Oracle® Performance Analyzer

Reference Guide

Release 4.5

May 2002

Part No. A80987-02

ORACLE®

Oracle Performance Analyzer Reference Guide, Release 4.5

Part No. A80987-02

Primary Author: Steven Roepke

Contributing Authors: Kevin Courtney, Greg Hall, Claudia Hawkins, and Joe Stolz

Contents

3 Multicurrencies

4 Overview of IDs

5 Data Verification ID

6 Table ID

7 Lookup Table ID

8 Allocation ID

14 Transformation ID

15 Batch ID

16 Party Profitability Process ID

Send Us Your Comments

Oracle Performance Analyzer Reference Guide, Release 4.5

Part No. A80987-02

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

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

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

- Electronic mail: your_product@us.oracle.com
- FAX: telephone number   Attn: Oracle Your Product Name Here
- Postal service:

  Oracle Corporation
  Oracle Your Product Name Here Documentation
  Street Address, Mailstop
  City, State Postal Code
  Country

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

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

Preface

This reference guide describes the features of Oracle Performance Analyzer. This preface describes the following information about the reference guide:

- Intended Audience
- Organization
- Report-Related Changes
- Related Documents
- Conventions
- Customer Support Information

Intended Audience

This reference guide is written for Cost Analysts, Operational Managers, and others intending to use Performance Analyzer. It assumes familiarity with graphical user interfaces and knowledge of Oracle Financial Services Applications (OFSA) functionality, terminology, and navigation. It also assumes that Performance Analyzer has already been installed on the client/server system.

Organization

This manual is organized into the following chapters:

Chapter	Contents
Chapter 1	Performance Analyzer is introduced, providing Setup information, format, and menu descriptions.
Chapter 2	An overview of the performance analysis process is described, with discussions on Leaves and IDs.
Chapter 3	Describes implementation of multicurrencies in the in the Oracle Performance Analyzer profitability solution.
Chapter 4	An overview of IDs is described, with an explanation of the ID concept. In addition, related functions associated with IDs (such as importing and exporting) are covered. Tree related IDs are also addressed.
Chapter 5	Data Verification ID is defined. Editing the ID is also covered.
Chapter 6	Table ID is defined. Editing the ID is also covered.
Chapter 7	Lookup Table ID is defined. Setting up and editing are also covered.
Chapter 8	Allocation ID is defined, including working with multiple record allocation IDs and processing results.
Chapter 9	Data Filter ID is defined. Further explanation covers the processing of a Data Filter ID, as well as examples.
Chapter 10	Group Filter ID is defined. Further explanation covers the processing of a Group Filter ID.
Chapter 11	Tree Rollup ID is defined, with instructions on editing the ID. Tree Rollup ID examples are also covered.
Chapter 12	Tree Filter ID is defined, with instructions on editing and processing the ID.
Chapter 13	Formula ID is defined. Additional topics include fundamental concepts, functions, and examples.
Chapter 14	Transformation ID is defined, with instructions on usage, editing, and processing.

Chapter	Contents
Chapter 15	Batch ID is defined, covering serial and parallel Batch ID processing. Instructions on editing the ID are also included.
Chapter 16	Party Profitability Process ID is defined. Topics include setup, specific features, processes, procedures, definition of customer account tables, processing tables, global options, and diagnosis of problems and error messages.
Appendix A	Fiscal year information is provided.
Appendix B	Error Messages found in the Transformation ID are listed.
Appendix C	Routines that are necessary to integrate Oracle Activity Based Management with Performance Analyzer are described.
Appendix D	The OFSA insurance-specific data model is described. It is designed to integrate with the existing Performance Analyzer application and serve as a data warehouse for management reporting purposes.

This reference guide also contains a glossary and an index.

Report-Related Changes

Release 4.5 of OFSA fully integrates Oracle Discoverer with the following OFSA products:

- Oracle Financial Data Manager Balance & Control
- Oracle Performance Analyzer
- Oracle Risk Manager
- Oracle Transfer Pricing

With Release 4.5, Discoverer becomes the recommended reporting tool for OFSA.

This release supports both Discoverer and the reporting tool used in prior OFSA releases. You can find information about both reporting tools in the following documents:

Reporting Tool	Documentation
Oracle Discoverer	Oracle Discoverer User Guide Oracle Discoverer Administrator Guide
OFSA reporting tool	Oracle Portfolio Analyzer Reference Guide

Report-related chapters and appendixes have been removed from all OFSA reference guides and moved to the *Oracle Portfolio Analyzer Reference Guide*.

The following list describes the changes:

- Report ID
- Report Macros
- Report Runner Dialog
- Report Stratification ID, Subtotal ID, and Standard Reports

Related Documents

For more information about using Performance Analyzer, see the following related documents:

- *Oracle Risk Manager Reference Guide*
- *Oracle Rate Manager Reference Guide*
- *Oracle Financial Data Manager Balance & Control Reference Guide*
- *Oracle Financial Data Manager Administration Guide*
- *Oracle Financial Services Installation and Configuration Guide*

Conventions

This reference guide uses the following conventions:

Convention	Meaning
. . .	Vertical ellipsis points in an example mean that information not directly related to the example has been omitted.
. . .	Horizontal ellipsis points in statements or commands mean that parts of the statement or command not directly related to the example have been omitted.
boldface text	Boldface type in text indicates a term defined in the text, the glossary, or in both locations.
`bold monospace`	Bold monospace type in text indicates information that you type in.
Italics	Italics emphasize a word or phrase.
< >	Angle brackets enclose user-supplied names (for example, <Branch Name>).
[ ]	Brackets enclose function and terminal keys. In common syntax, brackets denote one or more optional items.
{ }	Braces are used to denote variables, and in command syntax, a choice within a mandatory item. Example of command syntax: *Warning: INIT file {filename} already exists.* Example of choices: {EXIT | QUIT}

Symbols

- Bullets indicate a list of items or topics.

1. Numbered lists are used for sequential steps in completing a procedure.

Orientation of Procedures

Procedures in OFSA reference guides are generally menu-driven rather than command- or icon-driven. Only occasionally is a reference to a toolbar or mouse action necessary because the action has no menu equivalent. If you prefer to use the toolbar icons, refer to Chapter 1, "Introduction".

Notes, Cautions, and Warnings

Certain information may be set off in boxes for purposes of emphasis:

- *Note* refers to interesting but incidental information about the product or information that may be important but of lesser degree than a *Caution* or *Warning*.
- *Caution* indicates the possibility of damage to a product, system, or data.
- *Warning* refers to a situation that is potentially hazardous to people.

Customer Support Information

Product support is available through Oracle Support Services. Contact your project manager for information about using the support options offered in your geographic region. These options may include the following:

- MetaLink (which provides online access to information about Technical Libraries, Patches, TARs, and Bugs and is available at metalink.oracle.com)
- Telephone support

1

Introduction

Isolating and analyzing business unit profitability is only the first step in building effective profitability management. To hold operational managers truly accountable for profitability, senior management must give them the power to identify the products that are profitable in each market segment, for each customer.

Oracle Performance Analyzer is the tool that delivers this power. Performance Analyzer links general ledger, account-level, and statistical data together to produce detailed financial statements for individual business units, products, and customers.

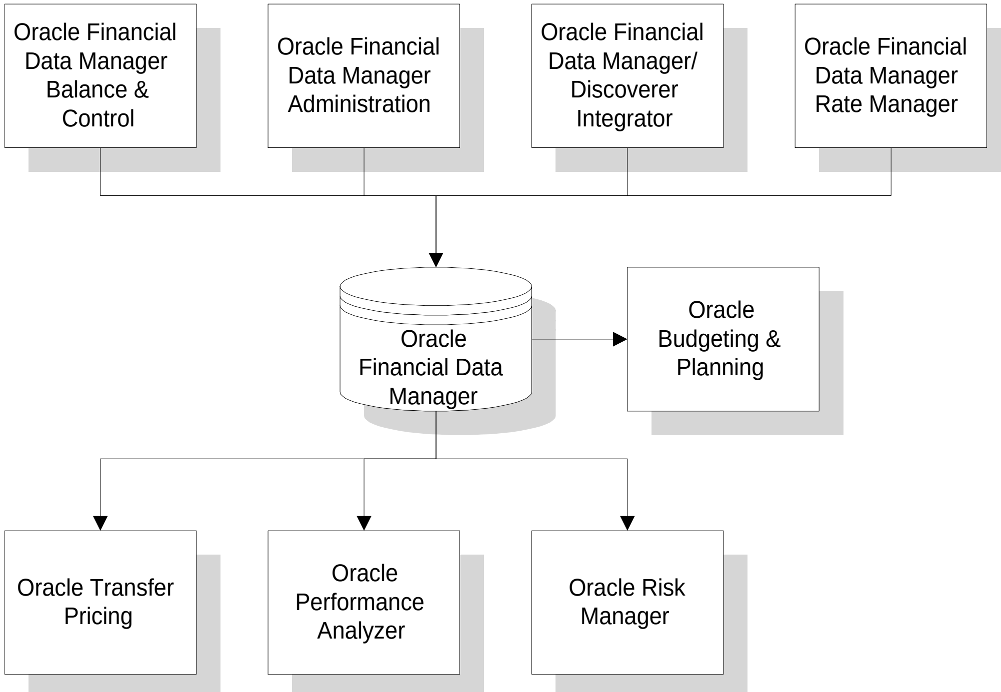
Combining powerful modeling and disciplined accounting, Performance Analyzer delivers complete, accurate, and inclusive profiles of profitability.

This chapter presents the following topics:

- Oracle Financial Services Overview
- Logging in to OFS
- OFS Applications
- Menus and Toolbars

Oracle Financial Services Overview

You can use Performance Analyzer as a standalone application or as part of the Oracle Financial Services (OFS) group of applications. As with all the other OFS applications, however, the Oracle Financial Data Manager provides the foundation for Performance Analyzer.



OFS Applications

OFS applications form a comprehensive decision support solution that significantly enhances transfer pricing, budgeting and planning, risk management, and performance measurement functions across a financial institution.

Oracle Financial Data Manager

Oracle Financial Data Manager (FDM) is a standalone data warehouse with prepackaged data elements for the financial services industry. FDM is also the foundation for the OFS applications. It provides the database structures necessary to support the individual business applications.

FDM includes Oracle Financial Data Manager Balance & Control, Oracle Financial Data Manager Administration, Oracle Financial Data Manager / Discoverer Integrator, and Oracle Financial Data Manager Rate Manager.

Oracle Financial Data Manager Balance & Control

Balance & Control validates, corrects, and aggregates data from the FDM.

Oracle Financial Data Manager Administration

FDM Administration manages the FDM, providing security and maintenance capabilities.

Oracle Financial Data Manager/Discoverer Integrator

Discoverer Integrator integrates the FDM database with Oracle Discoverer, which provides ad hoc reporting, analysis, and Web publishing capabilities.

Oracle Financial Data Manager Rate Manager

FDM Rate Manager manages interest rate, exchange rate, and currency information for the FDM.

Oracle Budgeting & Planning

Budgeting & Planning provides performance-based planning. It integrates cash flow balance sheet and net income forecasting capabilities with the scalability and customizable framework of Oracle Financial Analyzer, part of the Oracle Express group of data access and analysis tools.

Oracle Transfer Pricing

Transfer Pricing calculates a transfer rate for each account and a charge or credit for funds for each asset or liability.

Oracle Performance Analyzer

Performance Analyzer provides comprehensive and flexible cost and equity allocations. It measures product, business unit, and customer profitability.

Oracle Risk Manager

Risk Manager forecasts cash flows, interest income, and market value to manage rate risk.

Logging in to OFS

To log in to any OFS application, double-click the appropriate OFS icon from Windows. The OFS Login window appears.

In Windows NT or Windows 95, you can run multiple OFS applications, but you should run each application in its own memory space.

Menus and Toolbars

You can use menus or the toolbar icons to activate Performance Analyzer options.

Menus

A typical OFS menu bar consists of the following menu options:

- File
- Edit
- Process
- Setup
- Options
- Tools
- Window
- Help

File Menu

The options of the File menu are used primarily for file management. This includes creating new files, opening existing files, saving, deleting, and renaming files. This menu also contains commands for printing IDs and windows, and for exiting the application.

Save Select Save to save IDs as you are creating or modifying them. You are prompted to confirm the save.

You should save your work every few minutes. From the File menu, you can select Save, or you can click on the Save icon (the floppy disk) on the toolbar.

Delete Select Delete to remove the open and active ID from the disk. You are prompted to confirm the deletion.

If you want to delete more than one open ID, make each ID active by clicking anywhere within the window and delete it.

Delete Group To delete a group of IDs, complete the following steps. Each ID does not need to be open and active for you to delete the entire group.

1. From the file menu, select Delete Group. The Delete Group window appears.
2. From the ID Type list, select the type of ID that you want to delete. All IDs of this type appear in the list below the ID Type list.
3. Select the IDs you want to delete, and select the Add button to list them in the ID Type/ID Name box.
4. Repeat steps 1 and 2 for as many ID types as required. You can select and delete multiple IDs of multiple types all at once. The following table describes other features of the Delete Group window.

Feature	Description
Selection Buttons	■ Four buttons control the movement of IDs between the ID list box and the Selected IDs box. ■ Select Add to add the ID to the Selected IDs box. Select Select All to select all the IDs of the chosen type. ■ Select Remove to remove an ID from the list you have made in the Selected IDs box. ■ Select Remove All to remove all of them from the Selected IDs list.
Order Buttons	■ To change the order of IDs in the Selected IDs box, select an ID and click the up or down Order arrows located below the list box.
ID Information	You can view various data about each ID before you delete it. Select any ID name in the ID list box, and click the right mouse button. An information box for that ID appears.

5. When your list of IDs is complete, select Remove All.

Import/Export Group Refer to “Importing and Exporting IDs” in Chapter 4, "Overview of IDs" for a full discussion of this menu option.

Dependencies Refer to “Dependencies”Chapter 4, "Overview of IDs" in for a full discussion of this menu option.

Print Select Print to print the open and active ID or window. Select Yes to proceed, or No to cancel the print job.

Print Group To print a group of IDs, complete the following steps. It is not required that each ID in the group is open in order to print the entire group.

1. From the File menu, select Print Group.
2. Select the type of ID you want to print from the ID Type list. All IDs of this type appear in the list below the ID Type list.
3. Select the IDs that you want to print, and select the Add button to list them in the Selected IDs box.
4. Repeat steps 2 and 3 for as many ID types as required. You can select multiple IDs of multiple types.
5. To change the order of the listed IDs in the Selected IDs list box, select an ID and click the up or down Order arrows below the list.
6. When your list of IDs is complete, click OK to proceed with the group print.
7. Click Yes to proceed or No to cancel the print job.

Edit Menu

Use the Edit menu options to copy and move material from place to place, and to delete spreadsheet rows.

Search Use the Search option to locate a specific leaf value. There are three search options:

Search Option	Action
Search Field	Specify Leaf Value or Description.
Search String	Type in the Leaf Value or Description.
Search Options	Select Case Sensitive if you want your search to match the string exactly as you entered it. Select Continue Searching... if your search begins part way down the tree (at the cursor location) but you also want to search above. Click OK to begin the search. When the search is complete and successful, the search window closes and the found leaf ID is highlighted. If the search string is not found, an error message appears.

Process Menu

The Process menu contains the following options:

- Run
- Server Status
- SQL Talk

Run Select this option to begin any process.

Server Status Use this option to check the status of the processes running on the server.

SQL Talk

Use this option to interface with the database and call subprograms.

Setup Menu

The Setup menu contains one option: Leaves.

Leaves To access the Leaf Setup window, select Leaves from the Setup menu.

With Leaf Setup, you can view, edit, insert, renumber, or delete any Leaf Value from any Leaf Type in the database. For further information about the Leaf Setup window, refer to “Leaf Setup”, in Chapter 4, "Overview of IDs".

Options Menu

Use the Options menu to set up preferred screen displays, and to change users' passwords.

Tool Bar This is a toggle option; that is, it turns the display of the toolbar on and off. When the option is “on,” a check mark is displayed.

Status Bar This is a toggle option; that is, it turns the display of the status bar on and off. When the option is “on,” a check mark is displayed.

Customize ID Toolbar You can customize the vertical toolbar using this option. For more information, see the section "Toolbars" in this chapter.

Change Password Use this option to change the password for the Current User of this OFS application. When you select Change Password, the Change User Password window appears.

Option	Description
Login name	The name of the current user appears here.
Current password	This box appears blank. Enter the current password for the current user. Asterisks appear as you type, for security reasons.
New password	This box appears blank. Enter the new password for the current user. Asterisks appear as you type, for security reasons. Click OK to execute the change. Click Cancel to cancel the operation and close the window.

Tree Bar When you open a product-based or currency-based ID, the tree bar appears. Select a product or currency from the split window. If you worked with another product- or currency-based ID during the current work session, the selections default to the previous product or currency you selected. Otherwise, the selections default to No Product (0) and Default Currency (000). For all currency-based IDs except Forecast Balance ID and Transaction Strategy ID, select Default Currency (000) to set up assumptions for all the currencies you use. You can move, resize, or close the tree bar.

Control Bar For information about the Control Bar, refer to "Spreadsheet Control Bar in Chapter 4, "Overview of IDs".

Tools Menu

Use the Tools menu to start the Discoverer applications: Oracle Discoverer User Edition, Oracle Discoverer Administration Edition, and Discoverer Integrator. The Tools menu is always present. If an application in not installed, however, its name is grayed out and it is not available for use.

Window and Help Menus

The Window menu options are for managing windows and icons, and for moving the cursor between windows.

The Help menu provides an online version of this reference guide.

Status Bar

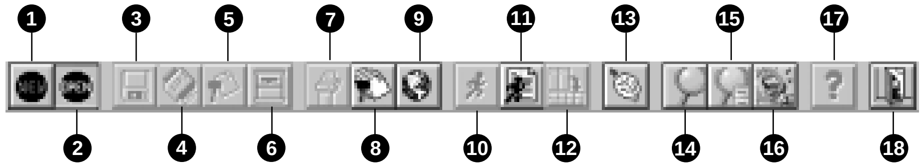
The Status Bar at the bottom of the screen displays information about current operations. Position the pointer on any Toolbar icon to display the name of the icon on the Status Bar.

Toolbars

The horizontal and vertical toolbars are panels of icons that support shortcuts for menu commands and IDs.

Horizontal Toolbar

The horizontal toolbar provides shortcuts to menu commands. Following is the horizontal toolbar and the function each icon activates. You cannot customize this toolbar.



	Icon	Description
1	New	Create a new ID or pattern
2	Open	Open an existing ID or pattern
3	Save	Save an ID or pattern
4	Print	Print results
5	Delete	Delete an ID or pattern
6	Close	Close an ID or pattern
7	Print Group	Print a group of IDs
8	Delete Group	Delete multiple IDs
9	Group Import/Export	Import or export multiple IDs
10	Run	Run an ID
11	Report Runner	Run an active report
12	Transform Data	Run transformation on selected data
13	Set Up Leaves	Define user-defined and system leaves
14	Discoverer User Edition	Open the user edition of Discoverer
15	Discoverer Administration Edition	Open the administration edition of Discoverer
16	Discoverer Integrator	Open the OFS/Discoverer Integrator application
17	Leaf Tree Bar	Display the leaf tree bar
18	Spreadsheet Control Bar	Automate entry of data for spreadsheet-enabled IDs
19	Help	Access online help
20	Exit	Exit the OFS application

Note: The Discoverer icons (14, 15, and 16) are grayed out if the application is not installed.

Vertical Toolbar

The Vertical toolbar provides shortcuts to the IDs to which you have access in the application. Space limitations may restrict the icons that you see on the screen. However, you can customize the toolbar for your specific needs.

To customize the vertical toolbar, complete the following steps:

1. From the Options menu, select Customize ID Toolbar.

   The Customize ID Toolbar window appears.

2. Locate the icon you want to add, delete, or move.

   Use the scroll bar on the Customize ID Toolbar window to find the icon that you want to add to the vertical toolbar. Or, find the icon on the vertical toolbar you want to delete or move.

3. Add, delete, or move the icon.

   To add an icon to the toolbar, select and then drag the icon from the Customize ID Toolbar window to the vertical toolbar.

   To delete an icon, drag the icon from the toolbar to the Customize ID Toolbar window.

   To change the order of the icons, drag them individually from one location on the toolbar to another.

2

Overview of Performance Management

An effective performance management system should include the following key requirements:

- Flexible, efficient access to account level (such as loan-by-loan, deposit-by-deposit) information
- Multiple views of the organization
- Matched rate funds transfer pricing at the account level of detail
- Flexible definition of cost accounting processes with strong accounting controls
- Powerful modeling capabilities

Oracle Performance Analyzer addresses these requirements by linking general ledger, account-level, and statistical data together in a unique manner. All data is linked together through user customized hierarchies (or trees). Each row in the database, whether a ledger row or an account level row, has a data value (or leaf value) for each dimension of profitability (or view of the organization).

Trees can be constructed for each dimension, thereby organizing a large volume of detailed data into a simple, yet powerful framework. These trees form the basis for generating complex cost allocations and all financial reports.

This chapter presents the following topics:

- Multiple Dimensions of Profitability
- Leaves
- Tree Rollup IDs
- Tree Filter IDs
- Allocation IDs

Multiple Dimensions of Profitability

Views represent the following four dimensions of profitability:

Organizational View

An Organizational View provides the measurement of profitability at any level in your organizational structure, such as Bank Level, Region Level, Branch Level, or Cost Center Level

Product View

A Product View provides the measurement of profitability of specific product lines, such as Retail Deposits, Retail CDs, or even certain retail CD offerings.

Customer View

A Customer View provides the measurement of profitability of various customer segments, such as age or socioeconomic segments.

Channel of Distribution View

A Channel of Distribution View provides the measurement of the profitability of various methods for conducting business, such as ATM activity, Teller Activity, or Mail-In Activity.

Leaves

Leaves are data codes that represent key dimensions across database tables in the Oracle Financial Data Manager (FDM) database. For example, leaves can represent Cost Centers, Product codes, Chart of Account codes, or account transactions. Leaf values can be assigned either through the data extract process or directly in Balance & Control through the Data Correction Processing ID. Because you must give each leaf value some meaning (for example, assign a description and indicate if the account is an asset, liability, or equity), you must define all new leaves through the Setup Leaves menu.

The Purpose of Leaves

Leaves provide two major benefits:

- Leaves enable you to Create Customized Hierarchies, or Trees, that facilitate both data segmentation and aggregation. Leaves are the basis upon which you can build Tree Rollups, Tree Filters, and Subtotal IDs.
- Leaves define the relationship between the Ledger Data and the Instrument Data. Leaf values tie your source instrument and ledger data to these Tree Rollups, thereby defining the relationship between instrument and ledger data. This provides for an unlimited degree of flexibility in reporting and performing cost allocations across several dimensions.

Tree Rollup IDs

Tree Rollup IDs are custom hierarchies created from your leaves. You can build as many types of Tree Rollup IDs as you have leaf columns defined, such as an Organizational Tree or Product Tree. Within each type of tree (for example, each leaf column), you can construct an unlimited number of different tree rollups for organizing your data in different ways.

The Purpose of Tree Rollup IDs

Tree Rollups form a logical way of segmenting and summarizing data and therefore form the basis for several other IDs:

- Tree rollups aid in the generation of flexible summary level financial reports.
- Tree rollups can create flexible and easy-to-maintain allocation rules in an Allocation ID.
- Tree rollups can filter (narrow the focus) of any reporting or allocation process through a Tree Filter ID.

How to Create and Edit Tree Rollup IDs

See Chapter 4, "Overview of IDs".

Tree Filter IDs

Tree Filter IDs are data filters built in the context of a Tree Rollup ID. Selecting particular Leaves or Nodes (for example, subtotal levels) on a hierarchy includes only the detail data underneath that particular leaf or node.

The Purpose of Tree Filter IDs

Tree Filter IDs enable you to focus on detail data for a particular part of a hierarchy. For example, you can use a tree filter to include only bank branches in the Western Region for a report or allocation. If your organizational Tree Rollup ID is already constructed, you do not need to remember which specific branches make up the Western Region. The Western Region, with possibly hundreds or thousands of branches, is already defined in your tree rollup. You select it when specifying your report or allocation. Performance Analyzer includes the correct branches.

If you want to drill down into the Western Region for more precise analysis, however, you can drill down as far as you need to go. You determine the level of detail.

How to Create and Edit Tree Filter IDs

For a full description of Tree Filters, see Chapter 12, "Tree Filter ID". For a general discussion of tree-related IDs, see Chapter 4, "Overview of IDs".

Allocation IDs

Defining your Allocation Methodology

Defining the methodology for allocating expenses and crediting or debiting accounts is critical to the performance measurement process. After you have defined the methodology, you create Allocation IDs to implement it.

What Are Allocation IDs?

Allocation IDs are rules that enable you to transfer amounts between cost centers, or push down an expense from a high level in the organization to any level of detail below. An allocation rule uses data in the FDM database and adds new data to the existing FDM database.

The Purpose of an Allocation ID

Use an Allocation ID any time you want to move data between different leaf values, such as cost centers, chart of accounts, or products. Also, use an Allocation ID anytime you need to push data to a lower level of detail. For example, you can use an Allocation ID to charge a loan branch and credit a loan servicing center for processing each branch's loans. An Allocation ID can access tree filters, data filters, and individual leaves in order to create any type of allocation.

How to Create and Edit Allocation IDs

See Chapter 8, "Allocation ID".

Testing Your Allocation IDs

After you have defined an Allocation ID, you can test the results of the allocation without actually performing the allocation. You do so by using the **Preview** option in the Allocation ID.

Performing an Allocation Process

Once you are satisfied with the previewed results of the allocation, you can process an Allocation by itself or as part of a Batch ID. See Chapter 15, "Batch ID".

Reversing an Allocation Process

If you find a mistake in your allocation definition, you can reverse the results of the allocation by using the Undo option in the Allocation ID. For more information on the Undo option, see Chapter 8, "Allocation ID".

3

Multicurrencies

This chapter describes implementation of multicurrencies in the in the Oracle Performance Analyzer profitability solution. Specifically, it covers:

- Enhancements in Oracle Performance Analyzer in OFSA Release 4.5 for the implementation of multicurrencies in the profitability solution
- How currency translation is used in the calculation of Oracle Performance Analyzer
- Use of multicurrency output from Performance Analyzer in relation to LEDGER_STAT, Instrument, and Transaction tables.

This chapter is divided into the following sections:

FDM Rate Manager Currency Definitions and Calculating

Oracle Financial Data Manager Rate Manager (FDM Rate Manager) is a rate management utility enabling the user to manage complete interest rate, exchange rate, and currency data with a high degree of security and control. As part of the FDM database, it expands upon and replaces the Historical Rates ID and serves all applications in the Oracle Financial Services (OFS) group of applications.

FDM Rate Manager handles all currency definitions for OFS applications. Currencies are conveniently referred to by code and written description. A comprehensive list of ISO-defined currencies is included, and the user can define and add his or her own.

Upon installation, one currency is active and ready for processing, the one identified by the user's organization as the functional currency. The user can activate other currencies and define and activate newly-created currencies when he or she needs them.

The user can define, input, and maintain exchange rates between any two active currencies. In addition to the standard floating exchange relationships, special fixed relationships are available, such as those within the European Monetary Union (EMU). FDM Rate Manager complies with EMU structure and legislation, and supports changes in currency rates as well as additions and deletions of the currencies of member countries.

Exchange rates between currencies used in Performance Analyzer allocations should be reciprocal. The rate of exchange between two currencies must be arbitrage free. Performance Analyzer does not calculate any currency exchange offsets for non-reciprocal exchange rates.

Functional Currency

At the time of installation, FDM Rate Manager requires the installer to designate a functional, or primary, currency for the organization. A Swiss multinational bank would therefore designate the Swiss franc as its functional currency. Only one functional currency is allowed per organization. Once assigned, it cannot be changed. The functional currency is always an active currency.

Performance Analyzer allocations can read and write in multicurrencies. However, Performance Analyzer performs internal calculations in memory for allocations in the functional currency. This means that allocations do a double conversion.

When multicurrency is off, all ISO Currency Code values are written in the functional currency.

Currency Codes

ISO Currency Codes compose a predefined list of the world's major currencies. ISO Currency Codes are seeded in the FDM database at installation. ISO Currency Codes are the attributes that define the currency of any row in the database. ISO Currency Code values used in Performance Analyzer must be designated active in FDM Rate Manager. ISO Currency Code can have user-defined values. User-defined ISO Currency Codes are set up in FDM Rate Manager.

Users must designate ISO Currency Codes as *reporting currencies* in order to activate the calculation of conversion rates. Simply designating the currency as active is not sufficient.

Code 002

The ISO Currency Code value of *002* represents statistical LEDGER_STAT data. An ISO Currency Code value of *002* means no currency basis. An ISO Currency Code of *002* is used in situations where the row is statistical and has no currency associated with the value. A row representing square footage in a building could have a 002 ISO Currency Code and does not need a currency association. Assigning an ISO Currency Code to a statistic is useful for matching rows that have the same leaf values but differ only by ISO Currency Code.

Table Structure Changes

In OFSA release 4.0, all the instrument tables had a Currency Code column. OFSA Release 4.5 changes the Currency Code column to ISO Currency Code. In addition, the ISO Currency Code column appears in the LEDGER_STAT and Transaction tables.

Functionality Breakout

LEDGER_STAT Load

The FDM database comes with a process for loading monthly LEDGER_STAT data that maintains the leaf integrity and updates the year-to-date balance fields. The LEDGER_STAT load process changes slightly to accommodate multicurrency implementations in the OFSA Release 4.5.

The column property characteristic of the Financial Element ID leaf on a LEDGER_STAT row determines whether a value needs currency translation. If column property is balance or balance weighted object, then the value is translated. If other column properties are chosen, then no translation occurs on the value.

The above rule leads to two checks during the LEDGER_STAT load process. If no ISO Currency Code value exists on data with the Financial Element ID column property defined as balance or an invalid code is entered, then ISO Currency Code is defaulted to the functional currency code. If no ISO Currency Code value exists on data with Financial Element ID column property other than balance, such as statistic, then ISO Currency Code is defaulted to *002*. If data has an existing ISO Currency Code, then it is loaded as is. Statistics can have ISO Currency Codes in different currencies. Statistics with multiple ISO Currency Codes enable the *by leaf* functionality in the Allocation ID to function properly.

Allocation ID

In OFSA Release 4.5, the allocation engine is enhanced to enable processing of balances in multicurrencies. ISO Currency Code is added to all dialogs that list and ask for leaf values. ISO Currency Code works similar to other leaves in OFSA. A Macro exists in the ISO Currency Code dialog for functional currency. In addition, right clicking on the Leaf Number or Description of the Currency leaf in the edit box enables the use of macros for <All>, <Same as Filter>, <Same As %Dist>, and <Same As %Dist-Leaf>. The allocation engine can read multiple input currencies and output multiple result currencies.

Allocation Engine Process Processing Flow

The allocation engine:

1. Reads input data
2. Converts all input data to the functional currency. By having all inputs in the same currency, calculations, such as percent distributions, are created on apples-to-apples basis.
3. Performs the allocation calculations. The output results of the allocation can be in any currency.
4. Converts the results of calculations from functional currency to the appropriate debit or credit currency for that allocation.

Input rows to an allocation are checked to determine whether currency translation is necessary. The checks for whether an item needs translation vary between LEDGER_STAT and the Instrument and Transaction tables.

When reading rows from LEDGER_STAT, the user translates values based on the Financial Element ID. The column property characteristic of Financial Element ID determines whether a value needs currency translation. If column property is *balance or balance weighted* object, then the value is translated. If other column properties are chosen, then no translation occurs on the value. Currency translation needs are determined row by row in LEDGER_STAT.

Allocations updating the LEDGER_STAT table view ISO Currency Code as a leaf. For example, in a simple constant type allocation, if two rows in LEDGER_STAT share identical leaf values but differ in currency type, the expected result writes two rows when performing a multicurrency allocation. If multicurrency operations are not enabled, the same two rows are accumulated into a single row because ISO Currency Code is not viewed as a leaf type column.

Performance Analyzer extracts conversion rates from the FDM Rate Manager tables. Conversion rates are created from each currency to the functional currency. A rate is entered for each currency on the basis of the ending exchange rate for that date, corresponding to the As Of Date in Performance Analyzer.

How Allocations Handle Currency Conversion in a Systematic Way

The following is an example of how allocations handle currency conversion in a systematic way. A row of data is read in. If the row is a balance, the engine accesses the conversion rate between the row's currency and the functional currency for a date that matches the As Of Date in the current Configuration ID. If a rate for that date is found, it is used. If no conversion rate exists for that date but an earlier conversion rate is found, that earlier rate is used. When a matching date or earlier date is found a message is logged in OFSA_MESSAGES table, *message_cd = 1104, "An exchange rate was found*." If no conversion rate is found between the specified currencies, a different warning message is posted in the OFSA_MESSAGES table, *message_cd = 1100, "No exchange rate found. Defaulting to 1.0 for exchange rate*." Using the retrieved rate, the row is converted to the functional currency. Next, math operations specified in the allocation rule are performed on the row. Finally, the row is ready to be written back to LEDGER_STAT. The ISO Currency Code of the target row is determined from the Debit/Credit part of the allocation rule and the appropriate conversion rate is used when the row is written to LEDGER_STAT.

> **Note:** Math operations with constants assume that the constant value is in the functional currency.

When instrument and transaction records are read, values are translated based on column properties. The OFSA_TAB_COLUMNS table contains the column OFSA_DATA_TYPE_CODE. All tables and columns are stored in the OFSA_TAB_COLUMNS table. If the OFSA_DATA_TYPE_CODE for a column is a *1*, balance, then the value from that column is translated in allocations. If the OFSA_DATA_TYPE_CODE value is other than *1*, no translation occurs for that value in allocations. Instrument and transaction tables determine translation of values on a column-by-column basis. When writing from column to column within an instrument or transaction table row, no currency conversion is performed. When writing from one instrument table to another, a conversion is made from the row's currency into the functional currency. The conversion rate of the target instrument row is used when the results are written.

When allocation results write a new row to LEDGER_STAT, the output financial element determines if the value is translated. If the Financial Element has a column_property_cd that has a currency_basis_flg (in the ofsa_column_property_dsc table) of 0, then no translation is done on the value. If currency_basis_flg is 1, then translation occurs. ISO_CURRENCY_CD is whatever is determined by the debit/credit rule, such as <Same As Filter>, in all cases. An output row may have *002* from a constant in the Debit/Credit or from macros such as <Same As Filter>.

Allocations writing to a row in an Instrument or Transaction table translate all values written to a row to the existing ISO Currency Code for that row. As an example, allocations with USD results are translated to DEM if the instrument or transaction table row has an ISO Currency Code of DEM.

Process Matrix

The table below shows examples of inputs and outputs with the new allocation logic for multicurrencies.

