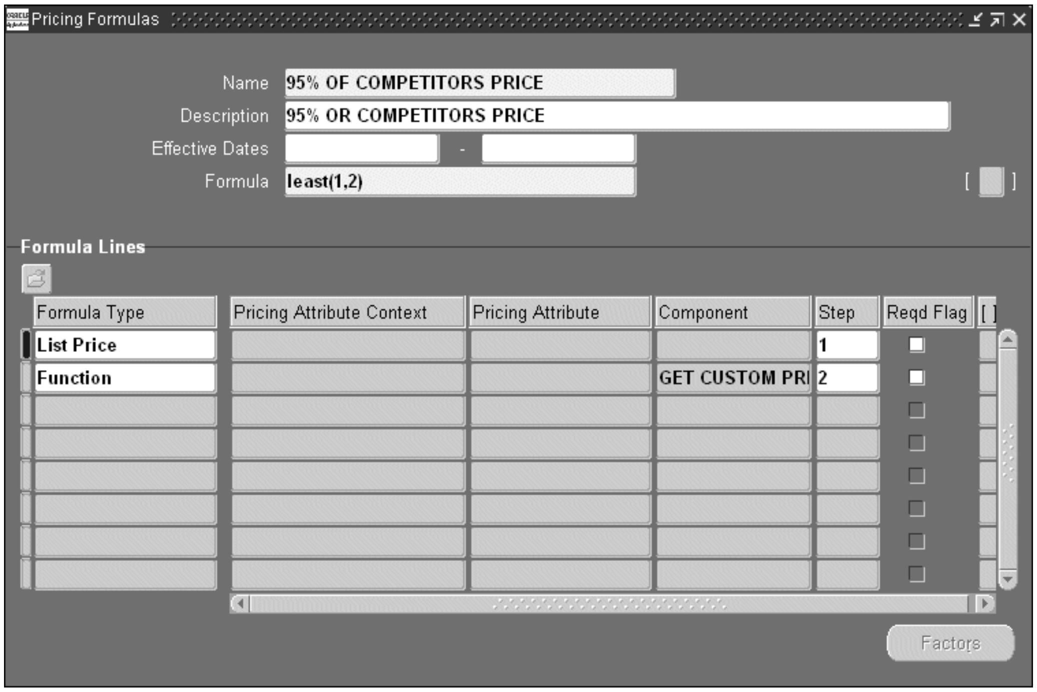
2. Set the value of profile QP: Get Custom Price Customized.

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 displays. Set the profile at site level.

3. Create a formula to use the get_custom_price function.

   The follow image depicts the Pricing Formulas window:

   Pricing Formulas window



Pricing Formulas

Name	95% OF COMPETITORS PRICE
Description	95% OR COMPETITORS PRICE
Effective Dates	-
Formula	least(1,2)

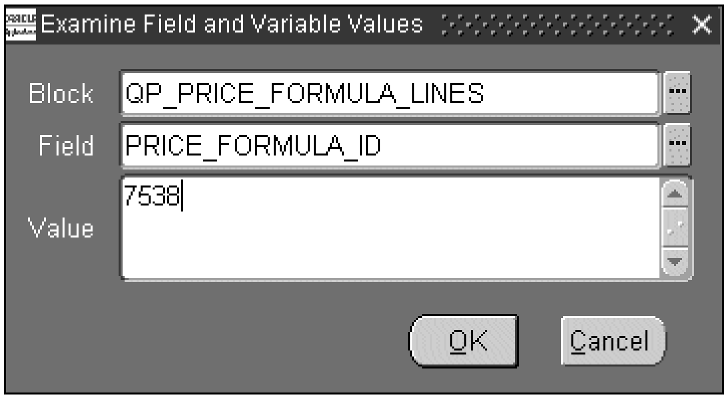
Formula Lines

Formula Type	Pricing Attribute Context	Pricing Attribute	Component	Step	Reqd Flag
List Price				1	☐
Function			GET CUSTOM PRI	2	☐

   Remember to note the formula_id by using Help, Examine. Use this formula ID in the get_custom_price function to identify the formula.

   The following image depicts the Examine Field and Variable Values window that pops up from the Pricing Formulas window in Oracle Advanced Pricing.

   Examine Field and Variable Values window



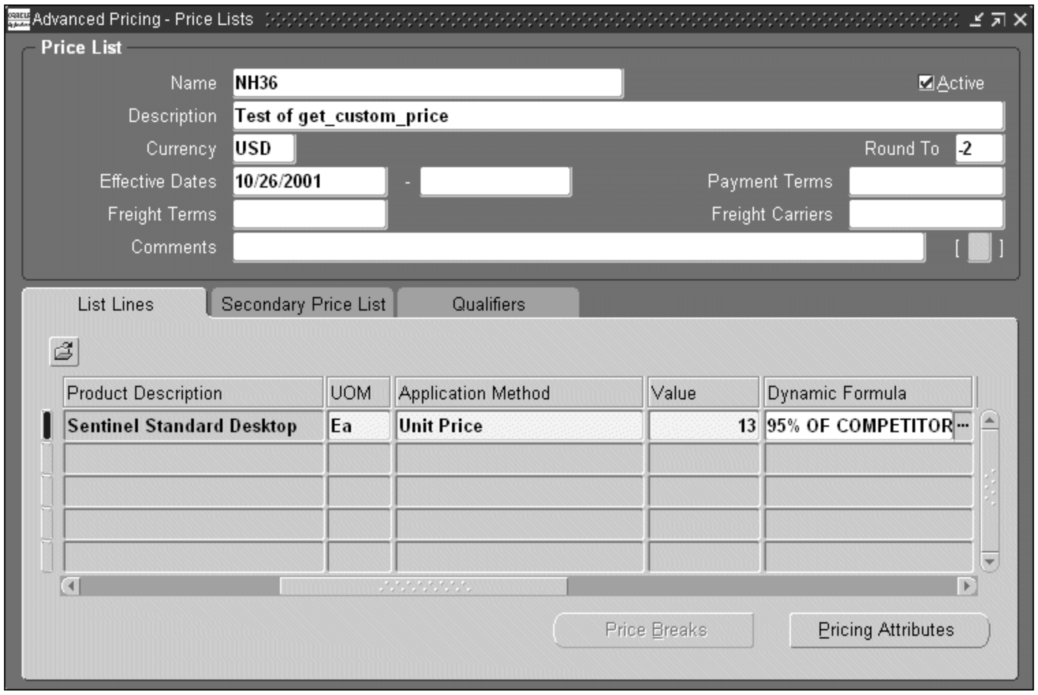
Examine Field and Variable Values

Block	QP_PRICE_FORMULA_LINES
Field	PRICE_FORMULA_ID
Value	7538

4. Attach this formula to the price list line.

The following image depicts the Price Lists window in Oracle Advanced Pricing.

Price Lists window



Advanced Pricing - Price Lists

Price List

Name	NH36	✓ Active	
Description	Test of get_custom_price		
Currency	USD	Round To	-2
Effective Dates	10/26/2001 -	Payment Terms	
Freight Terms		Freight Carriers	
Comments			

List Lines | Secondary Price List | Qualifiers

Product Description	UOM	Application Method	Value	Dynamic Formula
Sentinel Standard Desktop	Ea	Unit Price	13	95% OF COMPETITOR

Get_Custom_Price_Customized

The following sample code describes 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 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 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).

If 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 been 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. May 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_ATT
RS_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/p
rimary key for the sample Formula 1*2
     THEN
         Loop through the PL/SQL table of records passed by the En
gine as an input parameter and containing Pricing Attributes and P
roduct          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 At
tribute Value is the Inventory Item Id v_requested_item:= p_req_li
ne_att      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 fo
rmula       is attached.
     v_weight:= p_req_line_attrs_tbl(i).value;
     For this combination of Special Context and Attribute, the At
tribute Value is the Step Number of the formula line
     IF p_req_line_attrs_tbl(j).attribute_type=QP_GLOBALS.G_SPECIA
L_ATTRIBUTE_TYPE
     and p_req_line_attrs_tbl(j).context=QP_GLOBALS.G_SPECIAL_CONT
EXT
     and p_req_line_attrs_tbl(j).attribute=QP_GLOBALS.G_SPECIAL_AT
TRIBUTE1 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.

16 Events and Phases

This chapter covers the following topics:

Overview

Pricing events and phases enable you to configure Oracle Advanced Pricing so 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 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:

Event	Action	Function Which Calls the Pricing Engine
Batch	Batch Processing	Order Import
Book	Book Order	Order is booked
FTE: Apply Modifier	FTE: Apply Modifiers to Price	This event is only used by the Oracle Transportation Execution application to price their transactions.
FTE: Price Line	FTE: Price a Transportation Line	This event is only used by the Oracle Transportation Execution application to price their transactions.
ICBATCH	INV: Batch Processing for Intercompany Transfer Pricing	This event is only used by the Oracle Inventory application for pricing their transactions.
Line	Enter Order Line	User exits the order line
Order	Save Order Line	User saves order
Price	Fetch List Price	User enters item quantity and unit of measure
Price Load	Price a Logistics Load	
Reprice Line	Reprice line	Order is shipped, prior to invoice
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 only be assigned to one phase.

The following table summarizes the seeded pricing phases in Oracle Advanced Pricing:

Phase Sequence	Name	Level	List Type	Incompatibility Resolve Code	Freeze Override
0	List Line Base Price	Line	Standard Price List	Precedence	N/a
10	List Line Adjustment	Line	N/a	Best Price	N/a
30	All Lines Adjustment	N/a	N/a	Best Price	N/a
40	Header Levels Adjustments	Order	N/a	Precedence	N/a
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	N/a	Freight and Special charge List	Precedence	Yes
80	Modifiers for BOOK Event	N/a	N/a	Precedence	N/a

Assigning Pricing Phases

Key Implementation Decision: What phases and events meet my business requirements?

Events separate the order cycle into points attachable to pricing actions. You can use events and phases to exercise control over what pricing actions are taken and when this occurs. You must decide to which phases your discounts and promotions (modifiers) belong.

You can map a pricing event to several pricing phases, and a pricing phase can be assigned to more than one pricing event. 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 event, those phases must be added to the batch event.

Key Implementation Decision: Have I defined my modifier groupings so that the seeded phases are sufficient for my pricing actions?

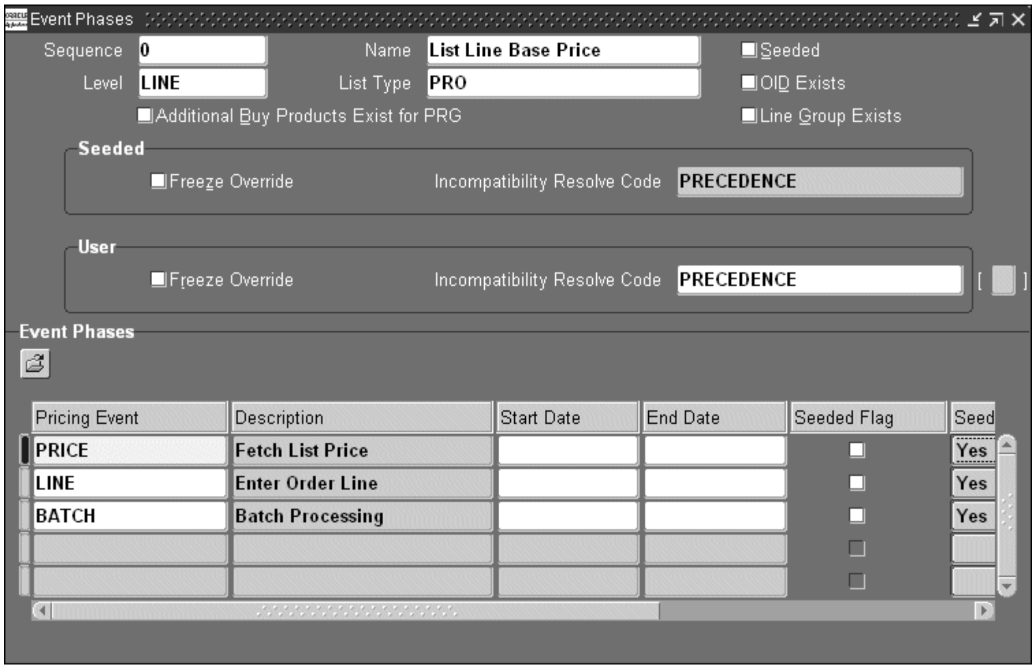
You can accomplish this through the Event Phases window in the Setup menu. Remember the following:

- Place line level discounts in line adjustments 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



Sequence	0	Name	List Line Base Price	☐ Seeded	
Level	LINE	List Type	PRO	☐ OID Exists	
☐ Additional Buy Products Exist for PRG				☐ Line Group Exists	

Seeded

☐ Freeze Override — Incompatibility Resolve Code: PRECEDENCE

User

☐ Freeze Override — Incompatibility Resolve Code: PRECEDENCE

Event Phases

Pricing Event	Description	Start Date	End Date	Seeded Flag	Seed
PRICE	Fetch List Price			☐	Yes
LINE	Enter Order Line			☐	Yes
BATCH	Batch Processing			☐	Yes
				☐	
				☐	

The following describe 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 execution order of the phases when there are multiple phases in an event.

Name

This field is required and appears to Modifier Setup window users when assigning a modifier to a phase. The field name should describe 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.

Seeded and User

Two sections appear below the Sequence, Name, Level, and List Type fields: one entitled Seeded and the other entitled User. These two sections are used to differentiate the seeded values from the user-entered values. In the User section, you can update the freeze override flag and incompatibility resolve code.

OID Exists

This cannot be entered by the user. This box is checked if there are any modifiers of the type other item discount for a particular phase.

Line Group Exists

This cannot be entered by the user. This box is checked if there are modifiers in the level group of lines for a particular phase.

Additional Buy Products Exist

This cannot be entered by the user. This box is checked if you defined modifiers for promotional goods or other item discounts and you will define additional buy products.

Freeze Override Flag

This flag provides additional control over freezing lines on your transaction. If calculate price sent from the calling application on the request line is set to P, the pricing engine looks at this Freeze Override Flag on the phase. If it is checked, the pricing engine applies eligible modifiers in this phase to the request line. If it is not checked, the modifiers in this phase are not considered for application to the request line. This flag can also be updated on seeded pricing phases.

Incompatibility Resolve Code

This field is required. Incompatibility resolve code refers to the method used to determine which modifier is selected when multiple modifiers in the same exclusivity or incompatibility group are eligible to be applied to the same pricing request line. The incompatibility resolve codes are as follows:

- Best price: The modifier which gives the lowest price (most advantageous) to a customer on the given pricing request line is applied.
- Precedence: The modifier with the lowest (most specific) precedence on the given pricing request line is applied

The incompatibility resolve code value can be updated on seeded pricing phases.

Pricing Event

This field assigns the pricing event you are linking to a phase. Multiple pricing events may be entered for a single phase. Valid values are (pricing event lookup):

- Batch
- Book
- Order
- Line
- Price
- Ship

> **Note:** In Event Phases window, if you select the PRICE event for a phase, and the phase is already attached to a promotional modifier type (Coupon Issue, Item Upgrade, Other Item Discount, Promotional Goods, Term Substitution) then the following error message displays:

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.

Start Date

This field is optional. Enter the date for the event to start using the pricing phase.

End Date

This field is optional. Enter the date for the event to stop using the pricing phase.

Seeded Search Flag

This flag is used by the pricing engine to decide whether the engine should perform a search for price or modifier lists in addition to any 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; it 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 only be set to No 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 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. This can be done by setting the User Search Flag to Yes.

Example

Passed price list = Corporate

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.

17
Pricing Engine Request Viewer window

This chapter covers the following topics:

Overview

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, or Oracle Contracts Core. Information about the latest pricing request is displayed 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 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. 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.

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 occur when a pricing call is made:

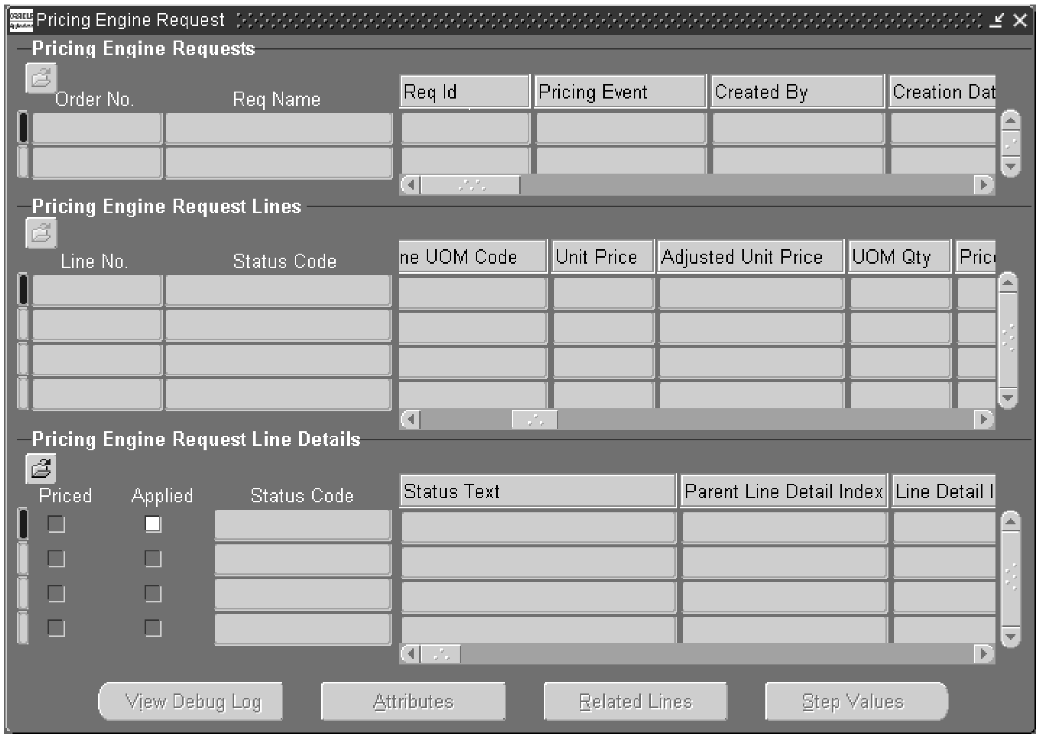
1. The calling application makes a call to the Build Attribute Mapping Rules package to generate the attributes defined by the Attribute Mapping Function.
2. The calling application then calls the pricing engine with the attributes generated by attributes mapping.
3. The pricing engine processes the request. 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 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 are displayed 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



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 is displayed. **Note:** Depending on the version of Oracle Pricing installed, the order and line numbers for orders created in prior releases may not display 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.
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 wish 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	This refers to the Calculation_Flag in 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: • N: Search engine: If you do not want the engine to calculate the selling price. • C: Calculate engine: If you are passing the adjustment records to the engine and you want the engine to recalculate the selling price, without retrieving new adjustments. • Y: Both calculate and search engine: Regular engine call. Retrieves new adjustments and calculates the selling price
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.
Rounding Flag	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. • If set to Y , engine should apply the rounding factor defined in the price list. • If set to N, unrounded figures would be returned. • If set to Q, then refer to the value of the profile QP: Selling Price Rounding Options.
GSA Check Flag	Indicates whether the pricing calculation engine should test for GSA Violations: 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: • *Yes*: Price Calculation engine tests for GSA violations, any violating request lines are returned to the calling application with a status of GSA violation. • *No*: Do not test for GSA violations. The value of this field is controlled by profile in the requesting systems.
GSA Dup Check Flag	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.
Temp Table Insert Flag	This flag is set to Y, if the calling application directly inserts data into the pricing temporary tables, set to No if the pricing engine inserts into Temporary tables.

Manual Discount Flag	This 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: • *Yes*: All the manual discounts will be returned. All the automatic discounts that get deleted as part of incompatibility processing will be returned as manual discounts. • *No*: All automatic and manual discounts will go through incompatibility processing and one of them in each incompatibility group will be returned. In this process an automatic discount might get deleted and a manual discount might get selected.
Source Order Amount Flag	If set to Y, Indicates to the pricing engine to source the order amount. It means the calling application will provide the order amount. If set to N, The Pricing engine should calculate the order amount.
Public API Call Flag	Indicates if the public API is being used to call the Pricing Engine. If set to Y, 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.
Manual Adjustments Call Flag	Indicates if the pricing engine should consider manual discounts. The values are set up in the profile option QP: Return Manual Discounts: • Yes: Apply the manual discounts. This means that the unit price is not calculated by pricing engine. • No: Do not apply the manual discounts. This means that the new unit price is calculated by pricing engine. 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 depicts the setup of two modifiers with different precedence levels: **Modifier Name: Manual _1** • Incompatibility Level: Incompatibility 1 • Precedence: 100 **Modifier Name: Manual _2** • Incompatibility Level: Incompatibility 1 • Precedence: 200 When QP: Return Manual Discounts = Yes, then the 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).
Check Cust View Flag	This is used for internal engine use.
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.

Field Name	Description
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: • N: New record created (All 'N' records are returned back from the pricing engine. These are success records) • X: Unchanged (Default status when the line is passed to the pricing engine for processing) • D: Deleted • U: Updated • IPL: Invalid price list (When passed in price list is not found, then an error is given) • GSA: GSA violation • FER: Error processing formula • OER: Other error • CALC: Error in calculation engine • UOM: Failed to price using unit of measure • INVALID_UOM: Invalid unit of measure • DUPLICATE_PRICE_LIST: Duplicate price list • INVALID_UOM_CONV: Unit of measure conversion not found • INVALID_INCOMP: Could not resolve incompatibility • INVALID_BEST_PRICE: Could not resolve best price.
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: • ORDER • LINE
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	Indicates the degree to which the price is frozen. Allowable values, based on lookup type CALCULATE_PRICE_FLAG are: • Y (Calculate Price): Apply all prices and modifiers to the request line. • N (Freeze Price): Do not apply any prices or modifiers to the request line. Consider the volume of the request line when processing LINEGROUP modifiers for other lines. • P (Partial Price): Apply prices and modifiers in phases whose freeze override flag is Y.
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	Indicates whether line has been processed by engine or not. Possible values: • *No*: Not processed • *Yes*: Processed Used for internal engine use.
Processed Code	Internal code which 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 for internal engine use.
Group Amount	Sum of the price of group of lines. Used for internal engine use.
Line Amount	Price for the line quantity. Used for internal engine use.
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. Pricing engine will try to apply the eligible manual adjustments if it can and calculate the new price or if it can't 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 will have the structure 'Request Type Code-Order Id-Line Id'. Used for internal engine use.
Hold Code	Whenever the limit is adjusted or exceeded and 'limit_hold_flag' is 'Y' the engine sets value of this field to 'LIMIT'. Used for internal engine use.
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 for internal engine use.
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 doesn't 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 doesn't 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 doesn’t exist for primary price list.
Usage Pricing Type	Indicates the usage pricing type. Allowable values are: • Regular • Billing • Authoring

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 displays 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: • Yes: Applicable when the attribute context is a list or list line • No: Not applicable when the attribute context is a list or list line

Status Code	Indicates returned status. Possible Values are: • New record created (All 'N' records are returned back from the pricing engine. These are success records) • Unchanged (Default status when the line is passed to the pricing engine for processing) • Deleted • Updated • Invalid price list (When passed in price list is not found, then an error is given) • GSA violation • Error processing formula • Other error • Error in calculation engine • Failed to price using unit of measure • Invalid unit of measure • Duplicate price list • Unit of measure conversion not found • Could not resolve incompatibility • Could not resolve best price
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
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: • Adjustment percent (for discounts) • Adjustment amount (for discounts) • Adjustment New Price (for discounts) • UNIT_PRICE (for price lists) • PERCENT_PRICE (for price lists) • LUMPSUM
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 can't be modified.
Pricing Phase	The pricing phase which 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 for internal engine use. If selected, it indicates that the selected modifier is an ASK FOR modifier.
Processed	Used for internal engine use. Indicates whether line has been processed by engine or not. Possible values: • No: Not processed • Yes: Processed

Created From SQL	Indicates which cursor was used by the engine to select the modifier. Used for internal engine use. Possible values: • PRODUCT_ONLY • EXCLUDED_PRODUCT_ONLY • QUALIFIER_ONLY • PRODUCT_QUALIFIER_ONLY • PRODUCT_PRICING_ONLY • PRODUCT_QUALIFIER_PRICING_ONLY • INSERTED IN SECONDARY LIST HEADER SEARCH • INSERTED IN VALIDATED LIST_HEADER_SEARCH1 • INSERTED IN NOT VALIDATED LIST_HEADER_SEARCH1 • INSERTED IN VALIDATED ASKED FOR PROMOTION SEARCH • INSERTED IN NOT VALIDATED QUAL_LIST_HEADER_SEARCH • INSERTED BY CREATE_QUALIFIER_FROM_LIST
Line Quantity	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.
Product Precedence	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.
Best Percent	Modifier percentage that gives the best price. Used for internal engine use.
Primary UOM	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.
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.
Estim 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 for internal engine use.

Group Amount	Sum of the price of group of lines. Used for internal engine use.
Process Code	This is set by the engine and used for selecting lines for calculation. Used for internal engine use. Possible values: • *N*: New • *D*: Deleted • *X*: Unchanged
Updated Flag	This value is passed in by the calling application. Used for internal engine use.
Limit Code	The value of this field is set to ADJUSTED when limit is adjusted, set to EXCEEDED when the limit is exceeded. Used for internal engine use.
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.

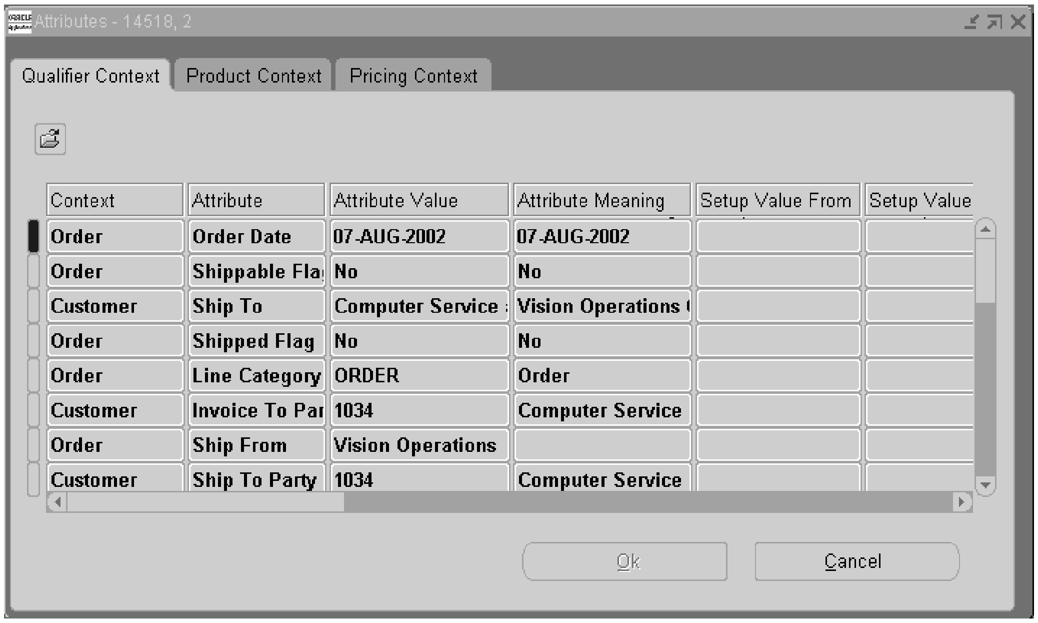
Use this table when Oracle Order Management does not return an expected discount or adjustment and the list line ID do not appear in line details region.

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



Context	Attribute	Attribute Value	Attribute Meaning	Setup Value From	Setup Value
Order	Order Date	07-AUG-2002	07-AUG-2002		
Order	Shippable Fla	No	No		
Customer	Ship To	Computer Service :	Vision Operations (		
Order	Shipped Flag	No	No		
Order	Line Category	ORDER	Order		
Customer	Invoice To Par	1034	Computer Service		
Order	Ship From	Vision Operations			
Customer	Ship To Party	1034	Computer Service		

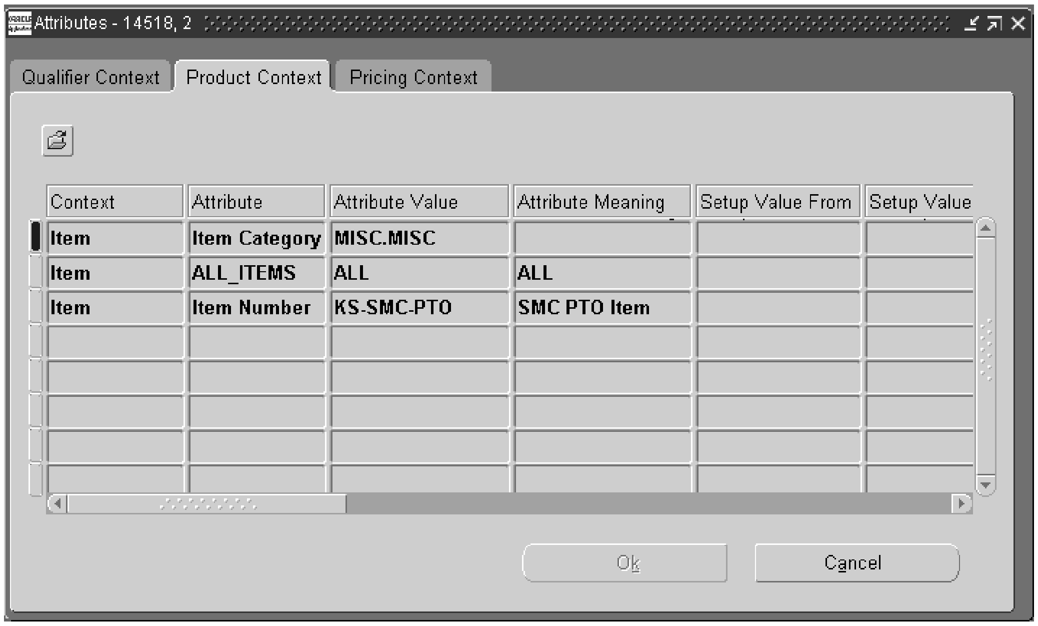
Columns in this tab include:

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: • *Yes*: Applicable when the attribute context is a list or list line • *No*: Applicable when the attribute context is a list or list line
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	This is set by the engine and used for selecting lines for calculation. Used for internal engine use. Possible values: • N: New • D: Deleted • X: Unchanged.
Distinct Qualifier Flag	To determine qualification engine sets this flag to indicate that this is unique qualifier.
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.

Product Context tab

The following image depicts the Product Context tab:

Product Context tab: Attributes window



Context	Attribute	Attribute Value	Attribute Meaning	Setup Value From	Setup Value
Item	Item Category	MISC.MISC			
Item	ALL_ITEMS	ALL	ALL		
Item	Item Number	KS-SMC-PTO	SMC PTO Item		

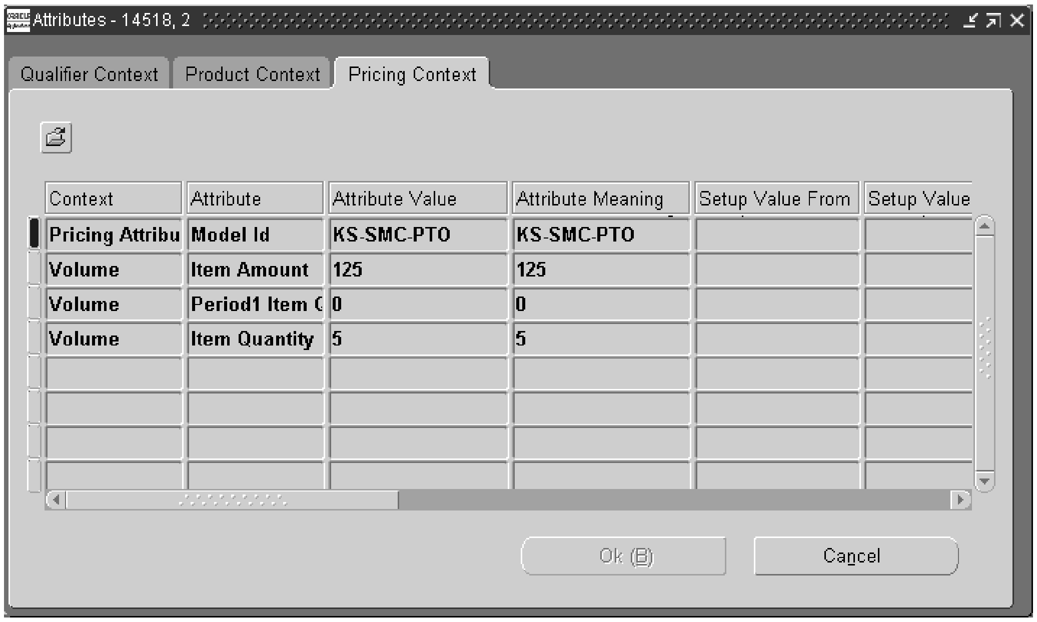
Columns in this tab include:

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: • *Yes*: Applicable when the attribute context is a list or list line • *No*: Applicable when the attribute context is a list or list line
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 in-compatibility resolution. This field is not updateable by 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 etc. for which the price or modifier is defined.
Processed Code	This is set by the engine and used for selecting lines for calculation. Used for internal engine use. Possible values: • N: New • D: Deleted • X: Unchanged.
Excluded Flag	If set to Yes, it indicates that product value was defined as an excluded item on the modifier line.
Group Qty	Sum of the quantity of group of lines. Used for internal engine use.
Group Amount	Sum of the price of group of lines. Used for internal engine use.
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 depicts the Pricing Context tab:

Pricing Context tab: Attributes window



Context	Attribute	Attribute Value	Attribute Meaning	Setup Value From	Setup Value
Pricing Attribu	Model Id	KS-SMC-PTO	KS-SMC-PTO		
Volume	Item Amount	125	125		
Volume	Period1 Item (	0	0		
Volume	Item Quantity	5	5		

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.
Applied Flag	The lists or list lines that this pricing event or a prior pricing event applied. Allowable values are: • Yes: Applicable when the attribute context is a list or list line • No: Applicable when the attribute context is a list or list line
Data Type	Indicates the data type of the pricing attribute value or qualifier attribute value.
Processed Code	This is set by the engine and used for selecting lines for calculation. Used for internal engine use. Possible values: • N: New • D : Deleted • X: Unchanged.
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.

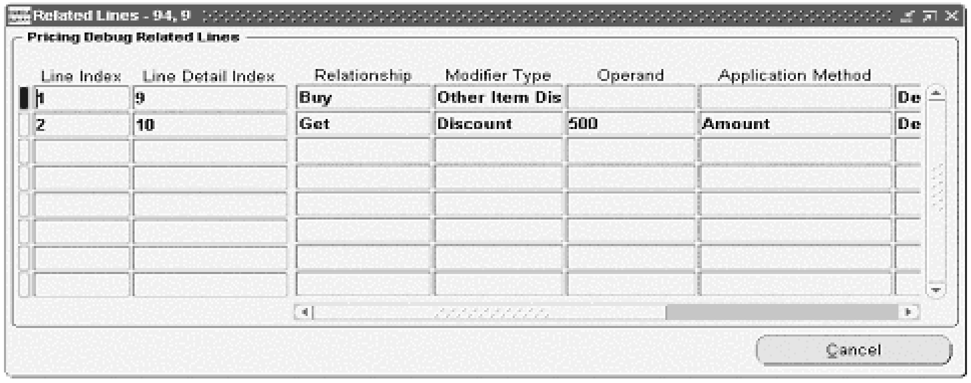
Related Lines window

This window maps to 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 depicts the Related Lines window:

Related Lines window



Line Index	Line Detail Index	Relationship	Modifier Type	Operand	Application Method	
1	9	Buy	Other Item Dis			De
2	10	Get	Discount	500	Amount	De

Columns in this window are described in the following table:

Field Name	Description
Line Index	PL/SQL unique identifier for request line.
Line Detail Index	PL/SQL unique identifier for request detail line.
Relationship	Type of relationship between pricing lines. Allowable values are: • BUY • GET
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 qp_lookups table.
Operand	Value of pricing request detail line, for example, 10 currency unit list price with 3% discount.
Modifier number	Modifier list number.
Modifier Name	Modifier list name.
Application Method	Type of operand. Eligible values are: • Adjustment percent (for discounts) • Adjustment amount (for discounts) • Adjustment New Price (for discounts) • Unit Price (for price lists) • Percent Price (for price lists) • Lumpsum • Block Price • Break Unit Price

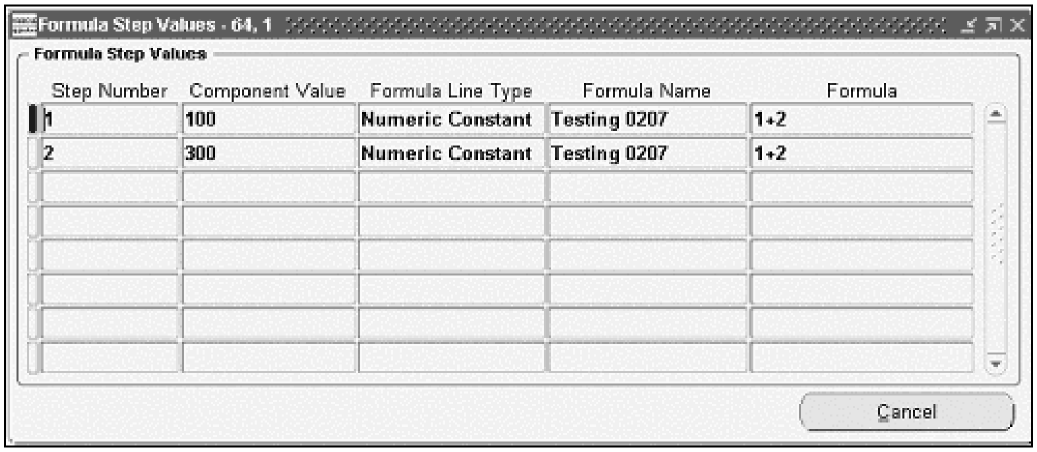
Formula Step Values window

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 QP_FORMULA_STEP_VALUES 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 depicts the Formula Step Values window:

Formula Step Values window


Formula Step Values - 64, 1

Formula Step Values

Step Number	Component Value	Formula Line Type	Formula Name	Formula
1	100	Numeric Constant	Testing 0207	1+2
2	300	Numeric Constant	Testing 0207	1+2

Columns in this window are described in the following table:

Step Number	Step number corresponding to a formula line.
Component Value	Evaluated value of a formula step.
Formula Line Type	Type of the formula line. Possible values are: • FUNC: Function • LP: List Price • ML: Factor List • MV: Modifier Value • NUM: Numeric Constant • PLL: Price List Line • PRA: Pricing Attribute
Formula Name	Name of the Pricing Formula.
Formula	Indicates the 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.

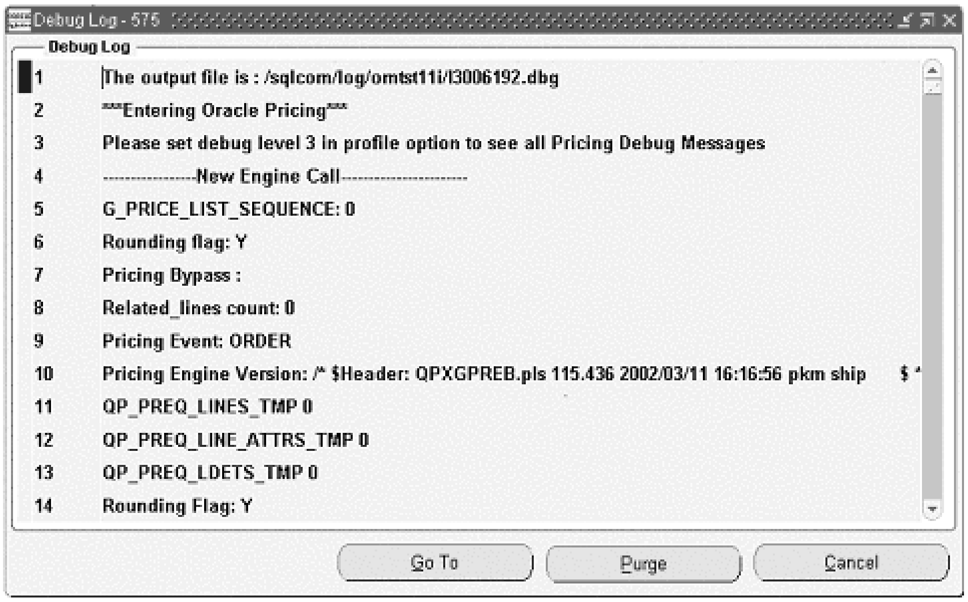
Searching the Debug Log (Using GOTO button)

You can do a query 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



```
1    The output file is : /sqlcom/log/omtst11i/l3006192.dbg
2    ***Entering Oracle Pricing***
3    Please set debug level 3 in profile option to see all Pricing Debug Messages
4    ----------------New Engine Call---------------------
5    G_PRICE_LIST_SEQUENCE: 0
6    Rounding flag: Y
7    Pricing Bypass :
8    Related_lines count: 0
9    Pricing Event: ORDER
10   Pricing Engine Version: /* $Header: QPXGPREB.pls 115.436 2002/03/11 16:16:56 pkm ship     $ *
11   QP_PREQ_LINES_TMP 0
12   QP_PREQ_LINE_ATTRS_TMP 0
13   QP_PREQ_LDETS_TMP 0
14   Rounding Flag: Y
```

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.

1. 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 display are listed in the following tables:

The following table lists the three success codes for line:

Pricing Status Code	Error
N	New record created
U	Updated
X	Unchanged

The following table displays the available internal processing status:

Pricing Status Code	Error
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

The following table lists codes that require action from the calling application:

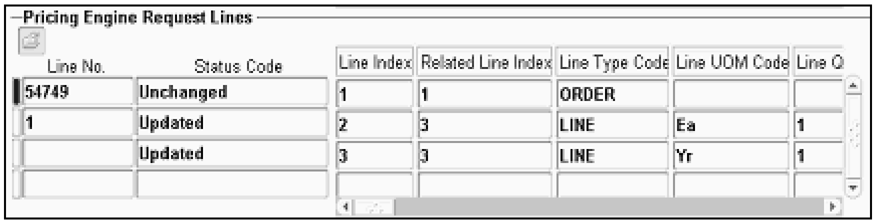
Pricing Status Code	Error
IPL	Invalid price list
GSA	GSA violation
NMS	Item not found in primary or secondary price list
FER	Error processing formula
OER	Other errors
S	System generated message
CALC	Error in calculation engine
UOM	Failed for price unit of measure
INVALID_ UOM	Invalid unit of measure
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 Service Lines relationship. The fields Related Line Index and Line Index in the Pricing Engine Request Lines region are related. By looking at the values of Related Line Index and Line Index we can identify if the lines are related; for example, Service Item and Serviceable Item.

Relationship between the Service Item and Serviceable Item:

Pricing Engine Request Lines region



Pricing Engine Request Lines

Line No.	Status Code	Line Index	Related Line Index	Line Type Code	Line UOM Code	Line Q
54749	Unchanged	1	1	ORDER		
1	Updated	2	3	LINE	Ea	1
	Updated	3	3	LINE	Yr	1

In this 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).

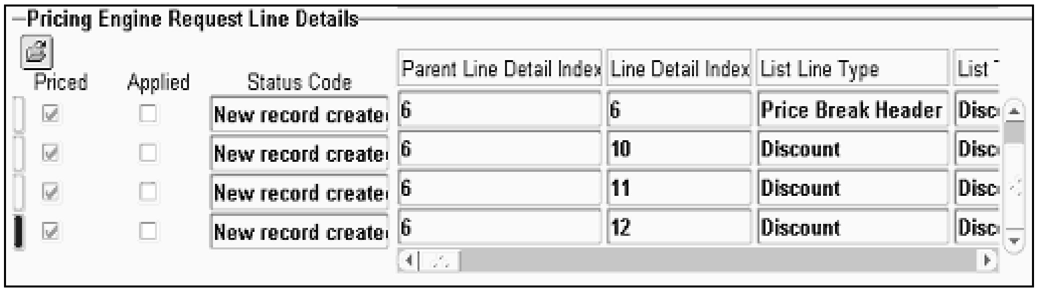
Viewing Price Break lines

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 we can identify if there is a Price Break. Price Break setup has a Price Break Parent Record, which has a line type called PBH. This PBH record can have more than 1 child record that actually define the breaks.

The following graphic displays a Price Break setup where the Price Break Parent Record (Line Detail Index 6) has a line type called Price Break Header.

This PBH record has three child records (Line Detail Index 10, Line Detail Index 11, and Line Detail Index 12) which define the breaks.

Pricing Engine Request Line Details region



Pricing Engine Request Line Details

Priced	Applied	Status Code	Parent Line Detail Index	Line Detail Index	List Line Type	List
✓	☐	New record create	6	6	Price Break Header	Disc
✓	☐	New record create	6	10	Discount	Disc
✓	☐	New record create	6	11	Discount	Disc
✓	☐	New record create	6	12	Discount	Disc

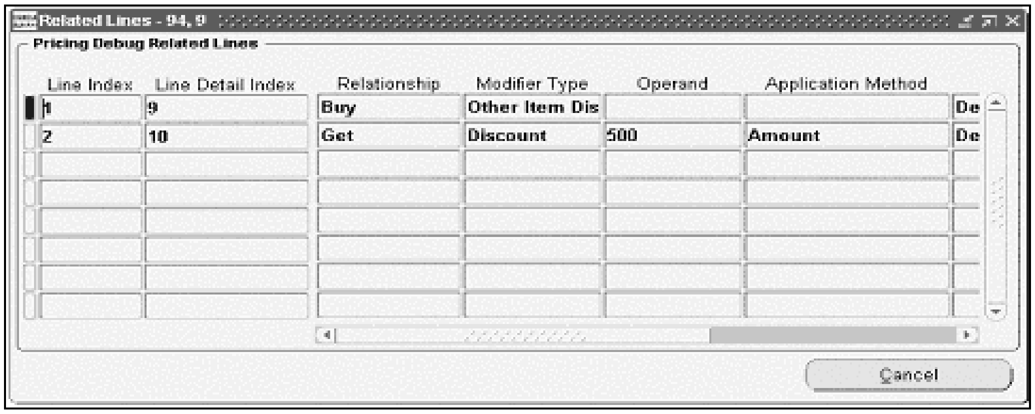
Viewing Other Item Discounts (OID)

In this case 2 request lines are passed to the pricing engine such as Buy ITEM1 and get $500 off on ITEM2.

In this example both ITEM1 & ITEM2 need to be ordered on two order lines. So 2 request lines are created and passed to the pricing engine. When the engine processes the other Item Discounts, it creates discount line for -$500 on the second request line. Also a relationship record is created 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 $500 off line. The relationship is as follows:

Related Lines window



Line Index	Line Detail Index	Relationship	Modifier Type	Operand	Application Method	
1	9	Buy	Other Item Dis			De
2	10	Get	Discount	500	Amount	De

Viewing Promotional Goods Discount (PRG)

In this case only original buy item request line is passed to the pricing engine such as Buy ITEM1 and get ITEM2 for free.

In this example only 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. In the process engine does the following things:

- A new request line (LINE RECORD) is created (Line Index 3).
- 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).
- A new Price List Line (LINE DETAIL RECORD) is created for the new request line (Line Index 3 - Line Detail Index 3).
- A new adjustment line for 100% discount is created for the new request line (LINE DETAIL RECORD) (Line Index 3 - Line Detail Index 4).
- A new relationship line (RELATED LINES RECORD) between the original PRG Line Detail Line and the new 100% off line detail is created (Line Detail Index 2 - Line Detail Index 4).
- 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.