Row	Table and Field Types	Starting Value	Input ISO Currency Code	Multiply by Constant	LEDGER_ STAT result	Instrument Balance Column Result	Output ISO Currency Code
1	LEDGER_STAT with FE 100, Ending Balance	100	USD	1	100		USD
2	LEDGER_STAT with FE 10000, Statistical	100	USD	1	100		USD
3	LEDGER_STAT with FE 100, Ending Balance	100	FRF	1	(100/7.4)*1=13.51		USD
4	LEDGER_STAT with FE 10000, Statistical	100	FRF	1	100		USD

Row	Table and Field Types	Starting Value	Input ISO Currency Code	Multiply by Constant	LEDGER_ STAT result	Instrument Balance Column Result	Output ISO Currency Code
5	LEDGER_STAT with FE 100, Ending Balance	100	USD	1	(100*1) / .135=740.74		FRF
6	LEDGER_STAT with FE 10000, Statistical	100	FRF	1	100		FRF
7	Instrument table CUR_PAR_BAL field	100	USD	1		100	USD
8	Instrument table CUR_NET_ RATE field	7.5	USD	1		7.5	USD
9	Instrument table CUR_PAR_BAL field	100	FRF	1		100	FRF
10	Instrument table CUR_NET_ RATE field	7.5	FRF	1		7.5	FRF
11	Instrument table CUR_PAR_BAL field	100	FRF	1		(100 / 7.4)*1=1351	USD
12	Instrument table CUR_NET_ RATE field	7.5	FRF	1		7.5	USD

The above examples assume that the functional currency is USD and the exchange rate is 0.135 FRF to 1.0 USD.

Rows 1 through 6 are LEDGER_STAT records, while rows 7 through 12 are detail, Instrument or Transaction, records. Where USD records are involved in LEDGER_STAT processing, rows 1 and 2, no translation occurs. Rows 3 through 6 show foreign currency records where the balance rows are translated, rows 3 and 5, but the statistical rows are not translated, rows 4 and 6.

Rows 7 through 12 are detail records. Because the detail records read input and produce output on the same record for rows 7 through 10, no translation occurs. Row 11 shows an example of instrument record translation where the input currency differs from the output. Row 12 shows how statistical records are not translated.

Filter On

In OFSA Release 4.5, ISO Currency Code is added to the Edit Filter On dialog that asks for leaf values. ISO Currency Code works similarly to other leaves in OFSA. A list of ISO Currency Code values is available for selecting any active currency code in the system. A Macro exists in the ISO Currency Code dialog for functional currency. In addition, right clicking on the Leaf Number or Description of the Currency leaf in the edit box enables the use of macros such as <All>.

Allocation Methods

As a general principle, all allocation rule calculations are performed in the functional currency. Values from a row in a table are brought into memory and converted first to functional currency. Then the rule is applied. When the row is written to the target table, the Currency of the target row is queried to determine the conversion factor. Then the row is written out in the proper currency.

Table ID

The assumption is that all data entered in Table IDs is in functional currency. This works well because all values are converted to functional currency prior to doing * or / operations on the Table ID. For other methods of table use, see Chapter 7, "Lookup Table ID". The user can use a Lookup Table ID and do look-ups on the ISO Currency Code.

PCT Distribution

All inputs to allocation calculations are translated first. Then the calculation is performed. For percent distribution calculations, all input data is read and translated to the functional currency. Then the distribution calculation is performed. For example, if the functional currency is USD and other records used in the percent distribution have DEM and FRF currencies, all records are read and translated to USD. Then the percent distribution algorithm is calculated. Translating all currencies to functional gives the percent distribution a consistent basis.

PCT Distribution Leaf

Percent distribution by leaf works the same as the percent distribution calculations above. All inputs to allocation calculations are translated first. Then the calculation is performed. For percent distribution by leaf calculations, all input data is read and translated to the functional currency. Then the distribution by leaf calculation is performed. For example, if the functional currency is USD and other records used in the percent distribution by leaf have DEM and FRF currencies, all records are read and translated to USD. Then the percent distribution by leaf algorithm is calculated. Translating all currencies to functional gives the percent distribution by leaf a consistent basis.

Constant

The assumption is that all data entered as a constant value is in the functional currency. This is particularly important for plug allocations and allocations that add or subtract a constant value.

By Leaf

By leaf aligns two LEDGER_STAT records together for calculations by matching all leaves between the rows except for the leaf listed in the *by leaf* dialog. Statistics can have ISO Currency Codes in different currencies. Statistics with multiple ISO Currency Codes enable the by leaf functionality in the allocation ID to function properly by keeping the ISO Currency Code in balance and statistical rows matched.

Field

Field works on Instrument and Transaction tables. Instrument and Transaction table allocations determine if currency translation is necessary on a column-by-column basis. Whether a target column needs currency translation is determined by the value of the OFSA Data Type Cd defined for that column on that table. OFSA Data Type Cd = 1, balance, is translated. All other OFSA Data Type Cd values are not translated.

Macro

The implementation of multicurrency has no effect on macros.

Debit/Credit

In OFSA Release 4.5, ISO Currency Code is added to the Debit / Credit dialogs that ask for leaf values. ISO Currency Code works similarly to other leaves in OFSA. A list of ISO Currency Code values is available for selecting any active currency code in the system. A Macro exists in the ISO Currency Code dialog for functional currency. In addition, right clicking on the Leaf Number or Description of the Currency leaf in the edit box enables the use of macros for <All>, <Same as Filter>, <Same As %Dist>, and <Same As %Dist-Leaf>.

Special attention is required when writing results to statistical Financial Element rows in LEDGER_STAT. The allocation engine reads in all currencies and converts them to function currency before processing calculations. If the calculated result is written to a statistical row, no conversion on the result occurs before output.

As an example, the database has a functional currency of USD. An allocation is attempting to create ratios from DEM balances as a product with statistical row data in LEDGER_STAT. The allocation calculates the result by converting the DEM balance to USD and then producing the ratio. The output of the calculation, however, has no conversion because it is written to a statistical Financial Element row. This may not be the intent of the above allocation rule. If the results of the ratio are intended to reflect the same currency as the input, then a balance Financial Element definition is required.

Upgrading from Previous OFSA Releases

For OFSA 4.5 Release 2 for Performance Analyzer, several processing changes were made to both the engine and internal storage tables. When the multicurrency option is turned on in OFSA 4.5 Release 2, allocations created in previous releases of OFSA will not run. After upgrading to OFSA 4.5 Release 2, each Allocation ID will display an error message indicating that users need to select currency codes before the allocation will run properly. Users must open each allocation, select currency codes, and save changes before the allocation ID will run in OFSA 4.5 Release 2.

Importing and exporting of Allocation IDs into OFSA 4.5 Release 2 from previous releases is not supported.

4

Overview of IDs

An Oracle Financial Services Application identification (OFSA ID) is a type of dialog box that supports the set up of assumptions information, reporting specifications, or processing specifications. With OFSA IDs, you have the flexibility to define as many types of assumptions, report specifications, or processing specifications as needed.

You can use an ID immediately after creating it and then discard it without saving it. You can also save it and then modify and resave it under another name for use in other OFS applications.

This chapter presents the following topics:

- Creating an ID
- Opening an Existing ID
- Closing an ID
- Saving an ID
- Deleting an ID
- Leaf Setup
- Tree-Related IDs
- Importing and Exporting Data
- OFSA ID Dependencies
- Processing an ID
- Server Status
- Spreadsheet Control Bar

Creating an ID

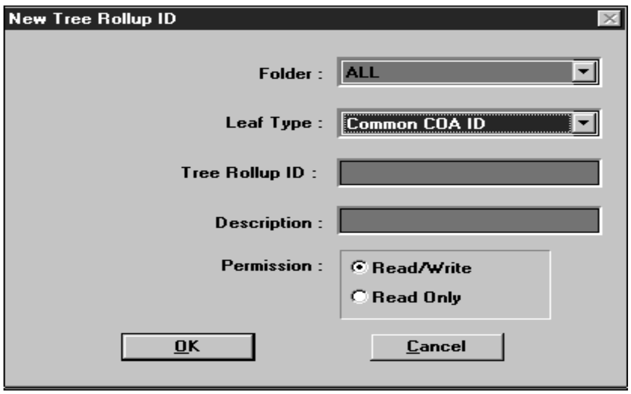
To create an ID, perform the following steps:

1. From the File menu, select New.

   A list of the IDs available for the product you are using appears.

2. From the list, select the ID you want to create and define.

   The New ID dialog box appears.



3. Select or type in each field, as required, using the following descriptions as a reference:

Field Name	Field Description
Folder	The folder default is set in Configuration ID. Until you set a new default in Configuration ID, the default option is <INDIVIDUAL>. This means that the user who logged on is the only user who can access this ID. You can make the ID available to a folder of users. If the user who logged on is a member of any folders, you can select the Folder options dialog box to view a list of those folders. Select a folder for this ID to make it available to all members of the folder. Select *<ALL>* to make it available to everyone.
Name of ID	Type a name for the new ID. When naming IDs, use alphanumeric characters only. Use an underscore (_) rather than a space.
Description	Type up to 60 characters describing the purpose of the ID. The description appears in the ID dialog box and is useful if others access the ID. This is an optional field.
Permission	
▪ Read / Write	Read / Write is the default.The individual ID creator always has Read / Write capability.
▪ Read Only	Read Only becomes an available option only when a folder has been selected as the folder logon option.

Note: Some IDs require additional information, such as leaf type or reporting currency. Select or type this information, as required.

---

4. Click OK to continue, or click Cancel at any time to cancel the operation and exit the dialog box.

Opening an Existing ID

To open an existing ID, perform the following steps:

1. From the File menu, select Open.

   A list of IDs available for the product you are using appears.

2. From the list, select the type of ID you want to open.

   The Select ID dialog box appears.

3. Select or type in each field, as required, using the following descriptions:

Field Name	Field Description
Folder	Click the arrow to view a list of folders. Select the name of the folder under which this predefined ID was created. The folder you select determines which ID choices are listed.
Additional criteria information	Before opening some types of IDs, you must provide additional information, such as a leaf type or a Tree Rollup ID. This data determines the list of previously defined IDs that appear for your selection.
Variable ID	Use the list to select the ID you want to open. (If you know the name of the ID, you can type the first letter of the ID to reach that portion of the alphabetized list, scroll down and select it.)

4. Click OK to continue, or click Cancel at any time to cancel the operation and exit the dialog box.

Closing an ID

From the File menu, select Close to close an open ID.

Saving an ID

From the File menu, select Save to save your ID.

Save As

You can copy IDs by saving them under a new name. Perform the following steps to copy an ID:

1. Select the ID that you want to copy.
2. From the File menu, select Save As....

   The New ID dialog box appears.
3. Type a folder name and description for the ID and any other data required, and click OK.

   The ID is copied and saved under the new name. The original ID remains unaltered.

Rename

To rename an ID, perform the following steps:

1. Open the ID that you want to rename.
2. From the File menu, select Rename.

   The Rename ID dialog box appears.
3. Type a new name and description for the ID and any other data required, and click OK.

   The ID is saved under the new name.

You can change the folder using this function, but only if the ID is actually renamed. This function does not perform a check for duplicate ID names in the folder.

You can also use Rename to change the Read / Write status for an ID.

Deleting an ID

To delete an ID, perform the following steps:

1. Open the ID that you want to delete.
2. From the File menu, select Delete.

   A Delete Confirmation dialog box appears.
3. Click Yes to confirm your decision, or click No to cancel the operation.

Delete a Group of IDs

To delete a group of IDs, perform the following steps:

1. From the File menu, select Delete Group.

   The Delete Group dialog box appears.

2. From the ID Type list, select the Type of ID.

   All IDs of that type appear in the large dialog box below it.

3. Select the IDs that you want to delete, and select Add to display them in the selected IDs list.

   You can continue to select different ID types and add them to the selected IDs list for deletion.

   Caution: Do not delete the active Configuration ID.

   When you create an ID, it has a set of assumptions. These assumptions are based on a specific modeling horizon, which is defined in the active Configuration ID.

   If the assumptions are based on a Configuration ID that does not exist, the data produced is inaccurate.

   ---

   a. To select multiple IDs of any type, hold down the Shift key while you click on them.
   b. To select all IDs of a chosen type, click Select All.
   c. To remove all IDs from the selected IDs list, click Remove All.
   d. Use the arrow buttons to change the position of an ID in the selected IDs list.
   e. Select the ID you want to move, and use the up or down arrow to move it up or down one position. IDs are deleted in the order that they appear in the list.

4. When you have selected all IDs for deletion, select Run from the Process menu.

   The Confirm Group Delete dialog box appears.

5. Click Yes to delete the IDs, or click No to cancel the operation.

Note: You cannot import or export IDs while Group Delete is open.

Leaf Setup

With Leaf Setup you can view, edit, insert (or, in other words, create), renumber, or delete any leaf value from any leaf type in the database.

Leaf type and *leaf column* are synonymous. A leaf type is any column registered in the database as a leaf column. For example, GL Account ID, Org Unit ID, Common COA ID, and Financial Element ID are all leaf types.

Leaf values compose the next highest level of categorization above detail account level. This level is known as the leaf level and provides the foundation for the structure of a Tree Rollup ID.

Leaf capacity is 200,000 leaves. This increased capacity enables institutions to model increasingly complex profitability scenarios.

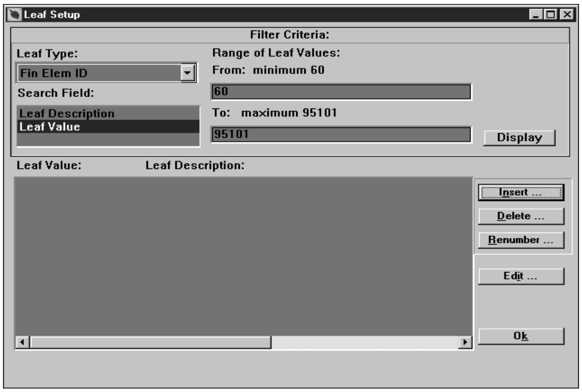
Note: You can display as many as 16,000 leaves at a time.

Creating a Leaf Setup Dialog Box

To create a Leaf Setup dialog box, perform the following steps:

1. From the Setup menu, select Leaves.

   The Leaf Setup dialog box appears.



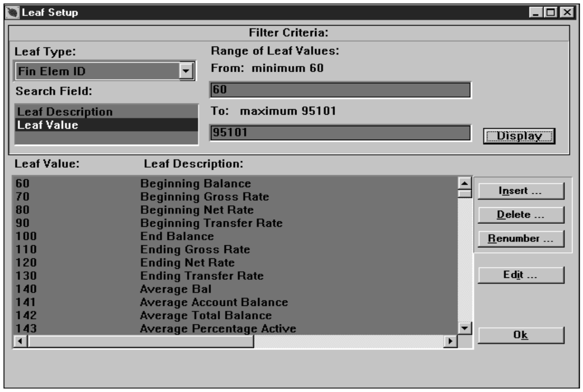
2. To view, edit, renumber, delete, or add any leaf value, select the leaf type that you want from the Leaf Type list.
3. Search for the leaf value that you want.

   You can search for a leaf value either by using Leaf Value or Leaf Description. Leaf Value is the default. Because you can display a maximum of 16,000 leaves at a time, Leaf Value is recommended when you do not know the exact leaf value or description.

   a. If you select Leaf Value, type the From and To Leaf Values in the Range of Leaf Values and then select Display.

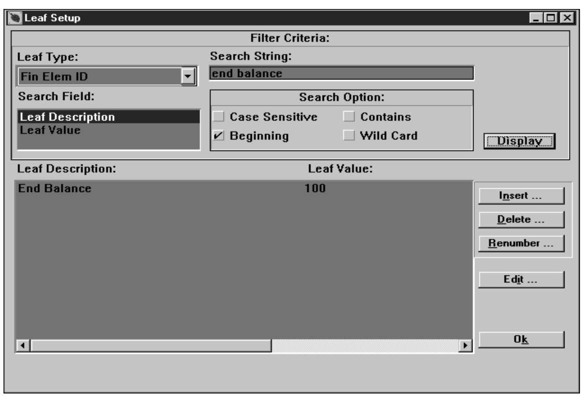
   Note: To search for an exact match, type the value in both the From and To fields.

   The Leaf Value and Leaf Description lists for the range that you specified appear.



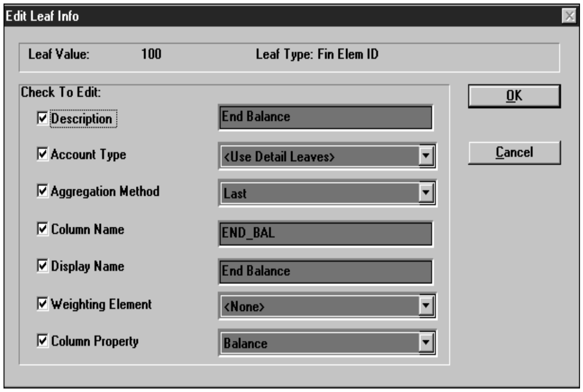
b. If you select Leaf Description, type the description in the Search String field and select a Search Option, if appropriate.

The Leaf Value and Leaf Description lists for the description that you specified appear.



4. To edit a Leaf Value or Leaf Description, select the Leaf Value that you want to edit and then click the Edit button.

The Edit Leaf Info dialog box appears.



Field Name	Field Description
Description	Leaf description, such as Average Net Rate
Account Type	Such as Dividends or Equity
Aggregation Method	Such as Average by Days Weighted or Beginning
Column Name	Physical column name, such as AVERAGE_NET_RATE
Display Name	Can be same or more detailed name as Description or Column name
Weighting Element	Balance used to weight the financial element. For example, Average Transfer Rate has a Weighting Element Average Balance.
Column Property	Values (Balance, Balance Weighted Object, Standard Rate, or Statistic) that determine whether the element requires currency conversion in processing or rate aggregation for reporting

Note: All non-common Chart of Accounts (COA) leaves need to be tied to the detail leaf. The detail leaf determines the account type and accrual basis.

5. To edit leaf information, select the appropriate Check to Edit options, make your changes, and click OK.
6. To renumber a leaf value, select the leaf value that you want to renumber and then click the Renumber button.

The Renumber Leaf Value dialog box appears.

7. Type the new number in the New Leaf Value field and then click OK.
8. To delete a leaf value, select the leaf value that you want to delete and then click the Delete button.

   The Confirm Leaf Delete dialog box appears.
9. Click Yes to delete the leaf value, or click No to cancel the operation.
10. To add a new leaf value, select the leaf type to which you want to add and then click the Insert button.

    The Enter New Values dialog box appears.
11. Type in the Leaf Value and Description and, depending on the leaf type, any other information about the new leaf.
12. Click OK to add the leaf value, or click No to cancel the operation.

Aggregation Methods for Financial Elements

Aggregation methods are applied to the summary financial information calculated at each event in order to generate financial element data for each modeling period.

The five different aggregation methods are:

- Average
- Accrual
- Sum
- First
- Last

Following are descriptions of the five aggregation methods.

Average Method

The average method calculates an average value (for example, Average Balance, Average Net Rate) over a modeling period. The calculation sums up the daily values and divides by the number of days in the modeling period.

Daily Average Balance = ∑(Daily Balance)/days in modeling period

All simulated events (originations, payments, prepayments, and repricings) are assumed to occur at the end of the event date. This implies that the balance and rate on the day of an event is counted as the value prior to any changes made by the event. Changes made influence the value of the next day.

Accrual Method

The accrual method determines how much accrual has occurred over the modeling period. The accrual method is determined by the code value in the detail record. Interest-in-advance instruments calculate interest accruals from the current payment date to the next payment date. Interest-in-arrears instruments calculate interest accruals from the current payment date to the previous payment date.

Daily Interest Accrual = Interest Cash Flow / number of days in payment

The example below demonstrates an interest accrual for an arrears record:

Example:

Payment Date	Interest Cash Flow	Days in Payment	Daily Accrual
January 15	950	31	30.64
February 15	900	31	29.03
March 15	850	28	30.36

Modeling Start Date = January 1

Modeling Period End Date	Accrual Calculation	Interest Accrual
January 31	15 days @ 30.64 + 16 days @ 29.03	924.08
February 28	13 days @ 29.03 + 15 days @ 30.36	832.79

Sum

The sum method adds together all financial element values associated with events occurring during the modeling period.

Principal Runoff = ∑(Principal Runoff)

First

The first aggregation method determines the value from the first event within a modeling period (for example, Beginning Balance).

Last

The last aggregation method determines the value from the last event within a modeling period (for example, Ending Balance).

Selecting an Aggregation Method

To select an Aggregation Method, perform the following steps:

1. From the Setup menu, select Leaves.

   The Leaf Setup dialog box appears.

2. From the Leaf Type list, select the leaf type you want.
3. Search for the leaf value that you want.

   You can search for a leaf value either by using Leaf Value or Leaf Description.

   a. If you select Leaf Value, type the From and To Leaf Values in the Range of Leaf Values and then select Display.

      Note: To search for an exact match, type the value in both the From and To fields.

      The Leaf Value and Leaf Description lists for the range that you specified appear.

   b. If you select Leaf Description, type the description in the Search String. field and select a Search Option, if appropriate.

      The Leaf Value and Leaf Description lists for the description that you specified appear.

4. From the Leaf Value and Leaf Description lists, select a leaf value and then click the Edit button.

   The Edit Leaf Info dialog box appears.

5. From the Check to Edit list, select Aggregation Method if it is not already selected.
6. Select an Aggregation Method and then click OK.

Tree-Related IDs

Many OFSA IDs function in the context of leaf columns. An example is the definition of modeling assumptions (such as Transfer Pricing methods or prepayments) on the basis of a Product ID (or product segmentation.).

Examples of tree- or leaf-related IDs include the Tree Rollup ID, Tree Filter ID, Transfer Pricing ID, and Prepayment ID. Tree Rollup IDs, for example, enable the user to *drill-down* to an appropriate level before making an assumption. The user can drill-down to the level of detail needed for the particular purpose.

The OFS applications in which these and similar IDs function include Performance Analyzer, Risk Manager, Transfer Pricing, and Balance & Control.

Hints on Editing Tree-related IDs

When editing Tree Rollup ID and Tree Filter ID, you can use the Options and Tree menus to format the appearance of your ID. For example, you can change typeface or specify which branches are to be included in the rollup. You can use the Edit menu to search for a specific node within the tree structure. For more information about these menus, see Chapter 11, "Tree Rollup ID" and Chapter 12, "Tree Filter ID".

Importing and Exporting Data

You can use IDs for importing and exporting data in Performance Analyzer:

Note: You can load Ledger Stat records using the server side process only. For information about loading Ledger Stat records using the server side process, see the *Oracle Financial Services Installation and Configuration Guide*.

Note: You must configure the ODBC drivers before you can import and export IDs. For information about configuring the ODBC drivers, see the *Oracle Financial Services Installation and Configuration Guide*.

Importing and Exporting IDs

You can use import/export functionality of OFSA to:

- Transport IDs from one FDM database to another
- Maintain external backup copies of key IDs
- Load data into FDM from an external database (.DBF) source

OFSA provides two methods of importing and exporting IDs:

- Individual ID import/export
- Dependent ID import/export

Individual ID Import/Export

IDs that do not contain other embedded IDs, such as Leaf Characteristics ID and Transfer Pricing ID, are imported and exported using the Individual ID method. These IDs are exported into a dBase file with the.DBF extension and can be imported only from a dBase file with the .DBF extension.

Note: Table ID and Allocations ID are an exception to this rule because they use Individual ID import/export. They may also contain other embedded IDs. Dependencies are not exported along with Table IDs or Allocations IDs. You must export all dependencies separately.

Individual ID import/export is available for the following IDs:

- Table ID
- Allocation ID
- Leaf Characteristics ID
- Transfer Pricing ID
- Historical Rates ID
- Discount Rates ID
- Maturity Strategy ID
- Pricing Margin ID
- Forecast Rates ID
- Result Detail ID
- Transaction Strategy ID
- Forecast Balance ID
- Prepayment ID

The Individual ID method of import replaces an existing ID of that type in the import database. You cannot import an ID using the Individual ID method without replacing an existing ID.

Caution: If you do not want to lose any existing IDs, create a new ID of the type being imported or use Save As from the File menu to create a copy of the existing ID before import.

To import an ID, open the ID that you want to overwrite and select File and Import.

Dependent ID Import/Export

IDs that contain other embedded IDs, such as Correction Processing ID, are imported and exported using the Dependent ID method. This method ensures that any dependent IDs required for correct operation of the exported ID are included in the export process.

Dependent ID import/export is available for the following IDs:

- Data Filter ID
- Tree Rollup ID
- Tree Filter ID
- Formula ID
- Correction Rule ID
- Correction Processing ID
- Group Filter ID

Any ID that is exported using the Dependent ID method creates a dBase file with the .CAT extension and one or more dBase files with extensions listed in the table below. Although the extensions differ from the standard .DBF, these files all use the dBase format. Importing these IDs requires that all files created in the export process reside in the same directory. Dependent ID import/export uses the following file extensions:

ID or Data Type	File Extension
Catalog of IDs	.CAT
Filter ID	.FLT
Report Leaves	.LF
Level Description	.LEV
Node Description	.NOD
Report Columns	.RCL
Correction Rule ID	.COR
Error Assignments	.ASS
Correction Processing ID	.PRC
Tree Filter ID	.TFT
Tree Rollup ID	.ROL
Formula ID	.FOR
Group Filter ID	.FGL

Note: If a dependent ID import fails during processing, change the file name extension from .DBF to its original form and extension and reprocess.

If a tree rollup ID that contains leaves not in leaf setup is imported, you receive a warning. If you elect to continue, the ID is imported but the leaves are not added to the leaf setup. You can use the Synchronize Instruments function to synchronize the Rollup ID with the leaf setup.

Group Import/Export

Use the Group Import/Export dialog box to import or export multiple IDs of various types. To import or export a batch of IDs, perform the following steps:

1. From the File menu, select Group Import/Export.

   The Group Import/Export dialog box appears. The Group Import/Export dialog box resembles a spreadsheet, where each row represents one import or export operation.

2. Click Insert Rows without highlighting any rows to add a row to the top of the spreadsheet.

   If any rows are highlighted, selecting Insert Rows adds an equivalent number of rows immediately before the highlighted section.

3. Click Delete Rows to remove any highlighted rows from the spreadsheet.

   If no rows are highlighted, selecting Delete Rows removes the first row from the spreadsheet.

You can use the Group Import/Export columns as follows:

Column Name	Column Description
Done	As IDs are imported or exported, a red check mark appears in the dialog box indicating that the job is complete.
Action	Click Import or Export for each ID on your list.
File Type	The only file type that you can select for Import or Export is dBase.
ID Type	Use the list to select ID types available for import or export.
ID Name	Use the list to view previously selected ID types. This field is active when exporting an ID or when importing an ID with the Individual ID method. When importing with the Dependent ID method, <not required> appears in the dialog box.
PC File Name	Double click in this field to open Select Import File if you are importing an ID or Select Export File if you are exporting an ID.

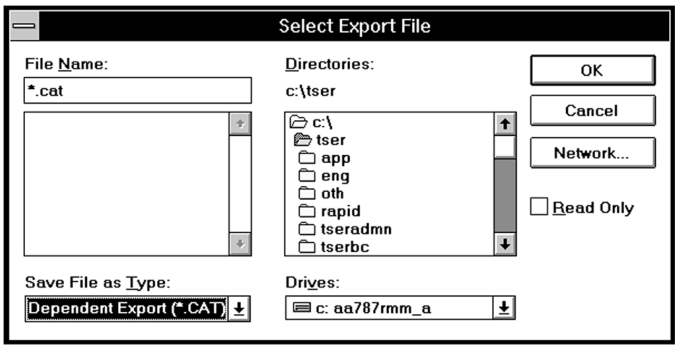
From the Process menu, select Run to run Group Import/Export.

OFSA processes the rows on your spreadsheet in sequential order, starting with the first row.

Export an ID with Group Import/Export

To export an ID using the Group Import/Export dialog box, perform the following steps:

1. Select Export in the Action list.
2. Select the ID Type and ID Name you want to export.
3. Double-click in the PC File Name cell to display the Select Export File dialog box.



4. Type a valid file name for the export and click OK.

   Individual ID exports must use .DBF as their file extension. Dependent ID exports must use .CAT as their file extension. If the file name that you have selected is not valid, you must select a new file name.

5. From the Process menu, select Run to run Group Import/Export.

   OFSA processes the rows on your spreadsheet in sequential order, starting with the first row.

   If the export file name that you have chosen already exists and the file is not read-only, you are asked if you want to overwrite the existing file.

6. Click Yes to replace the existing file with your export file.
7. Click No to cancel the export process and select a different name for your export file.

Export an ID from the File Menu

To export an ID using the Export option in the File menu, perform the following steps:

1. Open the ID that you want to export.
2. From the File menu, select Export.

   The Export ID dialog box appears.
3. Click Browse to bring up the same Select Export File dialog box used for exporting with Group Import/Export.
4. Type a valid file name in the export data File Name dialog box.
5. Click OK to export your ID.

Import an ID with Group Import/Export

To import an ID using the Group Import/Export dialog box, perform the following steps:

1. From the File menu, select Import/Export Group.

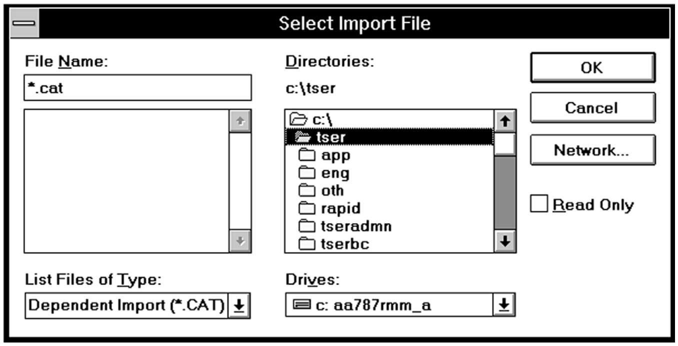
   The Group Import/Export dialog box appears.
2. Click Import in the Action dialog box.
3. Select the ID Type of the ID that you want to import.

   > **Caution:** Individual ID import overwrites the ID listed in ID Name with the contents of the imported ID. As a result, you lose any information contained in the ID being overwritten. If you do not want to lose any existing IDs, create a new ID of the type being imported or use Save As from the File menu to create a copy of the existing ID before import.

4. If the ID Type that you select relies on the Individual ID import method, select an existing ID to overwrite in the ID Name dialog box.

   If the ID Type you select relies on the Dependent ID import method, <not required> appears in the ID Name dialog box.
5. Double-click in the PC File Name dialog box.

   The Select Import File dialog box appears.



The Select Import File dialog box looks and functions almost the same as the Select Export File dialog box. The primary difference is in the file name validity check that is performed before you are returned to the Group Import/Export dialog box. A file name is valid only if it has the necessary file extension, .DBF for Individual ID imports and .CAT for Dependent ID imports, and if it is found in the designated directory. If a file does not have the necessary file extension, or if it is not found, you must select a new file name.

If you import an ID that uses the Dependent ID method, you are importing the original ID along with any embedded IDs used in its functionality. You import these IDs with the same <Name>/<Group> combination that they had in the original database.

6. From the Process menu, select Run to process all the rows on your Group Import/Export spreadsheet.

   If any of the <Name>/<Group> combinations for the IDs you are importing (or for any IDs embedded in your imported IDs) are already being used in your import database, the Import Warning dialog box appears. The Import Warning dialog box offers you three options of how to proceed when the ID you are importing or one of its dependent IDs already exists in your import database:

   - **Rename**

     To change the <Name>/<Group> combination of the ID that you are importing so it does not conflict with any IDs already present in the import database. After typing a new <Name>/<Group> combination for the ID that you are importing, click OK to continue with the import.

   - **Use Existing**

To use the existing ID in the database with the same <Name>/ <Group> combination as the ID that you are importing. You can use Use Existing when you need to import multiple IDs that share one dependency ID.

- **Cancel**

  To cancel the import process. Selecting Cancel before all the rows in the Group Import/Export dialog box have been processed generates the Import Warning dialog box.

  Click Yes to stop the Group Import/Export process. Click No to skip the current row on the Group Import/Export dialog box and continue processing the subsequent rows in sequential order.

  In either case, all IDs imported prior to your clicking Cancel remain in the import database.

Import an ID from the File Menu

To import an ID using the Import option in the File menu, perform the following steps:

1. Open the ID that you want to import.

   Caution: Individual ID import overwrites the ID listed in ID Name with the contents of the imported ID. As a result, you lose any information contained in the ID being overwritten. If you do not want to lose any existing IDs, create a new ID of the type being imported or use Save As from the File menu to create a copy of the existing ID before import.

2. From the File menu, select Import.

   The Import ID dialog box appears.

3. Click Browse to bring up the same Select Import File dialog box used for importing with Group Import/Export.
4. Type a valid file name in the import data File Name dialog box.
5. Click OK to import your ID.

   If your import uses the Individual ID method, you overwrite the open ID with the contents of the ID you are importing. As a result, you lose any information contained in the ID being overwritten.

6. If you encounter the Import ID dialog box, refer to the end of the previous section, "Export an ID with Group Import/Export", for instructions on how to proceed.

Rules, Limitations, and Hints

The Import/Export function is limited in scope. Keep in mind the following rules and limitations:

- If an ID is imported and the <Group> it was created in does not exist in the import database, the ID is placed in the default <Group> specified in the Configuration ID.
- Group Import/Export works only between databases that have identical structures. Different versions of the applications may use different database structures.
- Correction Processing IDs using the OFSA Cash Flow Edits cannot be imported.

When importing an ID of a given type, at least one ID of that type must already exist in the database or the import fails. No error message is displayed.

Note: Some .DBF files may require structural changes to import correctly into your FDM database. Before importing a non-OFSA.DBF into a particular database, export a similar type of ID to .DBF from that database. In your external database application, compare its structure to the non-OFSA file you want to import. Correct any structural differences in the non-OFSA file, and try the import it. If you experience problems, contact Oracle Support Services.

OFSA ID Dependencies

OFSA IDs often depend on other IDs for some or all of their data or usefulness. For example, a Transformation ID may depend on a particular Filter ID to select and focus data in a meaningful way. The Filter ID may have been created especially for that transformation and named in the transformation set up dialog box when the Transformation ID was created. If that Filter ID is deleted, then the transformation becomes unusable.

Caution: Consider OFSA ID Dependencies when you delete an ID. Deleting an ID may deprive a dependent ID of its meaning.

OFSA provides a Dependencies option in the File menu that enables you to view the dependencies factors of any active ID. This option shows both primary and secondary dependencies. For example, a Data Filter ID dependency may include a Processing ID as a primary dependency and a Batch ID as secondary dependency (depending on the Processing ID).

Also, when you click Delete ID or Delete Group of IDs, OFSA automatically checks for any dependencies, and enables you to review them before the deletion takes place. When you attempt to delete an ID, however, note that only primary dependencies are listed in the warning.

Note: Click Dependencies from the File menu if you plan to delete unfamiliar IDs.

Processing an ID

To process an ID, select Run from the Process menu. In most cases, the process proceeds immediately, without further prompting from the system.