18

Integrating with Oracle Advanced Pricing

This chapter covers the following topics:

Overview

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 Manual,* 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 :

1. See the Oracle Advanced Pricing User's Guide and the *Oracle Advanced Pricing Implementation Manual* to determine which pricing features need to be supported by your application.
2. 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.
3. 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 14-20 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.
4. 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.
5. 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 on 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: Oracle Oracle Order Management Suite APIs and Open Interfaces Manual.

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:

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: Oracle Oracle 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() callsets 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 looks at the public record structures listed in the attribute mapping rule. So, 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.

If there are no attribute mapping rules or record structure, the qualifiers and pricing attributes can 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_QUAL_TMP with qualifier_context=MODLIST.

qualifier_attribute=QUALIFIER_ATTRIBUTE4 and qualifier_attr_value_from holds the price_list_id. Set the validated_flag to Y, if you do not want the pricing engine to check for the qualifiers.

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 invetory_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 on 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 on 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 on the Append_ask_for procedure, see: *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 API structures and parameter descriptions, see: *Oracle Order Management Suite APIs and Open Interfaces Manual*.

7. If there are any manual adjustments to be applied or you need engine to consider any other adjustments, those records can be inserted into the pricing temporary table qp_preq_ldets_tmp with the applied_flag and updated_flag set to Y.

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 may not apply any order level adjustments.

   Set the line_type_code to ORDER for the summary line. (Repeat step# 4.3.1 through 4.3.6 for order header). Use p_pricing_type=H while calling QP_ATTR_MAPPING_PUB.build_context().

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 on 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:

   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 indicates the line level exceptions while pricing. These errors are populated in qp_preq_lines_tmp.pricing_status_code. The following three are the success codes for line:

   ```
   G_STATUS_NEW
   G_STATUS_UPDATED;
   G_STATUS_UNCHANGED
   ```

   The following needs 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_ERRORSG_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 on the different status codes, see: *Oracle Advanced Pricing Implementation Manual*, Troubleshooting.

Price List fetched for an 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). That means 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, 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 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 which 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), 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 offer by looking for an adjustment line of type PRG in qp_ldets_v. 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 good line.

  If you find a record with qp_ldets_v.list_line_type_code = IUE, there is 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.arithmet
  ic_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 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.

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 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 get 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% 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. So the calculated selling price will reflect the desired selling price.

  For example, if the calling application passes 3 request lines to the pricing engine and the unit selling price on the second request line is $80 and the unit list price is $100 and he wants to override the selling price to $90. In this case, the user has to pass 80 in 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 there are no surcharge adjustment, it tries to apply a manual overrideable discount adjustment. If there are no qualified manual overrideable adjustments, it returns an error status on the second request line and the pricing_status_text indicates that there are no manual adjustments. In case, there is an error during the back calculation, the engine returns an error status on the second 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 deleted.

Once they are deleted from the transaction tables, they will not get 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. 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 and sets the pricing_status_code on the request line to QP_PREQ_GRP.G_STATUS_GSA_VIOLATION in case of a GSA Violation. The calling application may handle the GSA violation appropriately, which means that a warning or an error message may be raised. In case of a GSA customer, 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 OM, if the GSA Violation profile is set to Warning, a warning message is raised. In case it 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

The rounding refers to the rounding of the list/selling price. Rounding is controlled by the rounding_flag on the control_record which is input to the pricing engine and the value of the profile QP: Selling Price Rounding Options. For more information, see: *Oracle Advanced Pricing Implementation Manual,* Profile Options and for information on the values for the rounding flag, see: Oracle Oracle Order Management Suite APIs and Open Interfaces Manual.

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 request line. The freight charge does not affect the selling price. Manual/Overridden charges that need to be applied, need to be passed to the pricing engine with applied_flag set to Y and updated_flag set to Y just like any manual adjustment (discount/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 at a later time. 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.

The coupons are stored in QP_COUPONS table and are marked redeemed when they are redeemed.

In order 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. In order 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. In order to prevent the pricing engine to do 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 OM, in case an order qualifies for a coupon, when the order is cancelled or if the item is returned, the coupon is not deleted. Also 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.

In order to setup an item upgrade modifier, there needs to be a item relationship defined between the buy item and the upgrade item in Inventory item relationships with 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% 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% discount to this new line. In case of buy 1 get 1 free, this discount is 100%.

The new line that has been 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 gets passed as an attribute in the qp_preq_line_attrs_tmp to that line. The discount on the free good line gets passed in the temp table qp_ldets_v. The order line which qualified for the PRG modifier is the parent modifier and the discount on the free good 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 their transaction tables.

During a reprice, the pricing engine public API QP_PREQ_PUB compares the existing free good line and 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 qualify 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 their transaction system.

When the free good 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 good lines. For more information on the implicit call for freight charges for promotional goods, see: Freight Charges for the free good line, page 18-12.

In cases where users do not want the free good line, they can remove it from the order. To make the pricing engine recognize that it does not need to create the free good again, complete the following steps:

1. Delete the free good 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 good 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 freegood lines. The same approach needs to be followed if the calling application wants to substitute the free good 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 freegood lines and the associations between the buy line and the freegood line.

Freight Charges for the free good line

The pricing engine does not apply freight charges on the free good line. To get freight changes for the free good item, a second pricing engine call needs to be made which will insert just the free good line with price_flag QP_PREQ_GRP.G_PHASE and the pricing engine will just search for freight charges for the free good line. The freight charges do not affect the price on the free good line.

The reason why the pricing engine cannot apply the freight charges (if any) on the free good line is because, the pricing engine does not have any attributes mapped for the free good line to look for freight charges. The pricing engine inserts the free good line with the pricing information and the product is the only attribute inserted against this line.

To get the freight charges for free good 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 good lines to the pricing engine.

If there are any line group-based modifiers, the order or quote can qualify for a free good 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:

- 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, we are making sure that the quote/order will qualify for all the applicable modifiers and that the price will not change on subsequent reprice.

For more information on QP_UTIL_PUB.Get_Order_Line_status, see:

Other Item Discount

Other Item discount (OID) behavior is very similar to the free goods or promotional goods as explained above except that the engine does not create a new order line. Other Item Discount is set up as buy item A and get x% discount on item B where both item A and item B are ordered.

This means, to qualify for the OID, both item A as well as 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 this information need to be stored in the calling application's transaction tables.

Pricing from Configurator

The configurator in Order Management (OM)/Order Capture (OC) uses OM/OC APIs to call pricing engine. In case of included items and config items, the unit_price is defaulted to zero. In case the pricing engine is called, the order lines having 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*, else it is *N*. The pricing engine does not calculate the unit_price in either case.

Service Item Pricing

If there is an item, which has price list line set as percentage of price of a parent line; for example, the price of a service item called gold support might be 10% of the price of a parent serviceable item called oracle 8I), pass the parent line in qp_preq_lines_tmp and give the relationship between the parent and the child in qp_preq_rltd_lines_tmp. Set the

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 duration of time, as in price a service item for a duration of 12 Months, pass the duration as uom_quantity in the structure qp_preq_lines_tmp. The unit price would be multiplied by the uom_quantity. Insert the relationships into qp_preq_rltd_lines_tmp.

The pricing engine fetches 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. Remember to delete/pass the service line when the parent line is deleted/updated.

Pricing also provides a time-UOM conversion API. For more information on Time-UOM conversion API, see: Oracle Oracle Order Management Suite APIs and Open Interfaces Manual.

Ask for promotions

The calling application can ask for a promotion to 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.

Terms Substitution

If a request line qualifies for a terms 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 e.g. 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 gets returned, in case there is a attribute mapping rule. Or make sure the old term gets passed to the pricing engine if there are no attribute mapping rules and if the attributes are passed directly.

Also, if the pricing engine does not pass the TSN modifier on subsequent reprice calls if the request line does not qualify for it any more, make sure to replace the old term before deleting the TSN modifier from the transaction table.

Cross Order Volume Based Modifiers

Pricing engine supports cross order based volume modifiers. These modifiers are setup with pricing or qualifier attributes that are operating unit specific. These are seeded attribute mapping rules under the Order Fulfillment PTE. Examples for pricing attributes are Period1 Item Quantity and Period1 Item Amount.

> **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.

There is a concurrent program cross order volume loader which looks into the Order Management tables for the order information and sums up cross order volume total for the a given operating unit and populates the OM tables OE_ITEM_CUST_VOLS_ALL and OE_ITEM_CUST_VOLS_ALL. This concurrent program needs to be run from time to time. For more information, see: *Oracle Advanced Pricing Implementation Manual,* Concurrent programs.

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 cross order volume attribute value the build_contexts API returns.

If the calling application is using a new request_type_code or a different global structure, then they need to do the following to make the pricing engine work for 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 information on the cross order volume attributes and attribute mapping, see: *Oracle Advanced Pricing User's Guide*, Modifiers setup.

Time-UOM Conversion

Pricing supports unit of measure (uom) conversion for price lists. The unit of measure conversion is done only if the primary uom flag in the price list line is checked. To do this conversion the advance pricing uses inventory uom conversion APIs.

This poses a problem when the units of measure are date and time as in MTH (month), YR (year), WK (week) etc. The inventory uom conversion holds a static conversion factor of 30 days per month. Since days is set as base uom, one year is converted to 12.16666… months by the inventory uom conversion API. (365 days/30days), instead of one year being converted to 12 months.

Pricing also provides option to use the OKS (Oracle Contracts) APIs to perform the uom conversion for date and time which does the conversion based on calendar year. This is controlled by a QP profile, QP: Time UOM Conversion.

Set this profile QP: Time UOM Conversion to Standard if you want the pricing engine to use the inventory API to perform the conversion for date/Time and set it to Oracle Contracts to use the APIs provided by Oracle Contracts to do time UOM conversion.

Remember that the effect of this profile set to Oracle Contracts can take place only if the product code 515: Oracle Contracts Service Module is fully Installed or Shared installed. Also, in cases that cause contracts APIs returns pricing quantity being null or 0, the standard inventory uom conversion takes place. These cases are QP: Time UOM Conversion profile value is null or when contracts APIs fail to generate correct pricing quantity, for example, Contract_Start_Date or Contract_End_Date being null, order uom is not time related, order uom code is not valid such as MI or MO.

Promotional Limits

Promotional limits are setup against modifiers. When a pricing engine call is made, the pricing engine qualifies the request line for suitable modifiers. If there are limits setup 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. In order 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 there are errors 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 a 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.

The limit balances need to be updated when an order line is returned, amended or cancelled. In case 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. Make sure you call the following API to update the limits consumption details:

```
QP_UTIL_PUB.Reverse_Limits (p_action_code             IN  VARCHAR
2,
     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 Oracle Order Management Suite APIs and Open Interfaces Manual.

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 Suite APIs and Open Interfaces Manual*. 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 may 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'
);
```

Remember to reprice the request if the price list or the currency changes on the order or quote which was priced earlier because, the pricing engine might return a different price based on the different currency.

Sample scripts to make pricing engine calls:

There are sample scripts to make pricing engine calls. The sample scripts can be found in $QP_TOP/patch/115/sql.

- 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 '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. The calling application to call an API provided by pricing is called QP_UTIL_PUB.Get_Price_List.

- 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) Th
en
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. The 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. There are three flags on the output record of this API:

All_Lines_Flag ('Y' or 'N')
This flag tells the call applications whether all lines should be passed to the engine. The value of this 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 ('Y' or 'N')
When this flag is set to 'Y', only the modified lines should be passed to the engine. This flag is 'Y' when there is only Line Level Modifiers.

Summary_Line_Flag ('Y' or 'N'):
The value of this flag is depended on the pricing setup with Order Level Modifiers. When there is only such a modifier, this flag will be set to 'Y' so that only the Order Summary Line should be passed to the engine.

Scenarios

Case 1
100 lines order with 5 lines changed.

Setup: All_Lines_Flag = 'N', Changed_Lines_Flag = 'Y', Summary_Line_Flag = 'N'

Result: Only 5 lines are passed to the pricing engine.

Case 2
100 lines 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
100 lines order with 5 lines changed 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
100 lines order with 5 lines changed and order level modifier.

Setup: All_Lines_Flag = 'N', Changed_Lines_Flag = 'Y', Summary_Line_Flag = 'Y'

Result: 5 lines and a summary line are passed to the pricing engine.

Case 5
100 lines order with no line changed.

Setup: All_Lines_Flag = 'N', Changed_Lines_Flag = 'N', Summary_Line_Flag = 'N'

Result: No lines are passed to the pricing engine.

Case 6
2 new lines are added to the 100 lines order.

Setup: All_Lines_Flag = 'N', Changed_Lines_Flag = 'Y', Summary_Line_Flag = 'N'

Result: Only the 2 new lines are passed to the pricing engine.

Oracle Service Contracts (OKS) Integration: Proration and Price List Locking

This section describes the usage proration (proration) and price list locking features available when Oracle Service Contracts (OKS) is integrated with Oracle Advanced Pricing (QP). Usage Proration is available to any calling application making usage calls. Price list locking is available only for use by Oracle Service Contracts. 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 (OKS) 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 prior 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 describes the process where 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 locked price list will be created with the name OKS LOCKED Corporate price list. 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. 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.

Related Topics

Usage Price Break Proration, page 7-2

Changes Related to the Proration feature

The Price list window displays two additional columns for price break entry 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, 1-1000 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.

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.
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

This feature is required by Oracle Service Contracts (OKS) to support prorated billing only for 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 Break_UOM+Pricing_attribute1 (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. If the contracts time_uom_conversion profile is set, then pricing will call the Contracts API for UOM conversion. The service_start_date and end_date from qp_preq_lines_temp as well as setup Break UOM, passed_Break_UOM should be passed to the Contracts API.

Note: For calling applications not using OKS, the pricing engine will use the conversion API from Inventory.

4. The API returns the conversion factor which should be used for proration. If the contracts profile is not set, then the standard UOM conversion results. For example, suppose the conversion = 1/3.
5. The Break From/To values will be converted and truncated based on the above conversion as displayed in the following table:

Original From	Modified From	Original To	Modified To
0	0	1000	Original_to * UOM conversionExample: 1000*(1/3) = 333
1001	1000*(1/3)+(original_break2_from - original_break1_to) Example: 1000*(1/3)+(1001-1000) = 334	2000	2000*(1/3) = 666
2001	667	3000	3000*(1/3) = 1000

6. The preceding Break From/To is used for break evaluation, and the pricing engine returns the setup Break UOM to the calling application.
7. OKS needs to pass context = BreakUOM, Attribute = Pricing_Attribute1, Value = Billing UOM to the pricing engine and pricing engine will calculate the proration.

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'.

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

  1001 - 2000, Value = 90

  2001 - 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, Value = 100
- 334 - 666, Value = 90
- 667 - 1000, Value = 80

Example 2

Profile QP: Break UOM Proration Allowed = 'N'

Set up a price break price list line for an item, such as AS999;

- Break UOM = QTR (Quarter)
- Break Type = POINT
- Break Lines for Volume Attribute Item Quantity:

  0 - 1000, Value = 100

  1001 - 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:

- Pricing Context = BREAK_UOM
- Attribute = PRICING_ATTRIBUTE1 and value = MTH (Month)

Expected Results:

The unit_price returned from pricing engine should be 100 (for line quantity 500) with no proration as profile is set to N.

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 = RANGE
- Break lines for Volume Attribute Item Quantity:

  0 - 1000, Value = 100

  1001 - 2000, Value = 90

2001 - 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' (i.e. 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, Value = 100
- 334 - 666, Value = 90
- 667 - 1000, Value = 80

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

  1001 - 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 5

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

  1001 - 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 6

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

  1001 - 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.

19

High Volume Order Processing

This chapter covers the following topics:

Overview of High Volume Order Processing (HVOP)

High Volume Order Processing (HVOP) enables to 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

High Volume Order Processing (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 *Oracle Order Management Open Interfaces, API,& Electronic Messaging Guide* or sample implementation for details.

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

20 Technical Considerations

This chapter covers the following topics:

- Basic versus Oracle Advanced Pricing
- Oracle Advanced Pricing Engine Processing
- Extendibility Features
- Performance Tuning Overview
- Diagnostics and Troubleshooting
- Summary of Pricing Engine Messages and Diagnosis
- Common Troubleshooting Problems in Pricing Setup windows

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 where 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

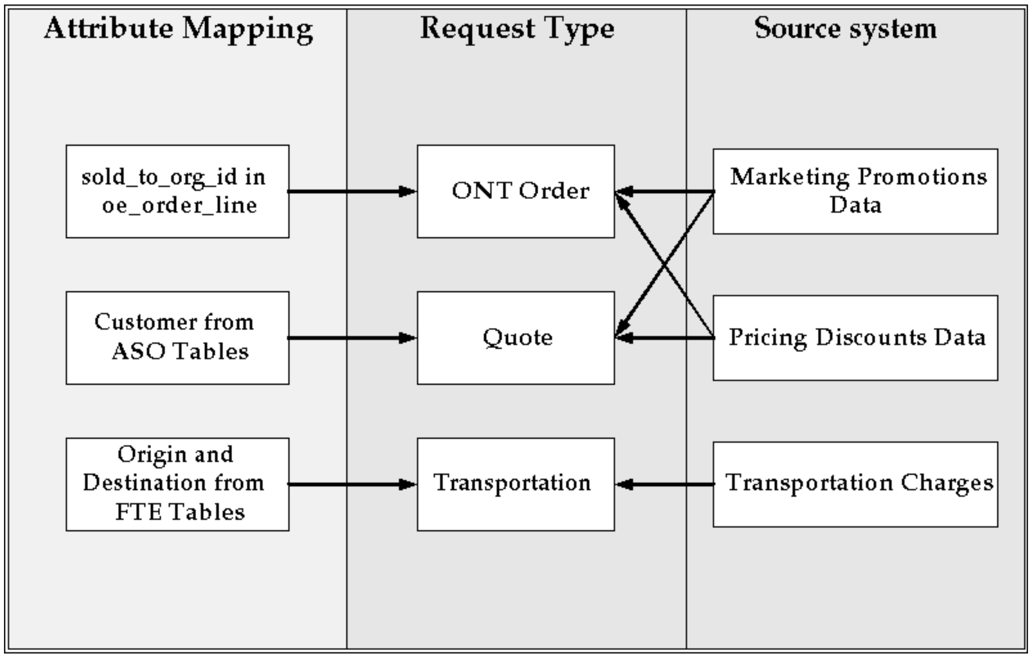
The search engine must know the data source to determine whether it is appropriate to price a transaction. For example, only price lists set up to price contracts should be considered when deriving a price for a contract line.

A source system identifier, that identifies the application which created the pricing data, is recorded on all price and modifier lists. Examples of this include Oracle Order Management, Oracle Marketing, Oracle Service Contracts, and iOracle iMarketing.

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/or price adjustments, benefits, or charges to be applied to the whole transaction or to specific transaction lines.

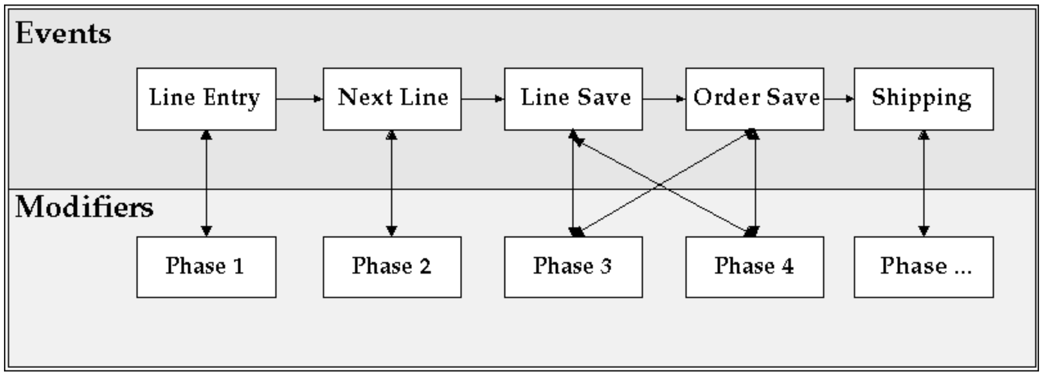
A pricing phase controls which modifiers are considered by the search engine and in what sequence they should be applied to the request. Some pricing phase attributes determine which modifiers can be placed in a phase. For example, if the list type of the pricing phase is charges, only list lines of 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 >= and <=)
- 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% discount is given provided that 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	99999999999

If the requirement is to give a 10% discount provided that 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	99999999999

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% 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	99999999999
2	Website domain	=	.co.uk	Not applicable

Pricing Attributes: Selecting What is Priced

Pricing attributes are user-defined attributes which define what is being priced or modified. Attributes may include factors which 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% discount
    2. 6 - 10 students, 12% discount
    3. 10 or more students, 15% discount
- All contact lenses are priced at $2.25 per pair.

Pricing attributes are defined in the Oracle Advanced Pricing descriptive flexfield - Pricing Contexts. Creating pricing attributes in different contexts allows attributes to be grouped according to their business use. The item context 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 a segment in the item context in the pricing context descriptive flexfield.

The segments are ordered to reflect the structure of the product pricing hierarchy, with the lowest level in the hierarchy having the lowest flexfield segment 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 flexfield represents a flattened view of the product pricing hierarchy.

Oracle Advanced Pricing includes the following seeded product hierarchy:

All (all products):	Enabled
Segment 1 - segment 20 item category flexfield:	Disabled
Item category key flexfield:	Enabled
Item	Enabled

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 pricing hierarchy but it excludes lower levels in the hierarchy from that modifier. In the product pricing hierarchy defined previously, if a 10% 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 which 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 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). There must always be a primary item in the modifier which can be combined with groups of other items. In the above 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.

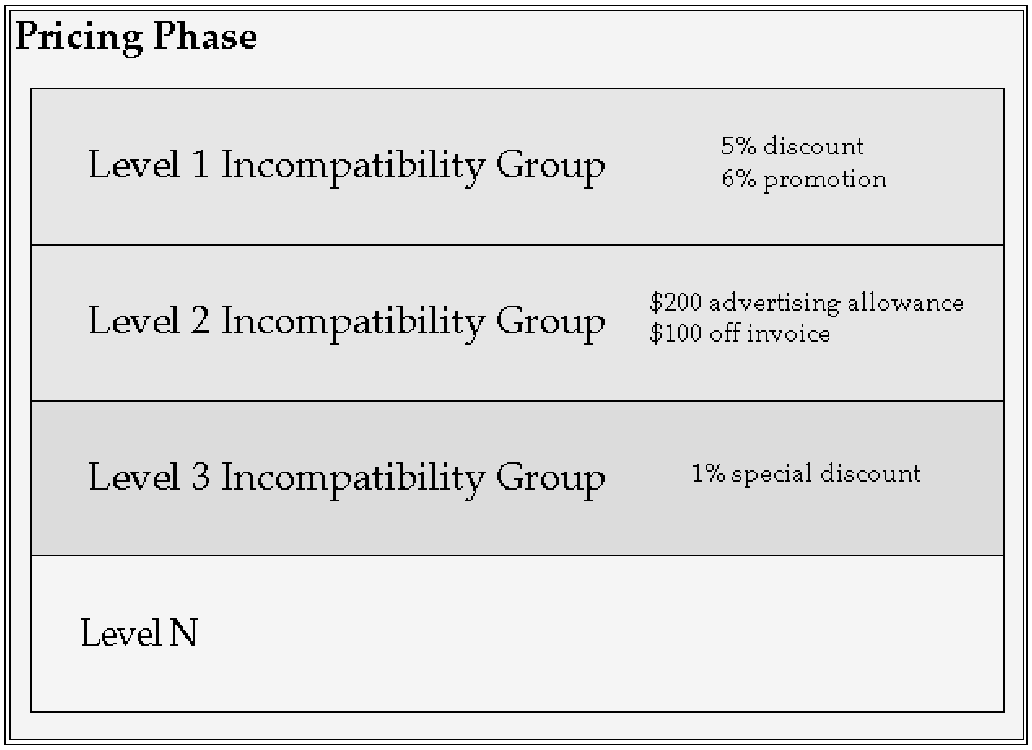
Additional modifiers may not be applied to a pricing request line if an exclusive modifier is applied to the request line in the same pricing phase. For example, a customer receives a 15% discount on an item and is not eligible for any other discount on the item.

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% discount on a tennis racquet may not be given if the customer is also eligible for “Summer Sports Promotion” that gives a 6% discount on all sporting goods.

Incompatibility between discounts is defined between modifiers at the same level in a discount application hierarchy. This is illustrated in the following image. In the image, Level 1 is base level discounts, Level 2 is specially negotiated discounts, and Level 3 is retrospective discounts/accruals.

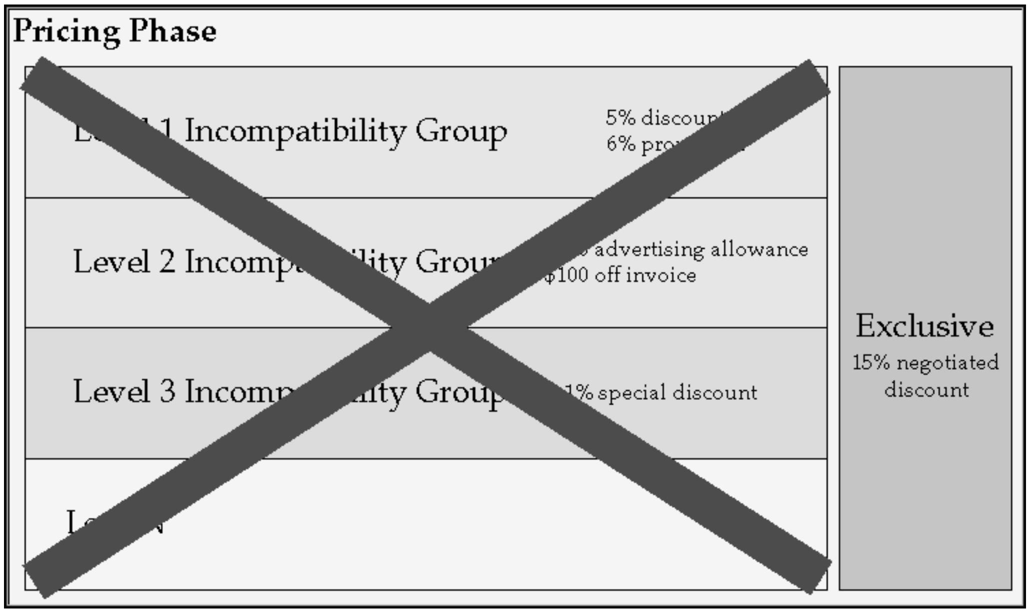
If the discounts of Levels 1 through 4 in the following image are set for the same product family (and therefore applied to the same pricing request line) a customer entitled to receive all of these discounts only receives one discount from each incompatibility group. For example, the customer could be eligible for a 5% discount, a 1% special discount, and a $100 off invoice.

Incompatibility Groups for a Pricing Phase


Pricing Phase
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% exclusive discount for the same product family and the search engine determines that this can be applied to the pricing request line, this is the only discount applied to the line in the current pricing phase. None of the discounts in Levels 1 through N are considered. This concept is illustrated in the following image.

Incompatibility Groups and Exclusivity rules



> **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:

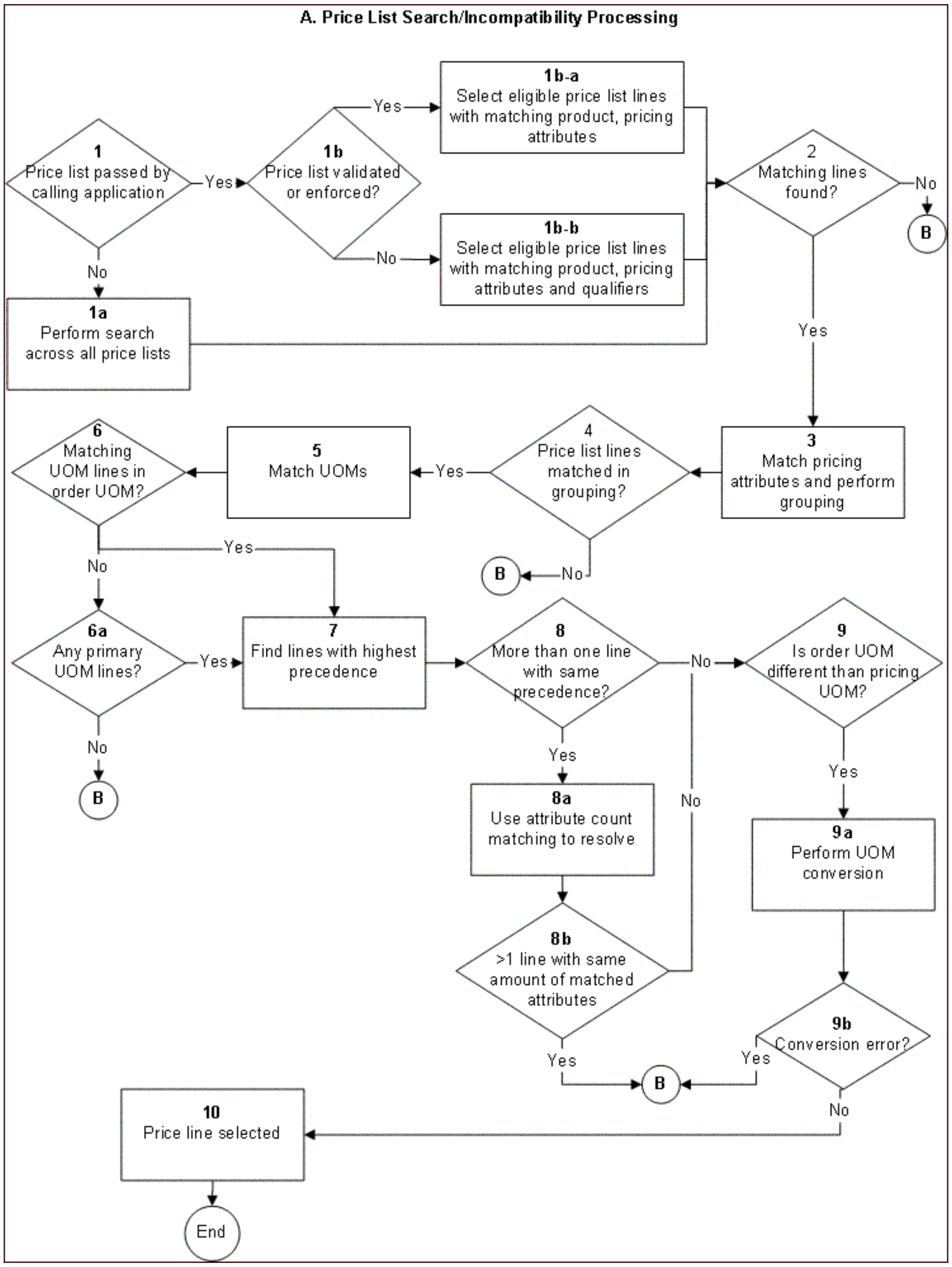
- Yes leads to Box 1a: Perform search across all price lists. Box 1a then progresses to Box 2.
- A no result for Box 1 leads to Box 1b, which asks whether the price list has been validated/enforced. Yes leads to Box 1b-a: Select eligible price list lines with matching product and pricing attributes. 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 there are matching UOM lines in the order UOM. Yes leads Box 7. No leads to Box 6a which asks whether there are any primary UOM lines. 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 there is more than one line with 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 there is more than one line with the same amount 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 there is a conversion error. 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


A. Price List Search/Incompatibility Processing
1 Price list passed by calling application
Yes
No
1a Perform search across all price lists
1b Price list validated or enforced?
Yes
No
1b-a Select eligible price list lines with matching product, pricing attributes
1b-b Select eligible price list lines with matching product, pricing attributes and qualifiers
2 Matching lines found?
No
B
Yes
3 Match pricing attributes and perform grouping
4 Price list lines matched in grouping?
Yes
No
B
5 Match UOMs
6 Matching UOM lines in order UOM?
Yes
No
6a Any primary UOM lines?
Yes
No
B
7 Find lines with highest precedence
8 More than one line with same precedence?
No
Yes
8a Use attribute count matching to resolve
8b >1 line with same amount of matched attributes
No
Yes
B
9 Is order UOM different than pricing UOM?
Yes
9a Perform UOM conversion
9b Conversion error?
Yes
No
10 Price line selected
End

The following image depicts price list search/incompatibility processing, part B. It contains diamond boxes, which represent decisions, and rectangular boxes, that represent processes.

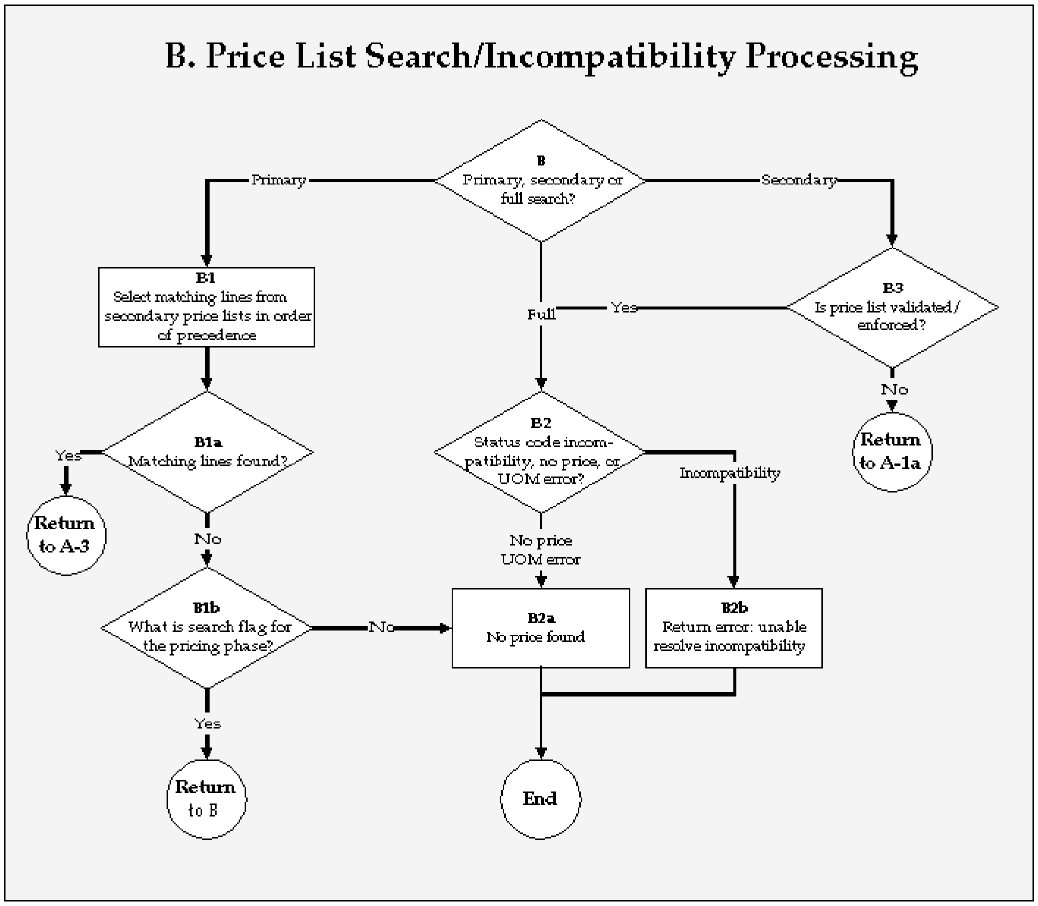
Box B asks if the search is primary, secondary or full.

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/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


B. Price List Search/Incompatibility Processing
B
Primary, secondary or full search?
Primary
Secondary
Full
B1
Select matching lines from secondary price lists in order of precedence
B3
Is price list validated/ enforced?
Yes
No
Return to A-1a
B1a
Matching lines found?
Yes
Return to A-3
No
B2
Status code incompatibility, no price, or UOM error?
Incompatibility
No price
UOM error
B1b
What is search flag for the pricing phase?
No
Yes
Return to B
B2a
No price found
B2b
Return error: unable resolve incompatibility
End

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. There are two methods which the search engine uses 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 depicts modifiers search/incompatibility processing. It contains diamond boxes, that 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, 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.

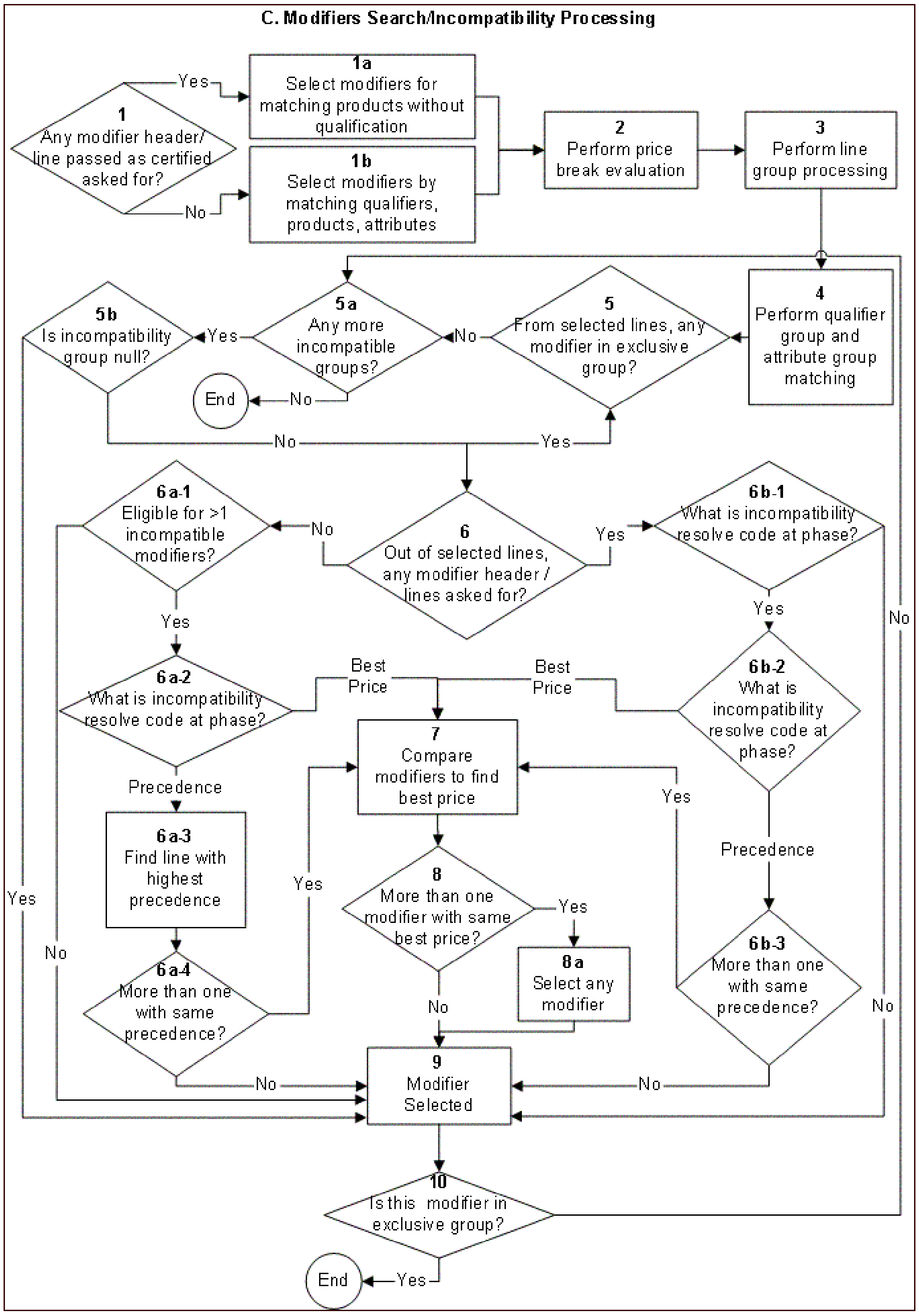
Box 4 leads to Box 5, which asks whether any modifiers of the selected lines are in an exclusive group. Yes leads to Box 6. No leads to Box 5a, which asks if there are any more incompatible groups. If no, then the process is complete. Yes leads to Box 5b which asks if the incompatibility group is null. 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, that 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, that asks for the incompatibility resolve code at phase. Best price leads to Box 7. Precedence leads to Box 6b-3. which asks whether there is more than one line with 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 that asks if there is more than one modifier with 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


C. Modifiers Search/Incompatibility Processing
1 Any modifier header/ line passed as certified asked for?
Yes
No
1a Select modifiers for matching products without qualification
1b Select modifiers by matching qualifiers, products, attributes
2 Perform price break evaluation
3 Perform line group processing
4 Perform qualifier group and attribute group matching
5 From selected lines, any modifier in exclusive group?
5a Any more incompatible groups?
5b Is incompatibility group null?
End
6 Out of selected lines, any modifier header / lines asked for?
6a-1 Eligible for >1 incompatible modifiers?
6a-2 What is incompatibility resolve code at phase?
6a-3 Find line with highest precedence
6a-4 More than one with same precedence?
6b-1 What is incompatibility resolve code at phase?
6b-2 What is incompatibility resolve code at phase?
6b-3 More than one with same precedence?
Best Price
Precedence
7 Compare modifiers to find best price
8 More than one modifier with same best price?
8a Select any modifier
9 Modifier Selected
10 Is this modifier in exclusive group?
End

Precedence and Specificity

The search engine always chooses the modifier which is the most specific; the modifier with the lowest precedence number is selected.

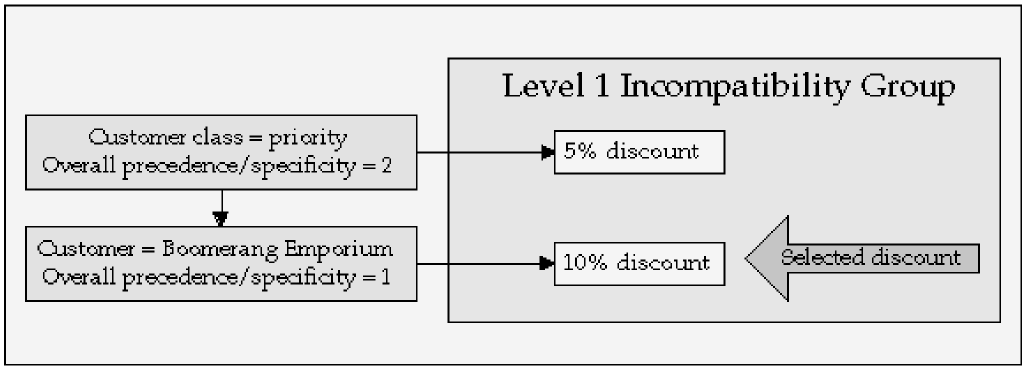
For example, all priority customers get a 5% discount on Product Family A. Customer Boomerang Emporium, although a priority customer, has negotiated a 10% discount on the same product family and is therefore not eligible to receive the 5% discount.

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 is the most specific discount, in this case the Boomerang Emporium negotiated discount rather 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 given, 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% discount	2 (customer class)	3	2
10% Boomerang Emporium discount	1 (customer)	3	1

The search engine selects the 10% Boomerang Emporium discount, as it has the lowest precedence (the most specific discount). This concept is illustrated in the following image.

Precedence and incompatibility


Level 1 Incompatibility Group
Customer class = priority
Overall precedence/specificity = 2
5% discount
Customer = Boomerang Emporium
Overall precedence/specificity = 1
10% discount
Selected discount

At time of setup, qualifier or product attribute precedence is defaulted from the sequence number in the qualifier and pricing descriptive flexfields. The sequence of the qualifier and product segments in and across contexts determines 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 flexfield, the item context in the pricing descriptive flexfield, and the most specific qualifiers and product attributes having the lower segment sequence numbers.

For example, when defining the customer pricing hierarchy, the lowest level in the hierarchy has the lowest flexfield segment sequence number. If the search engine needed to choose between a price for an item negotiated with a specific customer and a price for the same item which was given to an entire customer class, the engine would select the customer-specific price. The customer qualifier is more specific (assigned a lower sequence number) than the customer class qualifier. This is how the customer pricing hierarchy is flattened across the qualifier descriptive flexfield and how the product hierarchy is defined in the item context of the pricing descriptive flexfield.

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.

This type of incompatibility resolution is used only when the monetary value of the modifier is easily estimated, and is not used for price list lines and some types of modifiers. The table below depicts this concept.

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 estimated discount value and is list price *%.
Terms substitution	Estimated discount value of modifier such as Comparison Value. If not provided then best price is set to zero.
Item upgrade	Estimated discount value of modifier such as Comparison Value. If not provided then best price is set to zero.
Coupon issue	Estimated discount value of modifier such as Comparison Value. If not provided then best price is set to zero.
Other Item discount	Not included - always valued at zero
Promotional goods	Not included - always valued at zero
Freight Charges	Same as discount and surcharges.
Price list lines	Not included

The count of matched pricing attributes is considered if two or more price list lines are matched and have the same precedence. In this case the price list line with higher count is selected by the engine.

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 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(s).

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, etc.) 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 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 Manual,* Integration chapter.