Server Status

Most OFS applications can launch calculation-intensive software processes on a server. The server runs the process in the background, freeing the client PC to do other activities. Also, the server is often more powerful and can finish the process more quickly than the client PC.

When a server process is launched by an OFS application, the job request first is processed by RQ, the special OFS application that monitors OFSA server processing. RQ enters the job in a special table, and sends the job to the server. RQ then monitors the progress of the job on the server until it has completed, logging that information in its table.

When an OFS application launches a server job, the Server Status Update dialog box appears. Getting its data from RQ, the dialog box displays the progress of your server jobs or, optionally, displays all jobs on the server. Also, you can terminate any of your processes that are running on the server.

In Oracle Financial Data Manager (FDM) Administration, the capabilities of the Server Status Update dialog box have been expanded. The Server Status Update dialog box:

- enables you to terminate *any* job running on the server, not just those jobs you have launched yourself
- incorporates a test function to validate the ADMN - RQ - SERVER links
- provides a *housekeeping* capability, enabling you to delete completed jobs from the RQ table

Using the Server Status Update Dialog Box

To open the Server Status Update dialog box, select Server Status from the Process menu. The Server Status Update dialog box appears.

View your jobs or all

- See your jobs

  Select this option to view the status of your own server jobs.

- See all jobs

  Select this option to view the status of *all* jobs running on the server.

Stop Job

Select any of your own server jobs, and click this option to stop it. This function is not a Pause. Once stopped, the job cannot be continued. You must restart it from its source application. This option does not cancel jobs that have not begun execution.

Note: In this OFS application, the Server Status Update dialog box enables you to stop your own server jobs only. In Oracle Financial Data Manager (FDM) Administration, the dialog box enables you to cancel of any job running on the server.

Insert Test Request

This option starts a simple job to *ping* the server, testing network connections and protocols. Click Stop Job to halt the test, and click Remove YOUR Finished Jobs to delete the job from the display. If the test fails, the Cannot Launch Request error message appears.

Remove YOUR Finished Jobs

Select this option to delete all finished jobs from the display. The Request Clean Up verification box appears when the jobs have been removed. You click OK to close it.

Caution: Selecting **Remove YOUR Finished Jobs** does not halt processing of a server job in progress. The job is deleted from the display (and from the RQ table) but continues to run to completion on the server.

Server Polling Indicator

At the left side of the dialog box is a blue wheel that moves up and down the dialog box. The wheel indicates that OFSA is polling the database.

The Status Display

Through this interface you can monitor active server jobs.

Interface	Definition
Job Description	Name of the ID being processed
Table Name	Name of the table being processed against
Step / Page	Step or Page being processed
Host	Not currently implemented
Record	Number of records being processed
User	User's login name
Title	Description or status of the process
Status	Indicates whether of not the job has completed
Job Return Status	Indicates successful process or an error code. Possible return codes include:
	■ Making request (Indicates that the client has made a request to the server, but it has not yet been acted upon)
	■ No INI found (Indicates that, on startup, the server process could not find the server ini file. This file should be located in the same directory as RQ (usually /bin/rq under the OFSA install directory). Either the file does not exist (being deleted or moved after RQ started) or it is not readable (someone changed the modes on the file)
	■ None: Running (Job is currently running)
	■ Failed on Fork (Normally indicates that RQ was unable to execute the requested application. This can occur if the path in the server ini file is incorrect (for example, the OFSA software suite is not installed normally or is moved after it is installed). This message could also indicate that the permissions on the executable are incorrect.)
	■ None: Canceled (Indicates the process has been canceled, either from the client or from the server, with signal 1, 2, or 15 (SIGHUP, SIGINTR, or SIGTERM, respectively).
	■ Job Return Number (Job returned an unrecognized status)
	■ Normal (Indicates the job terminated normally)

Interface	Definition
	■ Bad Usage (Indicates that the subprocess failed to start because the parameters were not passed correctly. With the current releases of OFSA, this should not happen.)
	■ Session Failure (Indicates that the application was not able to create a session. Check the server log file for additional information.)
	■ No memory (Indicates that the server application ran out of memory)
	■ Internal Error (Indicates that some error occurred within the server application. This is normally a database error. Check the server log file for additional information.)
	■ Internal Error (Indicates that some error occurred within the server application. This is normally a database error. Check the server log file for additional information.
	■ Connect Failure (Indicates that the server process was unable to connect to the database. An incorrect user/password is probably the reason. Check the log file for more information. This can occur if the password is changed after starting up the application but before running a server process.)
	■ Rights Violation (Indicates that the specified user does not have the proper rights to run the program. This can occur if an administrator changes the rights for a user after the user started a client application but before the server application is launched.
	■ Signaled: <number> (Indicates that the process stopped running due to a signal. Check the log file for additional information. The number shown is the negative of the number that caused the process to halt. The most common number is -11.)
Request Date	Date and time when request is inserted into OFSA Request Queue.The date and time are from the client PC.
Start Date	Date when OFSA Request Queue launches process
Start Time	Time when OFSA Request Queue launches process
End Date	Date and time when process completes
Priority	Processing priority in order of importance or urgency

Changing a Password

To change your password, perform the following steps:

1. From the Options menu, select Change Password.

   The Change Password dialog box appears, with your login name (such as the name of the current Oracle Financial Data Manager (FDM) Administration user) at the top.

2. Type your current password.

   For security reasons, asterisks appear instead of the characters you type.

3. Type your new password, and then click OK.

   You are prompted to confirm the new password.

4. Retype the password exactly as you did the first time.

5. Click OK when you are done, or click Cancel at any time to close the Change Password dialog box and return to the main window.

> **Note:** OFSA passwords are case-sensitive. When you log on to the database via any OFS application, you must type your password exactly as you typed it here.

Spreadsheet Control Bar

The **Spreadsheet Control** Bar, available through the Stratification ID, automatically calculates and enters incremental ranges of data into your spreadsheets, such as for interest rates or amounts. The Spreadsheet Control Bar offers five incremental methods for automatically calculating ranges of data.

Increment Methods

Following are descriptions of the five Increment Methods and their uses:

Increment by Value

The increment is applied as a constant value amount (such as 1.00 or -1.50) that is cumulatively added to the value of the Start Value. Type an Increment Value and then click Run.

Increment by Percent

The increment is calculated as a cumulative percentage increase over the Start Value (with 100 percent specified by the value 100). Type an Increment Value and then click Run.

Add to by Value

A constant amount is added to the current value of each of the values in the ID. (For example a value of 1.5 adds 1.500 percent to each value.) This increment does not accumulate. Type an Increment Value and then click Run.

Add to by Percent

Each value is increased by a constant percentage over its current value (with a 100 percent increase specified by the value 100). This increment does not accumulate. Type an Increment Value and then click Run.

Interpolate

Interpolate inserts into the **From/To** columns an evenly-divided series of values calculated between a Start and End value, leaving the selected first and last row values intact.

Interpolation in Auto Increment

Interpolation enables you to create interpolated buckets based on a start- and end-value. This method does not update the end value specified in the Auto Increment dialog box, however. You must manually enter that value.

Increment a Range of Values

To automatically increment a range of values, perform the following steps:

1. From the horizontal tool bar, click the Spreadsheet Control Bar icon to open the Control Bar.

   Note: You can move the dialog box to another, perhaps more convenient, location on your window. Place your mouse pointer in the Control Bar's title bar, and hold the left mouse button while you drag the dialog box to a new location.

2. Select a range of values in the spreadsheet.

   In the Control Bar, the value from the first row in your selected column is transferred to the Control Bar's Start Value dialog box:

   The Method list is at the top of the Control Bar. It displays Inc by Value when it becomes active. From the Method list, you can open a list of five increment Methods.

3. Select the increment method that you want to use and then click Run from the Process menu.

5

Data Verification ID

When you need to edit or verify the data in a database, you often need to view that data at the lowest level possible: the row and column level. The Data Verification ID enables you to define the specific table, columns, and rows you want to view.

The results of the Data Verification ID appear in the form of a virtual spreadsheet that displays a maximum of 32,000 records. Because databases may be much larger than this, you should use predefined Data Filter IDs to narrow your focus on the database. Refer to Chapter 9, "Data Filter ID" for instructions on how to create a Data Filter ID.

This chapter discusses the following topics:

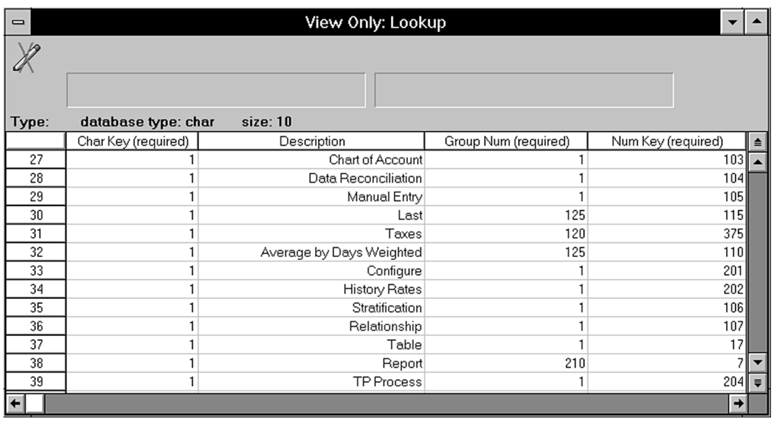
Creating and Defining a Data Verification ID

Use the following instructions to create and define a Data Filter ID. See Chapter 4, "Overview of IDs" for further explanation about ID creation and maintenance.

To create a Data Verification ID, perform the following steps:

1. From the File menu, select New and Data Verification ID to display the New Data Verification ID dialog box.
2. Type name for the ID in the Data Verification ID field.
3. Type a description for the ID in the Description field.
4. Select the permissions for the ID.
5. Click OK to continue or Cancel to exit.

   The Data Verification ID dialog box appears.


View Only: Lookup

Type: database type: char size: 10

	Char Key (required)	Description	Group Num (required)	Num Key (required)
27	1	Chart of Account	1	103
28	1	Data Reconciliation	1	104
29	1	Manual Entry	1	105
30	1	Last	125	115
31	1	Taxes	120	375
32	1	Average by Days Weighted	125	110
33	1	Configure	1	201
34	1	History Rates	1	202
35	1	Stratification	1	106
36	1	Relationship	1	107
37	1	Table	1	17
38	1	Report	210	7
39	1	TP Process	1	204

6. From the Table Types list, select the type of table with which you want to work.

   Your selection determines which tables are available in the Tables list.
7. From the Table list, select the table with which you want to work.

Note: The Data Verification ID does not support Risk Manager result detail tables.

8. From the Filter ID list, select the Data Filter ID or Group Filter ID for the subset of data you want to view.

   Note: The virtual spreadsheet displays up to 32,000 records. If the records you want to view are outside this range, you must refine your filter (Data Filter ID or Group Filter ID) to narrow the focus of your view.

   ---

9. Select up to three columns in the Sort By box to define the sort order you want.
10. In the Columns box, select the columns you want to include in your view and click Add.

    The columns move to the Defined View box.

    - To add all the columns to the Defined View box, click Select All and then click Add.
    - To delete a column from the Defined View, select the column and click Remove.
    - To remove all the columns from the Defined View, click Remove All.

    Note: If a column name contains a substring that includes the name of another column (such as, Remaining Term and Remaining Term Multiplier), then you must add the column with the shorter name to the defined view first.

    ---

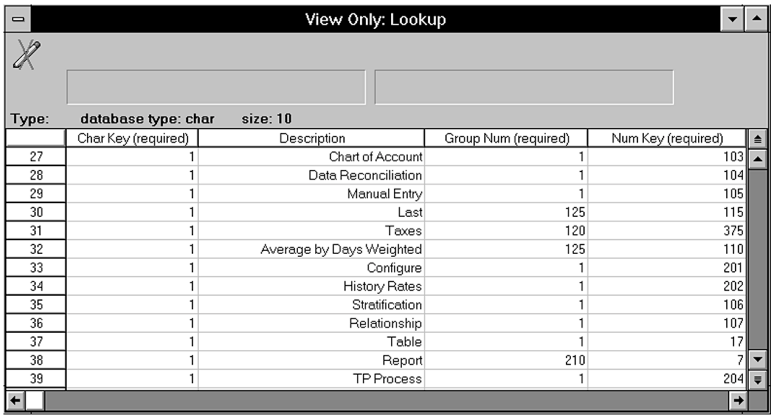
11. Use the arrows next to the Defined View box to organize the columns in the order you want to see them.

    Each click of an arrow moves the selected columns one position. Raising a column in the list moves it to the left in the spreadsheet.
12. Save the ID.

Running a Data Verification ID

To view the results of your Data Verification ID definitions, complete the following steps:

1. Select the Run option from the Process menu.
2. Review the results.



	Char Key (required)	Description	Group Num (required)	Num Key (required)
27	1	Chart of Account	1	103
28	1	Data Reconciliation	1	104
29	1	Manual Entry	1	105
30	1	Last	125	115
31	1	Taxes	120	375
32	1	Average by Days Weighted	125	110
33	1	Configure	1	201
34	1	History Rates	1	202
35	1	Stratification	1	106
36	1	Relationship	1	107
37	1	Table	1	17
38	1	Report	210	7
39	1	TP Process	1	204

Edit Spreadsheet On/Off

The Edit spreadsheet On / Off option is available only in Oracle Financial Data Manager Balance and Control. If this feature is on, you can edit the data at row level. The yellow pencil icon with the red X over it indicates that the Oracle Financial Services (OFS) application does not allow editing.

What the Results Show

In the View window, all values display as they are stored in the database. For example, the view displays code values numerically, not with the code value translation. This is essential for quality control of the data. If you have invalid codes, simply displaying an invalid code description is not sufficient. The actual code value is necessary for determining the source of the problem.

Editing a Data Verification ID

To edit a Data Verification ID, perform the following steps:

1. From the File menu, select Open and Data Verification ID.

   The Select Data Verification ID dialog box appears.

2. Select the Folder and Data Verification ID you want to edit.
3. Make your desired changes.
4. Save the ID.

Reviewing a Data Verification ID Example

This example creates a Data Verification ID to verify the margins on adjustable rate mortgages. To include only adjustable rate mortgages in the table view, you must first create a Data Filter ID (MTG_ARM) that meets these characteristics. See "Data Filter ID" in Chapter , "Reviewing a Data Filter ID Example" for instructions on creating the Data Filter ID for this example.

To create the Data Verification ID, perform the following steps:

1. Select New and Data Verification ID.
2. Select <ALL> for Folder.
3. Type ADJ_RT_MTG_MAR in the Data Verification ID field.
4. Type Verify Adj Rt Mtg Margins as the description.
5. Select Read / Write for Security.
6. Click OK.

   The Data Verification ID dialog box appears.

7. Select Client Data Tables Table type.
8. Select the Mortgages table.
9. Select Interest Rate Code, Product Type Code, Margin, and Current Gross Rate from the Columns box.
10. Click Add to include these columns in the Defined View box.
11. In the Filter box, select MTG_ARM.

12. Select Interest Rate Code from the first Sort By box to view the margins based on the interest rate code.
13. Run the Data Verification ID to view and verify the results.

6

Table ID

Table IDs are used in conjunction with Allocation IDs for two purposes:

- To distribute balances in the LEDGER_STAT table
- To perform a lookup table function against instrument tables

As a distribution tool, they function in a way that is similar to an Allocation ID percent distribution (see Chapter 8, "Allocation ID"). When you use a Table ID as a lookup table, you input specific leaf values and a specific amount associated with them. If a matching row is found, the allocation performs a math operation, updating a result column. You can also define and use Helper leaves in a Table ID to distribute balances in a way that is more readily understood within your organization.

This chapter presents the following topics:

Create a Table ID

To create a Table ID, complete the following steps:

1. From the File menu, select New and Table ID.

   The New Table ID dialog box appears.

2. Select a Folder from the list.
3. Type a descriptive name for the ID in the Table ID field.

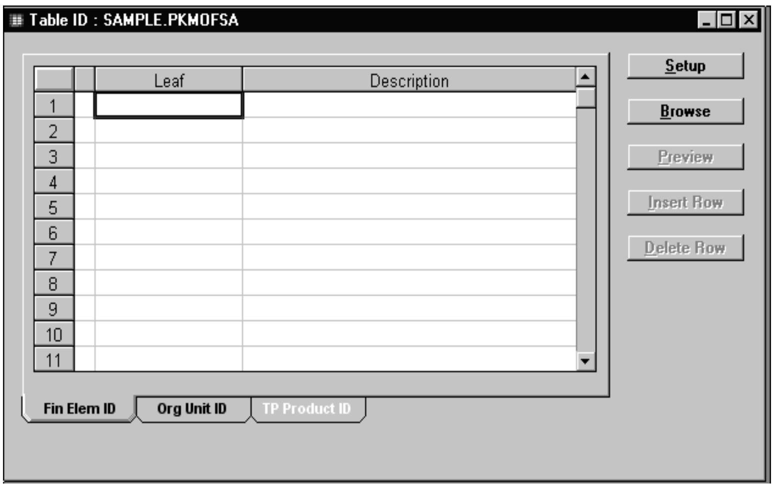
4. Type a description for the ID in the Description field.

   This is an optional field.

5. Select either the Read / Write option or the Read Only permission level.

6. Click OK.

   The Table ID dialog box appears with the name of the ID displayed at the top of the dialog box.

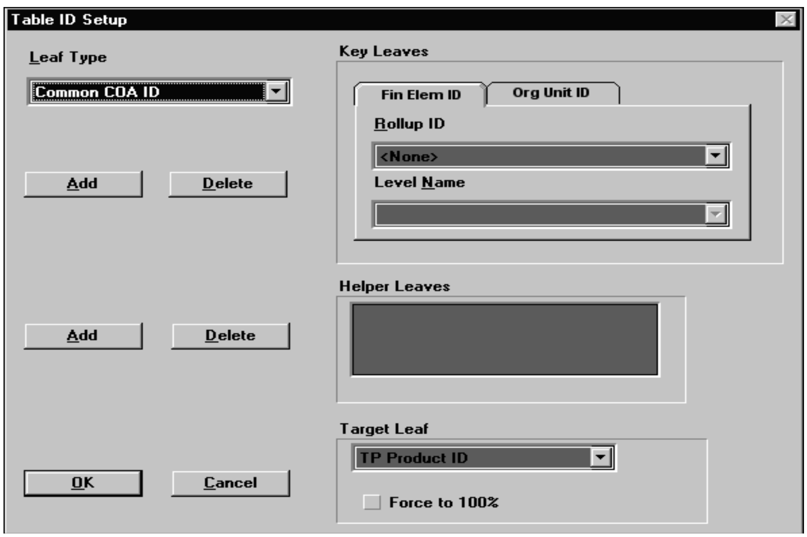


Define a Table ID

Setup

Caution: If you already entered values in the table and then make changes to the leaf types in the Table ID Setup dialog box, you will lose some data in the table.

You must first define the leaf types to be used in the allocation. Click the Setup button on the Table ID dialog box. The Table ID Setup dialog box appears.



Leaf Type

The Leaf Type list is used to add Leaves, or *tabs*, to your Table ID. Select the leaf type you want from the list.

Key Leaves

The Key Leaves pane displays the leaf types that are included in the Table ID. Key leaf tabs are used in the lookup process. Rows that match the Filter On set of the Allocation ID are joined with the Table ID. If an input row leaf value matches a leaf value specified in the first tab of the Table ID, the row is checked against the second tab leaf values, and so on. If the row matches across all key leaf tabs, the row is processed by the Allocation ID. If an input row does not match across all tabs in the Table ID, it is not updated by the Allocation ID. This is useful in performing selective updates in a mixed data input set.

Click Add to add the selected leaf type to the Key Leaves box. The leaf type selected is displayed as a folder and the Leaf Type name appears on the folder tab.

To remove a Leaf Type, click the tab in the Key leaves pane and then click Delete.

Rollup ID

Within Table IDs, you can specify actual leaf values or Tree Node in a predefined Rollup ID. To specify the actual leaf value, select <None> for a Rollup ID. To specify a Tree Node, select a predefined Tree Rollup ID.

Level Name

Select a rollup tree level from the Level Name list. This establishes which nodes are available on the leaf tab in the Table ID. A Table ID can have as many as three key leaf tabs in a Table ID.

Helper Leaves

Helper Leaves store additional detailed information that is useful in performing an allocation. The Helper Leaves pane shows selected Helper Leaves. A Helper leaf appears as one or more tabs in the Table ID just like a regular leaf except it has an amount column.

To add a Helper leaf, select the leaf from the Leaf Type list and click the Add button next to the Helper Leaves pane. To remove a Helper Leaf, highlight the leaf type in the Helper Leaves pane and click the Delete button next to the Helper Leaves pane.

See the *Oracle Financial Services Installation and Configuration Guide* for information on creating Helper Leaves.

Target Leaf

Target leaves are used only in allocations that distribute to the LEDGER_STAT table. Distribution allocations can be viewed as first performing a lookup on one or more leaf values (key leaves). When a match is found, the target leaf tab becomes active. The matching input row's amount is distributed to all of the account leaf values specified in the target leaf tab. This means that a new row is created in LEDGER_STAT for each row specified in the target leaf tab. The leaf values specified in the Allocation IDs Debit Item typically match (that is, use the <Same As Filter> macro) for all Key leaf types and use the <Same As Table> macro on the leaf type that matches the leaf type of the Target Leaf. Select the Target Leaf type from the list, or <None> if the Table ID does not use a Target Leaf.

Force to 100%

Click the Force to 100% button to calculate a percent to total for the amounts entered for the Target Leaf section.

If this button is not checked, Performance Analyzer uses the exact amounts entered as the factors for the allocation. This does not ensure a 100 percent allocation of the balances from the input rows.

Click OK to save the setup specifications and return to the main Table ID dialog box. Click Cancel if you do not want to save the changes made in the Table ID Setup dialog box.

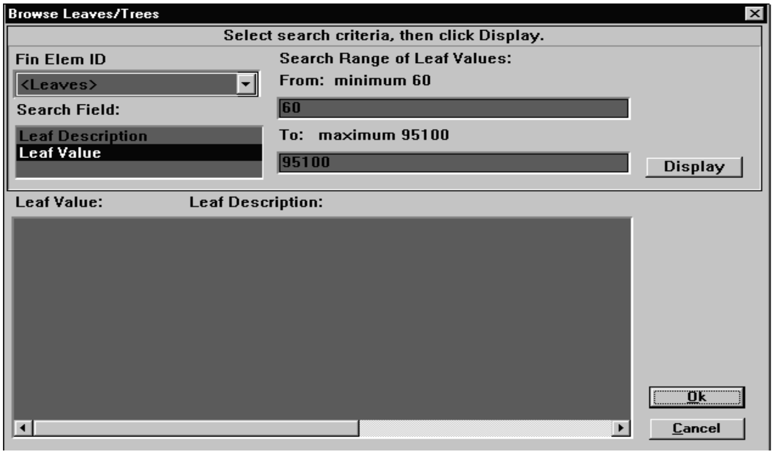
Entering Leaf Values

Each leaf type defined in the Table ID Setup dialog box is represented in the Table ID as a folder tab. If the Table ID is a lookup table, the first tabs are the lookup tabs and the last tab provides an additional lookup and contains the amounts to use in the allocation rule. If the table has a target leaf, the first few tabs are used for lookup and the last tab provides the Target Leaf values as well as an amount to use for distribution. When Helper Leaves are included, an Amount column is also added in addition to the Amount column that is on the Target Leaf tab.

You can type a leaf or tree node value directly into a row of a tab. When you click on another cell, a description appears in the Description column.

Browse

You can also browse from a list of leaf values that are appropriate for the tab. Click a row and then click the Browse button to open the Browse Leaves / Trees dialog box.



Use the Browse Leaves / Trees dialog box to locate the leaf values that you want to display in the table. If you did not set up the leaf tab to use a Rollup ID in the Table ID Setup dialog box (see Define a Table ID), you can enter only a single leaf value per row. You can locate leaves in the Browse dialog box searching either on the Leaf Description or the Leaf value. For value searches, adjust the search range of leaf values and click Display. The list of available leaves displays below. For Description string searches, use the percent symbol (%) as the wild card symbol. Click on the desired leaf value and click OK. The selected value is inserted in the Leaf column for you.

If your leaf tab uses a Rollup ID, you can insert only a Rollup node on a row. To search for a Rollup node, click on a row and then click the Browse button to open the Browse Leaves / Trees dialog box. You can search for a node using rollup description or node. Type your criteria and click Display. Click the desired node value and then click OK. The selected value is inserted in the Leaf column for you.

If you have a one tab lookup table, you add leaf values and associated amounts on the tab. If you have a multilevel Table ID and you have chosen all leaf or tree node values for the first leaf type or tier, you add second level lookup values. Rows that match a leaf value on tier one can match one or more values on tier two. Click a leaf value on the first tier and then click the second tier tab to add secondary level lookup leaf values. The + or **-** sign in the left column indicates whether leaf values have been defined for the next lower level leaf tier. A **-** sign means that no values were assigned and a + sign means that one or more leaf values were assigned.

The amount column is on the lowest level tab (and on Helper Leaf tabs). For lookup Table IDs, the amount column contains the lookup amount that is used in the allocation rule.   For distributions, this column holds the amount to be distributed to target ledger rows when a lookup match is made. As noted above, the distribution is generally made as a percent to total, depending on whether the Force to 100% button is checked or not.

For each leaf value, enter the percentage to be allocated to that leaf. You can enter either a straight percentage or values from which a percent to the total can be calculated.

For example, if you want to allocate 15% of the total amount to Chart of Account Account 141, 25% to 142, and so on, you can enter the percentages directly.

		Leaf	Description	Amount
1	-	141	Mortgage Fixed	0.150000
2	-	142	Mortgage Adjustable	0.250000
3	-	145	Mortgage Loan Provision	0.100000
4	-	311	Mortgage Fixed Income	0.300000
5	-	312	Mortgage Adjustable Inco	0.200000
6				
7			Total	1.000000

Or, if you want to allocate the total amount based on the actual volumes in the accounts, you can enter those volumes.

		Leaf	Description	Amount
1	-	141	Mortgage Fixed	37.500000
2	-	142	Mortgage Adjustable	62.500000
3	-	145	Mortgage Loan Provision	25.000000
4	-	311	Mortgage Fixed Income	75.000000
5	-	312	Mortgage Adjustable Inco	50.000000
6				
7			Total	250.000000

Oracle Performance Analyzer then calculates the percent to total for each leaf value, which in this case is 37.5/250=15% for 141, and so on. The results of these two tables are the same.

Note: To calculate a percent to total, click the Force to 100% button in the Table ID Setup dialog box. Otherwise, Performance Analyzer allocates 250 times the total amount of the input.

Table IDs can also be used as *Lookup Tables*. For example, you want to assign a rate but it varies by Org Unit ID and Common COA ID. In the Table ID Setup dialog box, specify Org Unit ID and Common COA ID in the Key Leaves pane and specify *None* in the Target Leaf pane.

This setup enables you to enter a rate for every unique combination of Org Unit ID and Common COA ID. When this Table ID is used in an Allocation ID, the specific rate is used.

Preview

Use Preview in the Table ID dialog box to see the total percentage that will be allocated to the leaf values assigned in the Table ID. Click the tab corresponding to the leaf type from which the percentages should be calculated. Then click the Preview button. The Compute Table Factors dialog box appears. In the Preview Level box, select the leaf level for which to calculate the percentages. The leaf values and corresponding factors appear in the bottom box.

Insert Row/Delete Row

Use Insert Row or Delete Row in the Table ID dialog box to insert or delete a row of leaf values. To insert a row between rows, select the row below which you want to insert and click Insert Row. To delete a row, select the row and click Delete Row.

Helper Leaves

When setting up a Table, you can select from all of the leaf types defined in the database as fields as well as helper leaves. Unlike the other leaf types, helper leaves are not represented as columns in the data table. These leaves are used only in the Table ID to help distribute balances to lower levels of detail. Helper Leaves can be selected only in the Helper Leaves pane of the Table ID Setup dialog box. Because no data tables refer to these leaf types, it is not possible to distribute costs either from or to these leaves. Enter values for the helper leaves as described in "Entering Leaf Values".

For example, you may want to break the total salary expense down further into the expense for each employee. You can do this using a user-defined Element leaf. Also, you can distribute the salary expense further by specifying various tasks performed per employee. For this you would use the Function leaf, since it relates to a task or function rather than a financial element. Define the leaf values and descriptions in the Leaf Setup dialog box.

Table ID Example: Using Helper Leaves

Note: Helper Leaves are introduced on an individual discretionary basis during the implementation phase at any given institution.

You can allocate total salary expense in more detailed fashion than in the Table ID example. In the Mortgage Loan center, there are three different job positions: Loan Officer, Credit Analyst, and Administrative Assistant. The total salary expense for these three employees is held in the Officer Salaries expense account. The percent distribution of the total salary expense is:

Employee	Percent Distribution
Loan Officer	45%
Credit Analyst	30%
Administrative Assistant	25%

Each employee allocates a different proportion of his or her time to the two products. Therefore, each employee's percentage of the total salary expense should be allocated to the products uniquely. Assume each employee allocates time to the products as follows:

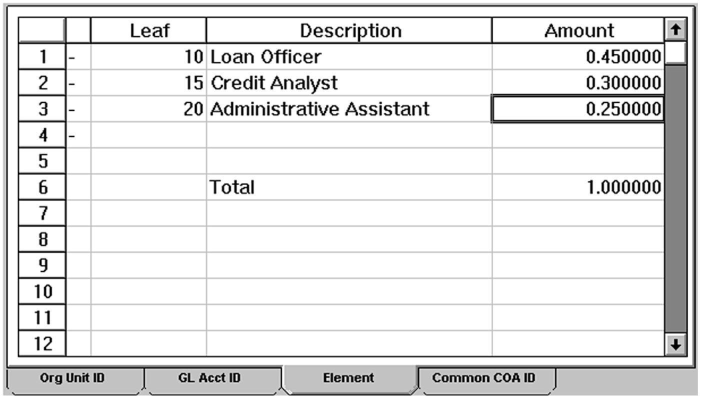
	Mortgage Fixed	Mortgage Variable
Loan Officer	35%	65%
Credit Analyst	50%	50%
Admin. Asst.	40%	60%

A Helper Leaf can be used to hold this percent distribution for the different job positions. In this example, the jobs were defined in the Element leaf.

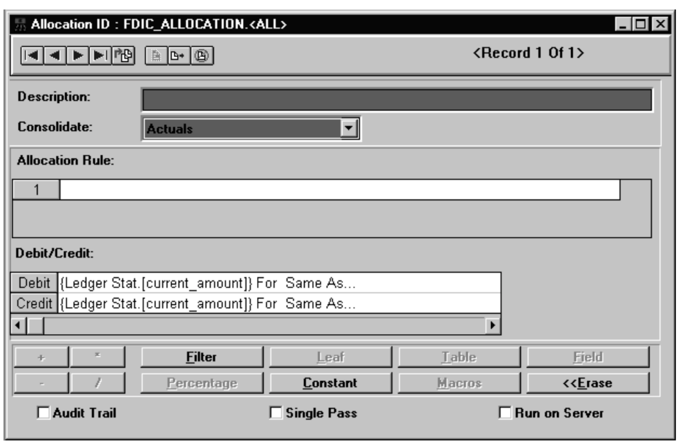
To allocate total salary expense, perform the following steps:

1. In the Table ID Setup dialog box, add the Element leaf type to the Helper Leaves box.
2. Type the Org Unit and GL Account leaf values.
3. Click OK to return to the Table ID dialog box.

4. Type the employee positions used to distribute the salary expense. Select the row for GL Account 800201 and select the Element tab. On the Element dialog box, type the leaf value and the percent of the total salary expense associated with each job.



5. For each job position, click the Common COA ID tab and type the percentage of each job's salary that should be allocated to each of the products.



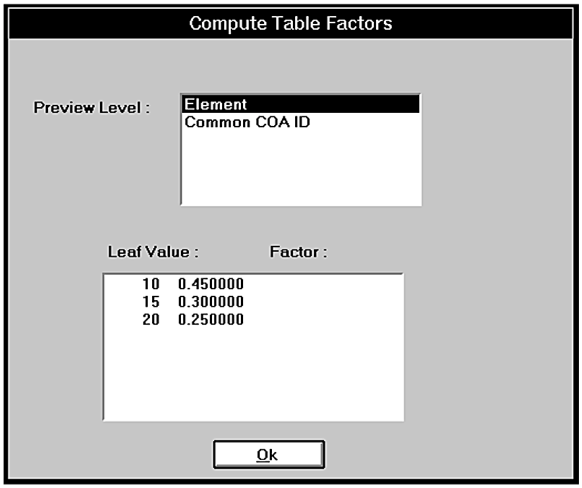
6. Repeat these steps for the Credit Analyst and the Administrative Assistant.

   Once the table is complete, you can use Preview to determine the total percentage that will be allocated to each product.

7. Click the GL Account ID tab to open the GL Account dialog box.

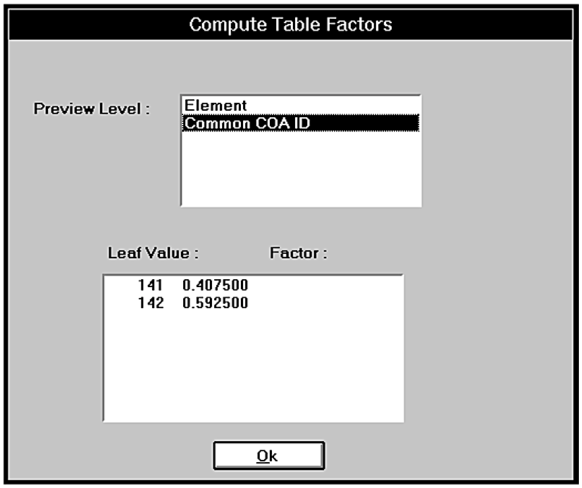
8. Highlight the Officer Salary row and click the Preview button to the right of the table.

The Complete Table Factors dialog box appears.



The Complete Table Factors dialog box indicates the percentage of the total salary expense that will be allocated to each Element.

9. Select the Common COA ID line in the Preview Level box to see the percentage of the total salary expense that will be allocated to each product.



The Factors represent the total percentage of the salary expense allocated to the products, for all job positions.

Table ID Example: Table ID

You can allocate the total expenses in the Mortgage Loans Cost Center to various products (held in the Common Chart of Account leaf), based on predetermined percentages. Assume the following:

The Mortgage Loans Center (1500) incurs three expenses each month.

GL Account		
800201	Officer Salaries	$10000
801560	401-K Plan Costs	$250
801570	Medical and Other Benefits	$800

Officer Salaries should be distributed to products based on the following percentages:

- 75% to Mortgage Fixed ($7500)
- 25% to Mortgage Adjustable ($2500)

401-K Plan Costs and Medical and Other Benefits should be distributed to products based on the following percentages:

	401-K	**Medical**
40% to Mortgage Fixed	$100	$320
60% to Mortgage Adjustable	$320	$480

To create a Table ID for this allocation, perform the following steps:

1. Define the structure for the Table ID in the Table ID Setup dialog box.

   The expenses are captured in unique Org Unit/GL Account combinations and are distributed to products using the Common COA leaf type.