Pricing Engine Flow

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)
```

Pricing Engine Code

```
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 Pha
ses)
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
```

Invoking Application Integration Code

- Compare the returned modifiers with existing adjustments and perform update if necessary.
- Call local calculation engine to get selling price.

For more information, see: Oracle Technical White Paper *Integrating with R11i Advanced Pricing.*

Extendibility Features

Oracle Advanced Pricing contains a number of extendibility features that are documented in the Oracle Technical White Paper *Extending 11i Advanced Pricing for Your Business - Part 1: Using Attributes Mapping.*

Oracle Advanced Pricing APIs

Oracle Advanced Pricing contains several 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 Oracle*MetaLink*. These can be studied and modified for specific customer needs. For a complete listing of Oracle Advanced Pricing public APIs, see: *Oracle Manufacturing Suite APIs and Open Interfaces Manual.*

Performance Tuning Overview

Meeting performance requirements of e-businesses has been a major design objective throughout Oracle Advanced Pricing development. Significant effort has been applied to optimizing the Oracle Advanced Pricing engine call. Advanced tuning techniques have been applied in the pricing engine. There are several recommendations that, if followed, ensure optimal performance when using Oracle Advanced Pricing.

Related Topics

Optimal Performance, page D-1

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.

Diagnostics and troubleshooting

Diagnosing Method	When to use
Pricing Engine Request Viewer window in Sales Order Pad. How to use: 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.	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). This is a quick method to verify the data passed to the pricing engine and data that was returned by the pricing engine. This method determines whether the expected qualifiers and pricing attribute are sourced. Since the Pricing Engine Request Viewer requests are stored in the permanent pricing debug tables, users can query previous pricing debug requests.

Diagnosing Method	When to use
Debug output How to for Oracle Order Management users: Set OM: Debug Directory to directory listed in util_file_dir in init.ora. Set OM: Debug_Level to at least 5. How to for other users: Set profile QP:Debug to Yes. Set OM:Debug Directory to directory listed in util_file_dir in init.ora. Search the output file in the directory mentioned in above profile based on time-stamp.	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. **Warning:** 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.
script: qp_list_line_detail.sql How to Use: Get the price list line ID/modifier line ID from the Price List/ Modifier setup window. Open the Price List/Modifiers window. Select the price list line/modifier line: Help > Diagnostics > Examine > Pick the LIST_LINE_ID field from the Field LOV Note the value. This list_line_id value must be provided as input to the script qp_list_line_detail.sql. Login to apps/apps@sid. 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. Make sure that the script outputs a <list line id>.lst file. Alternately, you can run this script as a concurrent program and view the output in the concurrent request output file: 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.	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. When the user knows the specific price list line/ modifier line that should be selected by the engine, but it is not selected. When the possibility exists that the denormalized columns are not being properly updated due to unusual user exceptions.

Diagnosing Method	When to use
Verify setup using Oracle Advanced Pricing set up windows	Certain setup information has an impact on pricing engine process. Verify that the following columns have appropriate values: Incompatibility Resolve Code in event phase Search_flag in event phase Automatic_flag, active_flag in modifiers and price lists Pricing Phase in modifier Certain profile values impact pricing engine processes. Verify that the following profiles are appropriately set: QP: Get Custom Price Customized QP: Blind Discount Option QP: Verify GSA Violations QP: Return Manual Discounts
Verify engine control record and other record structure	For power users: If you call the pricing engine directly, refer to the *Oracle Manufacturing Suite APIs and Open Interfaces Manual* 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.

Item and UOM not on Price List

Probable Cause	How to Debug
The price list header is inactive.	Check Active box on price list header.
The price list header is ineffective as of the pricing date.	Check 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 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.	Check 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 use of that price list, resulting 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 sourced.
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.
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 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.	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. 2. Price list header currency is different from the order currency. Check currency.

Cannot Resolve Incompatibility Between Price List X and Price list Y

Probable Cause	How to Debug
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 there are qualifiers for the passed-in price lists. Also verify that the price list is active. If the user intends 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 then update the precedence appropriately.

Invalid UOM

Probable Cause	How to Debug
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.
There is no other matching price list line having primary UOM flag checked.	If user intends to define price list in primary UOM and expects the engine to convert pricing quantity, then verify that the primary flag of the price list line is checked.

Invalid UOM Conversion

Probable Cause	How to Debug
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 getting selected based on the previous table: Item and UOM not on Price List.
There is no conversion defined in mtl_uom_ conversions, between ordered UOM and primary UOM.	If user expects pricing to convert the pricing quantity from ordered UOM to the pricing UOM, verify that proper conversion is defined in Oracle Inventory.

Invalid Formula, Error Returned by QP_FORMULA_PRICE_CALC_PVT.Calculate

Probable Cause	How to Debug
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 there is a matching factor line.
Formula is not a valid mathematical expression supported by the database sql.	Verify that the formula is a valid expression.
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 formula exists and effective as of pricing date.
Formula does not have at least one component.	Verify that Formula has at least one component.
QP_CUSTOM.Get_Custom_Price( ) function does not exist or 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 getting non-numeric values whereas it is expecting a numeric value.	a) Ensure that the pricing attributes used in the formula calculation always have the number valueset attached to it. The steps to verify are: 1. Go to Oracle Pricing Responsibility. 2. Go to Setup > Attribute Management > Context and Attributes. 3. Query the context and attribute used in the formula setup. 4. For that attribute, select if any number valueset is attached. If not attach any number valueset to it. b) Another way to check is to change the formula by making use of the function TO_NUMBER. This works only if the attribute value returned has only numeric characters. It will fail if any alphabetic characters are present in the attribute value.
There could be undefined step numbers in the formula.	Verify that all the step numbers in the formula has been 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: *Item and UOM not on Price List*.
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, then execute 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 there is any modifier in the exclusive group.
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.
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

Probable Cause	How to Debug
Incompatibility resolution code is set incorrectly.	Refer to previous table: *Item and UOM not on Price List*.
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

Probable Cause	How to Debug
Manual discounts are not available.	Save the order line or move the cursor out of the line and back; this causes Oracle Order Management to fire the pricing engine modifier phase.
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.	Overtype the unit selling price and increase the price. This applies overridable surcharges, whereas decreasing the price applies overridable 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 is caused due to 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 qp_qualifiers table. If user has selected ALL headers, then the program takes time.

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.	Execute 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 occurs when someone makes 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).	Execute 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 there is a patch available to correct the seeded rule. If the error is "Encountered the symbol '_' when expecting…" then determine the relevant patch to be applied.

Freight Charges Integration Messages and Errors

Cost to Charge Conversion is not Returned by the Engine

Probable Cause	How to Debug
There may be qualifiers 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 setup 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 if the freight charge that you expect is the maximum freight charge or 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 Setup windows

Price List Problems

The following is a list of possible problems and tips on how to solve them.

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
- QPXPACRL.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 occurs 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 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 checked. 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 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 5

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

Problem

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 which has currency as USD and 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.

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 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. Execute 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. Execute 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 beginning to apply the patch.

A
Windows and Navigator Paths

This appendix covers the following topics:

- Windows and Navigator Paths

Windows and Navigator Paths

Refer to the following sources for other window and application information:

- *Oracle Order Management User's Guide*
- *Oracle Application User's Guide*

The following table lists the navigation path for each window or HTML page in Oracle Advanced Pricing. The term SSWA (Self Service Web Application) identifies a navigation path in the HTML user interface. Square brackets indicate a [button name or link] that must be clicked to display the window or HTML page.

Window Name	Navigation Path
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 (alternate)	Modifiers > Modifier Incompatibility Setup > [Modifiers]
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] **Alternate**: SSWA > Oracle Pricing Administrator >Price List Maintenance > [Advanced Search]
Archive Pricing Entities	Archive Purge Pricing Entities > Archive Entity
Assign Attributes	Setup > Attribute Management > Attribute Linking and Mapping > [Assign Attributes]

Window Name	Navigation Path
Attribute Defaulting Rules	Setup > Attribute Mapping > [Defaulting Rules...]
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] **Alternate**: 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]
Context Setup	Setup > Attribute Mapping > Context and Attributes
Copy Modifiers	Modifiers > Copy Modifiers
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 List: General Information (HTML UI)	SSWA > Oracle Pricing User responsibility > Home > Oracle Advanced Pricing: Home page [Shortcuts: Create Price List] **Alternate**: SSWA > Oracle Pricing User responsibility > Home > Oracle Advanced Pricing: Home page [Lists tab] > Create List
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] **Alternate**: SSWA > Oracle Pricing User responsibility > Home > Oracle Advanced Pricing: Home page [Lists tab] > Create List > Create Price List: General Information page [Next]
Defaulting Condition Validation Templates	Setup > Attribute Mapping > [Defaulting Condition Templates...]
Defaulting Setup	Setup > Attribute Mapping

Window Name	Navigation Path
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] **Alternate**: 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
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
Find Requests (alternate)	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]

Window Name	Navigation Path
Log file: request id	View Concurrent Requests > [Find] > [View Log...]
Modifier Lists	SSWA > Oracle Pricing User > Home > Lists {T}, Administrator Responsibility > Modifier Lists
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
Oracle Pricing Lookups	Setup > Lookups
Personal Profile Values	Setup > Profiles > [Find]
Price Breaks	Pricing Agreements > Price Breaks
Price Breaks (alternate 1)	Modifiers > Modifier Setup > [Define Details*] > Price Breaks
Price Breaks (alternate 2)	Price Lists > Price List Setup > [Price Breaks]
Price List Maintenance (HTML UI)	SSWA > Oracle Pricing Administrator Responsibility > Price List Maintenance **Alternate**: 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
Pricing Agreements	Pricing Agreements
Pricing Attributes	Pricing Agreements > [Pricing Attributes]
Pricing Attributes (alternate)	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]

Window Name	Navigation Path
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]
Qualifiers (HTML UI)	SSWA > Oracle Pricing User responsibility > Find or create Modifier/Price List > List Qualifiers > Qualifiers page [Append Group] **Alternate**: 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]
Segments Summary (Attachment context)	Setup > FlexFields > [Segments]
Segments Summary (alternate 1)	Setups > FlexFields > [Segments] > [New]
Segments Summary (alternate 2)	Setups > FlexFields > [Segments] > [Open]
Source Systems	Setups > Source Systems
Submit a New Request	Reports
Submit a New Request (alternate)	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)	SSWA > Oracle Pricing User responsibility > Lists > Modifier Lists > Modifier Lists Search page (complete search for modifier) > [Update] > Modifier Lines (from Navigation bar) > [Update] **Alternate**: SSWA > Oracle Pricing User responsibility > Home page (Recently Created Modifier Lists) > [Update] > Modifier Lines (from Navigation bar) > [Update]
Update (Modifier Type): Modifier Lines (HTML UI)	SSWA > Oracle Pricing User responsibility > Lists > Modifier Lists > Modifier Lists Search page (complete search for modifier) > [Update] > Modifier Lines (from Navigation bar) **Alternate**: SSWA > Oracle Pricing User responsibility > Home page (Recently Created Modifier Lists) > [Update] > Modifier Lines (from Navigation bar)
Update (Modifier Type): General Information (HTML UI)	SSWA > Oracle Pricing User responsibility > Modifier Lists Search page (after completing a search) > [Update] **Alternate**: SSWA > Oracle Pricing User responsibility > Home page (Recently Created Modifier Lists) > [Update]
Value Sets	Setups > FlexFields > [Segments] > [Value Set]
Value Sets (alternate 1)	Setups > FlexFields > [Segments] > [New] > [Value Set]
Value Sets (alternate 2)	Setups > FlexFields > [Segments] > [Open] > [Value Set]

B
Attribute Seed Data

This appendix covers the following topics:

- Overview of Attribute Seed Data
- Complex Maintenance Repair and Overhaul PTE Attributes
- Demand Planning PTE Attributes
- Intercompany Transaction PTE Attributes
- Logistics PTE Attributes
- Order Fulfillment PTE Attributes
- Procurement PTE Attributes

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-4
- Intercompany Transaction PTE Attributes, page B-5
- Logistics PTE Attributes, page B-8
- Order Fulfillment PTE Attributes, page B-11
- Procurement PTE Attributes, page B-38

To find an attribute(s), 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.

A key of the short names and definitions used in the attribute tables are provided in the following table:

Short name	Definition
AM	Attribute Mapping
AMM	Attribute Mapping Method
Attrib	Attribute
ASO	Oracle Order Capture
CMRO	Complex Maintenance Repair and Overhaul
Lmt	Limits
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

The following section displays the attributes for the Complex Maintenance Repair and Overhaul (CMRO) Pricing Transaction Entity (PTE).

Pricing Attributes

There are no seeded pricing attributes for this PTE.

Product Attributes

The following table lists the seeded product attributes for the Complex Maintenance Repair and Overhaul PTE:

Context	Attribute	Prec	Mapped	Value Name	Level	AMM	Lmt	Req Type
ITEM	Item Number	220	PRIC ING_ ATTR IBUTE1	_	LINE	UE	Y	ASO
ITEM	Item Number	220	PRIC ING_ ATTR IBUTE1	_	LINE	UE	Y	OKC
ITEM	Item Number	220	PRIC ING_ ATTR IBUTE1	_	LINE	UE	Y	ONT
ITEM	Item Number	220	PRIC ING_ ATTR IBUTE1	_	LINE	UE	Y	IC
ITEM	Item Number	220	PRIC ING_ ATTR IBUTE1	_	LINE	UE	Y	PO

Qualifier Attributes

The following table lists the seeded qualifier attributes for the Complex Maintenance Repair and Overhaul PTE:

Context	Attribute	Prec	Mapped	Value Name	Level	AMM	Lmt	Req Type
CUSTOMER	Party ID	360	QUAL IFIER_ ATTR IBUTE16	QP_ PARTY	BOTH	AM	Y	ASO
CUSTOMER	Party ID	360	QUAL IFIER_ ATTR IBUTE16	QP_ PARTY	BOTH	AM	Y	OKC
CUSTOMER	Party ID	360	QUAL IFIER_ ATTR IBUTE16	QP_ PARTY	BOTH	AM	Y	ONT
CUSTOMER	Party ID	360	QUAL IFIER_ ATTR IBUTE16	QP_ PARTY	BOTH	AM	Y	ONT
CUSTOMER	Party ID	360	QUAL IFIER_ ATTR IBUTE16	QP_ PARTY	BOTH	AM	Y	OKC
CUSTOMER	Party ID	360	QUAL IFIER_ ATTR IBUTE16	QP_ PARTY	BOTH	AM	Y	ASO

Demand Planning PTE Attributes

The following section displays the attributes for the Demand Planning Pricing Transaction Entity (PTE).

Pricing Attributes

There are no seeded pricing attributes for this PTE.

Product Attributes

There are no seeded product attributes for this PTE.

Qualifier Attributes

The following table lists the seeded qualifier attributes for the Demand Planning PTE:

Context	Attrib	Prec	Mapped	Value Name	Level	AMM	Lmt	Req Type
ORDER	Blanket Number	600	QUAL IFIER_ ATTR IBUTE3	QP_ BLANKET_ NUMBER	LINE	UE	Y	ASO
ORDER	Blanket Number	600	QUAL IFIER_ ATTR IBUTE3	QP_ BLANKET_ NUMBER	LINE	UE	Y	ASO
ORDER	Blanket Number	600	QUAL IFIER_ ATTR IBUTE3	QP_ BLANKET_ NUMBER	LINE	UE	Y	ONT
ORDER	Blanket Number	600	QUAL IFIER_ ATTR IBUTE3	QP_ BLANKET_ NUMBER	LINE	UE	Y	ONT
PARTY	SALES ORGAN IZAT ION	170	QUAL IFIER_ ATTR IBUTE3	QP_ SALES_ ORGAN IZAT ION	LINE	UE	Y	ONT
PARTY	SALES ORGAN IZAT ION	170	QUAL IFIER_ ATTR IBUTE3	QP_ SALES_ ORGAN IZAT ION	LINE	UE	Y	ONT

Intercompany Transaction PTE Attributes

The following section displays the attributes for the Intercompany Transaction Pricing Transaction Entity (PTE).

Pricing Attributes

There are no seeded pricing attributes for this PTE.

Product Attributes

The following table lists the seeded product attributes for the Intercompany Transaction PTE:

Context	Attribute	Prec	Mapped	Value Name	Level	AMM	Lmt	Req Type
ITEM	Item Number	220	PRICING_ATTRIBUTE1	_	LINE	AM	Y	ASO
ITEM	Item Number	220	PRICING_ATTRIBUTE1	_	LINE	AM	Y	ONT
ITEM	Item Number	220	PRICING_ATTRIBUTE1	_	LINE	AM	Y	OKC
ITEM	Item Number	220	PRICING_ATTRIBUTE1	_	LINE	AM	Y	IC
ITEM	Item Number	220	PRICING_ATTRIBUTE1	_	LINE	AM	Y	PO

Qualifier Attributes

The following table lists the seeded qualifier attributes for the Intercompany Transaction PTE:

Context	Attribute	Prec	Mapped	Value Name	Level	AMM	Lmt	Req Type
GLOBAL_PROCUREMENT	Procuring Operating Unit	1	QUALIFIER_ATTRIBUTE1	HR_OPERATING_UNITS	LINE	AM	Y	IC
GLOBAL_PROCUREMENT	Receiving Operating Unit	2	QUALIFIER_ATTRIBUTE2	HR_OPERATING_UNITS	LINE	AM	Y	IC
GLOBAL_PROCUREMENT	Vendor Id	3	QUALIFIER_ATTRIBUTE3	Number	LINE	AM	Y	IC
GLOBAL_PROCUREMENT	Vendor Site Id	4	QUALIFIER_ATTRIBUTE4	Number	LINE	AM	Y	IC
INTERCOMPANY_INVOICING	Customer Id	3	QUALIFIER_ATTRIBUTE3	INV_IC_CUSTOMER	LINE	AM	Y	IC

Context	Attribute	Prec	Mapped	Value Name	Level	AMM	Lmt	Req Type
INTERCOMPANY_INVOICING	Customer Site Id	4	QUALIFIER_ATTRIBUTE4	INV_IC_CUSTOMER_SITE	LINE	AM	Y	IC
INTERCOMPANY_INVOICING	Selling Organization Id	2	QUALIFIER_ATTRIBUTE2	HR_OPERATING_UNITS	LINE	AM	Y	IC
INTERCOMPANY_INVOICING	Shipping organization Id	1	QUALIFIER_ATTRIBUTE1	HR_OPERATING_UNITS	LINE	AM	Y	IC
ORDER	Blanket Number	600	QUALIFIER_ATTRIBUTE3	QP_BLANKET_NUMBER	LINE	UE	Y	ASO
ORDER	Blanket Number	600	QUALIFIER_ATTRIBUTE3	QP_BLANKET_NUMBER	LINE	UE	Y	ASO
ORDER	Blanket Number	600	QUALIFIER_ATTRIBUTE3	QP_BLANKET_NUMBER	LINE	UE	Y	ONT
ORDER	Blanket Number	600	QUALIFIER_ATTRIBUTE3	QP_BLANKET_NUMBER	LINE	UE	Y	ONT
ORDER	Order Date	510	QUALIFIER_ATTRIBUTE1	QP: Date	BOTH	AM	Y	ASO
ORDER	Order Date	510	QUALIFIER_ATTRIBUTE1	QP: Date	BOTH	AM	Y	ASO
ORDER	Order Date	510	QUALIFIER_ATTRIBUTE1	QP: Date	BOTH	AM	Y	OKC
ORDER	Order Date	510	QUALIFIER_ATTRIBUTE1	QP: Date	BOTH	AM	Y	ONT
ORDER	Order Date	510	QUALIFIER_ATTRIBUTE1	QP: Date	BOTH	AM	Y	OKC

Context	Attribute	Prec	Mapped	Value Name	Level	AMM	Lmt	Req Type
ORDER	Order Date	510	QUALIFIER_ATTRIBUTE1	QP: Date	BOTH	AM	Y	IC
ORDER	Order Date	510	QUALIFIER_ATTRIBUTE1	QP: Date	BOTH	AM	Y	ONT
PARTY	SALES ORGANIZATION	170	QUALIFIER_ATTRIBUTE3	QP_SALES_ORGANIZATION	LINE	UE	Y	ONT
PARTY	SALES ORGANIZATION	170	QUALIFIER_ATTRIBUTE3	QP_SALES_ORGANIZATION	LINE	UE	Y	ONT

Logistics PTE Attributes

The following table lists the seeded pricing attributes for the Logistics Pricing Transaction Entity (PTE):

Pricing Attributes

The following table lists the seeded pricing attributes for the Logistics PTE:

Context	Attributes	Prec	Mapped	Value Name	Level	AMM	Lmt
Logistics	Multipiece Flag	5	PRICING_ATTRIBUTE10	QP: Yes/No	Line	UE	Yes
Logistics	Commodity	10	PRICING_ATTRIBUTE1	QP: Number	Line	UE	Yes
Logistics	Container Type	20	PRICING_ATTRIBUTE2	QP_CONTAINER_TYPE	Line	UE	Yes
Logistics	Service Type	30	PRICING_ATTRIBUTE3	QP_SERVICE_TYPE	Line	UE	Yes
Logistics	Additional Service	40	PRICING_ATTRIBUTE4	QP_ADDITIONAL_SERVICE	Line	UE	Yes

Context	Attributes	Prec	Mapped	Value Name	Level	AMM	Lmt
Logistics	HAZARD_ CODE	50	PRIC ING_ ATTR IBUTE5	—	Line	UE	Yes
Logistics	FRE IGHT_ CLASS	60	PRIC ING_ ATTR IBUTE6	QP_FRE IGHT_ CLASS	Line	UE	Yes
Logistics	Origin Zone	70	PRIC ING_ ATTR IBUTE7	QP: Number	Line	UE	Yes
Logistics	Destination Zone	80	PRIC ING_ ATTR IBUTE8	QP: Number	Line	UE	Yes
Logistics	Total Shipment Quantity	90	PRIC ING_ ATTR IBUTE9	QP: Number	Line	UE	Yes
Volume	Item Quantity	800	PRIC ING_ ATTR IBUTE10	QP: Number	Line	UE	Yes
Volume	Item Amount	810	PRIC ING_ ATTR IBUTE12	QP: Number	Line	UE	Yes
Volume	Total Item Quantity	880	PRIC ING_ ATTR IBUTE20	QP: Number	Line	UE	Yes

Product Attributes

The following section displays the seeded product attributes for the Logistics PTE:

Context	Attributes	Prec	Mapped	Value Name	Level	AMM	Lmt	Req Type
ITEM	All Items	315	PRICING_ATTRIBUTE3	QP: ITEM_ALL	LINE	UE	Y	ASO
ITEM	All Items	315	PRICING_ATTRIBUTE3	QP: ITEM_ALL	LINE	UE	Y	ONT
ITEM	All Items	315	PRICING_ATTRIBUTE3	QP: ITEM_ALL	LINE	UE	Y	OKC
ITEM	All Items	315	PRICING_ATTRIBUTE3	QP: ITEM_ALL	LINE	UE	Y	PO

Qualifier Attributes

The following section displays the seeded qualifier attributes for the Logistics PTE:

Context	Attributes	Prec	Mapped	Value Name	Level	AMM	Lmt	Req Type
MODL IST	Price List	140	QUAL IFIER_ ATTR IBUTE4	QP_SRS_ PRICE_ LIST_ NAME	LINE	UE	Y	ASO
MODL IST	Price List	140	QUAL IFIER_ ATTR IBUTE4	QP_SRS_ PRICE_ LIST_ NAME	LINE	UE	Y	ONT
MODL IST	Price List	140	QUAL IFIER_ ATTR IBUTE4	QP_SRS_ PRICE_ LIST_ NAME	LINE	UE	Y	ONT
MODL IST	Price List	140	QUAL IFIER_ ATTR IBUTE4	QP_SRS_ PRICE_ LIST_ NAME	LINE	UE	Y	ASO
ORDER	Blanket Number	600	QUAL IFIER_ ATTR IBUTE3	QP_ BLANKET_ NUMBER	LINE	UE	Y	ASO
ORDER	Blanket Number	600	QUAL IFIER_ ATTR IBUTE3	QP_ BLANKET_ NUMBER	LINE	UE	Y	ONT
ORDER	Blanket Number	600	QUAL IFIER_ ATTR IBUTE3	QP_ BLANKET_ NUMBER	LINE	UE	Y	ONT
ORDER	Blanket Number	600	QUAL IFIER_ ATTR IBUTE3	QP_ BLANKET_ NUMBER	LINE	UE	Y	ASO
PARTY	SALES ORGAN IZAT ION	170	QUAL IFIER_ ATTR IBUTE3	QP_ SALES_ ORGAN IZAT ION	LINE	UE	Y	ONT
PARTY	SALES ORGAN IZAT ION	170	QUAL IFIER_ ATTR IBUTE3	QP_ SALES_ ORGAN IZAT ION	LINE	UE	Y	ONT

Order Fulfillment PTE Attributes

The following section displays the attributes for the Order Fulfillment Pricing Transaction Entity (PTE):

Pricing Attributes

The following table lists the seeded pricing attributes for the Order Fulfillment PTE:

Context	Attribute	Prec	Mapped Value	Value Set Name	Level	AMM	Lmt	Req Type
BREAK_UOM	Usage UOM	500	PRICING_ATTRIBUTE1	-	LINE	AM	N	OKS
PRICING ATTRIBUTE	Administration Cost	780	PRICING_ATTRIBUTE17	QP: Number	LINE	AM	Y	ONT
PRICING ATTRIBUTE	Customer Item	720	PRICING_ATTRIBUTE11	-	LINE	AM	Y	ONT
PRICING ATTRIBUTE	Duty Cost	760	PRICING_ATTRIBUTE15	QP: Number	LINE	AM	Y	ONT
PRICING ATTRIBUTE	Estimated Transportation Charge	830	PRICING_ATTRIBUTE24	QP: Number	LINE	AM	N	ONT
PRICING ATTRIBUTE	Estimated Transportation Price	820	PRICING_ATTRIBUTE23	QP: Number	LINE	AM	N	ONT
PRICING ATTRIBUTE	Export Cost	750	PRICING_ATTRIBUTE14	QP: Number	LINE	AM	Y	ONT
PRICING ATTRIBUTE	Freight Cost	770	PRICING_ATTRIBUTE16	QP: Number	LINE	AM	Y	ONT
PRICING ATTRIBUTE	Grade	782	PRICING_ATTRIBUTE19	OPM_QC_GRADE	LINE	AM	Y	ONT
PRICING ATTRIBUTE	Handling Cost	740	PRICING_ATTRIBUTE13	QP: Number	LINE	AM	Y	ONT
PRICING ATTRIBUTE	Insurance Cost	730	PRICING_ATTRIBUTE12	QP: Number	LINE	AM	Y	ONT
PRICING ATTRIBUTE	Model Id	790	PRICING_ATTRIBUTE1	QP_MODEL_ITEM_ID	LINE	AM	Y	ASO

Context	Attribute	Prec	Mapped Value	Value Set Name	Level	AMM	Lmt	Req Type
PRIC ING ATTR IBUTE	Model Id	790	PRIC ING_ ATTR IBUTE1	QP_ MODEL_ ITEM_ID	LINE	AM	Y	OKC
PRIC ING ATTR IBUTE	Model Id	790	PRIC ING_ ATTR IBUTE1	QP_ MODEL_ ITEM_ID	LINE	AM	Y	ONT
PRIC ING ATTR IBUTE	Parent List Price	710	PRIC ING_ ATTR IBUTE10	QP: Number	LINE	AM	Y	ASO
PRIC ING ATTR IBUTE	Parent List Price	710	PRIC ING_ ATTR IBUTE10	QP: Number	LINE	AM	Y	ONT
PRIC ING ATTR IBUTE	Transportation Charge	810	PRIC ING_ ATTR IBUTE21	QP: Number	LINE	AM	Y	ONT
VOLUME	Blanket Amount	900	PRIC ING_ ATTR IBUTE6	QP: Number	LINE	AM	Y	ONT
VOLUME	Blanket Line Amount	920	PRIC ING_ ATTR IBUTE7	QP: Number	LINE	AM	Y	ONT
VOLUME	Blanket Line Quantity	940	PRIC ING_ ATTR IBUTE8	QP: Number	LINE	AM	Y	ONT
VOLUME	Period1 Item Amount	850	PRIC ING_ ATTR IBUTE13	QP: Number	LINE	AM	N	ASO
VOLUME	Period1 Item Amount	850	PRIC ING_ ATTR IBUTE13	QP: Number	LINE	AM	N	ONT
VOLUME	Period1 Item Quantity	820	PRIC ING_ ATTR IBUTE3	QP: Number	LINE	AM	N	ONT
VOLUME	Period1 Item Quantity	820	PRIC ING_ ATTR IBUTE3	QP: Number	LINE	AM	N	ASO

Context	Attribute	Prec	Mapped Value	Value Set Name	Level	AMM	Lmt	Req Type
VOLUME	Period2 Item Amount	860	PRIC ING_ ATTR IBUTE14	QP: Number	LINE	AM	N	ASO
VOLUME	Period2 Item Amount	860	PRIC ING_ ATTR IBUTE14	QP: Number	LINE	AM	N	ONT
VOLUME	Period2 Item Quantity	830	PRIC ING_ ATTR IBUTE1	QP: Number	LINE	AM	N	ASO
VOLUME	Period2 Item Quantity	830	PRIC ING_ ATTR IBUTE1	QP: Number	LINE	AM	N	ONT
VOLUME	Period3 Item Amount	870	PRIC ING_ ATTR IBUTE15	QP: Number	LINE	AM	N	ASO
VOLUME	Period3 Item Amount	870	PRIC ING_ ATTR IBUTE15	QP: Number	LINE	AM	N	ONT
VOLUME	Period3 Item Quantity	840	PRIC ING_ ATTR IBUTE11	QP: Number	LINE	AM	N	ONT
VOLUME	Period3 Item Quantity	840	PRIC ING_ ATTR IBUTE11	QP: Number	LINE	AM	N	ASO
VOLUME	Period3 Order Amount	610	QUAL IFIER_ ATTR IBUTE11	QP: Number	BOTH	AM	Y	ASO
VOLUME	Period3 Order Amount	610	QUAL IFIER_ ATTR IBUTE11	QP: Number	BOTH	AM	Y	ONT
VOLUME	Period3 Order Amount	610	QUAL IFIER_ ATTR IBUTE11	QP: Number	BOTH	AM	Y	ONT

Product Attributes

The following section displays the product attributes for the Order Fulfillment PTE:

Context	Attributes	Prec	Mapped	Value Name	Level	AMM	Lmt	Req. Type
ITEM	All Items	315	PRICING_ATTRIBUTE3	QP: ITEM_ALL	LINE	AM	Y	OKC
ITEM	All Items	315	PRICING_ATTRIBUTE3	QP: ITEM_ALL	LINE	AM	Y	PO
ITEM	All Items	315	PRICING_ATTRIBUTE3	QP: ITEM_ALL	LINE	AM	Y	ASO
ITEM	All Items	315	PRICING_ATTRIBUTE3	QP: ITEM_ALL	LINE	AM	Y	ONT
ITEM	Item Category	290	PRICING_ATTRIBUTE2	n/a	LINE	AM	Y	ASO
ITEM	Item Category	290	PRICING_ATTRIBUTE2	n/a	LINE	AM	Y	OKC
ITEM	Item Category	290	PRICING_ATTRIBUTE2	n/a	LINE	AM	Y	ONT
ITEM	Item Category	290	PRICING_ATTRIBUTE2	n/a	LINE	AM	Y	PO
ITEM	Item Number	220	PRICING_ATTRIBUTE1	n/a	LINE	AM	Y	IC
ITEM	Item Number	220	PRICING_ATTRIBUTE1	n/a	LINE	AM	Y	PO
ITEM	Item Number	220	PRICING_ATTRIBUTE1	n/a	LINE	AM	Y	ASO
ITEM	Item Number	220	PRICING_ATTRIBUTE1	n/a	LINE	AM	Y	OKC

Context	Attributes	Prec	Mapped	Value Name	Level	AMM	Lmt	Req. Type
ITEM	Item Number	220	PRICING_ATTRIBUTE1	n/a	LINE	AM	Y	ONT
ITEM	SEGMENT_1	291	PRICING_ATTRIBUTE4	n/a	LINE	AM	N	ASO
ITEM	SEGMENT_1	291	PRICING_ATTRIBUTE4	n/a	LINE	AM	N	ONT
ITEM	SEGMENT_1	291	PRICING_ATTRIBUTE4	n/a	LINE	AM	N	OKC
ITEM	SEGMENT_10	300	PRICING_ATTRIBUTE13	n/a	LINE	AM	N	ASO
ITEM	SEGMENT_10	300	PRICING_ATTRIBUTE13	n/a	LINE	AM	N	OKC
ITEM	SEGMENT_10	300	PRICING_ATTRIBUTE13	n/a	LINE	AM	N	ONT
ITEM	SEGMENT_11	301	PRICING_ATTRIBUTE14	n/a	LINE	AM	N	ASO
ITEM	SEGMENT_11	301	PRICING_ATTRIBUTE14	n/a	LINE	AM	N	ONT
ITEM	SEGMENT_11	301	PRICING_ATTRIBUTE14	n/a	LINE	AM	N	OKC
ITEM	SEGMENT_12	302	PRICING_ATTRIBUTE15	n/a	LINE	AM	N	ASO
ITEM	SEGMENT_12	302	PRICING_ATTRIBUTE15	n/a	LINE	AM	N	OKC

Context	Attributes	Prec	Mapped	Value Name	Level	AMM	Lmt	Req. Type
ITEM	SEGMENT_12	302	PRICING_ATTRIBUTE15	n/a	LINE	AM	N	ONT
ITEM	SEGMENT_13	303	PRICING_ATTRIBUTE16	n/a	LINE	AM	N	ASO
ITEM	SEGMENT_13	303	PRICING_ATTRIBUTE16	n/a	LINE	AM	N	OKC
ITEM	SEGMENT_13	303	PRICING_ATTRIBUTE16	n/a	LINE	AM	N	ONT
ITEM	SEGMENT_14	304	PRICING_ATTRIBUTE17	n/a	LINE	AM	N	ASO
ITEM	SEGMENT_14	304	PRICING_ATTRIBUTE17	n/a	LINE	AM	N	OKC
ITEM	SEGMENT_14	304	PRICING_ATTRIBUTE17	n/a	LINE	AM	N	ONT
ITEM	SEGMENT_15	305	PRICING_ATTRIBUTE18	n/a	LINE	AM	N	ASO
ITEM	SEGMENT_15	305	PRICING_ATTRIBUTE18	n/a	LINE	AM	N	OKC
ITEM	SEGMENT_15	305	PRICING_ATTRIBUTE18	n/a	LINE	AM	N	ONT
ITEM	SEGMENT_16	306	PRICING_ATTRIBUTE19	n/a	LINE	AM	N	ASO
ITEM	SEGMENT_16	306	PRICING_ATTRIBUTE19	n/a	LINE	AM	N	ONT

Context	Attributes	Prec	Mapped	Value Name	Level	AMM	Lmt	Req. Type
ITEM	SEGMENT_16	306	PRICING_ATTRIBUTE19	n/a	LINE	AM	N	OKC
ITEM	SEGMENT_17	307	PRICING_ATTRIBUTE20	n/a	LINE	AM	N	ASO
ITEM	SEGMENT_17	307	PRICING_ATTRIBUTE20	n/a	LINE	AM	N	ONT
ITEM	SEGMENT_17	307	PRICING_ATTRIBUTE20	n/a	LINE	AM	N	OKC
ITEM	SEGMENT_18	308	PRICING_ATTRIBUTE21	n/a	LINE	AM	N	ASO
ITEM	SEGMENT_18	308	PRICING_ATTRIBUTE21	n/a	LINE	AM	N	OKC
ITEM	SEGMENT_18	308	PRICING_ATTRIBUTE21	n/a	LINE	AM	N	ONT
ITEM	SEGMENT_19	309	PRICING_ATTRIBUTE22	n/a	LINE	AM	N	ASO
ITEM	SEGMENT_19	309	PRICING_ATTRIBUTE22	n/a	LINE	AM	N	ONT
ITEM	SEGMENT_19	309	PRICING_ATTRIBUTE22	n/a	LINE	AM	N	OKC
ITEM	SEGMENT_2	292	PRICING_ATTRIBUTE5	n/a	LINE	AM	N	ASO
ITEM	SEGMENT_2	292	PRICING_ATTRIBUTE5	n/a	LINE	AM	N	OKC

Context	Attributes	Prec	Mapped	Value Name	Level	AMM	Lmt	Req. Type
ITEM	SEGMENT2	922	PRICING_ATTRIBUTE5	n/a	LINE	AM	N	ONT
ITEM	SEGMENT3	1120	PRICING_ATTRIBUTE23	n/a	LINE	AM	N	ASO
ITEM	SEGMENT3	1120	PRICING_ATTRIBUTE23	n/a	LINE	AM	N	OKC
ITEM	SEGMENT3	1120	PRICING_ATTRIBUTE23	n/a	LINE	AM	N	ONT
ITEM	SEGMENT2	933	PRICING_ATTRIBUTE6	n/a	LINE	AM	N	ASO
ITEM	SEGMENT2	933	PRICING_ATTRIBUTE6	n/a	LINE	AM	N	OKC
ITEM	SEGMENT2	933	PRICING_ATTRIBUTE6	n/a	LINE	AM	N	ONT
ITEM	SEGMENT2	944	PRICING_ATTRIBUTE7	n/a	LINE	AM	N	ASO
ITEM	SEGMENT2	944	PRICING_ATTRIBUTE7	n/a	LINE	AM	N	OKC
ITEM	SEGMENT2	944	PRICING_ATTRIBUTE7	n/a	LINE	AM	N	ONT
ITEM	SEGMENT2	955	PRICING_ATTRIBUTE8	n/a	LINE	AM	N	ASO
ITEM	SEGMENT2	955	PRICING_ATTRIBUTE8	n/a	LINE	AM	N	ONT

Context	Attributes	Prec	Mapped	Value Name	Level	AMM	Lmt	Req. Type
ITEM	SEGMENT_295 5		PRICING_ ATTRIBUTE8	n/a	LINE	AM	N	OKC
ITEM	SEGMENT_296 6		PRICING_ ATTRIBUTE9	n/a	LINE	AM	N	ASO
ITEM	SEGMENT_296 6		PRICING_ ATTRIBUTE9	n/a	LINE	AM	N	ONT
ITEM	SEGMENT_296 6		PRICING_ ATTRIBUTE9	n/a	LINE	AM	N	OKC
ITEM	SEGMENT_297 7		PRICING_ ATTRIBUTE10	n/a	LINE	AM	N	ASO
ITEM	SEGMENT_297 7		PRICING_ ATTRIBUTE10	n/a	LINE	AM	N	OKC
ITEM	SEGMENT_297 7		PRICING_ ATTRIBUTE10	n/a	LINE	AM	N	ONT
ITEM	SEGMENT_298 8		PRICING_ ATTRIBUTE11	n/a	LINE	AM	N	ASO
ITEM	SEGMENT_298 8		PRICING_ ATTRIBUTE11	n/a	LINE	AM	N	ONT
ITEM	SEGMENT_298 8		PRICING_ ATTRIBUTE11	n/a	LINE	AM	N	OKC
ITEM	SEGMENT_299 9		PRICING_ ATTRIBUTE12	n/a	LINE	AM	N	ASO
ITEM	SEGMENT_299 9		PRICING_ ATTRIBUTE12	n/a	LINE	AM	N	OKC

Context	Attributes	Prec	Mapped	Value Name	Level	AMM	Lmt	Req. Type
ITEM	SEGMENT_9	299	PRICING_ATTRIBUTE12	n/a	LINE	AM	N	ONT
ITEM	iStore Section	312	PRICING_ATTRIBUTE24	AMS_ISTORE_SECTION	LINE	AM	Y	ONT
ITEM	iStore Section	312	PRICING_ATTRIBUTE24	AMS_ISTORE_SECTION	LINE	AM	Y	ASO

Qualifier Attributes

The following section displays the seeded qualifier attributes for the Order Fulfillment PTE:

Context	Attributes	Prec	Mapped	Value Name	Level	AMM	Lmt	Req Type
ASO PARTY INFO	Bill to Party Site	3	QUALIFIER_ATTRIBUTE11	QP_SHIP_TO_PARTY_SITE	BOTH	AM	Y	ASO
ASO PARTY INFO	Customer Party	1	QUALIFIER_ATTRIBUTE1	QP_CUSTOMER_PARTY	BOTH	AM	Y	ASO
ASO PARTY INFO	Ship to Party Site	2	QUALIFIER_ATTRIBUTE10	QP_SHIP_TO_PARTY_SITE	BOTH	AM	Y	ASO
CUSTOMER	Account Type	340	QUALIFIER_ATTRIBUTE12	QP_ACCOUNT_TYPE	BOTH	AM	Y	ASO
CUSTOMER	Account Type	340	QUALIFIER_ATTRIBUTE12	QP_ACCOUNT_TYPE	BOTH	AM	Y	ASO
CUSTOMER	Account Type	340	QUALIFIER_ATTRIBUTE12	QP_ACCOUNT_TYPE	BOTH	AM	Y	OKC
CUSTOMER	Account Type	340	QUALIFIER_ATTRIBUTE12	QP_ACCOUNT_TYPE	BOTH	AM	Y	OKC

Context	Attributes	Prec	Mapped	Value Name	Level	AMM	Lmt	Req Type
CUSTOMER	Account Type	340	QUAL IFIER_ ATTR IBUTE12	QP_ ACCOUNT_ TYPE	BOTH	AM	Y	ONT
CUSTOMER	Account Type	340	QUAL IFIER_ ATTR IBUTE12	QP_ ACCOUNT_ TYPE	BOTH	AM	Y	ONT
CUSTOMER	Agreement Name	210	QUAL IFIER_ ATTR IBUTE7	QP_ AGREEMENT_ NAME	BOTH	AM	Y	ASO
CUSTOMER	Agreement Name	210	QUAL IFIER_ ATTR IBUTE7	QP_ AGREEMENT_ NAME	BOTH	AM	Y	ASO
CUSTOMER	Agreement Name	210	QUAL IFIER_ ATTR IBUTE7	QP_ AGREEMENT_ NAME	BOTH	AM	Y	ONT
CUSTOMER	Agreement Name	210	QUAL IFIER_ ATTR IBUTE7	QP_ AGREEMENT_ NAME	BOTH	AM	Y	ONT
CUSTOMER	Agreement Type	240	QUAL IFIER_ ATTR IBUTE8	QP_ AGREEMENT_ TYPE	BOTH	AM	Y	ASO
CUSTOMER	Agreement Type	240	QUAL IFIER_ ATTR IBUTE8	QP_ AGREEMENT_ TYPE	BOTH	AM	Y	ASO
CUSTOMER	Agreement Type	240	QUAL IFIER_ ATTR IBUTE8	QP_ AGREEMENT_ TYPE	BOTH	AM	Y	ONT
CUSTOMER	Agreement Type	240	QUAL IFIER_ ATTR IBUTE8	QP_ AGREEMENT_ TYPE	BOTH	AM	Y	ONT
CUSTOMER	Bill To	280	QUAL IFIER_ ATTR IBUTE14	QP_ INVO ICE_TO_ ORGS	BOTH	AM	Y	ASO
CUSTOMER	Bill To	280	QUAL IFIER_ ATTR IBUTE14	QP_ INVO ICE_TO_ ORGS	BOTH	AM	Y	ASO

Context	Attributes	Prec	Mapped	Value Name	Level	AMM	Lmt	Req Type
CUSTOMER	Bill To	280	QUAL IFIER_ ATTR IBUTE14	QP_ INVO ICE_TO_ ORGS	BOTH	AM	Y	OKC
CUSTOMER	Bill To	280	QUAL IFIER_ ATTR IBUTE14	QP_ INVO ICE_TO_ ORGS	BOTH	AM	Y	OKC
CUSTOMER	Bill To	280	QUAL IFIER_ ATTR IBUTE14	QP_ INVO ICE_TO_ ORGS	BOTH	AM	Y	ONT
CUSTOMER	Bill To	280	QUAL IFIER_ ATTR IBUTE14	QP_ INVO ICE_TO_ ORGS	BOTH	AM	Y	ONT
CUSTOMER	Customer Class	310	QUAL IFIER_ ATTR IBUTE1	QP_ CUSTOMER_ CLASS	BOTH	AM	Y	ASO
CUSTOMER	Customer Class	310	QUAL IFIER_ ATTR IBUTE1	QP_ CUSTOMER_ CLASS	BOTH	AM	Y	ASO
CUSTOMER	Customer Class	310	QUAL IFIER_ ATTR IBUTE1	QP_ CUSTOMER_ CLASS	BOTH	AM	Y	OKC
CUSTOMER	Customer Class	310	QUAL IFIER_ ATTR IBUTE1	QP_ CUSTOMER_ CLASS	BOTH	AM	Y	OKC
CUSTOMER	Customer Class	310	QUAL IFIER_ ATTR IBUTE1	QP_ CUSTOMER_ CLASS	BOTH	AM	Y	ONT
CUSTOMER	Customer Name	260	QUAL IFIER_ ATTR IBUTE2	QP_ CUSTOMERS	BOTH	AM	Y	ASO
CUSTOMER	Customer Name	260	QUAL IFIER_ ATTR IBUTE2	QP_ CUSTOMERS	BOTH	AM	Y	OKC
CUSTOMER	Customer Name	260	QUAL IFIER_ ATTR IBUTE2	QP_ CUSTOMERS	BOTH	AM	Y	ONT

Context	Attributes	Prec	Mapped	Value Name	Level	AMM	Lmt	Req Type
CUSTOMER	Customer Name	260	QUALIFIER_ATTRIBUTE2	QP_CUSTOMERS	BOTH	AM	Y	OKC
CUSTOMER	Customer Name	260	QUALIFIER_ATTRIBUTE2	QP_CUSTOMERS	BOTH	AM	Y	ASO
CUSTOMER	Customer Name	260	QUALIFIER_ATTRIBUTE2	QP_CUSTOMERS	BOTH	AM	Y	ONT
CUSTOMER	GSA	100	QUALIFIER_ATTRIBUTE15	QP: Yes/No	BOTH	AM	Y	ASO
CUSTOMER	GSA	100	QUALIFIER_ATTRIBUTE15	QP: Yes/No	BOTH	AM	Y	ASO
CUSTOMER	GSA	100	QUALIFIER_ATTRIBUTE15	QP: Yes/No	BOTH	AM	Y	OKC
CUSTOMER	GSA	100	QUALIFIER_ATTRIBUTE15	QP: Yes/No	BOTH	AM	Y	OKC
CUSTOMER	GSA	100	QUALIFIER_ATTRIBUTE15	QP: Yes/No	BOTH	AM	Y	ONT
CUSTOMER	GSA	100	QUALIFIER_ATTRIBUTE15	QP: Yes/No	BOTH	AM	Y	ONT
CUSTOMER	Invoice To Party Site	400	QUALIFIER_ATTRIBUTE18	QP_INVOICE_TO_PARTY_SITE_ID	BOTH	AM	Y	ASO
CUSTOMER	Invoice To Party Site	400	QUALIFIER_ATTRIBUTE18	QP_INVOICE_TO_PARTY_SITE_ID	BOTH	AM	Y	ASO

Context	Attributes	Prec	Mapped	Value Name	Level	AMM	Lmt	Req Type
CUSTOMER	Invoice To Party Site	400	QUALIFIER_ATTRIBUTE18	QP_INVOICE_TO_PARTY_SITE_ID	BOTH	AM	Y	OKC
CUSTOMER	Invoice To Party Site	400	QUALIFIER_ATTRIBUTE18	QP_INVOICE_TO_PARTY_SITE_ID	BOTH	AM	Y	OKC
CUSTOMER	Invoice To Party Site	400	QUALIFIER_ATTRIBUTE18	QP_INVOICE_TO_PARTY_SITE_ID	BOTH	AM	Y	ONT
CUSTOMER	Invoice To Party Site	400	QUALIFIER_ATTRIBUTE18	QP_INVOICE_TO_PARTY_SITE_ID	BOTH	AM	Y	ONT
CUSTOMER	Party ID	360	QUALIFIER_ATTRIBUTE16	QP_PARTY	BOTH	AM	Y	ASO
CUSTOMER	Party ID	360	QUALIFIER_ATTRIBUTE16	QP_PARTY	BOTH	AM	Y	OKC
CUSTOMER	Party ID	360	QUALIFIER_ATTRIBUTE16	QP_PARTY	BOTH	AM	Y	ONT
CUSTOMER	Party ID	360	QUALIFIER_ATTRIBUTE16	QP_PARTY	BOTH	AM	Y	ONT
CUSTOMER	Party ID	360	QUALIFIER_ATTRIBUTE16	QP_PARTY	BOTH	AM	Y	ASO
CUSTOMER	Party ID	360	QUALIFIER_ATTRIBUTE16	QP_PARTY	BOTH	AM	Y	OKC
CUSTOMER	Sales Channel	320	QUALIFIER_ATTRIBUTE13	QP_SALES_CHANNEL_CODE	BOTH	AM	Y	ASO

Context	Attributes	Prec	Mapped	Value Name	Level	AMM	Lmt	Req Type
CUSTOMER	Sales Channel	320	QUAL IFIER_ ATTR IBUTE13	QP_ SALES_ CHANNEL_ CODE	BOTH	AM	Y	ASO
CUSTOMER	Sales Channel	320	QUAL IFIER_ ATTR IBUTE13	QP_ SALES_ CHANNEL_ CODE	BOTH	AM	Y	OKC
CUSTOMER	Sales Channel	320	QUAL IFIER_ ATTR IBUTE13	QP_ SALES_ CHANNEL_ CODE	BOTH	AM	Y	OKC
CUSTOMER	Sales Channel	320	QUAL IFIER_ ATTR IBUTE13	QP_ SALES_ CHANNEL_ CODE	BOTH	AM	Y	ONT