2. Click OK to return to the Table ID dialog box.
3. On the Org Unit dialog box, type the Mortgage Loans Cost Center:

		Leaf	Description
1	+	1,500	Mortgage Loans

4. Highlight Org Unit 1500 and click the GL Acct ID tab. Type the GL Accounts for the expenses to be allocated:

		Leaf	Description
1	-	800,201	Officer Salaries
2	-	801,560	401-K Plan Cost
3	-	801,570	Medical and Other Benefi

5. Highlight GL Account 800201 and select the Common COA ID tab.

6. Type the products to which the Officer Salaries will be allocated, as well as the percentages used to calculate the allocated amounts.

		Leaf	Description	Amount
1	-	141	Mortgage Fixed	0.750000
2	-	142	Mortgage Adjustable	0.250000
3				
4			Total	1.000000
5				
6				
7				
8				
9				
10				
11				
12				

Org Unit ID | GL Acct ID | Common COA ID

7. Define how the 401-K Plan expenses will be allocated.

8. Select the GL Acct ID tab and highlight a cell in the 401-K Plan Cost row.

9. Select the Common COA ID tab again.

A blank spreadsheet appears. There is a separate Common COA definition dialog box for each of the leaf values in preceding tier (in this case, GL Account).

10. Type the products and percentages used to allocate the 401-K Plan expense.

		Leaf	Description	Amount
1	-	141	Mortgage Fixed	0.400000
2	-	142	Mortgage Adjustable	0.600000
3				
4			Total	1.000000

11. Repeat this step for the Medical and Other Benefits Expense.

The Table ID is now ready to use in an Allocation ID. (See Chapter 8, "Allocation ID".)

7

Lookup Table ID

Lookup Table IDs are user-defined tables in the Oracle Financial Data Manager (FDM) database containing key columns and return columns. The key columns on a Lookup Table ID must all be matched to columns in the Source table. A Lookup Table ID can have multiple return columns.

Lookup Table IDs are used in conjunction with Allocation IDs to return columns from other tables for use in calculations or allocations. Lookup Table IDs enable user-defined tables to source data for allocation processing.

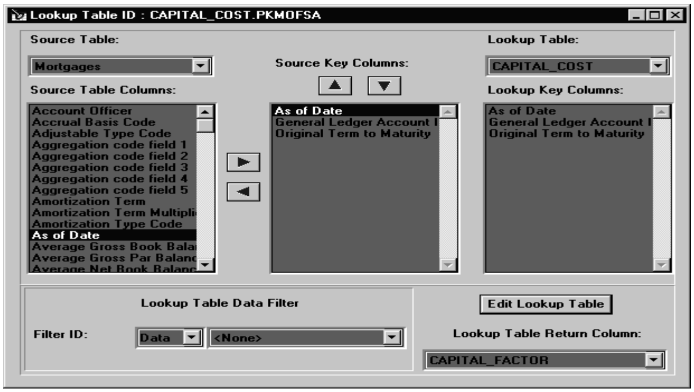
Lookup Table IDs provide greater flexibility than Table IDs, enabling users to key on numeric, character, or date column data types for lookup data. Table IDs are limited to leaf fields as the source keys in a lookup process.

An example of using Lookup Table IDs is the retrieval of capital allocation factors for assigning capital to products within the organization. The capital allocation factors are stored in a user-defined table with a key on date, product, and term and a column holding the factor.

This chapter is divided into the following sections:

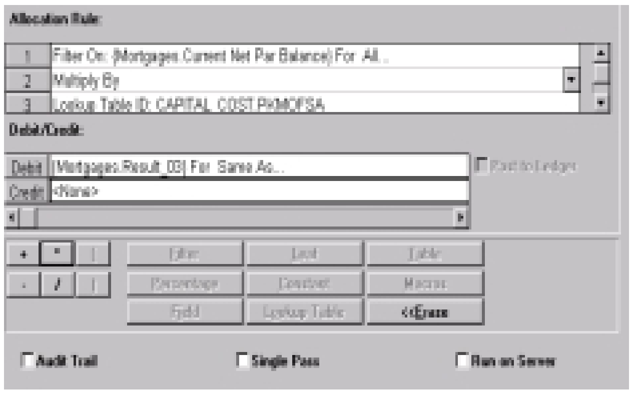
Define a Lookup Table ID

To define a Lookup Table ID, select the New icon on the horizontal toolbar and then click the Lookup Table ID icon on the vertical toolbar. Alternatively, from the File menu, select New and then select Lookup Table. The Lookup Table ID window appears:



Lookup Table ID Usage

Lookup Table IDs are part of Allocation ID processing. Lookup Table IDs are end operators for Allocation IDs similar to Percent, Table, and Field buttons. Unlike other end operators, multiple Lookup Table IDs are available in a single allocation page. Any single allocation page can have multiple Lookup Table IDs, but all Lookup Table IDs must contain the same Source table defined in the Lookup Table Id and the Filter On portion of the Allocation ID.



Source Table Indexes

By definition, the Lookup Table ID has an index on the key columns. Depending on the size of the Source table, an index on the fields used to key to the Lookup Table ID may be required. The user should consult his or her DBA for best practices on table indexing.

Lookup Table ID Registration

The Lookup Table ID lookup tables must be registered within the OFSA schema. The registered lookup tables must have a Table Classification Assignment of “PA Lookup Tables” and a Processing Key equal to the unique key on the table. The DBA defines the lookup tables and can load data similarly to instrument table data.

Set Up a Lookup Table ID

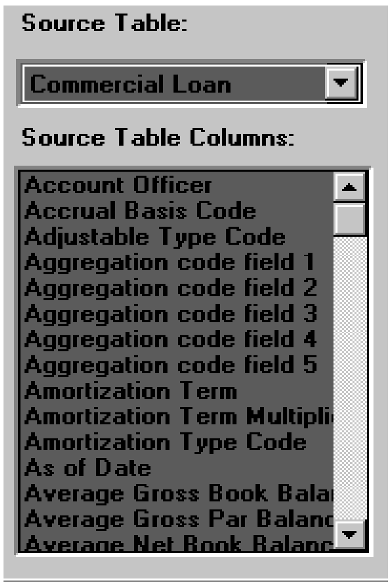
Perform the following steps to set up a new Lookup Table ID:

1. Define and register the lookup table.
2. Define a new Lookup Table ID.
3. Select a Source table.
4. Select a lookup table.
5. Select Source Key columns.
6. Edit or insert data in the lookup table, if necessary.
7. Select Lookup Table Return Column.
8. Add Lookup Table Data Filter, if necessary.
9. Add Lookup Table ID to Allocation ID for processing.

Source Tables

Source tables for a Lookup Table ID can be any registered instrument or transaction table in the FDM database. The Lookup Table ID is not supported on allocations to LEDGER_STAT.

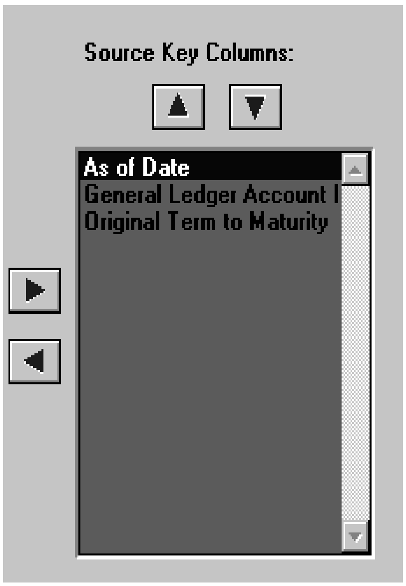
Select the source table from the list. Once a Source table is selected, a list of columns in the Source table appears in the Source Table columns list. Clicking on a column highlights it.



Source Key Columns

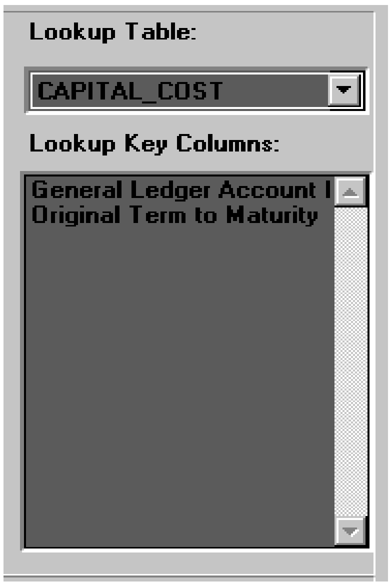
Columns in the Source Table Columns list are moved to the Source Key Columns list for use as keys in the lookup process. The right and left arrows move highlighted columns between the Source Table Columns list and the Source Key Columns list.

The Source Key Columns list matches the source keys with the lookup keys in sequential order. The top column in the Source Key Column list matches to the top column in the Lookup Key Columns list. The up- and down-arrows sort the columns in the Source Key Columns list. Use the up- and down-arrows to match Source Key Columns with the Lookup Key Columns.



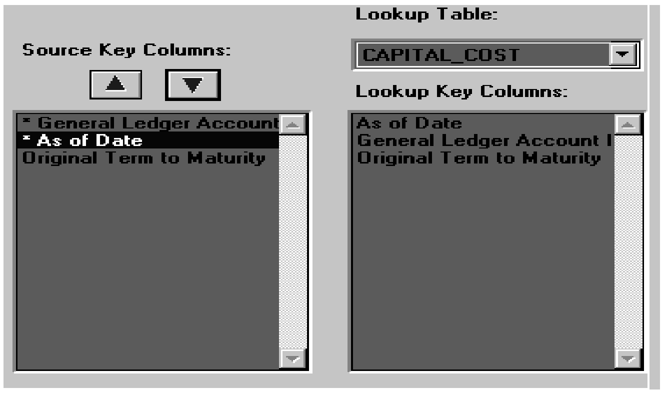
Lookup Key Columns

The Lookup Table list shows all registered lookup tables in the FDM database. A table must be registered with the table property *PA Lookup Table* in the Oracle Financial Data Manager Administration application to appear in the list. Selection of the lookup table from the list automatically populates the Lookup Key Column list. The Lookup Key Column list cannot be edited. The columns automatically appear in the order of the unique index on the table and cannot be rearranged.



Match of Source Key Columns

The match of Source Key Columns with Lookup Key Columns must contain the same data type. If a numeric Source Key Column is matched with a date Lookup Key Column, the interface identifies the mismatched data types with an asterisk. All mismatched column data types have asterisks.

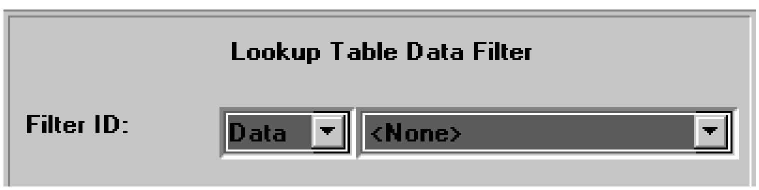


Lookup Table Return Column

Define the column returned from the key matching between the source table and lookup table in the Lookup Table Return Column list box. All available Return columns appear in this listed for the selected Lookup Table. Key columns on a lookup table are not available in the Lookup Table Return Column list.

Lookup Table Data Filter

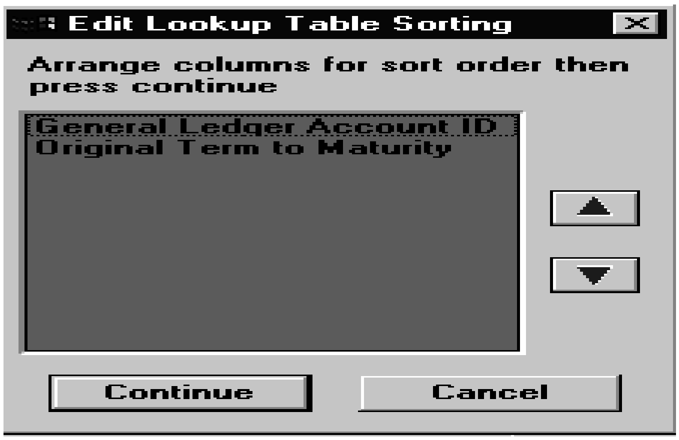
Factors stored in Lookup Table IDs may span multiple instrument and transactions tables in the FDM database. Data Filter IDs can be added to a Lookup Table ID minimizing the data used in any Lookup Table ID process. Select the Data Filter ID in the Lookup Table Data Filter list. The default Data Filter on a Lookup Table ID is *none*.



Edit Lookup Table

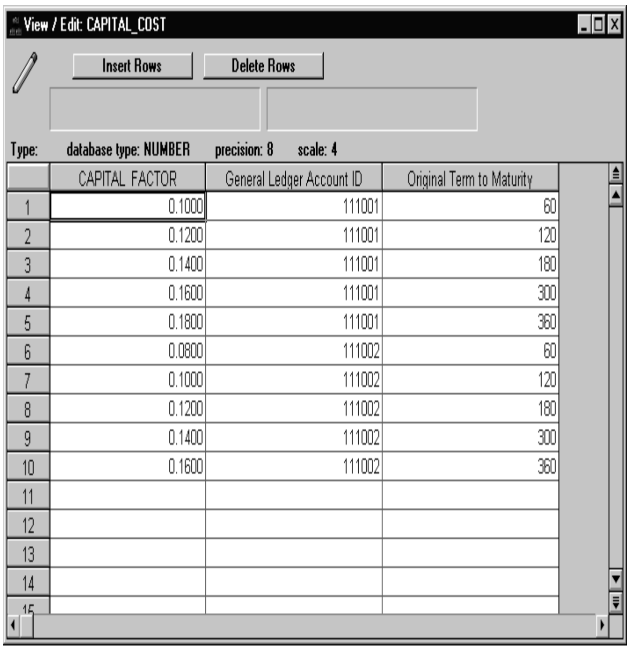
The Edit Lookup Table button enables users to edit, insert, and delete values in the lookup table. When the user clicks the Edit Lookup Table button, the Edit Lookup Table Sorting dialog box appears.

The Edit Lookup Table Sorting dialog box defines the sort columns and how the order data is retrieved in the edit lookup table spreadsheet. Use the up- and down-buttons in the dialog box to order the columns for data sorting. When all columns are in the preferred order, click Continue to open the View/Edit window.



View/Edit Window

The View / Edit window enables the user to change column data, delete rows, and insert rows into the lookup table. Double clicking on any cell highlights the data and enables editing. CTRL C copies data, and CTRL V inserts the copied data into other selected cells. When all cell changes are complete, clicking on the Save Icon saves changes to the FDM database. The View / Edit window is a good way to examine factors in the lookup table before processing.



8

Allocation ID

The Allocation ID provides you with the ability to move data among different leaf values or dimensions, such as cost centers, chart of accounts, or products. Allocation IDs are commonly used to allocate expenses from a high organizational level to a more detailed level. For example, you want to distribute the bank's total loan servicing expense from the Loan Servicing Center to a group of profit centers. There are several methodologies that can be used in the Allocation ID to distribute balances. An Allocation ID can access tree filters, tree nodes, data filters, individual leaves, Table IDs and database table columns to define any type of allocation.

See Appendix A, "Fiscal Year Information" for information on how the Allocation ID works with a LEDGER_STAT table defined as a Fiscal Year.

This chapter presents the following topics:

- Create an Allocation ID
- Components of an Allocation ID
- Multiple Record Allocation IDs
- Process an Allocation ID
- Allocation ID Examples

Create an Allocation ID

To create an Allocation ID, complete the following steps:

1. From the File menu, select New and Allocation ID.

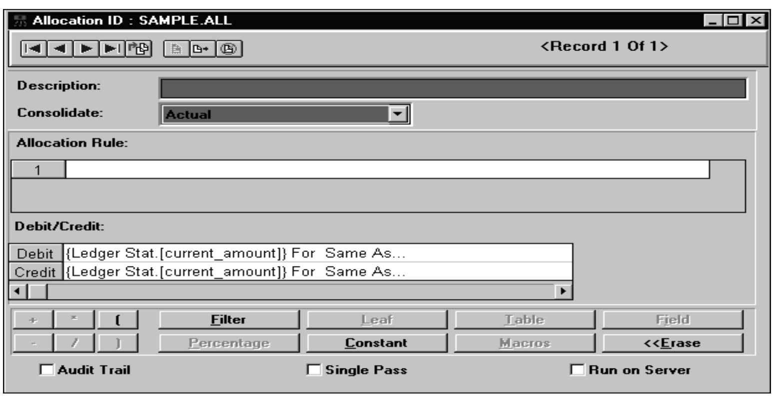
   The New Allocation ID dialog box appears.

2. Select a Folder from the list.
3. Type a descriptive name for the ID in the Allocation ID pane.
4. Type a description for the ID in the Description field.

   This is an optional field.

5. Select either the Read / Write option or the Read Only permission level.
6. Click OK.

   The Allocation ID window appears with the name of the ID displayed at the top of the dialog box.



Components of an Allocation ID

The Allocation ID comprises many components that are used to set assumptions for the resulting allocation.

Description

Specify a description for this allocation for memo purposes.

Consolidate

Select a consolidation code from the list that is to be written to all records created by this Allocation Rule.

The consolidation code defines whether the written balances are actual, budget, or forecast data. The consolidation code displayed value does not act as a filter criterion.

Allocation Rule

Build the Allocation Rule line by line, similar to building a formula. Use the buttons at the bottom of the dialog box to add items to the Allocation Rule. The buttons are enabled or disabled depending upon whether the option is valid to complete the allocation. Following is a description of the types of functionality available within an Allocation ID.

Operators

The first four buttons are the mathematical operators: addition (+), multiplication (*), subtraction (-) and division (/). These operators are used in conjunction with the following options:

Parentheses

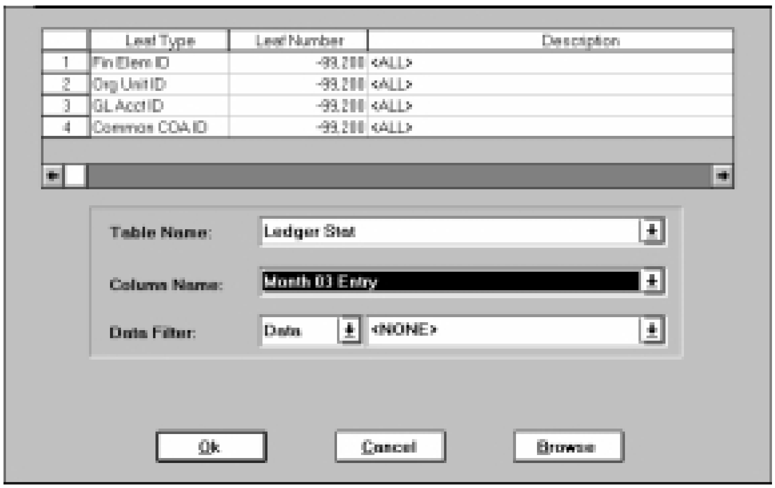
You can use parentheses to explicitly order math operations in your Allocation Rule. Math operators have a natural order of precedence. Multiplication and division take precedence over addition and subtraction. You can use parentheses to clearly define your Allocation Rule operations and to override the natural math operator precedence.

There are some rules that apply to using parentheses. All operator buttons are enabled or disabled depending upon context because some operations are illegal. You can start a rule with one or more parentheses but the next operations must be a Filter On. Also, Leaf, Table ID, and Percent of, which operate on a data set, must be the last operations in a rule and cannot be contained within parentheses.

Filter

The Filter option enables you to define a group of records from the database that is used to *drive* the allocation. For most Allocation Rules, this is the first item in the allocation definition. Select the Filter button and the Filter On row is added to the Allocation Rule dialog box.

To edit the Filter On criteria, select the row. The Edit Filter On Item dialog box appears:



In the top pane, called the Edit Window, are the list of all leaf types represented on the selected table defined as key leaves. You may enter a specific leaf value, a Tree Node, a Tree Filter, or a Macro. Use the <ALL> Macro to include all values of a particular leaf type. To operate on a specific leaf value, type the value directly in the Edit Filter On Item dialog box.

You can click on a Leaf Number and then right click with your mouse to produce a list of context-sensitive choices for the field. You can use this menu to select <All>, or click Browse to open the Browse Leaves/Trees dialog box. You can also click the Browse button to produce the Browse dialog box.

Note: When an Instrument table is selected, Financial Elem ID leaf disappears because the leaf is not represented in the instrument type tables.

Browse Dialog

In the Browse dialog box, you are given the option to select <Leaves>, <Tree Filters>, or Rollup IDs from a list. If you select <Leaves>, select the search field type, by leaf value or leaf description. Then adjust the search range of leaf values and click Display. The list of available leaves appears. For Description string searches, use the percent symbol (%) as the wild card symbol. Click on the desired leaf value and click OK. The selected value is inserted in the Leaf Number column for you.

You have the option to specify a Tree Rollup ID or Tree Filter ID relevant to the leaf type chosen. If you select a Rollup ID from the list, you must specify the Tree level and Leaf/Node appropriate for you filter criteria. You can search the rollup using the Rollup Description, Rollup Level, or Rollup Node. Select the appropriate search field, select a range or type in a description, and click the Display button to see your available choices. Select the Rollup node you want and click OK to insert the value in the Leaf Number column.

If you select <Tree Filters>, you can select Rollup IDs or Tree Filter IDs from the Search Field list. If you select Rollup IDs, a list of Rollup IDs relevant to the chosen leaf type appears. Click on a rollup and click Display to see Tree Filter IDs associated with the selected Rollup ID. If you search Tree Filter IDs, you can search for Tree Filter names from a list of filters in your database. Click Display with no criteria to see the full list of Tree Filters for your leaf type.

Click OK to return to the Edit dialog box. Now specify which table and column hold the data for the allocation. Select the table in the Table Name dialog box and the column in the Column Name dialog box.

Pct Of (Percent Of)

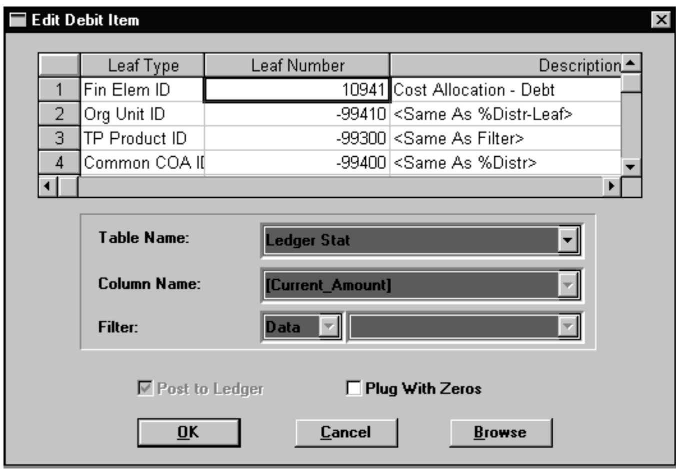
The Pct Of function distributes data that passes the Filter On criteria to a group of leaves on a percent to total basis. Pct Of is added as a row to the Allocation Rule. A Pct Of is the last item in a Allocation Rule.

Select the Pct Of line item to specify which leaves to use to calculate the percent to total. Follow the detailed instructions for the Edit Filter On dialog box. Also, select the appropriate table and column that hold the leaves specified to drive the percent to total calculation.

To calculate a percent distribution, select the Force to 100% box. A check appears in the selection box. This option is a toggle. Selecting it again turns it off. If the Force to 100% box is not selected, a percent distribution is not calculated. Instead, a simple multiplication, division, addition, or subtraction is calculated using all of the records that meet the Pct Of criteria.

Note: A Percent Distribution Allocation can use a different percentage distribution methodology. This enables you to control the percentage weighting factor by leaf. For example, to control the percentage weighting factor by an individual org unit (cost center) rather than by the entire org unit field, use the Debit & Credit macro called <Same_As_%_Dist_Leaf>.

For example, the Edit Debit Item dialog box shows the Debit of an expense based on a percent distribution of Product (Common COA ID) by Cost Center (Org Unit ID).



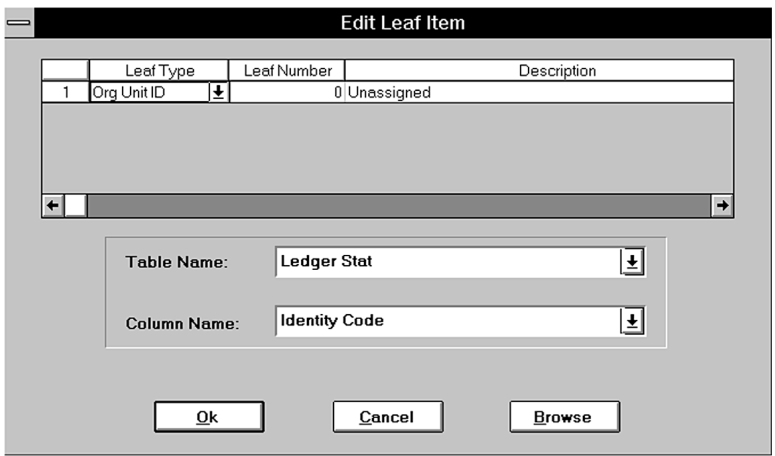
In cost center 1600, there are four products, which have a total balance of $1000, and in cost center 1700 there are five products with a total balance of $1500. The sum of all products regardless of these two cost centers and all other cost centers is $100,000. If there was no <Same As %Dist Leaf> macro specified on the Org Unit leaf, a normal percent distribution just by product would be performed, using $100,000 as the denominator for calculating the Percent to Total calculation. By selecting the <Same As %Dist Leaf> macro for the Org Unit leaf, however, a Per Org Unit Distribution is performed. Thus, $1000 would be used as the denominator for calculating the Percent to Total calculation for cost center 1600, and $1500 would be used as the denominator for cost center 1700. This provides for expense distribution broken out as a percent distribution per cost center.

Constant

Use this option to specify a constant amount to use in conjunction with an operator for all records that pass the Filter On criteria. Select Constant to add the row to the Allocation Rule pane. Double-click on the row to type in the constant value. Click Enter after the constant has been entered correctly. Follow these same steps to edit the constant.

Leaf

Specify a distinct leaf value or group of leaves to use in conjunction with an operator for all records that pass the Filter On criteria. Select the Leaf button to add the row to the Allocation Rule pane. Select the row to open the Edit Leaf Item dialog box.



Select the leaf type you want to define. Specify a single leaf value by typing it directly into the Edit dialog. You can right click your mouse on the row and click Browse or click the Browse button to produce the Browse Leaves/Trees dialog box. From this dialog box, you can select leaves, Tree Filters, or a Tree Rollups. You can find leaves using either the leaf description or the leaf value. Click the Display button to show the values that match your search criteria. For leaf value searches, adjust the search range of leaf values and click Display. The list of available leaves appears. For Description string searches, use the percent symbol (%) as the wild card symbol. Click on the desired leaf value and click OK. The selected value is inserted in the Leaf Number column.

To select a Tree Filter ID, select either Rollup IDs or Tree Filter IDs from the Select Field box. If you select Rollup IDs, a list of Rollups appears. Click the one you want and click on Display to show associated Tree Filter IDs. Click the Tree Filter ID that you want and click OK to insert the value in the Leaf Number column. If you click the Tree Filter IDs Search Field, you can search for a filter name from the list of tree filters in your database that is associated with your leaf type.

Specify the table and column that hold the data to be used in the calculation. Click OK to return to the Allocation ID window.

You can define only one leaf type in the Edit Leaf Item dialog box. All other leaf types are defined by the Filter On criteria. For example, assume the Filter On criteria focuses on ending balance records for G/L account 130510 in org unit 1300. If the Leaf criteria is for beginning balance, the leaf function focuses on all beginning balance records for G/L account 130510 in org unit 1300.

Leaf type allocations are used only against the LEDGER_STAT table. They are typically used to perform an operation between two sets of rows that differ in a single dimension, such as for Financial Element ID to perform a cost of funds calculation.

Table

Select Table to reference a Table ID that already has been created. A Table ID is added as a row to the Allocation Rule. It is the last item in an Allocation Rule. It is used to distribute amounts according to multiple leaves and percentages and to perform a lookup. See Chapter 6, "Table ID" for more information.

If there are several predefined Table IDs, click on the row for a list of Table IDs, and select the appropriate ID.

Macros

Select Macros to specify the accrual basis that is applied to the records that pass the Filter On criteria. It is added as a row to the Allocation Rule. This feature is most commonly used when calculating income from balances using annualized rates. Double-click the row (or click the arrow at the end of the row) for a list of accrual bases to appear. Select the appropriate macro from the list.

Field

Select Field to specify a table and field from the database to use in the Allocation calculation. A default Table/Column is added as a row to the Allocation Rule. The Allocation Rule accesses the specified field for the records that have passed the Filter On criteria. Select the Field row to open the Edit Field Item dialog box. Select the appropriate Table and Column. Click OK to return to the Allocation ID window. A field is typically used to multiply two columns within a single row in an instrument table update Allocation Rule.

Erase

Use this button to erase the last entry in the Allocation Rule. Select Erase and the delete confirmation dialog box appears. Click Yes to continue with the deletion. Click No to cancel the deletion.

You cannot insert a correction in an Allocation Rule. You must erase the rule from the end to the place where you want to insert a correction. You then add back the parts of the rule that you need.

Debit/Credit

The Debit/Credit section of the Allocation ID defines the records that are posted or updated as a result of processing the Allocation ID. When posting or updating records in the LEDGER_STAT table, the account type specified in Leaf Setup for a particular Financial Element ID or Common COA ID determines if the balances of the resulting records are added (positive balances) or subtracted (negative balances) from the database.

Note: The account type can be defined in the Financial Element ID. If you prefer to have the account type defined in the Common COA ID, however, you must set the Financial Element ID to <Use Detail Leaves>in the Leaf Setup financial element.

Edit Debit/Credit Item Dialogs

Allocation Rules write the results of the rule to tables in the database. An Allocation Rule writes differently to the LEDGER_STAT table than to an instrument table.

Allocations rules that write to LEDGER_STAT The first time an Allocation Rule is run, a new row is written to the LEDGER_STAT table and a new Identity Code is assigned to the Allocation Rule. (See Deleting Allocation Results for information on Undo and Identity Code.) If the allocation is run again for the same time period (As_Of_Date) or for another time period in the same fiscal year, the results are written to the same row that was created previously. When the allocation is run for a time period that falls in the next fiscal year, a new row is again created in LEDGER_STAT using the same Identity Code but for the new year (a different YEAR_S value). The leaf values used in creating new LEDGER_STAT rows are determined by the values set in the Edit Debit Item and Edit Credit Item dialog boxes. If an amount is being allocated to a specific account (reclassified), you can enter the new leaf value directly in the Leaf Number column. Other predefined macros can be used to assign leaf values for the resultant rows written.

When allocating money from the Filter On data set to a different account, the leaf values in the Edit Credit dialog box can be left as <Same As Filter>, creating a row that is equal in magnitude to the allocated amount but opposite in sign.

Edit Debit/Credit Macros The macros available for Edit Debit and Credit Items are:

<Same As Filter> (-99300) - denotes that the posted rows use the same leaf value that the input rows have in the Filter On set.

<Same As %Distr> (-99400) - denotes that the posted rows use the same leaf values as contained in the Percent Of set of the Allocation Rule for this leaf. See Pct Of (Percent Of) for information about creating Pct Of allocations.

<Same As %Distr-Leaf> (-99410) - denotes that the posted rows use the same leaf value as contained in the Percent Of set of the Allocation Rule, but a new row is generated for each distinct group in the Percent Of set.

<Same As Table> (-99450) - denotes that the posted rows use the leaf values as specified on the Target Leaf of the Table ID in the Allocation Rule. (See the description of Target Leaf tabs in Chapter 6, "Table ID".)

Note: You cannot use this macro with a Table ID that has no Target Leaf or for leaves other than the target leaf type.

Allocations rules that write to Detail Tables The first time a detail level Allocation Rule (to either Instrument or Transaction Summary tables) is run, a new Identity Code is assigned to the Allocation Rule. (See Deleting Allocation Results for information on Undo and Identity Code.) New rows are never written to detail level tables. Instrument table Allocation Rules match up ID_NUMBER and IDENTITY_CODE values in the Filter On set and the Debit/Credit set. Only rows that match are updated. The leaf values in the Debit and Credit Items have no impact on rows written if the target is a detail level table. Leaf values entered here are used if Post To Ledger is checked or if the target table is LEDGER_STAT. For Percent Distribution allocations by leaf to detail level tables, the macro <Same As %Distr-Leaf> is placed in the Leaf Number column in which you want to group the leaf values.

When performing detail level allocations that calculate values within a row (for example, by using a lookup table value or multiplying a rate column times a volume column), you typically set the Credit Item to Plug With Zero.

Edit Debit/Credit Leaf Browser Click on a leaf row and then right click the mouse button to produce a context-sensitive menu that can insert the appropriate macro value in the leaf row. You can also click Browse (or click the Browse button) to open the Browse Leaves/Trees dialog box. You use the Browse dialog box to locate the leaf value that you want to debit or credit. You can debit or credit only to a single leaf value, as opposed to a Tree Filter or Rollup. You can find leaves in the Browse dialog box by searching either on the leaf description or the leaf value. For value searches, adjust the search range of leaf values and click Display. The list of available leaves appears. For Description string searches, use the percent symbol (%) as the wild card symbol. Select the desired leaf value and click OK. The selected value is inserted in the Leaf Number column for you.

Posting Options

Available options within the Edit Debit Item and Credit Item dialog boxes are *Post To Ledger* and *Plug With Zero*. The *Post To Ledger* option enables you to suppress the Ledger posting when updating instrument records. The *Plug With Zero* option allows for complete suppression of either the debit or credit postings/updates.

Using the options together enables you to represent one-sided entries. These are typically used for posting statistical records to the LEDGER_STAT or for instrument level only allocations that are both typical in Account Level Profitability implementations.

Using LEDGER_STAT Macros

When defining the LEDGER_STAT table within the Allocation Rule, you have the option of using the LEDGER_STAT macros to define a column within the table. You can also access a column directly, for example, Month 2 Amount. These macros provide an alternative way to access data in the LEDGER_STAT table. The [Current_Amount] macro is the preferred way to refer to the current month column because it takes the fiscal year into account (see Appendix A, "Fiscal Year Information" for information on how the Allocation ID works with a LEDER_STAT table with a fiscal year).

The following macros are included:

Macro Name	Macro Description
[Current_Amount]	Represent the current month column of the LEDGER_STAT table as set in the Configuration ID
[Last_Mo_Amount]	Represents the previous month column of the LEDGER_STAT table
[Last_Mo_YTD_Amt]	Represents the previous month year to date column of the LEDGER_STAT table
[YTD_Amount]	Represents the current year to date column of the LEDGER_STAT table

All of the macros rely upon the As of Date set in the Configuration ID.

The macros are available only for use within the Allocation Rule and not the debit/credit section of the Allocation ID. As a result, LEDGER_STAT postings (as well as instrument updates) occur only for the current period.

Multiple Record Allocation IDs

Performance Analyzer provides the option of defining multiple Allocation Rules in a single Allocation ID. This is considered a multiple record allocation. (Each Allocation Rule is considered to be a record, or a *page*.)

For example, if you wanted to allocate several related expenses that each require a different allocation definition, you would use this option. You define the first Allocation Rule, or page, and then copy that rule to a new page and make the necessary changes for the second rule, and so on. It is convenient to use multiple page IDs to keep groups of Allocation Rules together as a single Allocation ID. Also, this reduces the number of Allocation IDs that need to be managed.

Managing and moving records

The toolbar in the Allocation ID window provides the same functionality as the options found in the Record menu:

- Add a record
- Copy a record
- Delete a single or multiple records
- Move to other records

First Record

Click this icon or select First Record from the Record menu to move directly to the first record in the Allocation ID.

Previous Record

Click this icon or select Previous Record from the Record menu to go to the previous record in the Allocation ID.

Next Record

Click this icon or select Next Record from the Record menu to advance to the next record in the Allocation ID.

Last Record

Click this icon or select Last Record from the Record menu to advance directly to the last record in the Allocation ID.

Goto a Record

Click this icon or select Goto a Record from the Record menu to move directly to a specific record.

Type the record number or use the arrows to enter the correct record number. Click OK and the selected record appears in the window.

Add a New Record

Click this icon or select Add New Record from the Record menu. A new, blank record is added as the last page of the Allocation ID.

Copy Current Record to New Record

Click this icon or select Copy Current Record from the Record menu. The current record is copied to the end of the Allocation ID as a new record.

Delete Current Record

Click this icon or select Delete Current Record from the Record menu to delete the current page of the Allocation ID. (You are prompted to confirm the deletion.)

Copy Fields

This feature enables you to copy allocation specifications from one record to another record (or multiple records) in the Allocation ID. It is a time-saver when you need to make the same change to all pages in a multiple record Allocation ID. You can copy any field defined in an Edit dialog box (such as the Filter On dialog box, Pct Of dialog box, and Debit dialog box).

Select the field that you want to copy, click the right mouse button, and select and Copy. When you are copying the Org Unit ID field from the Edit Filter On Item dialog box, for example, the Copy Field dialog box appears. Type the start and end records of the range to which you want to copy the selected field. Click OK to copy the value to the selected records.

Process an Allocation ID

To process an Allocation ID, select the Run option from the Process menu. The Run ID dialog box appears. Click OK to process the allocation and save the results directly to the database.

Select Preview to process the allocation and display the allocation results for review. When you have verified the results, you can select Print to send the results of the allocations to the printer. Click Save to write the results to the database. If you do not want to save the results, click Discard.

Click OK to return to the Allocation ID window.

Processing options

The Single Pass, Run on Server, and Audit Trail options offer three different process modes that affect the speed and convenience of running an Allocation ID.

Single Pass

Toggle this option ON to increase performance of appropriate multipage allocations. Normally, each allocation page is filtered, evaluated, and run page by page. Single Pass enables the allocation to evaluate the Filter On condition only once for all pages. Then the filter row is run against all pages. This process continues for each row of the Filter on set.

This option is available only for specific types of Allocation IDs. In general, the Single Pass option is available only for multipage allocations that have the same Filter On condition (or WHERE clause) on all of the pages.

Single Pass can be used on allocations against the LEDGER_STAT table. Detail level allocations that update multiple columns in an instrument table can run in Single Pass if each page can be updated by a bulk UPDATE command. This means that the allocation pages are performing rules that use only field, constant, macro, and Table ID operations against the same filter set.

Multipage allocations can run in Single Pass when they use constants, fields, and Percent distributions in any combination on the different pages.

Note: If you receive an error message running the Single Pass allocation, uncheck Single Pass and run the allocation again.

This option has the following constraints:

- The Filter on WHERE clause must be the same on all pages. The column in the Filter can be different on each page
- There must be no implied order of operation between the pages of an allocation. The results written from page one cannot be used in the calculations of subsequent pages.
- Instrument-level allocations cannot be run unless each page updates a different table. If all pages update different columns in the same table, however, you can run Single Pass.
- The allocation must not update the same instrument table column on different pages.
- LEDGER_STAT Leaf operations (for example, filter times leaf) are not allowed.
- Percent distribution allocations can be run Single Pass if the percent distribution set does not exceed the maximum PctOfBufferSize as defined in OFSPA.INI. See the *Oracle Financial Services Installation and Configuration Guide* for the default value of PctOfBufferSize.
- Some allocations run in Single Pass but cannot be run to Preview with Single Pass checked. If this happens, uncheck the check box and run the Preview. Afterwards, reselect the Single Pass check box.
- An allocation with a Table ID on one page is run in Single Pass only if all the other pages have a Table ID.
- Instrument-level allocations cannot be run in Single Pass if any page does Post to Ledger.

Run On Server

Toggle this option ON to process the Allocation ID on the server rather than on the client (local) machine. This option saves processing time. You cannot process the Allocation ID on the server, however, if you want to preview the results.

Note: Note that when you run an allocation for a second time in Run on Server mode, you are not prompted for "Already run this period, do you want to run again?"

Audit Trail

Toggle this option ON to write current allocation details to the Audit Trail table when your allocation is processed. This will allow you to create an Audit Trail report using a Report ID.

Toggle this option OFF if you do not want the Audit Trail table updated with details of the current allocations. The total results of your allocations are still recorded in the LEDGER_STAT table.

UNIX Processing Options

If your financial institution is running the OFS applications on a UNIX system, specific processing options are available.

UNIX Parallel Allocation

To process your allocations more quickly, you can run multiple process on a UNIX server. Several allocations can run in parallel. Also, allocations that process many rows can run faster if several processes work simultaneously on the data set.

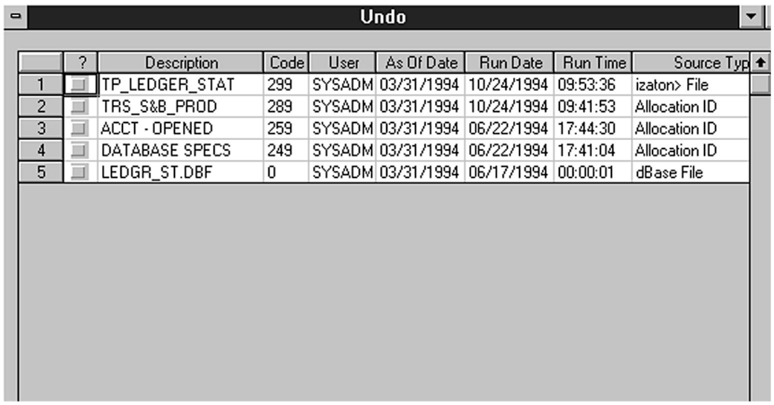
If you are running allocations on a UNIX server you can set up multiprocessing for your allocation IDs. See the *Oracle Financial Services Installation and Configuration Guide* for information on how to set up this option. In addition, see Chapter 15, "Batch ID" for information about setting up your system to run Allocation IDs in parallel.

Running too many processes on a UNIX server can result in overall slower performance. Optimal performance is achieved through a balance of the number of processors on the server and the amount of system memory it has, the size of the allocation Filter On sets, and the amount of parallelism used in processing.

You must also design your allocation Batch IDs so that an allocation that is dependent on a preceding allocation ID does not run until the first one has completed.

Deleting Allocation Results

If you process an allocation, save the results to the database, and later realize that the results were incorrect, you can use the Undo option. Undo enables you to delete allocation results. From the Process menu, select Undo. A dialog box appears listing all of the Allocation IDs that have been processed:



The Spreadsheet

The Allocations are listed and numbered, as in a spreadsheet. You can select records to delete by identity code, As of Date, and by using a data filter (which must have been defined previously.)

Most spreadsheet columns contain informational data provided by the system. Following is a description of the columns and their use:

Description:

The name of the Allocation ID or other source type, such as a dBase file.

Code:

The Identity Code number assigned to the allocation by the system when an allocation is run for the first time.

User:

The OFSA user who performed the allocation (actually, the login name, which may vary from the actual user name.)

As of Date:

The As of Date that was assigned at the time of the allocation.

Run date:

The date the allocation was processed.

Run time:

The time the allocation was processed.

Source type:

The output data source, usually an Allocation ID, but it could be a dBase file that corresponds to the ledger load file.

Table name:

The table on which the allocation was performed, primarily the LEDGER_STAT table or an Instrument table.

Column name:

The name of the column updated by the allocation.

of records:

The number of rows affected by the allocation, which may not equal the number of rows updated.

Data filter:

You can select predefined Data Filter IDs from the lists that are provided for each allocation row. If you want to undo a certain group of the results for a particular allocation, create and select a Data Filter ID that includes only those rows to undo.

?

You can undo one or multiple allocations simultaneously. Click the check box in the ? column of the spreadsheet for each allocation you would like to undo.

Process an Undo

Click the RUN icon on the horizontal toolbar to begin the Undo process. Click OK again to continue. Click Cancel to stop the undo process.

Allocation ID Examples

Two Allocation ID examples are provided.

Allocation ID - Percent Distribution

The following Allocation ID demonstrates a percent distribution Allocation Rule. This Allocation Rule distributes the total expense for Officers' Salaries for the Executive cost center to all org units based on the percent of each unit's head-count to the total. The following illustrates how data should be allocated in this example.

Total Officers' Salary

Org Unit 8200

$7750

Distribute to:

Org Unit 1200
Head Count=2
Allocated Exp.: $861.11

Org Unit 1100
Head Count=3
Allocated Exp.: $1291.67

Org Unit 1250
Head Count=3
Allocated Exp.: $1291.67

Org Unit 1500
Head Count=3
Allocated Exp.: $1291.67

Org Unit 1600
Head Count=2
Allocated Exp.: $861.11

Org Unit 1620
Head Count=2
Allocated Exp.: $861.11

Org Unit 1640
Head Count=3
Allocated Exp.: $1291.67

The total head count for all Retail centers is 18. The expenses allocated to center 1640 is 3 / 18 * $7750 = $1291.67.

Practice Creation of Allocation ID

Perform the following steps to create an Allocation ID for this distribution.

1. Create a New Allocation ID named Officers' Salary.
2. Select the Filter On button to add the Filter row to the Allocation Rule.

   This is where you define the total expense to be distributed.

   In this example, the expense is stored in the database as GL Account 800201 and Org Unit 8200. Select the Filter On row to view the Edit Filter On dialog box. Specify the appropriate Org Unit ID and GL Account ID by typing the values in the Leaf column of this dialog box. Or select Browse to see a list of all values of a particular leaf type, and select the appropriate values.(See Chapter 4, "Overview of IDs" for more information.)

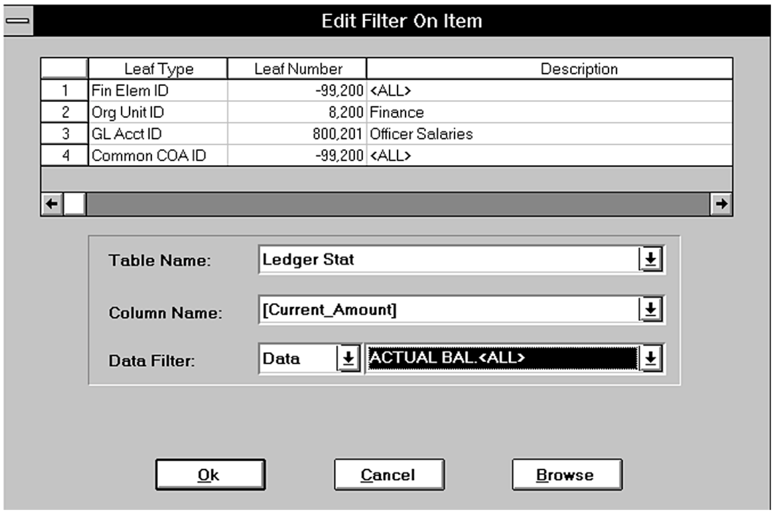
   Once you have defined which records to filter on from the database, specify the column that holds the value to use in the allocation. In this example, refer to the current month entry from the LEDGER_STAT table. Select LEDGER_STAT in the Table Name dialog box and select Current Amount in the Column Name dialog box.
3. Add a data filter to the Allocation Rule.

   Because the database holds actual, budget, and forecast data, you need to specify which type of data to perform the allocation on.

   The consolidation code field in the LEDGER_STAT table distinguishes between these types of data. If the allocation were run without a data filter, the Filter On criteria would return the sum of all actual, budget, and forecast data for the Officers' Salary account for org unit 8200. It would then allocate based on the sum of actual, budget, and forecast Head Count data. All resulting records would the be written to the database as Actuals (per the Consolidation type selected in the Consolidate field). This does not make sense, however. Instead, apply a data filter to the Allocation Rule that includes only Actuals records.

   For this example, add a Data Filter ID named ACTUAL.BAL.

Note: Note that this Data Filter ID had been created prior to defining the Allocation ID.



4. Click OK to return to the Allocation ID window.
5. Click the * (multiplication) button.
6. Click the Pct Of button.
7. Select the Pct Of row to open the Edit Pct Distribution Item dialog box.
8. Specify the Org Units and the GL Account that are used to calculate the percent distributions.

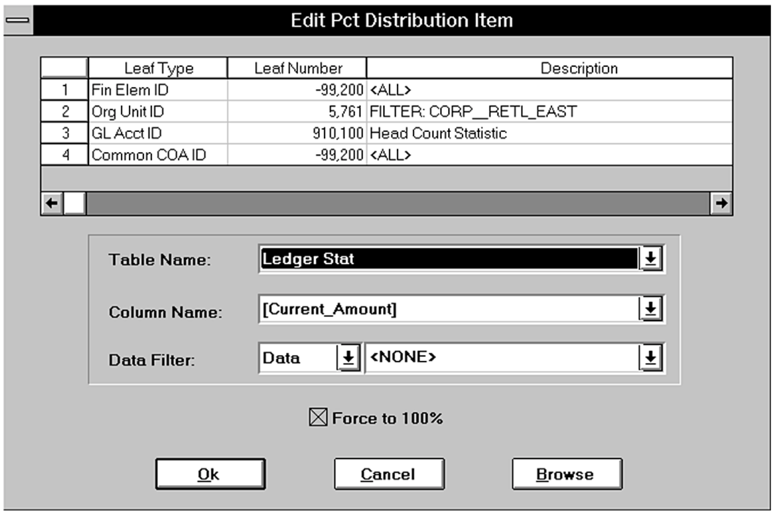
   For this example, the GL Account that holds the Head Count is 910100. As already explained, the salary expense should be distributed to the seven Retail org units. On the Edit Item dialog box, however, you can input only a single leaf value. In order to specify all seven Retail org units, you must use a tree filter that includes these seven org units. This tree filter should be created prior to creating the Allocation ID. (See Chapter 12, "Tree Filter ID".)
9. To specify the appropriate Tree Filter, highlight the Org Unit row and select Browse.

The Browse dialog box appears.

10. Select <Tree Filter> and then the appropriate Filter in the list.

11. Click OK to return to the Edit Pct Distribution Item dialog box.

12. Again, select the Replacement table and Current Amount column to reference the field that holds the Head Count data:



13. Click OK to return to the Allocation ID window.

Practice Definition of Accounts

To define the accounts that should be debited for the salary expense, perform the following steps:

1. Click the Debit row to open the Edit Debit Item.

2. Define the accounts that should be debited for the salary expense.

In this example, seven total accounts are to be debited: the same Officers' Salary account (GL 800201) for the seven retail centers. This GL account is most easily specified using the <Same As> macros. The GL account receiving the debits is the same as the account defined in the Filter On criteria, so use the <Same As Filter> macro to specify this account. The Org Units receiving the debits are those defined in the Pct Distribution, so use the <Same As Filter> macro to reference these org units.

3. Highlight the appropriate Leaf Type and right click mouse.
4. Select the macros in the Leaf / Node Desc dialog box.

> **Note:** You cannot select Tree Filters or Tree nodes in the Edit Debit Item and Edit Credit Item dialog boxes. You can select only single leaf values or macros.

5. Select the LEDGER_STAT table, Current Amount column as the column to store the allocated expenses.
6. Select the Credit row to open the Edit Credit Item dialog box.
7. Specify the accounts to credit.

   In this example, you would credit the Officers' Salaries account in Org Unit 8200. This represents the total expense that was allocated out to the retail org units. This credit keeps the ledger in balance. The total distributed expense was defined in the Filter On criteria.

   You should credit the same account or accounts. To do this, select the <Same As Filter> macro for all leaf types in the Edit Credit Item dialog box. Again, select LEDGER_STAT and Current Amount Entry column to hold the credit data.
8. Click OK to return to the Allocation ID window:
9. Process the Allocation ID.
10. Click the Run icon on the horizontal toolbar.
11. Select Preview when the Run ID dialog box appears.

    The results are sent to the window in a spreadsheet format. You are prompted to save the Allocation ID.
12. Save the Allocation ID.

Allocation ID - Using Instrument Tables

You can also reference information stored in instrument tables to perform allocations. For example, you want to allocate an expense for Loan Servicing to centers based on the number of outstanding commercial loans in each center. The charge for the Loan Servicing is $1.25 for each outstanding loan.

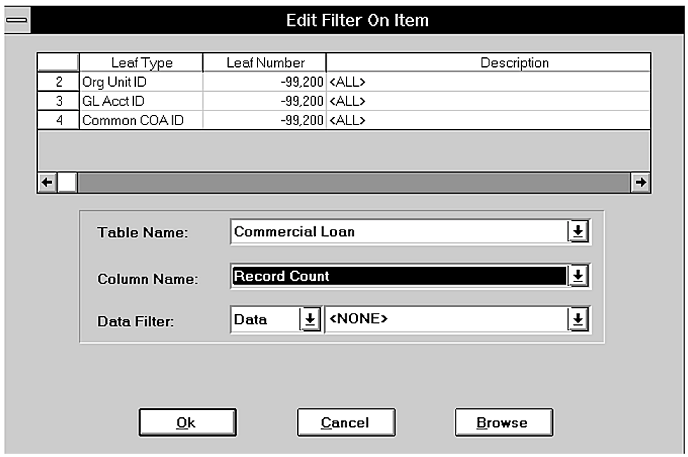
Practice Allocation of Expenses

To allocate this expense, perform the following steps:

1. In the Filter On criteria, capture the number of outstanding loans.

   This information is stored in the Record Count field of the Commercial Loan table.

2. Define the Filter On.



3. Multiply the data captured in the Filter On criteria by 1.25.

   The debits and credits are made to the LEDGER_STAT table. In this example, a debit is made to each center that has commercial loans. The debits are made to the Operating Expense GL Account in each center. A single offsetting credit is made to the Operating Expense GL Account in the Loan Servicing Center.

   The results of the allocation are as follows:

   The Metro/HQ Corporate center had a total of 495 commercial loans (495 * $1.25 = $618.75).

   The Regional Corporate center has a total of 505 commercial loans (505 * $1.25 = $631.25).

9

Data Filter ID

A Data Filter ID enables you to narrow the focus of your data for processing. It specifically targets the data you want to include or exclude for processing by another ID.

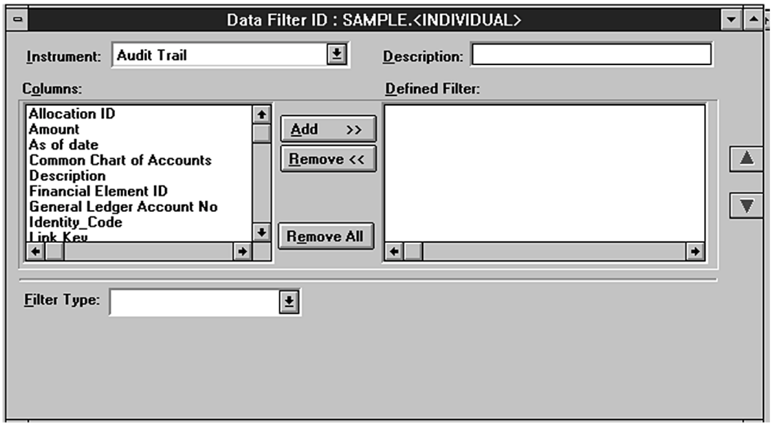
This chapter discusses the following topics:

Creating and Defining a Data Filter ID

To create a Data Filter ID, do the following:

1. From the File menu, select New and Data Filter ID to display the New Data Filter ID dialog box.
2. Type the name for the ID in the Data Filter ID field.
3. Type a description for the ID in the Description field.
4. Select the permissions for the ID.
5. Click OK to continue, or click Cancel to exit.

The Data Filter ID dialog box appears:



6. Select the instrument to include in your Data Filter ID from the list of instruments.

The instrument types are specific to your particular database. The following table lists the three types of instruments.

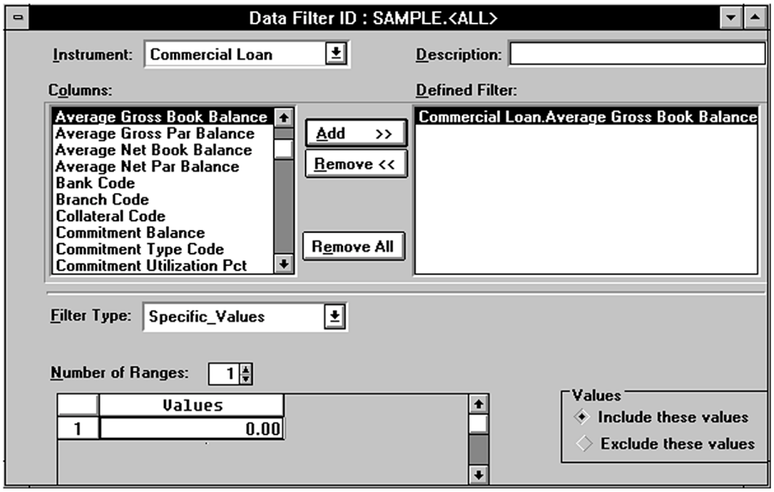
Instrument	Description
Portfolio	The Portfolio instrument type enables you to create a cross-instrument data filter. Portfolio fields are common to all instruments. All other instrument selections automatically narrow your focus to just that instrument type. If the filter criteria are the same for all instrument tables (for example, As of Date or Branch Code), you can use Portfolio as the instrument. If the filter criteria are different for all applicable instrument tables (such as Current Net Book Balance), define the filter criteria for the first instrument table, and then select the next instrument table from the instrument list and define its criteria, and so on.
Multiple tables	The Multiple Tables type instrument enables you to create a single data filter on multiple instrument tables. You must define the filter criteria for each instrument.
Formula	The Formula instrument type enables you to filter data against a selection derived from a predefined Formula ID.

7. Type or change the description of the purpose of the Data Filter ID.
8. Select one or more columns from the Columns box, and click Add to copy your selections to the Defined Filter box.

   The Defined Filter box displays the columns that you use to define filter criteria. To remove a column from the Defined Filter box, select it and click Remove. To remove all the columns, click Remove All.
9. Click on the first column in the Defined Filters box and select the Filter Type appropriate for that column. Then, enter the criteria appropriate for that column in the lower part of the dialog box. Repeat this step for each column in the Defined Filter box. See "Defining Filter Types and Criteria" in the following section for an explanation of the different types of filters and criteria that you can use with Data Filter IDs.
10. Save the ID.

Defining Filter Types and Criteria

The filter criteria options appear on the lower half of the dialog box when you select a column in the Defined Filter box and select a Filter Type. The criteria options change depending on the column and the type of filter that you select.



Filter Type

The five filter types differ based on the type of column you select. If the column represents a code, you can select a filter type of Code Values, Another Column, or formula ID. If the column represents a numeric field or date field (a non-code field), you can select a filter type of Ranges, Specific Values, Another Column, or Formula.

Filter Type	Description
Specific Values	This filter type presents the Values column. You can enter up to 60 specific values.
Ranges	This filter type offers From and To columns to enter ranges. You can enter the number of ranges that you want directly or use the spinner arrows to change the number of ranges dynamically. You can enter a maximum of 60 ranges.
Another Column	This filter type compares the selected column to another column.
Formula	This filter type compares a column to a value derived from a Formula ID.
Code Values	This filter type enables you to define specific code values for selection.

The filter type determines what criteria characteristics appear for the column that you select.

Filter Criteria

Criteria options change depending on the Filter Type that you select. The data must meet the definitions that you define in the filter criteria for each selected column. Otherwise, the data cannot pass the filter. For example, you may have two columns defined as follows:

Column	Filter Criteria
Current Balance	> 100,000
Current Rate	> 8.00

These columns indicate that a record must have a Current Balance greater than 100,000 *and* a Current Rate greater than 8.00 in order to pass the filter.

Include/Exclude After defining the specific Ranges or Values, you must choose whether you want to include or exclude the data that meets the defined criteria.

Operators You use operators for the filter types Another Column and Formula. The choices are:

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

After you have selected the operator, select the column or Formula ID as appropriate to complete the equation.

Code Values When you select Code Values as the filter type, the codes for the selected column appear in the Unselected Codes box. You can use the buttons to identify the codes you want to include or exclude in the filter.

Button	Description
Add	Moves the highlighted codes to the Selected Codes box
Remove	Moves the highlighted codes in the Selected Codes box to the Unselected Codes box
Select All	Moves all the codes to the Selected Codes box
Remove All	Moves all selected codes to the Unselected Codes box

Filter Criteria Rules

In order to get the results you want, you must define complete information in the Data Filter IDs. Complete information requires the following:

- Thorough Definitions

  You must thoroughly define the criteria for each column that you include in the data filter. All values, ranges, other column names, and formulas must be complete. If you specify that you want to filter on five specific values and then complete only the criteria for the first one, you can still save that Filter ID. However, an error message appears if you try to run an ID that uses the incomplete Filter ID.

- Order of Processing

  You can control the order of the application of the filter criteria (such as the order of the Columns in the Defined Filter box) by using the directional arrow buttons to the right of the Defined Filter box.

> **Note:** The end result of the filter is not dependent on the order of the application of the filter criteria. Each row must pass all criteria.

Implied Filter Criteria

Besides the filter criteria that you have defined, the following filter criteria may be automatically included in a process that you generate:

- As of Date

  Although the Oracle Financial Data Manager (FDM) database can contain data from an unlimited number of as-of-dates, most OFS applications automatically filter only those rows from the as-of-date that you define in the active Configuration ID.

- Row Level Security

  Your system administrator may limit the rows to which you have access.

Null Values in the Database

A null value in a column in the database is a column that has no data. You should take all necessary steps to avoid having null columns in the database for the following reasons:

- Any null value accessed in a formula results in a null value. For example:

Column X	Column Y
150	null

  If we define a formula as X + Y, the answer is 150 + null = null. All operations (such as +, -, /, *, ^) are handled in this same manner.

- You cannot define a filter to recognize null values. The Data Filter ID assumes that the database does not have any null values. Consequently, you cannot define a Data Filter ID to isolate all rows with a null value in a given column.

Note: Null <> 0, so filtering on 0 does not return rows with null values.

You can identify null values by using an SQL statement to select all rows WHERE <Column> IS NULL.

Running a Data Filter ID

Running the Data Filter ID generates an SQL statement but does not execute the statement. The application executes the SQL statement when you run an ID using the Data Filter ID.

To process a Data Filter ID, complete the following steps:

1. From the Process menu, select Run.

   The SQL statement that your Data Filter ID generates displays in a window.

2. Review the SQL statement to check the results of your Data Filter ID.
3. Click OK to close the SQL statement window.

Editing a Data Filter ID

To edit a Data Filter ID, do the following:

1. From the File menu, select Open and Data Filter ID.

   The Select Filter ID dialog box appears.

2. Select the ID you want to edit.
3. Make your desired changes.
4. Save the ID.

Using Data Filters on Multiple Tables

A Data Filter ID can contain criteria from different tables. Each line in the Defined Filter box is a part of the filter that can stand alone as its own filter. Also, each line or part of the defined filter can reference a different table.

A part of the filter can reference Portfolio as the table name. Portfolio is a proxy for whatever table the current ID is referencing. For example, if the ID using the Filter ID is a Data Correction ID, the application replaces Portfolio with the name of whatever table the Data Correction ID is correcting. The application may or may not apply each line of the filter to the data in a table.

The application applies the filter line if either of the following conditions is true:

- The table you name explicitly in the filter is the same as the table that the application is processing
- The table in the filter is Portfolio and the table the application is processing contains the referenced Portfolio field

The application does not apply the filter line if either of the following conditions is true:

- The table explicitly named in the filter is different from the table that the application is processing
- The table in the Data Filter ID is Portfolio and the table the application is processing does not contain the referenced Portfolio column

The examples in the following table demonstrate these rules. The Table and Column combination represents the filter. An **X** indicates that the application applies the filter to the table during processing.

Filter	**Commercial Loans Table**	**Ledger Stat Table**
Table: Commercial Loans Column: Current Gross Book Balance	**X**	
Table: Ledger Stat Column: Month 1		**X**
Table: Portfolio Column: Original Term to Maturity	**X**	
Table: Portfolio Column: Organizational Unit ID	**X**	**X**

Reviewing a Data Filter ID Example

This example demonstrates how to create a data filter that includes only adjustable rate mortgages. You can create a Data Verification ID using the new Data Filter ID that verifies the margins on your adjustable rate mortgages. See Chapter 5, "Data Verification ID" for more information.

To create the data filter, do the following:

1. From the File menu, select New and Data Filter ID.

   The New Data Filter ID dialog box appears.

2. Type **MTG_ARM** in the Data Filter ID field.
3. Type **Adjustable Rate Mortgages** in the Description field.
4. Click Read / Write as the permissions.
5. Click OK.

   The Data Filter ID dialog box appears.

6. Select Mortgages from the instrument list.
7. In the Columns box, select Interest Rate Code, and click Add.

   The Interest Rate Code column appears in the Defined Filter box.

8. Select Interest Rate Code in the Defined Filter box.

   This activates the Filter Type options. Code Values appears as the first choice because you have selected a code-type column.

9. Select No Index from the Unselected Codes box, and then click Add.

   The selection appears in the Selected Codes box.

10. Click Exclude These Codes in the Codes options box.
11. Select Process and Run to verify the SQL statement.
12. Save and close the ID.

When you create the Data Verification ID and apply this Data Filter ID, it limits the view to adjustable rate mortgages only.

10

Group Filter ID

The Group Filter ID enables you to group multiple data filters into a single ID. Using this ID, you can combine complex data groups in a single operation.

This chapter presents the following topics:

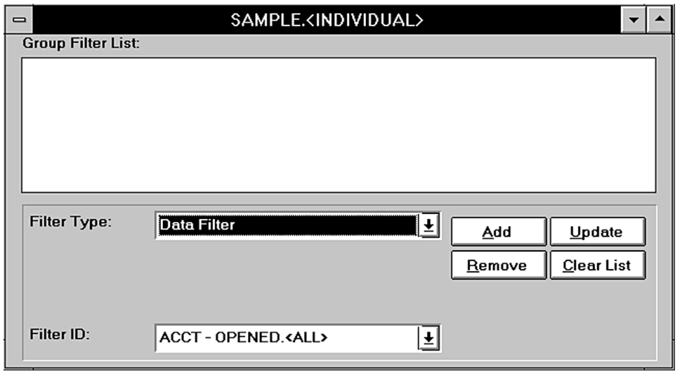
- Creating and Defining a Group Filter ID
- Editing a Group Filter ID

Creating and Defining a Group Filter ID

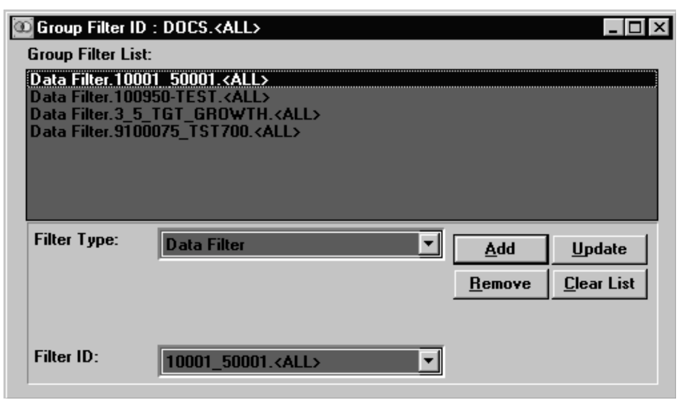
To create a Group Filter ID, complete the following steps:

1. From the menu bar select File, New, and Group Filter ID to display the New Group Filter dialog box.
2. From the Group list, select a group.
3. In the Group Filter ID field, type a descriptive title for the ID.
4. (Optional step) In the Description field, type a description that informs the user of the purpose of the ID.
5. In the Permission box, click either the Read / Write or Read Only button.
6. Click OK.

The Group Filter dialog box appears.



7. From the Filter Type list, select Data Filter.
8. From the Filter ID list, select a predefined Filter ID and click Add. The selected Filter ID appears in the Group Filter List box.
9. Repeat step 8 until you have selected all the Filter IDs that this Group Filter ID requires. When you finish, all the selected Filter IDs appear in the Group Filter List box.



10. Save the Group Filter ID.

Editing a Group Filter ID

To edit a Group Filter ID, complete the following steps:

1. From the menu bar, select File, Open, and Group Filter ID to display the Select Group Filter ID dialog box.
2. From the Group Filter ID list, select the filter you want to edit and click OK. The selected Group Filter ID appears.
3. Edit the Group Filter ID using the Add, Remove, Replace, and Clear buttons. Use these buttons to perform the following functions:

If you want to...	**then...**
Add a Filter ID from the Filter ID list to the Group Filter List,	select the desired Filter ID from the Filter ID list and click Add.
Remove a Filter ID from the Group Filter List,	select the Filter ID you want to remove and click Remove.
Replace a Filter ID in the Group Filter List with a new Filter ID,	1. Select a new Filter ID from the Filter ID list 2. Select the Filter ID in the Group Filter List that you want to replace and 3. Click Update.
Clear all Filter IDs from the Group Filter List,	click Clear List.

4. Save your edits.

11

Tree Rollup ID

Tree Rollup IDs define a hierarchical structure of detailed data in the Oracle Financial Data Manager (FDM) database. For example, you can build a model of the various levels into which products roll up and use cost centers as the base level of the institution. You use a Tree Rollup ID to define these hierarchical structures.

With a Tree Rollup ID, you can define multiple organizational, product, account, and other hierarchies by using leaves as the lowest level of detail in the hierarchical structure. Each row in the FDM database contains a leaf value for each dimension of profitability. These leaves serve as the lowest level in constructing Tree Rollup IDs.

For more information about Tree Rollup IDs and IDs in general, see Chapter 4, "Overview of IDs".

This chapter presents the following topics:

Creating a New Tree Rollup ID

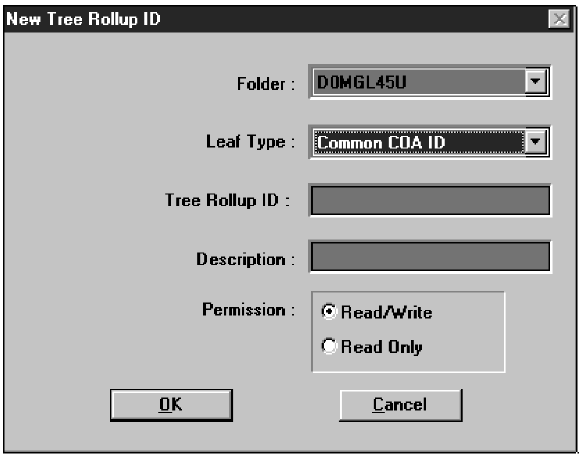
To create a Tree Rollup ID, perform the following steps:

1. From the File menu, click New.

   A list of the IDs that are available for the product you are using appears.