CUSTOMER	Sales Channel	320	QUAL IFIER_ ATTR IBUTE13	QP_ SALES_ CHANNEL_ CODE	BOTH	AM	Y	ONT
CUSTOMER	Ship To	250	QUAL IFIER_ ATTR IBUTE11	QP_SH IP_TO_ ORGS	BOTH	AM	Y	ASO
CUSTOMER	Ship To	250	QUAL IFIER_ ATTR IBUTE11	QP_SH IP_TO_ ORGS	BOTH	AM	Y	OKC
CUSTOMER	Ship To	250	QUAL IFIER_ ATTR IBUTE11	QP_SH IP_TO_ ORGS	BOTH	AM	Y	ONT
CUSTOMER	Ship To	250	QUAL IFIER_ ATTR IBUTE11	QP_SH IP_TO_ ORGS	BOTH	AM	Y	OKC
CUSTOMER	Ship To	250	QUAL IFIER_ ATTR IBUTE11	QP_SH IP_TO_ ORGS	BOTH	AM	Y	ASO
CUSTOMER	Ship To	250	QUAL IFIER_ ATTR IBUTE11	QP_SH IP_TO_ ORGS	BOTH	AM	Y	ONT
CUSTOMER	Ship To Party Site	380	QUAL IFIER_ ATTR IBUTE17	QP_SH IP_TO_ PARTY_ SITE_ID	BOTH	AM	Y	ASO

Context	Attributes	Prec	Mapped	Value Name	Level	AMM	Lmt	Req Type
CUSTOMER	Ship To Party Site	380	QUALIFIER_ATTRIBUTE17	QP_SHIP_TO_PARTY_SITE_ID	BOTH	AM	Y	ONT
CUSTOMER	Ship To Party Site	380	QUALIFIER_ATTRIBUTE17	QP_SHIP_TO_PARTY_SITE_ID	BOTH	AM	Y	ONT
CUSTOMER	Ship To Party Site	380	QUALIFIER_ATTRIBUTE17	QP_SHIP_TO_PARTY_SITE_ID	BOTH	AM	Y	OKC
CUSTOMER	Ship To Party Site	380	QUALIFIER_ATTRIBUTE17	QP_SHIP_TO_PARTY_SITE_ID	BOTH	AM	Y	OKC
CUSTOMER	Ship To Party Site	380	QUALIFIER_ATTRIBUTE17	QP_SHIP_TO_PARTY_SITE_ID	BOTH	AM	Y	ASO
CUSTOMER	Site Use	270	QUALIFIER_ATTRIBUTE5	QP_CUSTOMER_SITES	BOTH	AM	Y	ASO
CUSTOMER	Site Use	270	QUALIFIER_ATTRIBUTE5	QP_CUSTOMER_SITES	BOTH	AM	Y	ASO
CUSTOMER	Site Use	270	QUALIFIER_ATTRIBUTE5	QP_CUSTOMER_SITES	BOTH	AM	Y	OKC
CUSTOMER	Site Use	270	QUALIFIER_ATTRIBUTE5	QP_CUSTOMER_SITES	BOTH	AM	Y	ONT
CUSTOMER	Site Use	270	QUALIFIER_ATTRIBUTE5	QP_CUSTOMER_SITES	BOTH	AM	Y	ONT
CUSTOMER	Site Use	270	QUALIFIER_ATTRIBUTE5	QP_CUSTOMER_SITES	BOTH	AM	Y	OKC
CUSTOMER GROUP	Buying Groups	3	QUALIFIER_ATTRIBUTE3	AMS_BUYING_GROUP	BOTH	AM	Y	ONT

Context	Attributes	Prec	Mapped	Value Name	Level	AMM	Lmt	Req Type
CUSTOMER_GROUP	Buying Groups	3	QUALIFIER_ATTRIBUTE3	AMS_BUYING_GROUP	BOTH	AM	Y	ASO
CUSTOMER_GROUP	Buying Groups	3	QUALIFIER_ATTRIBUTE3	AMS_BUYING_GROUP	BOTH	AM	Y	ONT
CUSTOMER_GROUP	Buying Groups	3	QUALIFIER_ATTRIBUTE3	AMS_BUYING_GROUP	BOTH	AM	Y	ASO
CUSTOMER_GROUP	Lists	1	QUALIFIER_ATTRIBUTE1	AMS_LISTS	BOTH	AM	Y	ONT
CUSTOMER_GROUP	Lists	1	QUALIFIER_ATTRIBUTE1	AMS_LISTS	BOTH	AM	Y	ASO
CUSTOMER_GROUP	Lists	1	QUALIFIER_ATTRIBUTE1	AMS_LISTS	BOTH	AM	Y	ONT
CUSTOMER_GROUP	Lists	1	QUALIFIER_ATTRIBUTE1	AMS_LISTS	BOTH	AM	Y	ASO
CUSTOMER_GROUP	Segments	2	QUALIFIER_ATTRIBUTE2	AMS_SEGMENTS	BOTH	AM	Y	ONT
CUSTOMER_GROUP	Segments	2	QUALIFIER_ATTRIBUTE2	AMS_SEGMENTS	BOTH	AM	Y	ASO
CUSTOMER_GROUP	Segments	2	QUALIFIER_ATTRIBUTE2	AMS_SEGMENTS	BOTH	AM	Y	ONT
CUSTOMER_GROUP	Segments	2	QUALIFIER_ATTRIBUTE2	AMS_SEGMENTS	BOTH	AM	Y	ASO
MODLIST	Price List	140	QUALIFIER_ATTRIBUTE4	QP_SRS_PRICE_LIST_NAME	BOTH	AM	Y	ASO

Context	Attributes	Prec	Mapped	Value Name	Level	AMM	Lmt	Req Type
MODL IST	Price List	140	QUAL IFIER_ ATTR IBUTE4	QP_SRS_ PRICE_ LIST_ NAME	BOTH	AM	Y	ONT
MODL IST	Price List	140	QUAL IFIER_ ATTR IBUTE4	QP_SRS_ PRICE_ LIST_ NAME	BOTH	AM	Y	ONT
MODL IST	Price List	140	QUAL IFIER_ ATTR IBUTE4	QP_SRS_ PRICE_ LIST_ NAME	BOTH	AM	Y	ASO
ORDER	Blanket Id	700	QUAL IFIER_ ATTR IBUTE5	QP_ BLANKET_ HEADER_ ID	BOTH	AM	Y	ONT
ORDER	Blanket Id	700	QUAL IFIER_ ATTR IBUTE5	QP_ BLANKET_ HEADER_ ID	BOTH	AM	Y	ONT
ORDER	Blanket Line Number	800	QUAL IFIER_ ATTR IBUTE6	QP_ BLANKET_ LINE_ID	LINE	AM	Y	ONT
ORDER	Blanket Number	600	QUAL IFIER_ ATTR IBUTE3	QP_ BLANKET_ NUMBER	BOTH	AM	N	ASO
ORDER	Blanket Number	600	QUAL IFIER_ ATTR IBUTE3	QP_ BLANKET_ NUMBER	BOTH	AM	N	ONT
ORDER	Blanket Number	600	QUAL IFIER_ ATTR IBUTE3	QP_ BLANKET_ NUMBER	BOTH	AM	N	ONT
ORDER	Blanket Number	600	QUAL IFIER_ ATTR IBUTE3	QP_ BLANKET_ NUMBER	BOTH	AM	N	ASO
ORDER	Customer PO	440	QUAL IFIER_ ATTR IBUTE12	QP_ CUSTOMER_ PO	BOTH	AM	Y	ASO
ORDER	Customer PO	440	QUAL IFIER_ ATTR IBUTE12	QP_ CUSTOMER_ PO	BOTH	AM	Y	ASO

Context	Attributes	Prec	Mapped	Value Name	Level	AMM	Lmt	Req Type
ORDER	Customer PO	440	QUAL IFIER_ ATTR IBUTE12	QP_ CUSTOMER_ PO	BOTH	AM	Y	ONT
ORDER	Customer PO	440	QUAL IFIER_ ATTR IBUTE12	QP_ CUSTOMER_ PO	BOTH	AM	Y	ONT
ORDER	Freight Cost Type Code	580	QUAL IFIER_ ATTR IBUTE20	QP_FRE IGHT_ COST_ TYPE	LINE	AM	Y	ONT
ORDER	Line Category	460	QUAL IFIER_ ATTR IBUTE19	QP_L INE_ CATEGORY	LINE	AM	Y	ASO
ORDER	Line Category	460	QUAL IFIER_ ATTR IBUTE19	QP_L INE_ CATEGORY	LINE	AM	Y	ONT
ORDER	Line Type	450	QUAL IFIER_ ATTR IBUTE2	QP_L INE_TY PE	LINE	AM	Y	ASO
ORDER	Line Type	450	QUAL IFIER_ ATTR IBUTE2	QP_L INE_TY PE	LINE	AM	Y	ONT
ORDER	Order Category	480	QUAL IFIER_ ATTR IBUTE13	QP_ ORDER_ CATEGORY	BOTH	AM	Y	ASO
ORDER	Order Category	480	QUAL IFIER_ ATTR IBUTE13	QP_ ORDER_ CATEGORY	BOTH	AM	Y	ONT
ORDER	Order Category	480	QUAL IFIER_ ATTR IBUTE13	QP_ ORDER_ CATEGORY	BOTH	AM	Y	ONT
ORDER	Order Category	480	QUAL IFIER_ ATTR IBUTE13	QP_ ORDER_ CATEGORY	BOTH	AM	Y	ASO
ORDER	Order Date	510	QUAL IFIER_ ATTR IBUTE1	QP: Date	BOTH	AM	Y	ASO

Context	Attributes	Prec	Mapped	Value Name	Level	AMM	Lmt	Req Type
ORDER	Order Date	510	QUAL IFIER_ ATTR IBUTE1	QP: Date	BOTH	AM	Y	ONT
ORDER	Order Date	510	QUAL IFIER_ ATTR IBUTE1	QP: Date	BOTH	AM	Y	ONT
ORDER	Order Date	510	QUAL IFIER_ ATTR IBUTE1	QP: Date	BOTH	AM	Y	OKC
ORDER	Order Date	510	QUAL IFIER_ ATTR IBUTE1	QP: Date	BOTH	AM	Y	IC
ORDER	Order Date	510	QUAL IFIER_ ATTR IBUTE1	QP: Date	BOTH	AM	Y	OKC
ORDER	Order Date	510	QUAL IFIER_ ATTR IBUTE1	QP: Date	BOTH	AM	Y	ASO
ORDER	Order Type	470	QUAL IFIER_ ATTR IBUTE9	QP_ ORDER_ TYPES_ ALL	BOTH	AM	Y	ASO
ORDER	Order Type	470	QUAL IFIER_ ATTR IBUTE9	QP_ ORDER_ TYPES_ ALL	BOTH	AM	Y	ONT
ORDER	Order Type	470	QUAL IFIER_ ATTR IBUTE9	QP_ ORDER_ TYPES_ ALL	BOTH	AM	Y	ASO
ORDER	Order Type	470	QUAL IFIER_ ATTR IBUTE9	QP_ ORDER_ TYPES_ ALL	BOTH	AM	Y	ONT
ORDER	Pricing Date	530	QUAL IFIER_ ATTR IBUTE14	QP: Date	BOTH	AM	Y	ASO
ORDER	Pricing Date	530	QUAL IFIER_ ATTR IBUTE14	QP: Date	BOTH	AM	Y	ONT

Context	Attributes	Prec	Mapped	Value Name	Level	AMM	Lmt	Req Type
ORDER	Pricing Date	530	QUALIFIER_ATTRIBUTE14	QP: Date	BOTH	AM	Y	OKC
ORDER	Pricing Date	530	QUALIFIER_ATTRIBUTE14	QP: Date	BOTH	AM	Y	ASO
ORDER	Pricing Date	530	QUALIFIER_ATTRIBUTE14	QP: Date	BOTH	AM	Y	ONT
ORDER	Pricing Date	530	QUALIFIER_ATTRIBUTE14	QP: Date	BOTH	AM	Y	OKC
ORDER	Quote Source Code	630	QUALIFIER_ATTRIBUTE4		ORDER	AM	Y	ASO
ORDER	Request Date	520	QUALIFIER_ATTRIBUTE17	QP: Date	BOTH	AM	Y	ASO
ORDER	Request Date	520	QUALIFIER_ATTRIBUTE17	QP: Date	BOTH	AM	Y	ONT
ORDER	Request Date	520	QUALIFIER_ATTRIBUTE17	QP: Date	BOTH	AM	Y	ONT
ORDER	Request Date	520	QUALIFIER_ATTRIBUTE17	QP: Date	BOTH	AM	Y	ASO
ORDER	Ship From	540	QUALIFIER_ATTRIBUTE18	QP_SHIP_FROM	BOTH	AM	Y	ONT
ORDER	Ship From	540	QUALIFIER_ATTRIBUTE18	QP_SHIP_FROM	BOTH	AM	Y	ONT
ORDER	Shipment Priority Code	550	QUALIFIER_ATTRIBUTE16	QP_SHIPMENT_PRIORITY	BOTH	AM	Y	ASO

Context	Attributes	Prec	Mapped	Value Name	Level	AMM	Lmt	Req Type
ORDER	Shipment Priority Code	550	QUALIFIER_ATTRIBUTE16	QP_SHIPMENT_PRIORITY	BOTH	AM	Y	ONT
ORDER	Shipment Priority Code	550	QUALIFIER_ATTRIBUTE16	QP_SHIPMENT_PRIORITY	BOTH	AM	Y	ONT
ORDER	Shippable Flag	560	QUALIFIER_ATTRIBUTE10	QP: Yes/No	BOTH	AM	N	ASO
ORDER	Shippable Flag	560	QUALIFIER_ATTRIBUTE10	QP: Yes/No	BOTH	AM	N	ASO
ORDER	Shippable Flag	560	QUALIFIER_ATTRIBUTE10	QP: Yes/No	BOTH	AM	N	ONT
ORDER	Shippable Flag	560	QUALIFIER_ATTRIBUTE10	QP: Yes/No	BOTH	AM	N	ONT
ORDER	Shipped Date	590	QUALIFIER_ATTRIBUTE8	QP: Date	BOTH	AM	N	ONT
ORDER	Shipped Date	590	QUALIFIER_ATTRIBUTE8	QP: Date	BOTH	AM	N	ONT
ORDER	Shipped Flag	570	QUALIFIER_ATTRIBUTE11	QP: Yes/No	LINE	AM	N	ONT
ORDER	Source Type	490	QUALIFIER_ATTRIBUTE15	QP_SOURCE_TYPE	LINE	AM	Y	ONT
PARTY	SALES ORGANIZATION	170	QUALIFIER_ATTRIBUTE3	QP_SALES_ORGANIZATION	BOTH	AM	Y	ONT

Context	Attributes	Prec	Mapped	Value Name	Level	AMM	Lmt	Req Type
PARTY	SALES ORGAN IZAT ION	170	QUAL IFIER_ ATTR IBUTE3	QP_ SALES_ ORGAN IZAT ION	BOTH	AM	Y	ONT
SEGMENT	Target Segment	2	QUAL IFIER_ ATTR IBUTE2		BOTH	AM	Y	ONT
SEGMENT	Target Segment	2	QUAL IFIER_ ATTR IBUTE2		BOTH	AM	Y	ASO
SEGMENT	Target Segment	2	QUAL IFIER_ ATTR IBUTE2		BOTH	AM	Y	ONT
SEGMENT	Target Segment	2	QUAL IFIER_ ATTR IBUTE2		BOTH	AM	Y	ASO
STORE	MINIS ITE_ID	1	QUAL IFIER_ ATTR IBUTE1	QP_IBE_ MINIS ITES	BOTH	AM	Y	ASO
STORE	MINIS ITE_ID	1	QUAL IFIER_ ATTR IBUTE1	QP_IBE_ MINIS ITES	BOTH	AM	Y	ONT
STORE	MINIS ITE_ID	1	QUAL IFIER_ ATTR IBUTE1	QP_IBE_ MINIS ITES	BOTH	AM	Y	ONT
STORE	MINIS ITE_ID	1	QUAL IFIER_ ATTR IBUTE1	QP_IBE_ MINIS ITES	BOTH	AM	Y	ASO
TERMS	Freight Terms	640	QUAL IFIER_ ATTR IBUTE10	QP_FRE IGHT_ TERMS	BOTH	AM	Y	ASO
TERMS	Freight Terms	640	QUAL IFIER_ ATTR IBUTE10	QP_FRE IGHT_ TERMS	BOTH	AM	Y	ASO
TERMS	Freight Terms	640	QUAL IFIER_ ATTR IBUTE10	QP_FRE IGHT_ TERMS	BOTH	AM	Y	ONT

Context	Attributes	Prec	Mapped	Value Name	Level	AMM	Lmt	Req Type
TERMS	Freight Terms	640	QUALIFIER_ATTRIBUTE10	QP_FREIGHT_TERMS	BOTH	AM	Y	ONT
TERMS	Payment Terms	660	QUALIFIER_ATTRIBUTE1	QP_PAYMENT_TERMS	BOTH	AM	Y	ASO
TERMS	Payment Terms	660	QUALIFIER_ATTRIBUTE1	QP_PAYMENT_TERMS	BOTH	AM	Y	ASO
TERMS	Payment Terms	660	QUALIFIER_ATTRIBUTE1	QP_PAYMENT_TERMS	BOTH	AM	Y	OKC
TERMS	Payment Terms	660	QUALIFIER_ATTRIBUTE1	QP_PAYMENT_TERMS	BOTH	AM	Y	ONT
TERMS	Payment Terms	660	QUALIFIER_ATTRIBUTE1	QP_PAYMENT_TERMS	BOTH	AM	Y	ONT
TERMS	Payment Terms	660	QUALIFIER_ATTRIBUTE1	QP_PAYMENT_TERMS	BOTH	AM	Y	OKC
TERMS	Shipping Method	650	QUALIFIER_ATTRIBUTE11	QP_SHIP_METHODS	BOTH	AM	Y	ASO
TERMS	Shipping Method	650	QUALIFIER_ATTRIBUTE11	QP_SHIP_METHODS	BOTH	AM	Y	ONT
TERMS	Shipping Method	650	QUALIFIER_ATTRIBUTE11	QP_SHIP_METHODS	BOTH	AM	Y	ASO
TERMS	Shipping Method	650	QUALIFIER_ATTRIBUTE11	QP_SHIP_METHODS	BOTH	AM	Y	ONT
TERRITORY	Sales Territories	2	QUALIFIER_ATTRIBUTE2	AMS_SALES_ACCT_TERRITORIES	BOTH	AM	Y	ONT

Context	Attributes	Prec	Mapped	Value Name	Level	AMM	Lmt	Req Type
TERR ITORY	Sales Territories	2	QUAL IFIER_ ATTR IBUTE2	AMS_ SALES_ ACCT_ TERR ITORIES	BOTH	AM	Y	ASO
TERR ITORY	Sales Territories	2	QUAL IFIER_ ATTR IBUTE2	AMS_ SALES_ ACCT_ TERR ITORIES	BOTH	AM	Y	ONT
TERR ITORY	Sales Territories	2	QUAL IFIER_ ATTR IBUTE2	AMS_ SALES_ ACCT_ TERR ITORIES	BOTH	AM	Y	ASO
TERR ITORY	Trade Management Territories	1	QUAL IFIER_ ATTR IBUTE1	AMS_ TM_ OFFER_ TERR ITORIES	BOTH	AM	Y	ONT
TERR ITORY	Trade Management Territories	1	QUAL IFIER_ ATTR IBUTE1	AMS_ TM_ OFFER_ TERR ITORIES	BOTH	AM	Y	ASO
TERR ITORY	Trade Management Territories	1	QUAL IFIER_ ATTR IBUTE1	AMS_ TM_ OFFER_ TERR ITORIES	BOTH	AM	Y	ASO
TERR ITORY	Trade Management Territories	1	QUAL IFIER_ ATTR IBUTE1	AMS_ TM_ OFFER_ TERR ITORIES	BOTH	AM	Y	ONT
VOLUME	Line Volume	630	QUAL IFIER_ ATTR IBUTE15	QP: Number	LINE	AM	Y	ASO
VOLUME	Line Volume	630	QUAL IFIER_ ATTR IBUTE15	QP: Number	LINE	AM	Y	ONT
VOLUME	Line Weight	620	QUAL IFIER_ ATTR IBUTE14	QP: Number	BOTH	AM	Y	ASO

Context	Attributes	Prec	Mapped	Value Name	Level	AMM	Lmt	Req Type
VOLUME	Line Weight	620	QUAL IFIER_ ATTR IBUTE14	QP: Number	BOTH	AM	Y	ONT
VOLUME	Line Weight	620	QUAL IFIER_ ATTR IBUTE14	QP: Number	BOTH	AM	Y	ONT
VOLUME	Order Amount	500	QUAL IFIER_ ATTR IBUTE10	QP: Number	BOTH	AM	Y	ASO
VOLUME	Order Amount	500	QUAL IFIER_ ATTR IBUTE10	QP: Number	BOTH	AM	Y	ONT
VOLUME	Order Amount	500	QUAL IFIER_ ATTR IBUTE10	QP: Number	BOTH	AM	Y	ONT
VOLUME	Order Volume	650	QUAL IFIER_ ATTR IBUTE17	QP: Number	ORDER	AM	Y	ONT
VOLUME	Order Weight	640	QUAL IFIER_ ATTR IBUTE16	QP: Number	ORDER	AM	Y	ONT
VOLUME	Period1 Order Amount	590	QUAL IFIER_ ATTR IBUTE12	QP: Number	BOTH	AM	Y	ASO
VOLUME	Period1 Order Amount	590	QUAL IFIER_ ATTR IBUTE12	QP: Number	BOTH	AM	Y	ASO
VOLUME	Period1 Order Amount	590	QUAL IFIER_ ATTR IBUTE12	QP: Number	BOTH	AM	Y	ONT
VOLUME	Period1 Order Amount	590	QUAL IFIER_ ATTR IBUTE12	QP: Number	BOTH	AM	Y	ONT
VOLUME	Period2 Order Amount	600	QUAL IFIER_ ATTR IBUTE13	QP: Number	BOTH	AM	Y	ASO

Context	Attributes	Prec	Mapped	Value Name	Level	AMM	Lmt	Req Type
VOLUME	Period2 Order Amount	600	QUAL IFIER_ ATTR IBUTE13	QP: Number	BOTH	AM	Y	ONT
VOLUME	Period2 Order Amount	600	QUAL IFIER_ ATTR IBUTE13	QP: Number	BOTH	AM	Y	ONT
VOLUME	Period2 Order Amount	600	QUAL IFIER_ ATTR IBUTE13	QP: Number	BOTH	AM	Y	ASO
VOLUME	Period3 Order Amount	610	QUAL IFIER_ ATTR IBUTE11	QP: Number	BOTH	AM	Y	ASO
VOLUME	Period3 Order Amount	610	QUAL IFIER_ ATTR IBUTE11	QP: Number	BOTH	AM	Y	ASO
VOLUME	Period3 Order Amount	610	QUAL IFIER_ ATTR IBUTE11	QP: Number	BOTH	AM	Y	ONT
VOLUME	Period3 Order Amount	610	QUAL IFIER_ ATTR IBUTE11	QP: Number	BOTH	AM	Y	ONT

Procurement PTE Attributes

The following section displays the attributes for the Procurement (PO) Pricing Transaction Entity (PTE):

Pricing Attributes

The following table lists the seeded pricing attributes for the Procurement PTE:

Context	Attribute	Prec	Mapped Value	Value Set Name	Level	AMM	Lmt	Req Type
PO_PR ICING_ ATTR IBUTES	Item Revision	30	PRIC ING_ ATTR IBUTE2	QP: Text	LINE	AM	Y	PO
PO_PR ICING_ ATTR IBUTES	PO Vendor Item Number	50	PRIC ING_ ATTR IBUTE1	QP: Text	LINE	AM	Y	PO
PO_PR ICING_ ATTR IBUTES	Purchasing Org	10	PRIC ING_ ATTR IBUTE3	QP_PO_ ORG	ORDER	AM	Y	PO
PO_PR ICING_ ATTR IBUTES	Ship To Location	20	PRIC ING_ ATTR IBUTE5	QP_ PO_SH IP_TO_ LOCAT ION	LINE	AM	Y	PO
PO_PR ICING_ ATTR IBUTES	Ship To Org	40	PRIC ING_ ATTR IBUTE4	QP_PO_ SHIP_ TO_ORG	LINE	AM	Y	PO

Product Attributes

The following table lists the seeded product attributes for the Procurement PTE:

Context	Attribute	Prec	Mapped Value	Value Set Name	Level	AMM	Lmt	Req Type
ITEM	All Items	315	PRIC ING_ ATTR IBUTE3	QP: ITEM_ ALL	LINE	AM	Y	ASO
ITEM	All Items	315	PRIC ING_ ATTR IBUTE3	QP: ITEM_ ALL	LINE	AM	Y	ONT
ITEM	All Items	315	PRIC ING_ ATTR IBUTE3	QP: ITEM_ ALL	LINE	AM	Y	PO
ITEM	All Items	315	PRIC ING_ ATTR IBUTE3	QP: ITEM_ ALL	LINE	AM	Y	OKC
ITEM	Item Category	290	PRIC ING_ ATTR IBUTE2	_	LINE	AM	Y	ASO

Context	Attribute	Prec	Mapped Value	Value Set Name	Level	AMM	Lmt	Req Type
ITEM	Item Category	290	PRIC ING_ ATTR IBUTE2	_	LINE	AM	Y	OKC
ITEM	Item Category	290	PRIC ING_ ATTR IBUTE2	_	LINE	AM	Y	ONT
ITEM	Item Category	290	PRIC ING_ ATTR IBUTE2	_	LINE	AM	Y	PO
ITEM	Item Number	220	PRIC ING_ ATTR IBUTE1	_	LINE	AM	Y	ASO
ITEM	Item Number	220	PRIC ING_ ATTR IBUTE1	_	LINE	AM	Y	OKC
ITEM	Item Number	220	PRIC ING_ ATTR IBUTE1	_	LINE	AM	Y	IC
ITEM	Item Number	220	PRIC ING_ ATTR IBUTE1	_	LINE	AM	Y	PO
ITEM	Item Number	220	PRIC ING_ ATTR IBUTE1	_	LINE	AM	Y	ONT

Qualifier Attributes

The following table lists the seeded qualifier attributes for the Procurement PTE:

Context	Attribute	Prec	Mapped Value	Value Set Name	Level	AMM	Lmt	Req Type
PO_ BUYER	PO Agreement Number	200	QUAL IFIER_ ATTR IBUTE2	QP_PO_ AGREEMENT_ NUMBER	LINE	AM	Y	PO
PO_ BUYER	PO Agreement Type	100	QUAL IFIER_ ATTR IBUTE1	QP_PO_ AGREEMENT_ TYPE	LINE	AM	Y	PO
PO_ BUYER	PO Ship To Location	400	QUAL IFIER_ ATTR IBUTE4	QP_ PO_SH IP_TO_ LOCAT ION	LINE	AM	Y	PO
PO_ BUYER	PO Ship To Org	300	QUAL IFIER_ ATTR IBUTE3	QP_PO_ SHIP_ TO_ORG	LINE	AM	Y	PO
PO_ ORDER	PO Creation Date	200	QUAL IFIER_ ATTR IBUTE2	QP: Date	ORDER	AM	Y	PO
PO_ ORDER	PO Need By Date	400	QUAL IFIER_ ATTR IBUTE4	QP: Date	LINE	AM	Y	PO
PO_ ORDER	PO Order Type	300	QUAL IFIER_ ATTR IBUTE3	QP_PO_ ORDER_ TYPE	ORDER	AM	Y	PO
PO_SUP PLIER	PO Vendor	100	QUAL IFIER_ ATTR IBUTE1	QP_PO_ VENDOR	LINE	AM	Y	PO
PO_SUP PLIER	PO Vendor Site	200	QUAL IFIER_ ATTR IBUTE2	QP_PO_ VENDOR_ SITE	LINE	AM	Y	PO

C

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 are the steps that define how to calculate to calculate the . So 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.

Alternately, 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 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
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 list describes the names and setup details about 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.

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 Charge	Line 1
Step	1	Line 1
Formula Type	Numeric Constant	Line 2
Component	1	Line 2
Step	2	Line 2

D
Optimal Performance

This appendix covers the following topics:

- Overview
- Oracle Advanced Pricing Setup Considerations
- Qualifier Selectivity
- Qualifier Selectivity Examples
- Additional Tips for Better Performance
- Analyzing your Data Distributions Using a Script
- Technical Improvements

Overview

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 Advanced 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 - that 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 Advanced Pricing's internal tables, so the task of the Pricing Engine to provide maximum flexibility while delivering fast performance for e-businesses is very challenging.

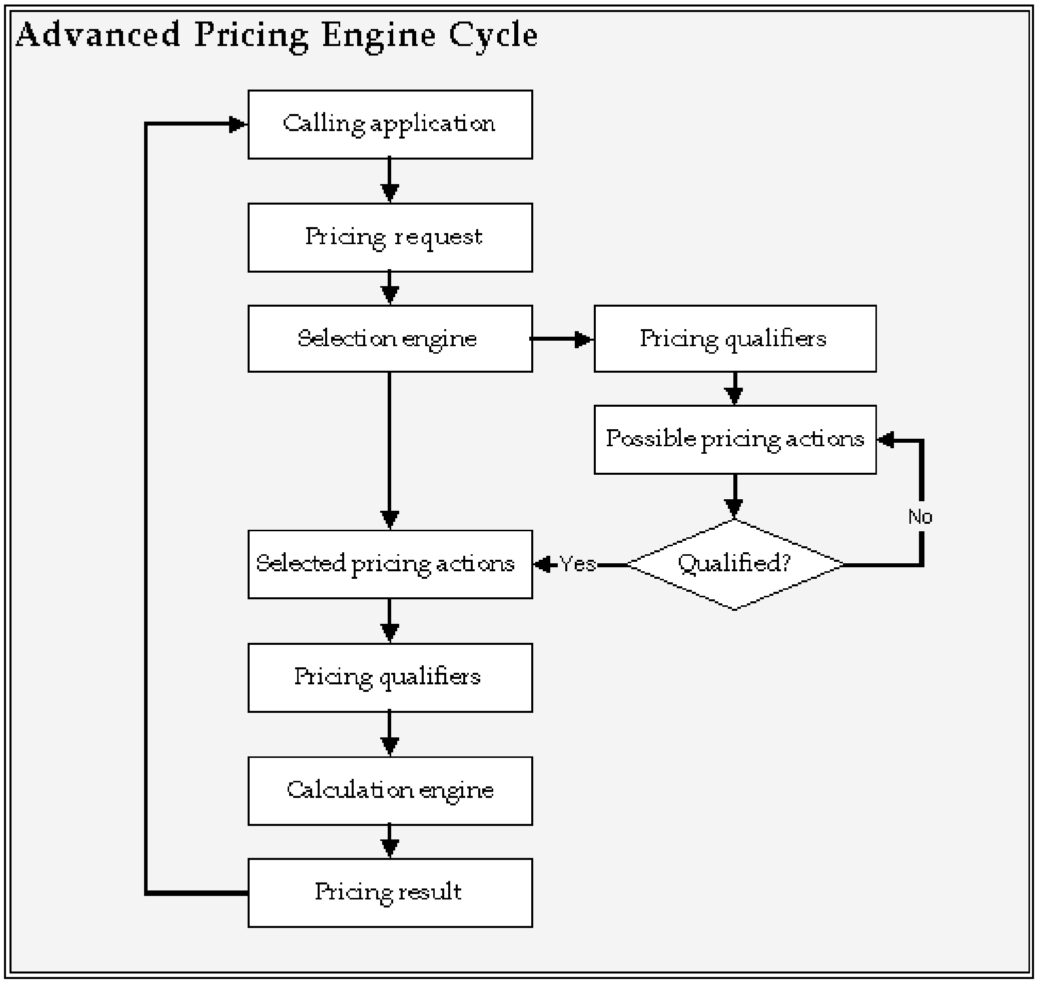
The pricing engine cycles each time a calling application makes a pricing request to the engine. Calling applications in Release 11*i* include Oracle iStore, Oracle Order Management, Oracle Contracts, Oracle TeleSales, Oracle Inventory, and Oracle Quoting.

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 Manual* 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:

Qualifier (Customer Class)	Qualifier Effective Dates	Price List	Effective Dates	Modifier	Exclusivity Group	Precedence
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
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
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 11*i* pricing engine performance. Consider merging these discount records into as few 11*i* modifier records as possible, and then use 11*i* 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 either through the use of price lists or modifiers at the higher level of 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 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 qualifiers that are redundant. 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.

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, there are several technical measures that 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

11*i* Oracle Advanced Pricing enables pricing engine execution to be broken up into phases, and enables each phase to be associated with an event in the calling application.

If there is no need to fetch or view the discounts at the time of entering the order lines, you can modify the event phases records to execute the discounting phases at the save. 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 ordered is entered, this technique can enhance pricing engine performance.

If you have unused pricing event phases, set the end date to a date in the past. 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 Oracle8i 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 Customer Name, site_use, 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.

Upgrade performance from Release 10.7/11 TO 11.I

Performance in upgrading the 10.7/11 pricing has been improved by providing:

- Parallel threads
- Bifurcation which enables you to upgrade the price lists and modifiers of active orders first and other data to be upgraded later at your convenience later. You may want to purge unneeded pricing data prior to the upgrade to simplify the upgrade process and improve the performance of the pricing engine.

If you purge the data, ensure that the data integrity of the transaction system is maintained. For example, you may want to purge the price list lines of the obsolete price list and upgrade just the price list header to maintain the data integrity.