2. Select the Tree Rollup ID.

   The New Tree Rollup ID dialog box appears.

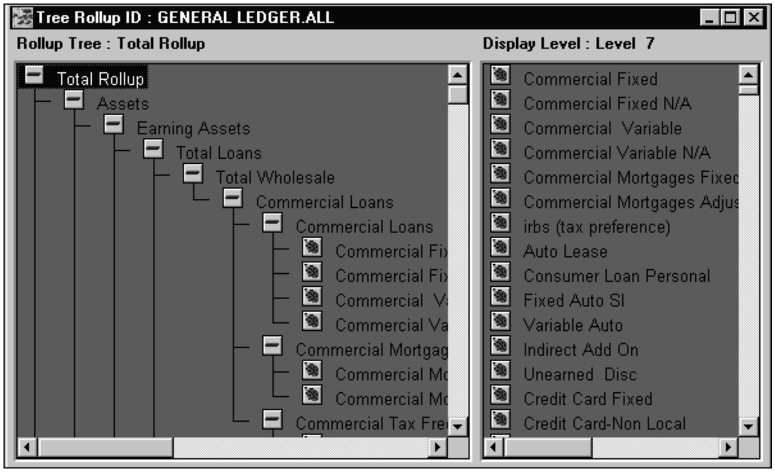


3. Select Folder and Leaf Type.
4. Type in the Tree Rollup ID and Description fields, as required.
5. Select the permission level of the ID, as required.

   Read / Write is the default.

6. Click OK to continue.

   The Tree Rollup ID window appears.



Using the Tree Rollup ID Window

The Tree Rollup ID window comprises the Rollup Tree and Display Level panes. It also provides access to the Edit, Options, and Tree menus.

Rollup Tree Pane

The Rollup Tree pane enables you to see and manipulate the tree structure at any level of the hierarchy.

Branch Expansion Indicators

Each node is represented by a description and an Indicator that indicates whether or not a branch can be expanded. The Expansion Indicator is a + (plus sign) if the branch can be expanded to display nodes at the next lower level, a **-** (minus sign) if the branch has already been expanded and all lower nodes are displayed, and blank if no lower nodes are attached.

Current Level Name

The name of the level currently selected in the Rollup Tree pane appears after *Rollup Tree* in the menu bar. The level name changes as different levels are selected.

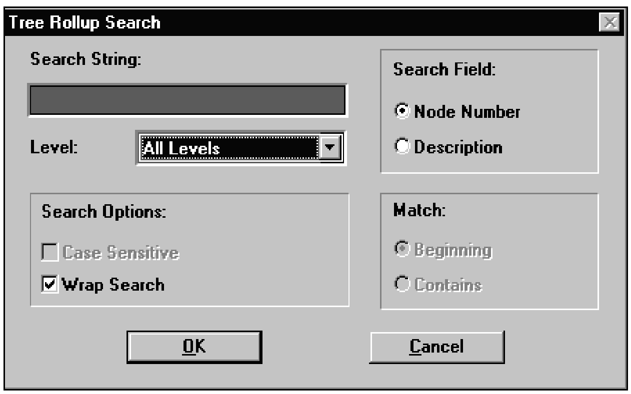
Display Level Pane

The Display Level pane enables you to see all the nodes attached to the selected node in the Rollup Tree at a specific level below the selected node. To set the level of the hierarchy that you want to display, select the Tree menu and Set Display Level.

Edit Menu

Search

If you want to locate a specific node within a Tree structure, you can use the Search option in the Edit menu. The Tree Rollup Search dialog box appears:



Field Name	Field Description
Search Field	Select the Node Number or Description for which you are searching.
Search String	Type the Node Number or Description.
Level	Select the Level of the node in the Tree. If you do not know the level, leave the default entry of All Levels.
Search Options	
■ Case Sensitive	Provides exact string matches when searching on descriptions only
■ Wrap Search	Performs the search of both node numbers and descriptions starting from the current cursor location on the Tree structure and continuing at the top of the Tree structure
Match	
■ Beginning	Searches node numbers or descriptions beginning with the string entered in the Search String field
■ Contains	Searches node numbers or descriptions containing the string entered in the Search String field

Note: Trees can show as many as 16,000 leaves and nodes at a time.

Search fails if the value you want is not within the 16,000 leaves range. The following message appears: "The tree is too large to show entirely. Collapse branches or use Search and Focus to view more." See the Tree Menu for more information about Collapse Branch.

Search Again

The Search Again option is available in the Edit menu and enables you to continue your search using the same criteria that you defined in Search.

Note: Search and Focus and Search and Focus Again are specifically for use on trees having more than 16,000 leaves.

Search and Focus

The Search and Focus option is available in the Edit menu. It uses the same dialog box as the Search option and takes you to the value itself.

Search and Focus Again

Like the Search and Focus option, the Search and Focus Again option is available in the Edit menu. It uses the same dialog box as the Search option and takes you to the value itself.

Search and Focus Again shows only the node or leaf for which you are searching. It does not show context or hierarchy. It is the only option that you can use with a large tree, that is, one containing more than 16,000 leaves.

Options Menu

You can use the Options menu to format the appearance of your Tree Rollup ID. The Options menu provides the following functions:

- Select Font
- Show Node Numbers on Left
- Show Node Numbers on Right

Select Font

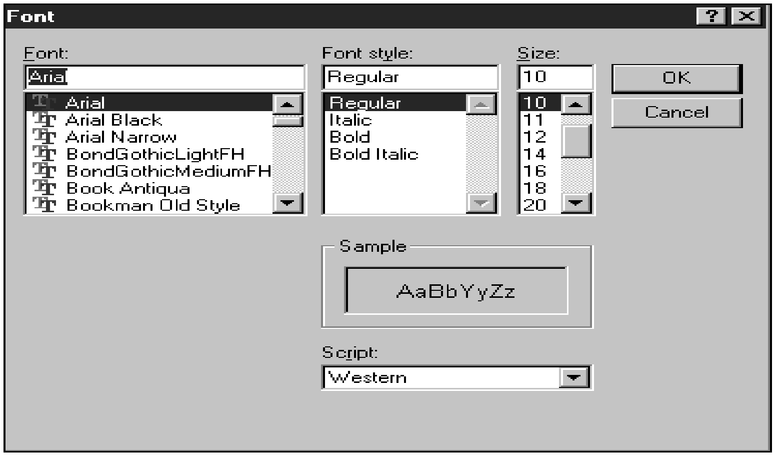
You can change the typeface in which your tree is displayed on your window. To try a new font:

1. Click Set Font.

   The default font is Font: System, Font Style: Bold, Size: 10. Each font is previewed in the Sample window at the bottom of the dialog box.

2. Select an appropriate Font, Font Style, Size, and Script, and then click OK.

   You select a smaller font to view more information within one window or a larger font to display less information per window but to view it more clearly.



Show Node Numbers on Left

You can select Show Node Numbers on Left if you are familiar with your Node Numbers and want to view them. This option displays the Node Numbers before the Node Descriptions.

This option is a toggle. To turn off the Node Numbers, select the option again.

Show Node Numbers on Right

You can select Show Node Numbers on Right if you are familiar with your Node Numbers and want to view them. This option displays the Node Numbers after the Node Descriptions.

This option is a toggle. To turn off the Node Numbers, select the option again.

Tree Menu

To change how the structure of a tree is viewed and to specify which branches are to be included in the rollup, you select Tree from the menu bar.

Note: Focus on Branch, Undo Focus, and Focus to Top are specifically for use on trees having more than 16,000 leaves.

Expand One Level

Expand One Level displays the branch and nodes attached one level below the selected node. You can also expand a branch by one level by selecting a node in the Tree Rollup and typing + (plus sign) or by double clicking the Branch Expansion Indicator when the indicator displays a + (plus sign).

Expand Branch

Expand Branch displays all levels of nodes attached under the node selected in the Tree Rollup. You can also expand a branch to show all lower levels of attached nodes by selecting a node in the Tree Rollup and typing Shift + * (plus sign and asterisk). This option is available only when the Branch Expansion Indicator displays a + (plus sign).

Expand All

Expand All displays every level of the entire Tree Rollup above and below the selected node. This option is available only when the Branch Expansion Indicator displays a + (plus sign).

Note: Expand All expands only the first 16,000 leaves and nodes. A Rollup ID can display a maximum of 16,000 leaves and nodes at a time.

Collapse Branch

Collapse Branch hides all levels of nodes attached under the selected node from the Rollup Tree pane. You can also collapse a branch by one level by selecting a node in the Rollup Tree pane and typing **-** (minus sign), or by double clicking the Branch Expansion Indicator when the indicator displays a **-** (minus sign).

Note: Focus on Branch, Undo Focus, and Focus to Top are specifically for use on trees having more than 16,000 leaves.

Focus on Branch

Focus on Branch changes the Rollup Tree pane from displaying the whole tree to displaying only the tree structure below the selected node.

Undo Focus

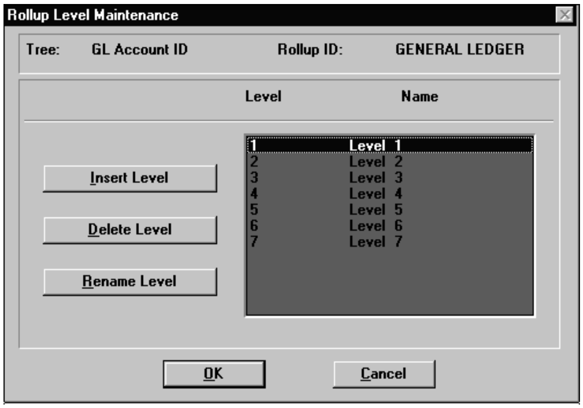
Undo Focus turns off the focus on a branch. This enables you to display all branches of the total Tree Rollup when you no longer want to focus on an individual branch.

Focus to Top

Focus to Top returns you directly to the top of the Rollup Tree pane. You do not need to return level by level.

Levels Maintenance

Levels Maintenance enables you to manipulate the level structure of a tree. Using this option, you can add, subtract, and rename levels of the Tree Rollup. Selecting this option opens the Rollup Level Maintenance dialog box.



- **Insert Level** enables you to insert a level into the hierarchy. To insert a new level, select the level in the Level Description box above which you want to add a level, and click Insert Level.

  Enter the name of the new level in the New Level Name box, and click OK. For more information, refer to the *Oracle Financial Services Installation and Configuration Guide.*

- **Delete Level** enables you to remove a level from the hierarchy. To remove a level, select the level you want to remove in the Level Description box and click Delete Level. You are prompted to confirm your decision to delete the level.

- **Rename Level** enables you to rename an already existing level. To rename a level, select the level you want to rename in the Level Description box and click Rename Level.

Enter the new name in the New Level Name box, and click OK.

When you have finished with Levels Maintenance, click OK to return to the main ID window. Click Cancel at any time to exit without changes.

Set Display Level

Set Display Level enables you to change the level displayed in the Display Level pane. Click this option to open the display level dialog box.

To change the Display Level, from the menu, select the level you want displayed.

Set Tree Level

Set Tree Level enables you to change the level that is displayed in the Rollup Tree pane. Click this option to open the display level dialog box.

Insert Node

Insert Node enables you to add an additional node to the Tree Rollup. To insert a node, select a node in the Tree Rollup under which you want to add a node, and click Insert Node. You can place new nodes anywhere in the Rollup Tree except below the leaf level or orphan node.

The Insert Node dialog box appears:

- **Parent Level** is the level of the parent node under which the new node is being added.
- **Parent Name** is the name of the parent node under which the new node is being added.
- **New Node Level** is the level at which the new node is being added.
- **New Node Name** enables you to enter the name of the new node.

  If you leave the New Node Name field blank, the new node appears in the Tree Rollup display, without a descriptive tag.
- **New Node ID** enables you to define the ID number that will be associated with the new node. This field does not require input because a unique default ID number is supplied by the system. If you decide to enter an ID number for the new node manually, however, this number must be unique.

Click OK when you have finished. Click Cancel at any time to exit without saving changes.

Rename Node

Rename Node enables you to rename or renumber an existing node. Select the desired node from the Tree Rollup, and select this option. The Rename/Renumber dialog box opens.

- **Name**

  Enter the new name in the Name box.
- **Number**

  Enter the new number in the Number box.

Click OK when you have finished. Click Cancel at any time to exit without changes.

Delete

Delete enables you to delete a node from the Tree Rollup. To delete a node, select the node you want to delete from the Tree Rollup and click Delete. The selected node and all nodes below it are deleted. The Leaves that are attached below the deleted node become orphan leaves in the orphan branch of the tree.

Copy

Copy enables you to copy an orphan leaf or group of orphan leaves from the orphan branch into another node in the Tree Rollup. To copy a single orphan leaf from the orphan branch, select the orphan leaf in the Tree Rollup or set the Display Level to Orphan Leaves and select the desired Orphan Leaf. Next, click Copy. Now select the node to which the orphan leaf is to be copied in the Tree Rollup, and use Paste to attach the leaf to the new node.

To copy several orphan nodes at one time to a single node in the Tree Rollup, hold the Shift key while selecting successive nodes. Then use Copy and Paste to attach the block of nodes at the new location.

Cut

Cut enables you to cut a leaf, a group of leaves, or a branch from the Tree Rollup.

To cut a portion of a Tree Rollup, select the leaf or node from the Tree Rollup, or set the Display Level. Next, click Cut. To reattach the cut portion of the tree to a different part of the tree, select the node to which the portion is to be copied in the Tree Rollup, and click Paste to attach it to the new node.

To cut several orphan nodes at one time, hold the Shift key while selecting successive nodes. Then click Cut if you want to reattach the nodes at a new location. Click Delete to cut the block of nodes permanently.

If a node that has leaves attached at a lower level is cut, or if leaves from the Tree Rollup are cut, the leaves are automatically placed at the end of the orphan leaves. If the leaves are reattached, or the node to which the leaves were attached are moved to a different part of the tree, the leaves that were placed at the end of the orphan branch are automatically moved.

Paste

Paste enables you to attach cut or copied orphan leaves, leaves, nodes, and branches to new nodes in the Tree Rollup. After you Cut or Copy, select the node to which you want to attach the data, and click Paste.

You can only perform a Paste after a Cut or Copy action has been performed. Paste can only be performed on the leaves or nodes copied or cut in the immediately previous cut or copy. If cuts are performed without a paste the node information from the first cut is lost and the leaves are placed at the end of the orphan leaves.

You can perform a Paste only after a Cut or Copy has been performed. Paste can be performed only on the leaves or nodes copied or cut in the immediately previous cut or copy. If you cut without a paste, the node information from the first cut is lost and the leaves are placed at the end of the orphan leaves.

Node Up

Node Up enables you to move a node up the list of nodes if more than one node is attached to a parent node. In order move a node above another node in the Tree Rollup window, select the node to move and click Node Up.

Node Down

Node Down enables you to move a node down the list of nodes if more than one node is attached to a parent node. In order move a node below another node in the Tree Rollup window, select the node to move and click Node Down.

Transform Rollup

Transform Rollup transforms the current Tree Rollup ID into a hierarchy table within the Oracle Financial Data Manager Reporting Data Mart. The new hierarchy table has the same name as the Tree Rollup ID.This table makes information available to the Reporting Data Mart user and for reporting purposes through Oracle Discoverer and other reporting tools. For more information, refer to Chapter 14, "Transformation ID".

Save the ID

Save the ID by clicking the Save icon on the horizontal toolbar, or choose Save from the File menu.

Editing a Tree Rollup ID

To open an existing Tree Rollup ID, perform the flowering steps:

1. From the File menu, click Open.

   A list of the IDs that are available for the product you are using appears.
2. Select Tree Rollup ID.

   The Select Tree Rollup ID dialog box appears.
3. Select Folder, Leaf Type, and Tree Rollup ID, as required.
4. Click OK.

   The Tree Rollup ID window appears.
5. Click in the pane that you want to search, either Rollup Tree or Display Level.

   If you select the Rollup Tree pane, you see the leaves in the context of the tree. If you select the Display Level pane, you see only the leaves.
6. From the Edit menu, click Search to search for the Tree Rollup ID you want to edit.

   The Tree Rollup Search dialog box appears.

   Note: Trees can show as many as 16,000 leaves and nodes at a time.

7. Select either Node Number or Description in Search Field.

8. Type the Search String, such as `Corporate`.
9. Select Level, Search Options, and Match, as appropriate.
10. Click OK.

    The Tree Rollup ID appears if your search is within 16,000 leaves. Search fails if the value you want is not within the 16,000 leaves range. The following message appears: “The tree is too large to show entirely. Collapse branches or use Search and Focus to view more.”

11. Click on the Tree Rollup ID.
12. With the Tree Rollup ID selected, copy or paste it as appropriate.

    For example, you want to cut one leaf, Regional Corporate, from Wholesale and move it to Branch Administration.

13. When your cutting or pasting is complete, select File and Save to save your Tree Rollup ID, or select File and Save As to save your edited Tree Rollup ID under a new name.

Example: Building a Product Tree

In this example, you use the Tree Rollup ID to build a product tree with five hierarchy levels.The tree structure is as follows:

Levels	Product Categories
LEVEL I	
Business Type	Personal
LEVEL II	
Market Family	Checking, Savings, Retirement
LEVEL III	
Market Type	Interest, Non-Int., Liquid, Time, IRAs
LEVEL IV	
Market Product	Regular, Money Market
LEVEL V	Star Checking, Super Checking,
Product Description	Money Mkt I, Money Mkt II

You must set the levels of the tree before you can build it.

Set the Levels of a Tree

To set the levels of a tree, perform the following steps:

1. Click New on the horizontal toolbar.

   A list of the IDs that are available for the product you are using appears.

2. Select the Tree Rollup ID.

   The New Tree Rollup ID dialog box appears.

3. Select All for Folder.

   Now everyone in the folder has access.

4. Select Common COA ID for Leaf Type.

   This is the Leaf Type that contains the information for products.

5. Type `PRODUCT` as the Tree Rollup ID name.
6. In Description, type `This Product Tree is for Deposits Only.`
7. Select Read / Write for Security.
8. Click OK.

   The Tree Rollup ID window appears.

9. Select Levels Maintenance from the Tree menu.

   The Levels Maintenance dialog box appears: The default names are the level numbers.Set the tree levels to define the structure of the tree.

10. To change the numbers to names, select each level, click Rename Level, and type the new name.

    The default number of levels for a new tree is set at five. You can add or subtract levels by clicking the Insert Level or Delete Level buttons. For this example, the number of levels is left at five. The maximum number of levels available is 15.

11. When you have finished naming the levels, click OK.
12. After setting the levels, build the tree.

Build a Tree

To build a tree, perform the following steps:

1. Highlight the parent branch into which you are going to insert the new branch.
2. Insert a new branch.

   You insert the first level of the sample, Business Type, by highlighting Total Rollup.
3. Click Insert Level (or select Insert Node from Tree menu).

   The Insert Node dialog box appears.
4. In the New Node Name box, type the description for this example, `Personal`, and click OK.
5. Repeat procedures 1 and 2 to insert all Level Two, Three, and Four branches.
6. The lowest level, Level Five, is the leaf level.

   For new trees, Leaves are under the Orphan branch.
7. Attach the leaves to each branch by one of two methods:
   - Use Copy and Paste from the Tree menu to copy the leaf from the Orphan branch to the target branch.
   - Use the mouse to drag the selected leaf from the Orphan branch and drop it on the appropriate branch.

   After the leaves are attached, the Tree Rollup ID window appears.
8. Save the new Tree Rollup ID.
9. Click the Exit icon at on the horizontal toolbar to exit the new ID.

12

Tree Filter ID

In Oracle Financial Services (OFS) applications, a Tree Filter ID narrows the focus of the tree hierarchy for processing and reporting. For example, if you want to report on only the products that roll up to the commercial loans portion of a product tree, you identify them with the Tree Filter ID.

A Tree Filter ID must have a predefined Tree Rollup ID as part of its definition, and it must be in the same group as the Tree Rollup ID. Tree Filter IDs are available only for data correction, transfer pricing, performance analyzer processing, and risk management processing.

For more information about Tree Filter IDs and IDs in general, see Chapter 4, "Overview of IDs".

This chapter presents the following topics:

- Creating a New Tree Filter ID
- Using the Tree Filter ID Window
- Editing a Tree Filter ID
- Example: Creating and Defining a New Tree Filter ID

Creating a New Tree Filter ID

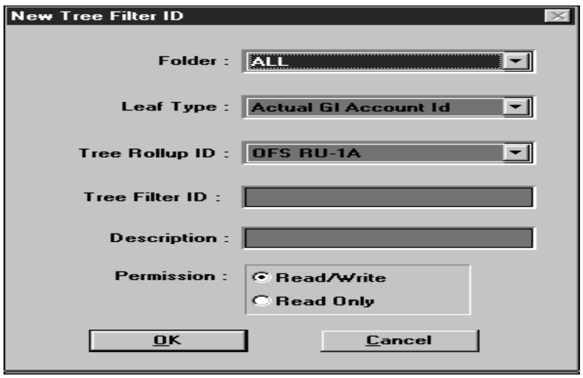
To create a new Tree Filter ID, perform the following steps:

1. From the File menu, click New.

   A list of the IDs that are available for the product you are using appears.

2. Select Tree Filter ID.

   The New Tree Filter ID dialog box appears.

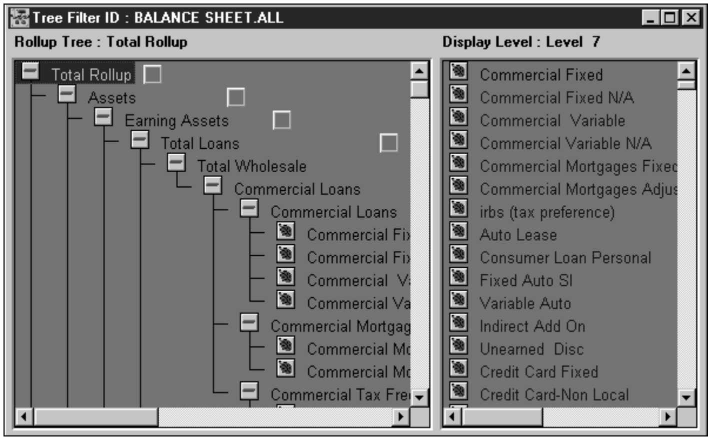


3. Select Folder, Leaf Type, and Tree Rollup ID.
4. Type in the Tree Filter ID and Description fields, as required.
5. Select the permission level of the ID, as required.

   Read / Write is the default.

6. Click OK to continue.

   The Tree Filter ID window appears.



Using the Tree Filter ID Window

Like the Tree Rollup ID window, the Tree Filter ID window comprises the Rollup Tree and Display Level panes. Also like the Tree Rollup ID window, it provides access to the Edit, Options, and Tree menus. Unlike the Tree Rollup ID window, the Tree Filter ID window has red and green flags that indicate nodes excluded from or included in the definition of the Tree Filter ID.

Rollup Tree Pane

The Rollup Tree pane enables you to see and manipulate the tree structure at any level of the hierarchy.

Branch Expansion Indicators

Each node is represented by a description and an Indicator that indicates whether or not a branch can be expanded. The Expansion Indicator is a + (plus sign) if the branch can be expanded to display nodes at the next lower level, a - (minus sign) if the branch has already been expanded and all lower nodes are displayed. It is blank if no lower nodes are attached.

Node Inclusion Indicator

Node Inclusion Indicators appear as green or red flags. If any nodes at lower levels of the branch are included in the Tree Filter ID, the Node Inclusion Indicator is green. If no nodes at lower levels of the branch are included in the Tree Filter, it is red.

Current Level Name

The name of the level currently selected in the Rollup Tree pane appears after *Rollup Tree* in the menu bar. The level name changes as different levels are selected.

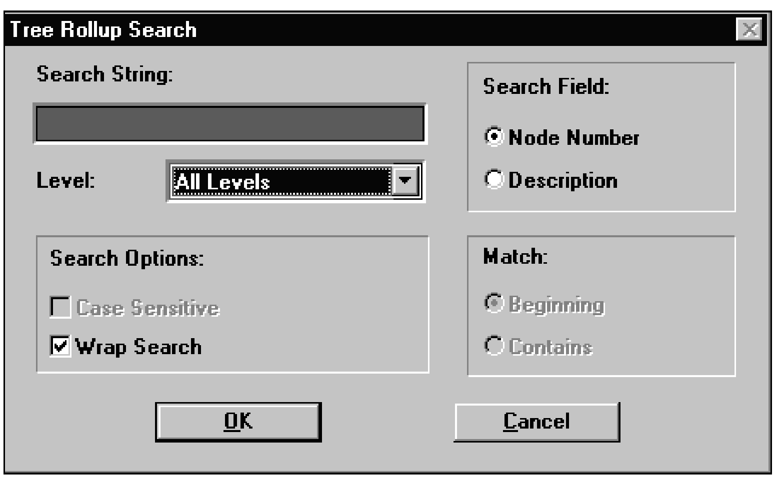
Display Level Pane

The Display Level pane enables you to see all the nodes attached to the selected node in the Rollup Tree at a specific level below the selected node. To set the level of the hierarchy that you want to display, select the Tree menu and Set Display Level.

Edit Menu

Search

If you want to locate a specific node within a Tree structure, you can use the Search option in the Edit menu. The Tree Rollup Search dialog box appears:



Field Name	Field Description
Search Field	Select the Node Number or Description for which you are searching.
Search String	Type the Node Number or Description.
Level	Select the Level of the node in the Tree. If you do not know the level, leave the default entry of All Levels.
Search Options	
■ Case Sensitive	This option provides exact string matches when searching on descriptions only.
■ Wrap Search	This option performs the search of both node numbers and descriptions starting from the current cursor location on the Tree structure and continuing at the top of the Tree structure.
Match	
■ Beginning	This option searches node numbers or descriptions beginning with the string entered in the Search String field.
■ Contains	This option searches node numbers or descriptions containing the string entered in the Search String field.

Note: Trees can show as many as 16,000 leaves and nodes at a time.

Search fails if the value you want is not within the 16,000 leaves range. The following message appears: “The tree is too large to show entirely. Collapse branches or use Search and Focus to view more.” See the Tree Menu for more information about Collapse Branch.

Search Again

The Search Again option is available in the Edit menu and enables you to continue your search using the same criteria that you defined in Search.

Note: Search and Focus and Search and Focus Again are specifically for use on trees having more than 16,000 leaves.

Search and Focus

The Search and Focus option is available in the Edit menu. It uses the same dialog box as the Search option and takes you to the value itself.

Search and Focus Again

Like the Search and Focus option, the Search and Focus Again option is available in the Edit menu. It uses the same dialog box as the Search option and takes you to the value itself.

Search and Focus Again shows only the node or leaf for which you are searching. It does not show context or hierarchy. It is the only option that you can use with a large tree, that is, one containing more than 16,000 leaves.

Options Menu

You can use the Options menu to format the appearance of your Tree Rollup ID. The Options menu provides the following functions:

- Select Font
- Show Node Numbers on Left
- Show Node Numbers on Right
- Show Include Attribute

Select Font

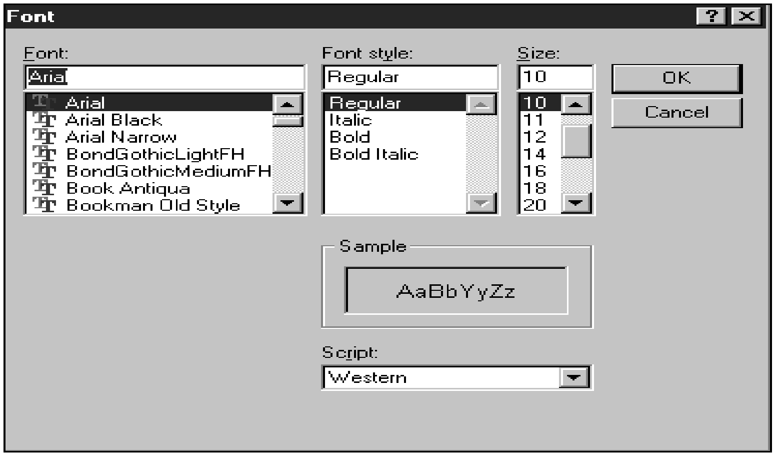
You can change the typeface in which your tree is displayed on your window. To try a new font:

1. Click Set Font.

   The default font is Font: System, Font Style: Bold, Size: 10. Each font is previewed in the Sample window at the bottom of the dialog box.

2. Select an appropriate Font, Font Style, Size, and Script, and then click OK.

   You select a smaller font to view more information within one window or a larger font to display less information per window but to view it more clearly.



Show Node Numbers on Left

You can select Show Node Numbers on Left if you are familiar with your Node Numbers and want to view them. This option displays the Node Numbers before the Node Descriptions.

This option is a toggle. To turn off the Node Numbers, select the option again.

Show Node Numbers on Right

You can select Show Node Numbers on Right if you are familiar with your Node Numbers and want to view them. This option displays the Node Numbers after the Node Descriptions.

This option is a toggle. To turn off the Node Numbers, select the option again.

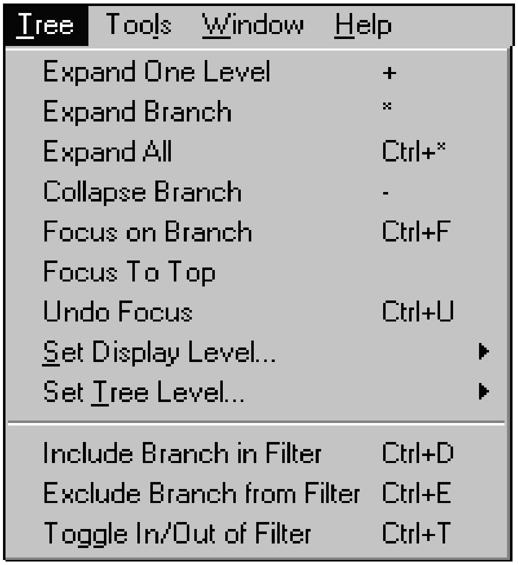
Show Include Attribute

You can select Show Include Attribute to activate the Node Inclusion Indicators and display the filter status of a node as you navigate through the hierarchy. A red flag indicates that the node has not been filtered. A green flag indicates that the node has been filtered.

Show Include Attribute is the default.

Tree Menu

To change how the tree structure is viewed and to specify which branches are to be included in the filter, select an option from the Tree menu.



Expand One Level

Expand One Level displays the branch and nodes attached one level below the selected node. You can also expand a branch by one level by selecting a node in the Tree Rollup and typing + (plus sign) or by double clicking the Branch Expansion Indicator when the indicator displays a + (plus sign).

Expand Branch

Expand Branch displays all levels of nodes attached under the node selected in the Tree Rollup. You can also expand a branch to show all lower levels of attached nodes by selecting a node in the Tree Rollup and typing Shift + * (plus sign and asterisk). This option is available only when the Branch Expansion Indicator displays a + (plus sign).

Expand All

Expand All displays every level of the entire Tree Rollup above and below the selected node. This option is available only when the Branch Expansion Indicator displays a + (plus sign).

Note: Expand All expands only the first 16,000 leaves and nodes. A Rollup ID can display a maximum of 16,000 leaves and nodes at a time.

Collapse Branch

Collapse Branch hides all levels of nodes attached under the selected node from the Rollup Tree pane. You can also collapse a branch by one level by selecting a node in the Tree Rollup and typing **-** (minus sign), or by double clicking the Branch Expansion Indicator when the indicator displays a **-** (minus sign).

Note: Focus on Branch, Focus to Top, and Undo Focus are specifically for use on trees having more than 16,000 leaves.

Focus on Branch

Focus on Branch changes the Rollup Tree pane from displaying the whole tree to displaying only the tree structure below the selected node.

Focus to Top

Focus to Top returns you directly to the top of the Rollup Tree pane. You do not need to return level by level.

Undo Focus

Undo Focus turns off the focus on a branch. This enables you to display all branches of the total Tree Rollup when you no longer want to focus on an individual branch.

Set Display Level

Set Display Level enables you to change the level that is displayed in the Display Level pane. Click this option to open the display level dialog box:

To change the Display Level, select the level you want to be displayed from the menu.

Set Tree Level

Set Tree Level enables you to change the level that is displayed in the Rollup Tree pane. Click this option to open the display level dialog box.

Include Branch in Filter

Include Branch in Filter selects an entire branch for inclusion in the Tree Filter ID window. To include an entire branch of a Rollup in the Tree Filter ID window, select the node at the top of the desired branch, and then select Include Branch in Filter.

If a node is included in the Tree Filter, then the Node Description is displayed in bold print. Otherwise, it is displayed in gray print.

Exclude Branch from Filter

Exclude Branch from Filter selects an entire branch to be excluded from the Tree Filter ID window if it has already been selected. To exclude an entire branch of a Rollup in the Tree Filter ID window, select the node at the top of the desired branch, and then select Exclude Branch from Filter.

Toggle In/Out of Filter

Toggle In / Out of Filter enables you to select or deselect a single node from an entire branch without changing the selection status of any other nodes in the branch. To include an individual node in a branch that is excluded or to exclude an individual node in a branch that is included, select the node to be toggled, and then select Toggle In / Out of Filter.

Save the ID

Save the ID by clicking the Save icon on the horizontal toolbar, or choose Save from the File menu.

Editing a Tree Filter ID

To edit a Tree Filter ID, complete the following steps:

1. From the File menu, click Open.

   A list of the IDs that are available for the product you are using appears.

2. Select Tree Filter ID.

   The Select Tree Filter ID dialog box appears.

3. Select Folder, Leaf Type, Tree Rollup, and Tree Filter IDs, as required.
4. Click OK.

   The Tree Filter ID window appears.

5. Click in the pane that you want to search, either Rollup Tree or Display Level.
6. From the Edit menu, click Search to search for the Tree Filter ID that you want to edit.

   The Tree Rollup Search dialog box appears.

   Note: Trees can show as many as 16,000 leaves and nodes at a time.

   ---

7. Select either Node Number or Description in Search Field.
8. Type the Search String.
9. Select Level, Search Options, and Match, as appropriate.
10. Click OK.

    The Tree Filter ID appears if your search is within 16,000 leaves. Search fails if the value you want is not within the 16,000 leaves range. The following message appears: “The tree is too large to show entirely. Collapse branches or use Search and Focus to view more.”

11. Click on the Tree Filter ID.
12. With the Tree Filter ID selected, edit it as appropriate.
13. When your editing is complete, select File and Save to save your Tree Filter ID, or select File and Save As to save your edited Tree Filter ID under a new name.

Example: Creating and Defining a New Tree Filter ID

This example explains how to create and define a new Tree Filter ID named Checking. The example uses the Rollup Tree ID named Product that was created in the Chapter 11, "Tree Rollup ID" chapter. To create the Tree Filter ID, perform the following steps:

1. From the File menu, select New.

   A list of the IDs that are available for the product you are using appears.

2. Select Tree Filter ID.

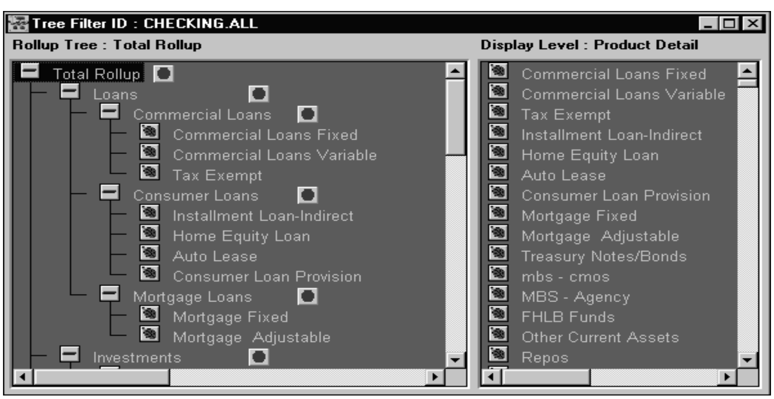
   The New Tree Filter ID dialog box appears.

3. Select All for Folder.

   Now everyone in the folder has access.

4. Select `Common COA ID` for Leaf Type.
5. Select Product for Tree Rollup ID.
6. Type **`CHECKING`** in the Tree Filter ID name box.
7. Type **`This filter is for checking profitability analysis`** in the Description box.
8. Select Read / Write for Security.
9. Click OK.

   The Tree Filter ID window appears.



10. Click Checking at level 2.

11. Select Include Branch in Filter from the Tree menu.

The Node(s) Inclusion Indicators turn green for all the Level 2 Checking branches.

12. Save the Tree Filter ID.

13 Formula ID

A Formula ID is a user-defined tool that supplements other IDs and enables you to further and more flexibly manipulate data. Formula IDs have three different uses:

- To specify a calculated column that the Oracle Financial Services (OFS) application derives from other columns in the database
- To calculate assignments in data correction
- To create calculated conditions in data and relationship filters.

For example, you want to calculate a weighted average rate that requires a calculation involving total net balance and the current rate on each individual account. You can use a Formula ID to define this calculation.

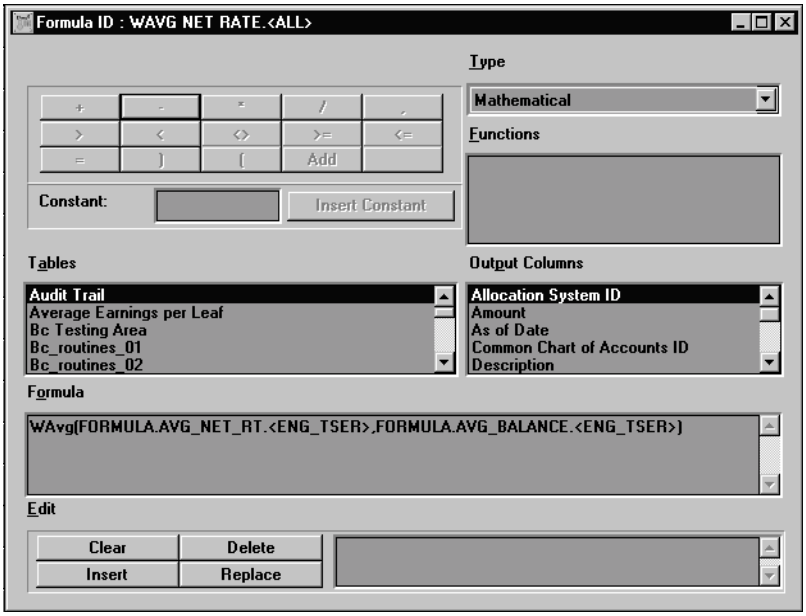
The OFS applications handle a Formula ID like any database column. The applications display all Formula IDs under the Formula selection in the instrument table lists, however.

This chapter discusses the following topics:

- Using the Formula ID Window
- Creating a Formula ID
- Editing a Formula ID
- Reviewing Formula ID Examples

Using the Formula ID Window

The Formula ID window comprises elements that you use to build formulas.



The formula elements are:

- Operators and Operands
- Function Types and Functions
- Constants
- Tables and Output Columns

Only those formula elements that you can add logically at the next point of a formula are available for use. The OFS application disables or hides the formula elements that are not applicable.

You use the formula elements as you need them. For example, you may need a formula to calculate the weighted average of your commercial loans. You must first select Mathematical as the function type and WAvg as the function before you select the columns under the Commercial Loans table. However, you can build a formula that does not require a function. It may need only operators and operands.

The following sections describe the formula elements and provide basic information on how to use them.

Operators and Operands

Operators and Operands are displayed as buttons across the top of the Formula ID window.

The operands available are left parenthesis, right parenthesis, and comma. Parentheses group segments of a formula to make logical sense. The comma separates statements of a function.

The mathematical operators available are:

- +
- -
- *
- /
- =
- >
- <
- <>
- >=
- <=

You can use these operators to apply mathematical operators to the formula.

Add enables you to add the currently highlighted database column or operator to build the formula.

Function Types and Functions

You select the type of function for your formula from the Type list. The choices are:

- Mathematical Functions
- Date Functions
- String Functions
- Other Functions

The type of function you select determines the choices available in the Function box. These unique functions of the Formula ID enable you to perform various operations on the data. The following tables list each available function. Detail on the operations of each function follows the table in which it appears.

Mathematical Functions

When you select Mathematical as the function type, you can use the following 11 functions from the Functions box:

Function	Notation
Absolute Value	ABS()
Ceiling	CEILING()
Greatest	GREATEST(column or expression, column or expression)
Least	LEAST(column or expression, column or expression)
Maximum	MAX()
Minimum	MIN()
Natural Log	LN(number)
Power	POWER(coeffecient, exponent)
Round	ROUND (number, precision)
Sum	SUM()
Weighted Average	WAvg (column being averaged, weight column)

The following descriptions provide detailed information about the mathematical functions, including examples.

Absolute Value: Returns the positive value of the database column

Example: ABS (-3.5) = 3.5.

ABS function syntax:

{ABS(} followed by {EXPR1 without any embedded or outermost left-right parentheses pair} followed by {)}

For example, ABS(F), ABS(F + C), ABS(F + C * R + F) are possible. However, ABS((F + C + R)), ABS((F + (MAX * CEILING))) are not possible.

Ceiling: Rounds a value to the next highest integer

Example: 3.1 becomes 4.0, 3.0 stays the same

Syntax: Ceiling(column or expression)

Greatest: Returns the greater of 2 numbers, formulas, or columns

Syntax: Greatest(column or expression, column, or expression)

Least: Returns the lesser of 2 numbers, formulas, or columns

Syntax: Least(column or expression, column or expression)

Maximum: Returns the maximum value of a -database column

Syntax: Max(Column)

Minimum: Returns the minimum value of a -database column

Syntax: Min(Column)

Note : You cannot use the Maximum and Minimum functions as calculated columns or in Data Correction Rules. The Maximum, Minimum, Sum, and Weighted Average functions are multi-row formulas. They use multiple rows in calculating the results.

Natural Log: Returns the natural logarithm of a number

Natural logarithms are based on the constant e (2.71828182845904).

Syntax: LN(number) where *number* is the positive real number for which you want the natural logarithm

Examples:

LN(86) equals 4.454347

LN(2.7182818) equals 1

Power: Raises one value to the power of a second

Syntax: Power(x, y) returns x raised to the power of y

POWER function syntax:

{POWER(} followed by {EXPR1 without any embedded or outermost left-right parentheses pair followed by {,} followed by {EXPR1 without any embedded or outermost left-right parentheses pair} followed by {)}

<u>Valid examples</u>:

POWER(F, R)

POWER(F + C * R, F / R)

<u>Invalid examples</u>:

POWER((F/R), F + R)

POWER((F + C), (C * R))

POWER(F + POWER, R)

POWER(MAX, C)

Round: Rounds a value to a number of decimal places

Syntax: Round(x, n) returns x rounded to n decimal places

Sum: Sums the total value of a database column. Sum is a multi-row function, in contrast to +, which adds 2 or more values in a given row (not column)

Syntax: Sum(Column).

Weighted Average: Takes a weighted average of one database column by a second column

Syntax: WAvg(Column A, Column B)

Example:

WAvg(DEPOSITS.CUR_NET_RATE,DEPOSITS.CUR_BOOK_BAL)

WAvg cannot appear in any expression.

If you have two formulas called F1 and F2, both of which are WAvg functions, then you can form a third formula F3 as F1 + F2. If F3 is chosen as a calculated column, then an error message appears and the SQL code is not generated for that column. This is similar for nested WAvg functions if F3 is WAvg and it has F1 or F2 or both as its parameters.

Date Functions

Four functions are available when you select Date as the function type:

Function	Notation
Build Date	BUILDDATE(CCYY,MM,DD)
Go Month	GOMONTH(date,number of months)
Month	MONTH(number)
Year	YEAR(date)

The following descriptions provide detailed information about the date functions, including examples.

Build Date: Requires three parameters, (CCYY,MM,DD) (century and year, month, day). It returns a valid data and enables you to build a date from components.

CAUTION: If the parameters are entered incorrectly, the date is invalid.

Example: BuildDate(95,11,30) is invalid (invalid century). BuildDate(1995,11,30) is valid.

Go Month: Advances a date by *x* number of months.

Syntax: GOMONTH(Date column, Number of months to advance)

Example:

GOMONTH(DEPOSITS.ORIGINATION_DATE,DEPOSITS.ORG_TERM)

Go Month does not know the calendar. For example, it cannot predict the last day of a month. Typical functionality is illustrated in the following table:

Date Column	# of Months	GOMONTH	Comment
1/31/94	1	2/28/94	Because 2/31/94 does not exist
1/15/94	2	3/15/94	Exactly 2 months:15th to 15th
2/28/94	3	5/28/94	Goes 28th to 28th: does not know that 31st is the end of May
6/30/94	-1	5/30/94	Goes back 30th to 30th: does not know that 31st is end of May

GOMONTH function syntax:

{GOMONTH(} followed by {F | C | R} followed by {,} followed by {EXPR1 without any embedded or outermost left-right parentheses pair} followed by {)}

Valid examples:

GOMONTH(F, F + R + C)

GOMONTH(F, R)

Invalid examples:

GOMONTH(F + (R + C), MAX)

GOMONTH((F * C), F)

GOMONTH followed by {+ | -} followed by {F | C | R} is the only expression possible with GOMONTH.

Month: Month(x) returns the month in x, where x is a numbered month.

Month(*Column*) returns the month in the column, where the column is a date column.

Examples:

Month(9) returns September.

Month(Origination Date) returns the month of the origination date.

Year: Year(x) returns the data for year x.

Year(*Column*) returns the year in the column, where the column is a date column.

Example: Year(Origination Date) returns the year of the origination date.

String Functions

Only one function is available when you select String as the function type:

Function	Notation	Description
Trim All	ALLTRIM()	Trims leading and following spaces, enabling the software to recognize numbers (entered in All Trim) as a numeric value, which can then be used in calculating

Other Functions

Two functions are available when you select Other as the function type:

Function	Notation
If statement	IF(logical_test, value_if_true, value_if_false)
Lookup	LOOKUP(original table column,lookup table column, return column)

The following descriptions provide detailed information about the other functions, including examples.

If: Use *If, then* logic in a formula. The syntax for the If function is:

If(Condition, Value if True, Value if False).

Example:

If(LEDGER_STAT.Financial= 110, LEDGER_STAT.Month 1 Entry,0)

IF function syntax:

{IF(} followed by EXPR2 followed by {> | < | <> | = | >= | <=} followed by EXPR2 followed by {{,} followed by EXPR followed by),} followed by EXPR}n followed by {)} where n = 1, 2, 3, ...

The IF function should always have odd number of parameters separated by commas. The first parameter is an expression followed by a relational operator, which is in turn followed by an expression.

For example, IF(((MAX + SUM) >= 30), F, POWER) is valid.

Note: Avoid embedding multiple individual formulas in subsequent formulas. This can create an invalid formula.

Lookup: Enables you to assign values equal to values in another table for data correction.

LOOKUP function syntax:

Lookup(O1,L1,O2,L2,...On,Ln,R) where

O=Column from Original table

L=Column from Lookup table

R=Column to be Returned

So the previous statement would read:

where O1=L1 and O2=L2... Returned value R

LOOKUP function should always have an odd number of parameters separated by commas and with a minimum of 3 parameters.

<u>Valid examples</u>:

LOOKUP(F, R, R)

LOOKUP(F, R, F, F, F)

<u>Invalid examples</u>:

LOOKUP(F)

LOOKUP(F, R)

LOOKUP(F + R, (F + R), MAX)

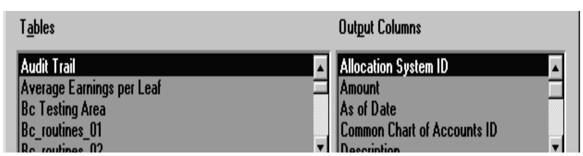
Note: Lookup is used exclusively for data correction.

Constants

The Constant box enables you to apply a constant value to the formula. You type a value into the box and click Insert Constant to insert the value of a constant into the formula.

Tables and Output Columns

The Tables box enables you to select the table that holds a specific database column you want to include in a formula. Once you select a table, the application displays the columns associated with that table in the Output Columns box.



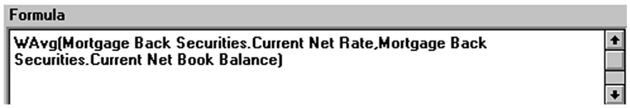
You can use the Formula table to access any previously defined formulas.

Output Columns (Data Elements)

To use a column (a data element) in the formula, select that column and then click Add in the Operand box, or double-click on the column name to add it to the formula.

Formula Box

The Formula box displays the formula as you create it.



Edit Options

The Edit box and option buttons provide full control of formula editing.

Clear

The Clear option clears the entire formula box of all logic.

Delete

The Delete option deletes the parts of the formula that you select. Select the formula element that you want to delete and click Delete.

> **Note:** You cannot delete any portion of a formula if the deletion results in leaving an invalid formula.

Insert

The Insert option inserts operators, operands, or constants into the formula at the cursor position. To insert an element, complete the following steps:

1. Place your cursor in the formula where you want to insert an element, and click Insert.
2. Select the item you want to insert. Double-click a column or function, or click the operator button and then click Add.

   Your insertion element appears in the Edit box for verification.
3. Press Insert again.

   A warning dialog appears.
4. Click Yes to insert, or click No to reselect your insertion element. You can click Cancel to cancel the operation.

Note: You cannot insert expressions, operators, or constants that result in leaving an invalid formula.

Replace

The Replace option replaces a formula element that you select. To replace a formula element, complete the following steps:

1. Select the formula element that you want to replace, and click Replace.
2. Select the replacement item. Double-click a column or function, or click the operator button and then click Add.

   Your replacement element appears in the Edit box for verification.
3. Click Replace again.

   A warning dialog appears.
4. Click Yes to confirm the replacement, or click No, and reselect your replacement element.
5. Click Cancel to cancel the operation.

Note: You cannot replace a selected element that results in leaving an invalid formula.

Null Values in the Database

A null value in a column in the database is a column that has no data. You should take all necessary steps to avoid having null columns in the database for the following reasons:

- Any null value accessed in a formula results in a null value. For example:

Column X	Column Y
150	null

If we define a formula as X + Y, the answer is 150 + null = null. All operations (such as +, -, /, *, ^) are handled in this same manner.

- You cannot define a filter to recognize null values. The Data Filter ID assumes that the database does not have any null values. Consequently, you cannot define a Data Filter ID to isolate all rows with a null value in a given column.

> **Note:** Null <> 0, so filtering on 0 does not return rows with null values.

You can identify null values by using an SQL statement to select all rows WHERE <Column> IS NULL.

Formula ID Creation Hints

The Formula ID is a flexible method for creating formulas. Consequently, you have the potential to define a formula that does not make sense. Consider the following recommendations when creating a Formula ID:

- Do not use a column out of context. For example, do not use a character field in a numeric calculation.
- Each formula should contain only columns from the same database table (except for certain Ledger_Stat table columns).
- An implied filter exists when creating a Formula ID with columns in the LEDGER_STAT table. LEDGER_STAT table formulas include if/else decode logic based on the Configuration ID as of date.

  Example: If the formula is Month_01 - 100, a filter includes Year_S equal to the as of date/year. This formula works as follows:

  Year_S = 1994, then Month_01 - 100, else 0.

Creating a Formula ID

Use the following instructions to create and define a Formula ID. See Chapter 4, "Overview of IDs" for further explanation about ID creation and maintenance.

To create a Formula ID, perform the following steps:

1. From the File menu, click New.

   A list of the IDs that are available for the product you are using appears.

2. Select the Formula ID.

   The New Formula ID dialog box appears.

3. Type the name for the ID in the Formula ID field.

   Note: Formula ID Names cannot contain spaces or special characters, such as dash (-) or slash (/).

4. Select Folder.
5. Type in the Formula ID and Description fields, as required.
6. Select the permission level of the ID, as required.

   Read / Write is the default.

7. Click OK to continue.

   The Formula ID window appears.

8. Build the formula using the formula elements.

   See "Using the Formula ID Window" in this chapter for information on using the formula elements in the Formula ID window.

9. Save the ID.

Editing a Formula ID

To edit a Formula ID, perform the following steps:

1. From the File menu, click Open.

   A list of the IDs that are available for the product you are using appears.
2. Select the Formula ID.

   The Select Formula ID dialog box appears.
3. Type the name of or select the ID and Formula ID that you want to edit.
4. Click OK to continue.

   The Formula ID window appears.
5. Make the editorial changes.
6. When you finish the editing, save the ID.

Reviewing Formula ID Examples

The following four examples demonstrate how you can use the Formula ID to achieve various results.

Example 1 Use *If, Then* Logic to Determine Average Daily Balance

The following steps create a Formula ID that calculates the Average Daily Balance for January from the Ledger Stat table.

1. From the File menu, click New.

   A list of the IDs that are available for the product you are using appears.

2. Select the Formula ID.

   The New Formula ID dialog box appears.

3. Select All from the Folder.
4. Type **`JAN_ADB`** in the Formula ID field.
5. Type **`Avg Dly Bal-Jan`** in the Description field.
6. Click Read / Write.
7. Click OK to continue.

   The Formula ID window appears.

8. Select Ledger Stat from the Tables box.
9. Select Other from the Type box.
10. Double-click If from the Functions box.
11. Select Financial Element ID from the Output Columns box and click Add in the Operand box.
12. Click the **`=`** (equal operator) from the Operand box.
13. Type **`140`** in the Constant box, and then click Insert Constant.
14. Click the **`,`** (comma operand) from the Operand box.
15. Select Month_01 from the Output Columns box and click Add in the Operand box.
16. Click the **`,`** (comma operand) from the Operand box.
17. Type **`0`** in the Constant box, and then click Insert Constant.

18. Click the **)** (close parenthesis operand) from the Operand box.

Example 2 Current Net Book Balance

The following steps create a Formula ID that calculates the current net book balance for Commercial Loans. This ID multiplies the current gross book balance by the percent owned (Gross Book Balance * Percent Owned):

> **Note:** The Oracle Financial Data Manager (FDM) database carries the percent sold, so the percent owned must be calculated.

1. From the File menu, click New.

   A list of the IDs that are available for the product you are using appears.
2. Select the Formula ID.

   The New Formula ID dialog box appears.
3. Select All from the Folder.
4. Type **`CL_NETBKBAL`** in the Formula ID field.
5. Type **`CL Net Book Bal`** in the Description field.
6. Click Read / Write.
7. Click OK to continue.

   The Formula ID window appears.
8. Select Commercial Loan from the Tables box.
9. Select Current Gross Book Balance from the Output Columns box, and then click Add in the Operand box.
10. Click the ***** (multiply operator) from the Operand box.
11. Click the **(** (open parenthesis operand) from the Operand box.
12. Type **`100`** in the Constant box and then click Insert Constant.
13. Click the **-** (minus operator) from the Operand box.
14. Select Percent Sold from the Output Columns box, and then click Add in the Operand box.
15. Click the **)** (close parenthesis operand) from the Operand box.
16. Click the **/** (divide operator) from the Operand box.

17. Type **`100`** in the Constant box, and then click Insert Constant.
18. Click the **`)`** (close parenthesis operand) from the Operand box.

Example 3 Average Current Net Book Balance

The following steps create a Formula ID that calculates the average current net book balance for each row in a Commercial Loan Stratification Report:

1. From the File menu, click New.

   A list of the IDs that are available for the product you are using appears.
2. Select the Formula ID.

   The New Formula ID dialog box appears.
3. Select All from the Folder.
4. Type **`CL_AVGNETBK`** in the Formula ID field.
5. Type **`CL Avg Net Bk Bal`** in the Description field.
6. Click Read / Write.
7. Click OK to continue.

   The Formula ID window appears.
8. Select Commercial Loan from the Tables box.
9. Select Mathematical from the Type box.
10. Double-click WAvg in the Functions box.
11. Select Current Net Book Balance from the Output Columns box, and then click Add in the Operand box.
12. Click the **`,`** (comma operand) from the Operand box.
13. Select Record Count from the Output Columns box, and then click Add in the Operand box.
14. Click the **`)`** (close parenthesis operand) from the Operand box.

Example 4 Weighted Average Current Net Rate

The following steps create a Formula ID that calculates the weighted average current net rate for each row in a Commercial Loan Stratification Report.

1. From the File menu, click New.

   A list of the IDs that are available for the product you are using appears.

2. Select the Formula ID.

   The New Formula ID dialog box appears.

3. Select All from the Folder.
4. Type **`CL_AVGNETRATE`** in the Formula ID field.
5. Type **`CL Avg Cur Net Rate`** in the Description field.
6. Click Read / Write.
7. Click OK to continue.

   The Formula ID window appears.

8. Select Commercial Loan from the Tables box.
9. Select Mathematical from the Type box.
10. Double-click WAvg in the Functions box.
11. Select Current Net Rate from the Output Columns box, and then click Add in the Operand box.
12. Click the **`,`** (comma operand) from the Operand box.
13. Select Current Net Book Balance from the Output Columns box, and then click Add in the Operand box.
14. Click the **`)`** (close parenthesis operand) from the Operand box.
15. Save and close the ID.

14

Transformation ID

Most tables in the Oracle Financial Data Manager (FDM) database are formatted for processing. Using your specifications, the Transformation ID restructures data from the processing tables and creates new tables for reporting and analysis. This new set of tables is called the Oracle Financial Data Manager Reporting Data Mart.

The Reporting Data Mart is a collection of tables within a database, optimized for use with reporting and analysis tools. It is created in the same database warehouse where the FDM processing data is stored. You can use the Reporting Data Mart with any reporting tool, without having to further manipulate your data. You can create, open, and edit the Transformation ID in any of the Oracle Financial Services (OFS) applications that use the Tree Rollup ID, including:

- Balance & Control
- Risk Manager
- Performance Analyzer

This chapter discusses the following topics:

- The Transformation Process
- Creating a Transformation ID
- Defining a Transformation ID

The Transformation Process

As the transformation process extracts data from the processing tables and restructures it into the Reporting Data Mart, the Transformation ID maintains the integrity of the data. In addition, the Transformation ID stores other transformation specifications, such as:

- Which transformations are performed
- What tables are designated for transformation
- What parameters or special processing options to apply

Pivoting

The Transformation ID uses two methods of pivoting to transform data:

- Financial Element
- Time Bucket

Financial Element Pivot

For processing, financial elements are stored in the FDM database in separate rows. The Financial Element Pivot extracts the financial elements from the FDM database and pivots them into columns. Each financial element in the original table receives its own column in the reporting and analysis table.

Time Bucket Pivot

A column exists for each time period in the FDM database. Reporting and analysis tools require that time periods exist as rows. The Time Bucket Pivot extracts the time periods from the FDM database and pivots them into rows.

The following tables require both the Financial Element and Time Bucket Pivots:

- Risk Manager Result Detail table (RES_DTL_XXX)
- Ledger Stat table (LEDGER_STAT)

Non-Pivot Transformations

Some Non-Pivot transformations perform filtering functions. Others run additional transformations on selected data. The Non-Pivot Transformations are:

- Dimension Filtering
- Data Filtering
- Rates

Dimension Filtering

With Dimension Filtering, you can select dimensions to exclude. Only Leaf Type Columns are valid dimensions for exclusion. When dimensions are excluded, the Transformation ID:

1. Creates the output table without the columns for the selected dimensions, and
2. Performs a sum aggregation of the data based on the remaining leaf columns

Data Filtering

With Data Filtering, you can specify the use of a filter on the input table. A filter limits the amounts of data for processing. If a filter is specified in the financial element column, then the output table is created with only those financial elements specified in the filter.

Rates

There are two types of rates used during processing:

- **Weighted Rates**

  Rates weighted with their associated balances are stored in the following tables:

  - Ledger Stat
  - Risk Manager Result

  The transformation process leaves these rates weighted.

- **Non-Weighted Rates**

  Real rates (non-weighted) cannot be aggregated unless they are weighted against a balance. These weighted rates are then saved to the output table.

  Occasionally, non-weighted rate data is saved in the LEDGER_STAT table. This rate data is primarily used with allocations. Reports are not generally created from these financial rate elements. The rate data is invalid for transformations that aggregate results from these financial elements. You should filter these financial elements using a data filter prior to running the Transformation ID.

Tree Rollup

The transformation for rollups is different than for other tables. Before the transformation rollup, data is stored in different tables. One rollup is combined and stored in the output table. Each transformation creates a separate table in which only one rollup is placed. This output table is also referred to as a Hierarchy Table.

Metadata and System Information Table

For each new table and column created during the transformation process, the Transformation ID inserts the appropriate rows into the FDM metadata tables to identify the new output table. The FDM metadata tables contain information for defining and tracking tables, columns, and other objects.

Creating a Transformation ID

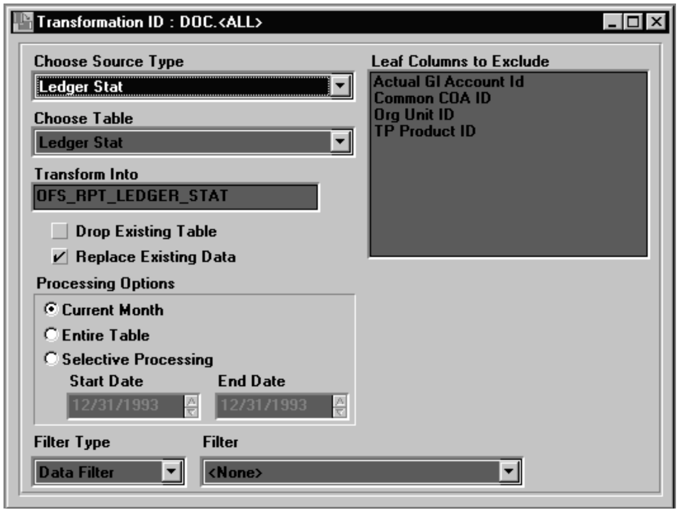
To create a new Transformation ID, complete the following steps:

1. Choose File, New, and Transformation ID.

   The New Transformation ID dialog box appears.

2. Choose a Folder from the list.
3. Press the Tab key to move to the next field.
4. Enter a descriptive name in the Transformation ID field.
5. Enter a description for your ID in the Description field. This is an optional step.
6. Check the Read / Write option.
7. Choose OK.

The Transformation ID dialog box appears with the name of the ID displayed at the top.



Defining a Transformation ID

The sources from which to process the Transformation ID are:

- Ledger Stat
- Risk Manager Result Detail
- Tree Roll

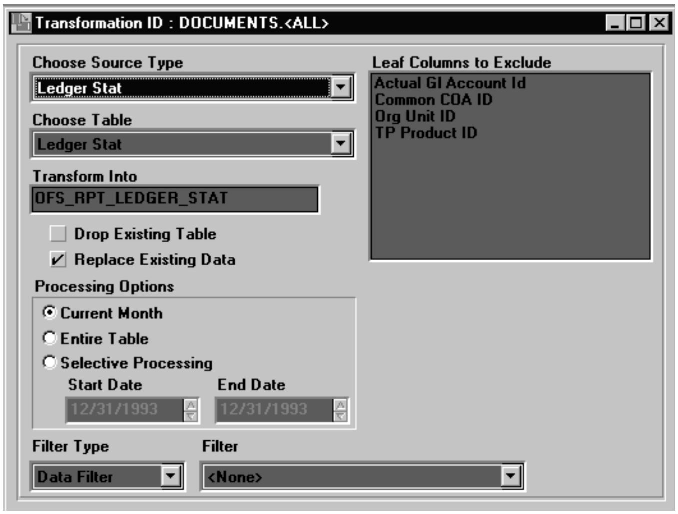
Note: The selection of tables available under Choose Source Types depends on which OFS products are installed.

Defining a Ledger Stat Transformation

The Ledger Stat Table stores General Ledger and statistical information.

To define a Transformation ID for a Ledger Stat table, complete the following steps:

1. Select Ledger Stat from the Choose Source Type list. Ledger Stat appears in the Choose Table field and the Ledger Stat dialog appears.



Note: The Ledger Stat Table appears as the default setting.

---

2. Modify the name of the output table in the Transform Into field to fit your requirements. This is an optional step.

   You cannot use the following characters in the Transform Into field:

   - , (comma)
   - % (percent sign)
   - " (quotes)
   - . (period)

   Note: The maximum amount of characters you can enter in the Transform Into field is 26.

The Transformation ID automatically calculates the storage parameters for the output table. These parameters can also be set manually by your database administrator. For more information on output tables, refer to the Transformation Output Table Creation section in the *Oracle Financial Services Installation and Configuration Guide.*

Note: By default, only the primary key index is created when the output table is generated. For instructions on how to automatically create additional indexes on output tables, refer to the Transformation section in the *Oracle Financial Services Installation and Configuration Guide*.

3. Check the The Drop Existing Table option to overwrite the existing output table.

The following tables explain processing variables, and the results of using them with the Drop Existing Table option:

If	Then
the Drop Existing Table Option is **checked**,	any table with the same name that appears in the Transform Into field is dropped.

If	And	Then
the Drop Existing Table option is **not checked**, and a valid table name appears in the Transform Into field,	either the Current Month or the Selective Processing option is **checked**,	newly processed data is added into the existing Transform Into table.

If	And	Then
the Drop Existing option is **not checked**,	the Entire Table option is **checked**, and a table exists with the name specified in the Transform Into field,	the Transformation ID stops all processing.

4. Check the Replace Existing Data option to overwrite any previously created data. The following tables explain several processing variables and the results of using them with the Drop Existing Table option:

If	Then
the Entire Table Processing Option is checked,	the Replace Existing Data option is not available.

If	And	Then
the Replace Existing Data option is checked,	new data is being added to an existing table,	any previous data that has identical dates as the new data is replaced.

If	And	Then
the Replace Existing Data option is not checked,	new data is being added to an existing table, and any previous data has identical dates as the new data being added,	the Transformation ID stops all processing.

5. Select one of the following Processing Options:
   - **Current Month**

     This option processes the data for the Current Month. The As of Date in the Active Configuration ID specifies the Current Month.
   - **Entire Table**

     This option processes all records within the Ledger Stat Table.
   - **Selective Reprocessing**

     You can specify a range of dates for processing with this option.
6. Choose a Filtering Type from the list if you want to include a filter.

   Note: Filtering narrows the focus of processing to include specific assumptions you define. Three filters used by the Ledger Stat Table include: Data, Folder, and Tree. A list of previously created filter IDs is associated with each filter type.

7. Choose an ID from the Filter list if you want to include a Filter ID.
8. Select the columns you want to exclude from processing in the Leaf Columns to Exclude field. All leaf columns within the Ledger Stat Table appear in the Leaf Columns to Exclude field. After you choose which columns to exclude, at least one column remains and cannot be selected.

   Note: Your filter cannot include any columns that are selected to be excluded during this step.

   The following table explains several processing variables that disable the Leaf Columns to Exclude field:

When	If	And	Then
the Transform Into Table already exists, and	the Drop Existing Table option is **not checked**,	the Current Month or the Selective Processing option is selected,	the Leaf Columns to Exclude field cannot be changed.

Risk Manager Result Detail Table

The Risk Manager Result Detail Table stores processing results from the Oracle Risk Manager product. To define a Risk Manager Result Detail Table, complete the following steps:

1. Select Risk Manager Result Detail from the Choose Source Type list. The Risk Manager Result Detail dialog appears.
2. Select a folder from the Folder list.
3. Select a leaf type from the Leaf Type list.
4. Select a Process ID from the list.
5. Modify the name of the output table in the Transform Into field to fit your requirements.

Depending on the type of information in the Result_Detail Table, the output tables are appended with either of the following characters:

Character	Description
_C	This table contains only the cash flow information (XXX_C).
_G	This table contains only the GAP information (XXX_G).
$C	This table contains only the aggregate cash flow information (XXX$C).
$G	This table contains only the aggregate GAP information (XXX$G).

Note: If your results are both Cash Flow and GAP, then output tables are created as appropriate with the extensions of _C and _G. For aggregated results, output tables with the extensions $C and $G are created.

6. Check the The Drop Existing Table option to overwrite the existing output table. The following table explains a processing variable and the result of using it with the Drop Existing Table option:

If	Then
the Drop Existing Table Option is **checked**,	any table with the same name that appears in the Transform Into field is dropped.

7. Select the names that appear in the Columns to Exclude field to exclude during processing. This field shows only valid leaf columns from the Result Detail Table. After you choose which columns to exclude, at least one column remains and cannot be selected.

Tree Rollup Table

The Transformation ID extracts data for one rollup and places the data into a customized table specific to that rollup.

To define a Tree Rollup Table, complete the following steps:

1. Select Tree Rollup from the Choose Source Type list. The Tree Rollup dialog appears.
2. Select a folder from the Folder list.
3. Select a leaf type from the Leaf Type list.
4. Select a Tree Rollup ID from the Tree Rollup ID list.
5. Modify the name of the output table in the Transform Into field to fit your requirements.
6. Check the The Drop Existing Table option to overwrite the existing output table. The following table explains a processing variable and the result of using it with the Drop Existing Table option:

If	Then
the Drop Existing Table Option is **checked**,	any table with the same name that appears in the Transform Into field is dropped.

15

Batch ID

A Batch ID enables you to perform batch processing of multiple IDs such as Processing ID or Transformation ID. With a Batch ID, you can group time-consuming tasks for processing after hours or on weekends.

This chapter presents the following topics:

- Defining a Batch ID
- Processing a Batch ID
- Editing a Batch ID

Defining a Batch ID

To define a Batch ID, perform the following steps:

1. From the File menu, select New - > Batch ID.

   The New Batch ID dialog box appears.