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 which uses Oracle 8*i* temporary tables. The performance of pricing engine statements are 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 certain 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 (metalink Note 130511.1 has a script) 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.
3) Significant amount of time is spent during Pricing	Pricing Performance patches may not have been applied	Contact your support analyst to obtain any know 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. Pin 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 there are any unselected qualifiers. Restructure the setup either by moving these qualifiers to Pricing attribute or by moving the qualifier from line to header. Use qpperf.sql Stmt # 10,16

Symptom	Probable Cause	How to fix it
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 hangs	ST patches for temporary table may not have been applied	Check metalink 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.
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 metalink 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 unchecking the active flag. Use qpperf.sql Stmt# 5.
14) Trace file is showing custom code getting executed many time as well as taking huge time	Inefficient Extension /custom code is being called by Attributes mapping or by get custom price. Caching not implemented in the custom code	Implement caching, tune the custom code. If you are not able to implement cache for order attributes at line level which change 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 build context concurrent program.

Symptom	Probable Cause	How to fix it
16) Many modifiers, price list lines coming into the mix in the pricing engine.	1) multiple "Not=" qualifiers without any selective qualifier in the qualifier group. 2) Excessive use of "ALL_ ITEMS" product element.	Too many lines of the same product element or same qualifier will reduce the selectivity of the engine. Evaluate alternate setup. 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. 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 and use cust_code as a qualifier.
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 data is 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 Purge Concurrent program.

Pricing Setup window

The following are performance-related improvements:

- 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(N
VL(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 I
S

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 18-1.

Performance Patches (subject to changes)

Following patches have been provided to improve the Performance of the Pricing

Engine. You should always check with support for any changes in this list of patches.

Pricing Patch Number	Pack E	Pack F	Pack G	11*i*5	Comments
1508982	-	-	-	-	Dec 2000 - in 11i.4
1806021	X	-	-	X	May 2001
2043597	X	X	-	X	Oct 2001 - in G

E

Case Study: Pricing Scenarios in the High-Tech Industry

This appendix covers the following topics:

- Company Background
- Applying Oracle Advanced Pricing
- Results

Company Background

Tech Emporium is a fictitious manufacturer of high-tech gadgets for the networking industry. Their 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

1. 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 wish 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.

Pricing Requirement	Pricing Rules
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 be 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% for all of their orders. HTG belongs is in the emerging growth customer class; it is not eligible for the VIP customer discount.

There is a 5% discount on products in the infrastructure product group and a 10% 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 they will receive the lesser of two discounts. For example, a customer can not receive a 5% 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% and 10% discount, then they should only get the 5% discount.

Pricing Requirement 3
Applying discounts depends on if an item has received special pricing promotions. Products in the infrastructure category get a 5% discount if the price is derived from the customer specific price list. Products in the tools category are entitled to a 10% 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% 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 a order level VIP discount. Therefore The 5% discount and 10% discount can only be applied to the order and not individual lines.

Pricing Action	Pricing Rules
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 allows 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 it 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. Each pricing rule can be mapped 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% and a 10% discount. A rule is set that the customer cannot receive both discounts; the customer should only receive the 5% 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% and 10% 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 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, all the lines on the order must be evaluated by the pricing engine.

Discount Example

One of Tech Emporium's pricing rules is that customers associated with the OEM customer class are entitled to a 3% 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% VIP discount is an order level discount, while the 5% and 10% product discounts are line level discounts.

5% 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% 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% Discount

The order level discount must be defined and a customer class qualifier attached. The discount always applies when the customer meets the requirement. An incompatibility group must not be defined. Customer class is the qualifier.

Results

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:

Item	Quantity	Price List	List Price	Selling Price	Price Adjustment
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:

Item	Quantity	Price List	List Price	Selling Price	Price Adjustment
Brainglo	1	Corporate	200.00	180.00	10% tools discount
Infratimers	1	HTG	150.00	135.00	10% tools discount

F

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 healthy fast food alternatives to traditional fast food. There are over two thousand Healthy Fast Food restaurants 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 amount 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
- Location 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 fluctuations in price, HFF only makes adjustments to the cost components on a monthly basis.

The final price a customer is charged for the Burger depends on the burger size, type of bun, and type of cheese. There is an extra charge for lettuce, tomato, onion, pickles, mustard, or other toppings. Each topping selected will add one cent to the final price.

There are three different burger 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, there are six different geographic areas. This 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.

A. 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

B. Identify and Setup 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:
2. Size of the Burger
3. Type of bun
4. Type of cheese
5. Number of toppings
6. Store/distribution center matrix
7. 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.

   See Overview of Attribute Management, page 14-1 for information on the setup for entered and sourced pricing attributes.
8. Setup entered attributes:
9. Size of the Burger
10. Type of bun
11. Type of cheese
12. Setup sourced attributes:
13. Number of toppings
14. Distribution center location
15. 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.

C. 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 burger was purchased and the distribution center from where the Burger was shipped also determine the Burger's final price.

HFF uses the following equation to determine the price of the Burger:

Equation: Burger size + type of bun + type of cheese + number of toppings + distribution center/store location

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 the formula detail:

Formula Type	Pricing Attribute Context	Pricing Attribute	Component	Step
Factor	Null	Null	Burger Size	1
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 information for 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 information for the type of bun:

Base Pricing Attribute Context	Base Pricing Attribute	Operator	Value From	Value To	Adjustment Factor
Garden	Bun	Equals	White	Null	.02
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 information for the type of cheese:

Base Pricing Attribute Context	Base Pricing Attribute	Operator	Value From	Value To	Adjustment Factor
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 information for the distribution costs:

Line	Attribute Context	Base Pricing Attribute	Operator	Value From	Value To	Adjustment Factor	Start Date	End Date
1	Distribution Costs	Region	Equals	Northeast	-	.21	01-Jan-2001	31-Jan-2001
2	Distribution Costs	Region	Equals	Northeast	-	.25	01-Jan-2001	31-Jan-2001
3	Distribution Costs	Region	Equals	Northeast	-	.27	01-Jan-2001	31-Jan-2001
4	Distribution Costs	Region	Equals	Northeast	-	.30	01-Jan-2001	31-Jan-2001
5	Distribution Costs	Region	Equals	Southeast	-	.11	01-Jan-2001	31-Jan-2001
6	Distribution Costs	Region	Equals	Southeast	-	.15	01-Jan-2001	31-Jan-2001
7	Distribution Costs	Region	Equals	Southeast	-	.17	01-Jan-2001	31-Jan-2001

Associate pricing attributes for line 1:

Associated Pricing Attribute Context	Associated Pricing Attribute	Operator	Value From	Value To
Store	Location	Equals	Baltimore	Null

Associate 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.

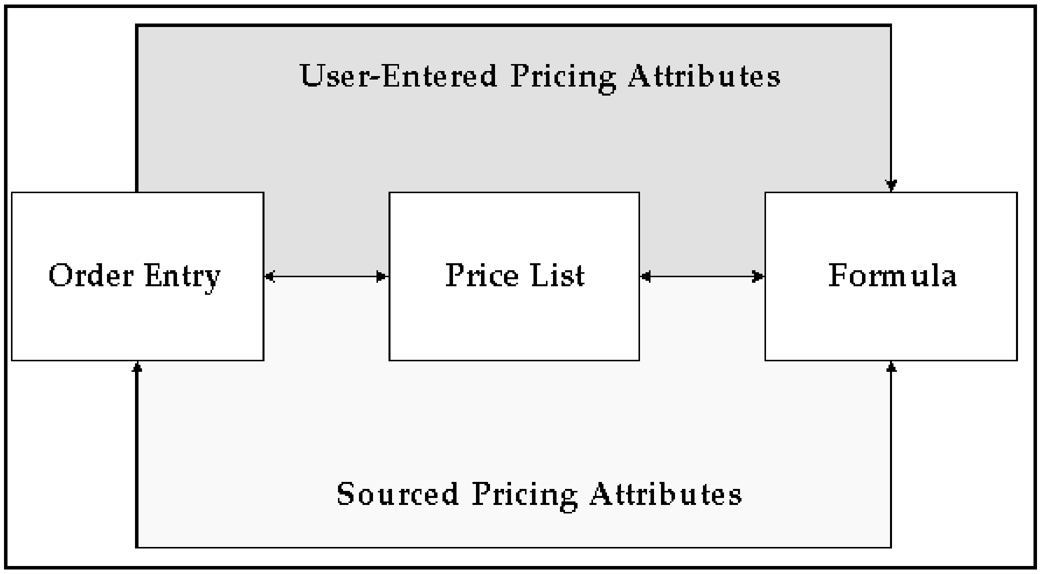
Identify Controls

HFF requires that the selling price of the Burger can change if the cost 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.

D. 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.

User-entered Pricing Attributes



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.

G

Lookups

This appendix covers the following topics:

- Lookups

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
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).

Code	Meaning
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.

Code	Meaning	Function
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.

Code	Meaning
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.

Code	Meaning	Function
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 U.I.'s.

The following table lists the default (seeded) values for this lookup type:

Code	Meaning	Function
%	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.
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.

Code	Meaning
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.

Code	Meaning
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.

Code	Meaning
DESC	Descriptive Flex Segment
KEY	Item Flex Segment
KEYFLEXFIELD	Descriptive Flex Segment
OEFORM	Field in OEFORM

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:

Code	Meaning	Function
=	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:

Code	Meaning	Function
=	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

Code	Meaning
FIXED	Fixed Date Type
PRICING_EFFECTIVITY_DATE	Pricing Effectively Date Type

Currency Conversion Method (CONVERSION_METHOD)

Access Level: System

Defines the currency conversion method.

Code	Meaning
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.

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

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.

Code	Meaning
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.

Code	Meaning
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.

Code	Meaning
CURRENCY	Currency
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.

Code	Meaning
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.

Code	Meaning
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:

Code	Meaning
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.

Code	Meaning
CRDEL	Create Deal List
CRDLT	Create Discount List
CRPRO	Create Promotion List
CRSLT	Create Surcharge List
DETDEL	View Deal List
DETDLT	View Discount List
DETPRO	View Promotion List
DETSLT	View Surcharge List
UPDDEL	Update Deal List
UPDDLT	Update Discount List
UPDLINES	Update
UPDPRO	Update Promotion List
UPDSLT	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 which are incompatible should be assigned to the same Incompatibility Groups, LVL1 - LVL3, and any modifier in a phase which 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.

Code	Meaning	Function
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.

Code	Meaning	Function
BEST PRICE	Best Price	Search Engine selects the Modifier which 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.

Code	Meaning
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.

Code	Meaning
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.

Code	Meaning
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.

Code	Meaning
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

Used to categorize the type of list which 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
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.

Code	Meaning	Function
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.

Code	Meaning	Function
CIE	Coupon Issue	Creation of a coupon which qualifies for a discount or promotional goods on a future request.
DIS	Discount	Reduces 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	Monetary charges which are calculated based on attributes of a request line, but which do not effect the selling price on the request line.
IUE	Item Upgrade	Substitution of one item for another on a request line, according to the pre-defined “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 line/s which are on the request.
PBH	Price Break Header	A series of base price or price adjustments which 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	Sets the base price of an item or level in product hierarchy.
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.

Code	Meaning	Function
SUR	Surcharge	Increases the list price, 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

Code	Meaning
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.

Code	Meaning
OPTCUR	Optional Currency

Organizer Formula Type (ORGANIZER_FORMULA_TYPE)

Access Level: System

Defines the formula types to be used in the Pricing Organizer.

Code	Meaning
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.

Code	Meaning
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.

Code	Meaning
N	No Products
P	Products
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 which determine which delimited break unit range/s 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 *total* volume falls.
RECURRING	Recurring	Volume break in which the modifier is given *for each* 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
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:

Code	Meaning
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.

Code	Meaning
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 etc. depends on the pricing phases which 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.

Code	Meaning	Function
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 which can be used to reprice an order line at any point during the order flow.
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;

this means that the adjustment amount for all modifiers in a bucket is calculated off 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.

Code	Meaning
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).

Code	Meaning
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.

Code	Meaning
CALC	Error in calculation engine
DUPLICATE_PRICE_LIST	Duplicate price list
D_PBH	Delete in Price Break Processing
FER	Error processing formula
GSA	GSA violation
INVALID_BEST_PRICE	Could not resolve best price
INVALID_INCOMP	Could not resolve incompatibility
INVALID_UOM	Invalid unit of measure
INVALID_UOM_CONV	Unit of measure conversion not found
IPL	Invalid price list
N	New record created
OER	Other error
OTHER_ITEM_BENEFITS	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.

Code	Meaning
M	Print Message
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.

Code	Meaning
D	Deleted
N	New
X	Unchanged

Proration Type (PRORATION_TYPE)

Access Level: System

Defines methods used to prorate discounts (none, category, all lines).

Code	Meaning
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.

Code	Meaning
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.

Code	Meaning
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.

Code	Meaning
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.

Code	Meaning
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.

Code	Meaning
=	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.

Code	Meaning
MISC	Miscellaneous

Rebate Payment Transaction Type Code (REBATE_TRANSACTION_TYPE_CODE)

Access Level: System

Defines the Rebate Payment Transaction Type Code.

Code	Meaning
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.

Code	Meaning	Function
BENEFIT	Benefit	Identifies those modifiers which are given as a benefit once the qualification criteria has been met.
COUPON	Coupon	Identifies the benefit which 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 which 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.

Code	Meaning
BUY	Buy
DETAIL_TO_DETAIL	Detail to Detail
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 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 which wishes to use Oracle Pricing should create a request type lookup code to identify their transaction.

The following table lists the default (seeded) value for this lookup type.

Code	Meaning	Function
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.
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.

Code	Meaning
NOREV	No Revision for this Agreement

Rounding Type (QP_ROUNDING_TYPE)

Access Level: User

Used by the engine for rounding the price.

Code	Meaning
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.

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.

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.

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.
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.

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

.

Code	Meaning
ACCESSORIAL	Accessorial Charge
SERVICE_SURCHARGE	Additional Service Charge

Time UOM Conversion (QP_TIME_UOM_CONVERSION)

Access Level: System

Defines the the options to set Time UOM Conversion either by Standard or by Oracle Contracts.

Code	Meaning
ORACLE_CONTRACTS	OKS Time Conversion
STANDARD	Standard

Types of Context (QP_CONTEXT_TYPE)

Access Level: Extensible

Defines the type of available contexts.

Code	Meaning
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.

Code	Meaning
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 attribute can be sourced.

Code	Meaning	Description
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
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.

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.

.

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.

Code	Meaning
No	No
Yes	Yes

Index