2. Complete each field, as required.

Note: For security reasons, Batch IDs are application-specific. You cannot use a Batch ID defined in one Oracle Financial Services (OFS) application in another OFS application. Accordingly, Batch IDs cannot have the same name, even though they appear in different OFS applications.

---

3. When you are done, click OK.

   The ID Type dialog box appears.

For more detailed information about defining IDs, refer to the ID Type information that follows and to the “Overview of IDs” chapter.

ID Type

When you select an ID type from the list, all the predefined IDs of that type are listed in the ID Type window.

ID Information Dialog

To locate the ID or IDs that you want, view summary information by selecting an ID and clicking the right mouse button. The ID Information dialog box opens. When you have finished viewing the data, click OK.

Selected IDs Window

Select the IDs you want to batch process and click Add. The ID names appear in the ID Name list. To select multiple IDs, hold the Shift key while selecting ID names. To add all of the IDs in your list, click Select All. When you have selected all the IDs you want to batch process, click Add. To remove an ID from the ID Names list, select the ID and click Remove. To remove all IDs from the ID Names list, click Remove All.

Start Date and End Date

These settings select the period in which you want the allocations or IDs to process. If you select Current Date, the system adopts the default As of Date. Choosing specific Start and End dates overrides the default As of Date.

Order of Process Arrows

The IDs are processed in the order in which they are listed in the Selected IDs window. To change the order, select an ID and use the up or down arrows to change its location in the list. When the order is correct, select Run from the Process menu.

Processing a Batch ID

You can process Batch IDs in serial or parallel mode.

In serial mode, each ID is processed one after another. Serial mode is the default setting.

In parallel mode, all IDs in Batch ID are run concurrently. Parallel processing may lengthen your processing time.

Note: If any IDs are dependent on other IDs, or if IDs access any of the same data, you cannot run parallel processing.

To enable parallel processing, you must modify your OSF.ini file. The appropriate parameter value in the [PARALLEL_BATCH_SERVER_PROCESSING] section of the .ini file must be changed to 1.

To process a Batch ID, perform the following steps:

1. Select Run from the Process menu. A spreadsheet window shows the status of each of the IDs being run.

Note: Processes that run on the server may show a complete status before they have actually completed. To check the status of processes running on the server, select Server Status from the Process menu.

---

2. Select Save from the File menu.

Editing a Batch ID

To edit a Batch ID, perform the following steps:

1. From the File menu, select Open - > Batch ID.

   The Select Batch ID dialog box appears.

2. Select the ID you want to edit.
3. With the ID open, make your changes following the principles described in "Defining a Batch ID."
4. Select Save from the File menu.

You can use the Save As option in the File menu to save your edited ID under a new name.

16

Party Profitability Process ID

As part of Oracle Performance Analyzer, the Party Profitability Process ID is a new business rule that offers a complete customer profitability solution. With the Party Profitability Process ID, users can define assumptions for both profitability measures and profitability reporting for customers and households.

Overview of Business Problem and Solution

To survive in the intensely competitive business environments of today, companies are devoting increased attention to the broad area of customer service and customer satisfaction. Key success factors are considered to be product quality, service levels, delivery lead-times, product flexibility, and diversity. In this lies the paradox of modern management. Many firms often claim that they are *customer focused* or *customer driven*, yet their underlying management accounting systems are not. Thus, there has been considerable criticism of the applicability and use of traditional management accounting practices in today's environment. These traditional systems fail to:

- Report information about what matters to the customer
- Measure the profitability of customers and the products they buy
- Encourage actions that improve the company's ability to profitably serve customers

Party profitability analysis gives organizations of all sizes the essential information for choosing what to do with whom, when, and how by answering the following questions:

- How many of your customers are profitable?
- How dependent are you on the most profitable customers?
- Which customers are unprofitable?
- Why are they unprofitable?
- What proportion of your resources do these customers consume?

The Oracle profitability solution answers all these questions and more.

This chapter covers specific features of a solution, including:

Party Profitability Setup

A complete implementation of the Party Profitability solution consists of the following steps:

1. Customer demographic data, account data, and account relationship data are loaded and made available in the appropriate data tables (FEM_PARTIES, account tables, and FEM_PART_ACCT_REL, respectively) using Oracle Warehouse Builder. As part of the loading process, unique customers are identified.
2. User-defined account tables, columns, and leaves are registered in Oracle Financial Data Manager (FDM) Administration. User-defined columns and leaves are added to the appropriate tables (account tables, PFT_PARTY_PROFIT_DETIAL, and FEM_PARTY_PROFITABILITY) and are mapped for proper customer account consolidation and aggregation.
3. Leaf values pertaining to leaves stored on customer accounts are loaded into Leaf Setup.
4. Solutions requiring funds transfer pricing continue to use the customer account (instrument) tables for this purpose.
5. Solutions requiring activity based costing allocations continue to use Oracle Activity Based Management (ABM) and Performance Analyzer to allocate costs to the appropriate customer accounts.
6. Customer account profit measure calculations may occur in the individual customer account tables or the PFT_PARTY_PROFIT_DETAIL table using Performance Analyzer.
7. Party Profitability IDs are set up to specification.

Data Requirements

The Party Profitability Business Rule requires that certain types of data are available before the rule can successfully run. That data includes:

- Customer and household demographic data (maintained in the FEM_PARTIES data structure)
- Customer account relationship data (maintained in the FEM_PARTY_ACCT_REL table)
- Customer account data (maintained in the FDM account tables)

Customer and Household Demographic Data

The FEM_PARTIES table maintains all of the demographic data for customers and households. The rows in the table represent unique customers and households that might have been defined using Oracle Pure Integrate and Oracle Warehouse Builder.

Party Profitability requires that FEM_PARTIES maintain the following columns:

Column	Definition
PARTY_ID	Unique Customer and Household Party identifier
PARTY_TYPE_CD	Identifies Customer as Individual, Business, or Household
PARTY_NUMBER	Unique Customer and Household Party identifier
PARENT_PARTY_NUMBER	Unique Household Party identifier for Individual Customers. Required value for customer records and required null for household records.
HEAD_OF_HOUSEHOLD_PARTY_NUMBER	Party number of head of household. Required value for customer and household.
POSTAL_CODE	Party postal code. Required value for Region Counting.

Note: The unique index includes PARTY_ID.

Note: Party Type Code (PARTY_TYPE_CD) maintains the following codes:

- BC - Business Customer
- FBC - Former Business Customer
- FIC - Former Individual Customer
- HH - Household
- IC - Individual Customer
- OTHER - Other
- PBC - Prospective Business Customer
- PIC - Prospective Individual Customer

Customer Account Relationship Data

Customer account relationship data is stored in the FEM_PARTY_ACCT_REL table. This data represents the information that joins the customer demographic records with the customer account records. This data defines the customer's relationship, primary or secondary, to the account. Each account has only one primary customer. All other customers related to an account are considered secondary customers. The account data can be loaded using functionality available in Oracle Pure Integrate and Warehouse Builder.

Party Profitability requires that FEM_PARTY_ACCT_REL maintains the following columns:

Column	Definition
AS_OF_DATE	Date represented by the data
ACCOUNT_TABLE_CD	Account table identifier
IDENTITY_CODE	Unique identifier for account types (typically the identifier of the account source system)
ID_NUMBER	Unique identifier of accounts (typically the account number)
PARTY_ID	Unique Customer and Household Party identifier
PARTY_NUMBER	Unique Customer and Household Party identifier
PRIMARY_REL_FLG	Flag signifying relationship type (*1* represents primary and *0* represents secondary)
PARTY_TYPE_CD	Identifies Customer as Individual, Business, or Household
PARENT_PARTY_ NUMBER	Unique Household Party identifier for Individual Customers

---

Note: The unique index includes AS_OF_DATE, ACCOUNT_ TABLE_CD, IDENTITY_CODE, ID_NUMBER, and PARTY_ID.

Note: See the Party Profitability Codes section of this chapter for more information on the account table identifiers.

Customer Account Data

Customer account data is the account-level information that is eventually aggregated for customer and household profitability. This data is stored in FDM account tables, which include instrument, service, and account tables. Account-level profitability is calculated on these tables. The results become the basis for aggregating profitability for customers and households. Given the flexibility of FDM ADMINISTRATION, users can define their own account tables. Using this feature, they can implement specific data tables for their own needs as well as provide a solution for non-financial services industries. For example, a telecommunications implementation can include customer account data tables such as Cellular, Land, and ISDN.

To be classified as Party Profitability Account tables and used in the Party Profitability process, customer account tables require the following columns:

Column	Definition
ID_NUMBER	Unique identifier of accounts (typically the account number)
IDENTITY_CODE	Unique identifier for account types (typically the identifier of the account source system)
IDENTITY_CODE_CHG	Required for Performance Analyzer Allocations
ISO_CURRENCY_CD	Currency Code
ACCOUNT_NUMBER	Account Number
AS_OF_DATE	Date represented by the data
RECORD_COUNT	Set to 1
ACCOUNT_GROUP_CD	Account Group identifier
GL_ACCOUNT_ID	General Ledger Account identifier
ORG_UNIT_ID	Organization Unit identifier
COMMON_COA_ID	Common Chart of Account

Note: The unique index includes IDENTITY_CODE and ID_NUMBER.

Note: See the Party Profitability Codes section of this chapter for more information on the Account Group identifiers.

In addition, each account table should have columns that contain the following data:

- Account Contribution
- Allocated Equity
- Contribution After Capital Charge
- Current Net Book Balance
- Equity Credit
- Interest Charge Credit
- Interest Income/Expense
- Loan Loss Provision (LLP)
- Net Fee Income
- Net Interest Margin (NIM)
- Open Account Flag
- Origination Date
- Primary Relationship Account Balance
- Return on Equity
- Total Account Expenses
- Total Transactions

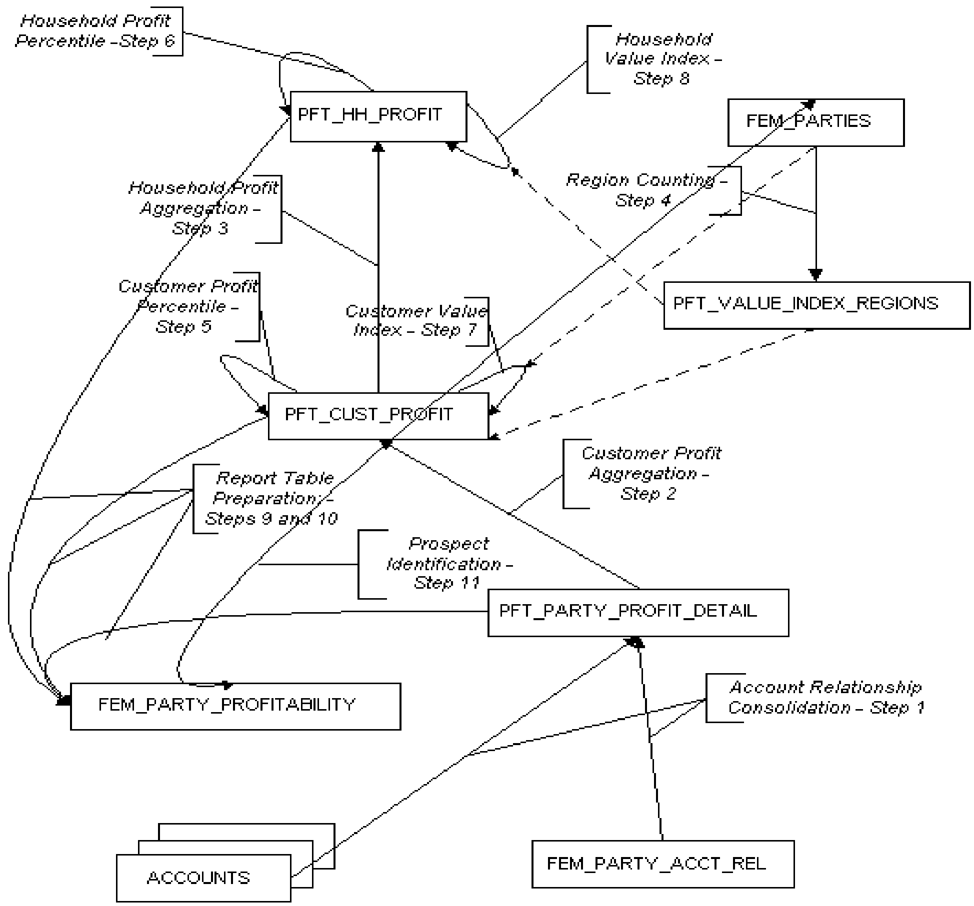
Party Profitability Processes

The Party Profitability Process ID provides 11 Process Steps. Using these Process Steps, users can process the existing customer and customer account data to produce customer profitability measures. Users can include custom processes to enhance the solution so that it better represents their specific needs. Users can also replace or supplement some of the default processes with their own implementations. As a result of the flexibility surrounding the process flow and the processes themselves, this solution opens itself to industry-neutral calculations of customer profitability measures. Industries such as telecommunications, retail, and manufacturing can use this solution for their own implementation of customer and household profitability. These users would only need to add industry-specific account tables and columns.

The Party Profitability solution provides the following processes:

1. Account Relationship Consolidation
2. Customer Profit Aggregation
3. Household Profit Aggregation
4. Region Counting
5. Customer Profit Percentile
6. Household Profit Percentile
7. Customer Value Index
8. Household Value Index
9. Report Table Preparation: Customer
10. Report Table Preparation: Households
11. Prospect Identification

The following diagram represents the process and data flow of the Party Profitability Solution:



Detailed Description of Processes

Account Relationship Consolidation

The Account Relationship Consolidation process consolidates the customer account data with the account relationship data to produce data that represents each customer account, account balance, profit measure and customer relation. The resulting data is loaded in the PFT_PARTY_PROFIT_DETAIL data table. User-defined columns and leaves defined on the account tables that are to be migrated to the PFT_CUST_PROFIT_DETAIL table must be mapped in the OFSA_COLUMN_PROPERTIES table.

The account table profitability components that are consolidated as a result of this Process Step include but are not limited to:

- Account Contribution
- Allocated Equity
- Contribution after Capital Charge (CACC)
- Total Account Transactions

Customer Return on Equity (ROE) is not aggregated as part of this Process Step. Instead, Customer ROE is calculated during the Customer and Household Report Table Preparation Process Steps. Again, any additional profitability components that should be consolidated must be mapped.

The Account Contribution data element is the only element required for non-financial service implementations. All other components listed above are part of the default solution for profitability. For other implementations requiring more or different data elements to define profitability, the solution requires those columns to be added as user-defined columns in the account tables and PFT_PARTY_PROFIT_DETAIL.

Options available as part of this process include loading secondary relationships and identifying orphans. Secondary relationships are not automatically loaded unless the option is chosen. Otherwise, only primary relationships are recognized for producing customer profit measures. There are two kinds of orphans:

- Orphan relationships are primary account relationships stored in the FEM_PARTY_ACCT_REL table that do not have associated customer information in the FEM_PARTIES table.
- Orphan accounts are customer accounts that do not have associated account relationships stored in the FEM_PARTY_ACCT_REL table.

Once orphans are identified, the orphan accounts and relationships are not processed in any of the remaining steps of the profitability model. Instead, they are stored in PFT_ORPHAN_ACCOUNTS and PFT_ORPHAN_ACCT_REL, respectively. Orphan relationships that do not have associated accounts are ignored and never identified for this solution.

Customer and Household Profit Aggregation

The Profit Aggregation processes aggregate the profitability results stored in the PFT_PARTY_PROFIT_DETAIL table and store the results in the Customer and Household profitability processing tables (PFT_CUST_PROFIT and PFT_HH_PROFIT). The process aggregates the profit contribution value across all accounts for unique customers and households. This value is made available in these processing tables for the purpose of calculating profit percentiles and value indexes for each customer and household. No options are available on this Process Step.

Region Counting

The Region Counting Process Step is an optional process that enhances the value index scoring process. This process counts the number of customers and households within each unique postal code to determine the postal codes with a saturation rate of 10 percent or more. Customers and households belonging to those regions receive points towards their score during the Value Index process. This process generates the count for each postal code and stores the data in the PFT_VALUE_INDEX_REGIONS table.

The Region Counting process requires that the Account Relationship Consolidation and Customer Profit Aggregation Process Steps are run. Only the Customer and Household Value Index Process Steps use the data produced by the Region Counting process. This process can be run in parallel with either of the profit percentile Process Steps. No options are available on this Process Step.

Customer and Household Profit Percentile

The Profit Percentile generated for each customer and household is a percentage ranking that represents the level of relative customer or household profitability. More specifically, customers and households are sorted by their calculated Profit Contribution and divided into 100 buckets, with the same number of customers/households in each bucket. Each customer and each household is assigned a profit percentile that is based on the bucket that it is placed in as a result of the process. The customers and households ranking in the highest 1 percent range are assigned a percentile value of 100, and those in the lowest 1 percent range are assigned a percentile value of 1. No options are available on this Process Step.

Customer and Household Value Index

The value index is a value between 0 and 10 (the higher the index, the higher the assumed value) that is assigned to each customer and household based on demographics, account relationships, and profitability characteristics. The default process maintains a value index algorithm that is optimal for financial service implementations. This process assigns points based on the following algorithm to derive the value index:

Customers/Households...	Receive...
Owning a primary checking (assigned account category) relationship	1 point
Owning a primary deposit (assigned account category) relationship and a primary loan relationship	1 point
Owning 3 primary deposit and/or loan (assigned account categories) relationships	1 point
Owning 4 primary deposit and/or loan (assigned account categories) relationships	2 points
Owning 5 or more primary deposit and/or loan (assigned account categories) relationships	3 points
Residing in an area where less than 10 percent of all primary customers reside	1 point
Residing in an area where 10 percent or greater of all primary customers reside	2 points
With a profit percentile between 40 percent and 69 percent	1 point
With a profit percentile between 70 percent and 89 percent	2 points
With a profit percentile between 90 percent and 100 percent	3 points

Options available in this Process Step include:

- Calculate Value Index including Profit Percentile
- Calculate Value Index including Region Counting

Enabling the value index calculation to include profit percentile and Region Counting follows the algorithm as presented above. Disabling the calculation to exclude profit percentile excludes the profit percentile ranking scores in the algorithm. The option to exclude Region Counting for calculating value index ignores the region value for calculating value index.

Note: When Region Counting is enabled, both customers and households receive at least one point. When Region Counting is disabled, customers and households may possibly receive a value index of zero.

Customer and Household Report Table Preparation

The Customer and Household Report Table Preparation Process Steps consolidate the customer and household profitability measures (stored in the PFT_CUST_PROFIT and PFT_HH_PROFIT tables) with the detailed account data (stored in the PFT_PARTY_PROFIT_DETAIL table). The steps store the results in the FEM_PARTY_PROFITABILITY table. Each row in the FEM_PARTY_PROFITAIBLTY table represents a customer or household and detailed information regarding specific account categories and account groups. The structure of the FEM_PARTY_PROFITABILITY table is optimal for reporting and provides continued use of the standard Customer Household Reports.

The two reporting steps, one for customer and one for household, can be run independently.

Each step provides one option:

- Include or exclude customer profitability
- Include or exclude household profitability

Including profitability for customer, household, or both enables the join between the following:

- Customer/household profitability measures stored in the customer and household profitability process tables (PFT_CUST_PROFIT and PFT_HH_PROFIT)
- Customer account detail data stored in the PFT_PARTY_PROFIT_DETAIL table

Excluding profitability for customer, household, or both creates rows in the FEM_PARTY_PROFITABILITY table that contain aggregated account balances but excludes percentile ranking and value index ranking.

For more information about multiprocessing for the Party Profitability Report Table steps, see the revised 4.5 Release of the *Oracle Financial Services Installation and Configuration Guide*, the "OFSA Multiprocessing" chapter.

Prospect Identification

The Prospect Identification Process Step identifies prospects and former customers within a given process period. Prospective customers are defined as customers in FEM_PARTIES that do not have any accounts or any account relationships for the current period. For these customers, the Prospect Identification Process Step changes the Party Type Code in FEM_PARTIES to *PBC* for business customers and to *PIC* for individual customers.

Former customers are defined as customers in FEM_PARTIES that have no accounts for the current period but that have one or more account relationships for the current period, suggesting that they previously had one or more accounts. For these customers, the Prospect Identification Process Step changes the Party Type Code in FEM_PARTIES to *FBC* for business customers and to *FIC* for individual customers.

Tables for Party Profitability Processes

For table recommendations for Party Profitability, see the revised 4.5 Release of the *Oracle Financial Services Installation and Configuration Guide*, the "FDM Database Performance Management" chapter.

Multicurrency

All currencies are translated to the functional currency when loaded from each account table into PFT_PARTY_PROFIT_DETAIL. All processing and reporting are done in functional currency from that point. Balance columns needing translation can be identified by OFSA_TAB_COLUMNS, with Data Type Code equal to 1.

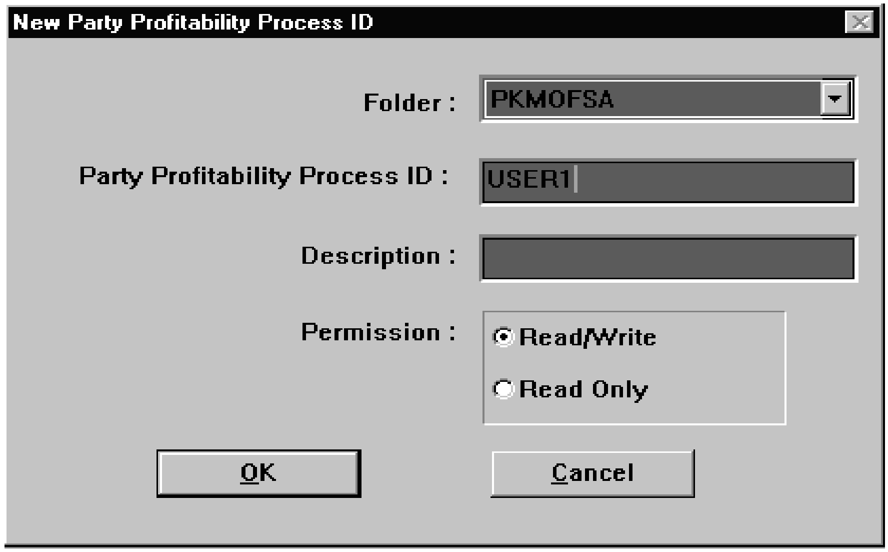
User Interface

The Party Profitability Process comprises the following dialog boxes:

- New Party Profitability Process ID
- Party Profitability Process ID
- Account Table Selection
- Option

New Party Profitability Process ID

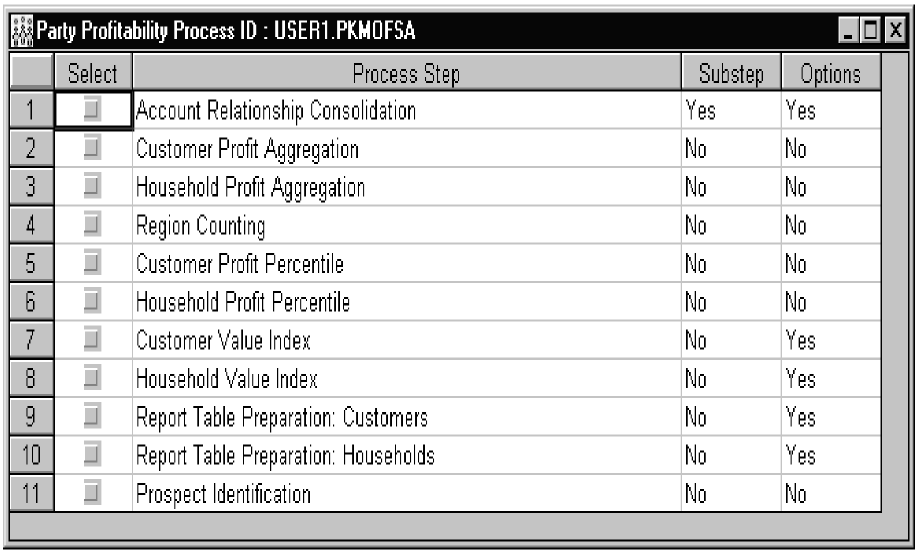
You use the New Party Profitability Process ID to create a party profitability business rule that produces customer and household profitability measures.



Field or Button Name	Field or Button Description
Folder	Contains a list of folders to which you have access as a member of a particular group. When you open the dialog box, your default folder appears. From the Folder list, you can select other folders to which you also have access.
Party Profitability Process ID	Name identifying the business rule
Description	Optional description of the business rule
Permission	Controls access to the business rule. Read / Write is the default.
OK	Saves the information your typed in and takes you to the Party Profitability Process ID dialog box
Cancel	Returns you to the Oracle Performance Analyzer window without saving the information you typed into the dialog box

Party Profitability Process ID

The Party Process ID dialog box lists the required steps for processing party profitability. You can choose the steps that are required for your processing needs. Once all selections are made and substeps and options are defined, you can choose the run icon to execute the steps. The steps are executed on the server so the server status screen immediately appears and displays the status of the Process Steps.

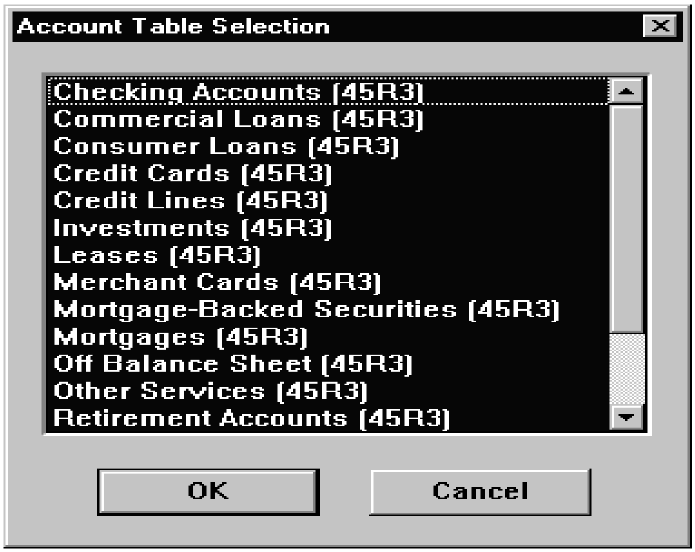

Party Profitability Process ID : USER1.PKMOFSA

	Select	Process Step	Substep	Options
1		Account Relationship Consolidation	Yes	Yes
2		Customer Profit Aggregation	No	No
3		Household Profit Aggregation	No	No
4		Region Counting	No	No
5		Customer Profit Percentile	No	No
6		Household Profit Percentile	No	No
7		Customer Value Index	No	Yes
8		Household Value Index	No	Yes
9		Report Table Preparation: Customers	No	Yes
10		Report Table Preparation: Households	No	Yes
11		Prospect Identification	No	No

Field or Button Name	**Field or Button Description**
Select	Indicates whether or not a Process Step is selected
Process Step	Name of each Process Step. Steps are predefined. You can select one or more of them. When you save your selections, a Save Error, "missing prerequisite step," message may appear. If so, you need to select the prerequisite or prerequisites for processing.
Substep	A *Yes* indicates that the corresponding Process Step has Substeps you can define for processing.
Options	A *Yes* indicates that the corresponding Process Step has Options you select for specific processing.

Account Table Selection

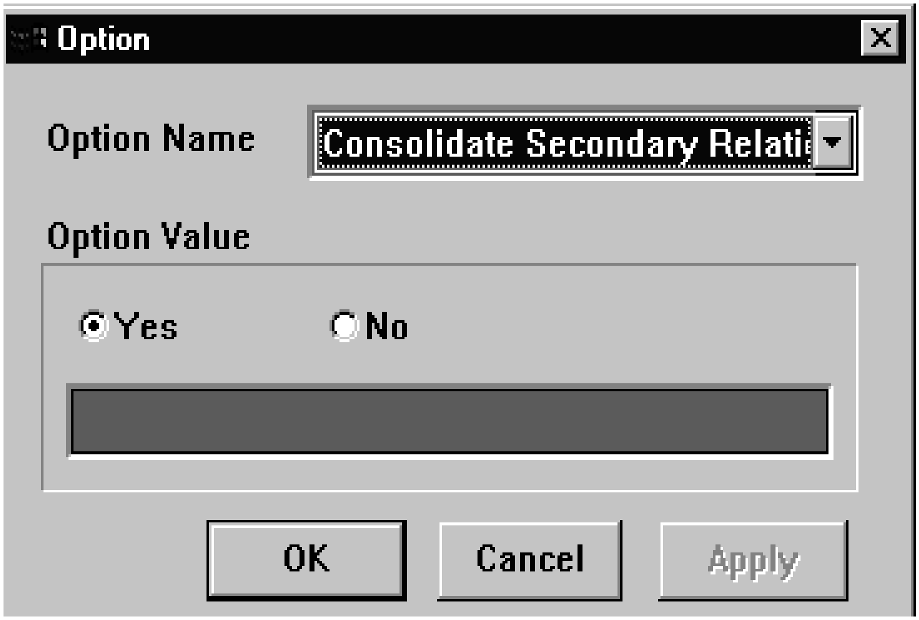
The Account Table Selection dialog box lists the account tables that have been defined as party profitability accounts. You highlight or select the account table or tables listed to identify the accounts to be consolidated during the Account Relationship Consolidation Process Step.



Field or Button Name	Field or Button Description
Account Table List	Contains a list of account tables that you can process. You can select one or more account tables at a time.
OK	Saves the information from your selection and takes you to the Party Profitability Process ID dialog box
Cancel	Takes you to the Party Profitability Process ID dialog box without saving any information you modified

Option

You use the Option dialog box to edit and apply Options Values. An option provides further definition on how to run the Process Steps.



Field or Button Name	Field or Button Description
Option Name	Contains a list of options you can select for processing
Option Value	Indicates whether a flag value or a text value is relevant to the selected option. The text box becomes valid for the Value Index Process Steps, enabling users to define their own algorithm for establishing Value Index. Users provide the name of the program in the text box.
OK	Keeps your changes and returns you to the Party Profitability Process ID dialog box
Cancel	Affects only the last change you had made before you applied the value
Apply	Applies or imposes your changes

Procedure

To create a Party Profitability Process ID, perform the following procedure:

1. If you have not already so, log in to Oracle Performance Analyzer.
2. From the Oracle Performance Analyzer window, select File, New, and Party Profitability Process ID.

   The New Party Profitability Process ID dialog box appears.
3. Select Folder name.
4. Type name of Party Profitability Process ID.
5. Type Description (optional).
6. Verify that Permission is Read / Write.
7. Select OK.

   The Party Profitability Process ID dialog box appears.
8. From the Select column, select a Process Step or Process Steps to include or exclude.
9. To edit Substeps, select (single click) a cell in the Substep column where Column value is Yes.

   The Account Table Selection dialog box appears.
10. Select the Account Table name or names.
11. When you have completed editing Substeps, select OK.

    The Party Profitability Process ID dialog box appears.
12. To edit Options, select (single click) a cell in the Options column where the Column value is Yes.

    The Option dialog box appears.
13. Select Option Name.
14. Edit Option Value.
15. When you have completed editing the selected Option, select Apply.
16. Repeat Steps 13 through 15 to edit any additional Options.
17. Select OK.

    The Party Profitability Process ID dialog box appears.

18. Save the ID.
19. Either run or close the ID.

Defining Customer Account Tables

Detailed customer account data is a key component for defining accurate customer profitability. FDM has three different types of customer account tables, including instrument, service, and account tables.

Instrument tables include:

- COMMERCIAL_LOAN
- CONSUMER_LOAN
- CREDIT_CARD
- MORTGAGES
- DEPOSITS
- TERM_DEPOSITS
- INVESTMENTS
- MORTGAGE_BACK_SEC

Service tables include:

- MC – Merchant Card
- CC – Credit Card
- IV – Investments
- CK **-** Checking
- TR **-** Trusts
- IL – Installment Loans
- RA **-** Retirements
- ML – Mortgage Loans
- DC – Debit Cards
- LS **-** Leases
- CD – Certificate of Deposits

- OD – Other Deposits
- CL – Commercial Loans
- CN – Credit Note
- SD – Safe Deposit
- OL – Other Loans
- OS – Other Services

Account tables include:

- FEM_CHECKING
- FEM_COMMERCIAL_LOANS
- FEM_CONSUMER_LOANS
- FEM_CREDIT_CARDS
- FEM_CREDIT_LINES
- FEM_INVESTMENTS
- FEM_LEASES
- FEM_MERCHANT_CARDS
- FEM_MORTGAGE_BACK_SEC
- FEM_MORTGAGES
- FEM_OTHER_SERVICES
- FEM_RETIREMENT_ACCOUNTS
- FEM_SAVINGS
- FEM_TERM_DEPOSITS
- FEM_TRUSTS
- FEM_WHOLESALE_FUNDING

The unique index for account tables includes ID_NUMBER and IDENTITY_CODE. Customer accounts must be unique by ID_NUMBER and IDENTITY_CODE across all account tables.

Party Profitability Account Table Classification

Every account table that you want to use for Party Profitability must be classified as a Party Profitability account table (Classification Code 90) using FDM Administration. The following columns are required for a table to receive this classification:

- AS_OF_DATE
- ID_NUMBER
- IDENTITY_CODE
- IDENTITY_CODE_CHG
- RECORD_COUNT
- GL_ACCOUNT_ID
- ORG_UNIT_ID
- COMMON_COA_ID
- ISO_CURRENCY_CD
- ACCOUNT_NUMBER
- ACCOUNT_GROUP_CD

See the full definitions of these columns in OFSA_TAB_COLUMNS_V or the revised 4.5 Release of the *Oracle Financial Data Manager Data Dictionary.* After adding any of these columns that are missing from the account table, you can register them in FDM Administration and then register the table for the Party Profitability account table classification. Only tables with this classification are visible in the Party Profitability Process Rule for inclusion in the Account Relationship Consolidation Process Step.

Party Profitability Code Definition

Party Profitability requires certain code definitions for processing and reporting, including account category, group, and table codes.

Account Category Codes

This code is used for high-level reporting summarization. Its initial values are Deposit, Loan, and Other. You can add values, or change existing values of this code. These values are defined in the PFT_ACCOUNT_CATEGORY_DSC table. There is no user interface for modifying this table. If you need to modify this table (MLS-enabled view), you must do it directly. Note that the RPT_DISPLAY_NAME_SUFFIX column is currently not used.

The Oracle-defined Customer Value Index and Household Value Index calculations (which are specific to financial services institutions) use the DEPOSIT_FLG and LOAN_FLG columns in this table to recognize Account Category code values that are deposits and loans, respectively.

ACCOUNT_CATEGORY_CD	ACCOUNT_CATEGORY	DEPOSIT_FLG	LOAN_FLG	RPT_FLEX_FIELD_NUMBER	RPT_DISPLAY_NAME_SUFFIX
10	Deposit	1	0	1	Deposit
20	Loan	0	1	2	Loans
30	Other	0	0	3	Other

Account Group Codes

This code is for lower-level reporting summarization, which corresponds approximately to the division of accounts into separate account tables. In fact, the initial seeded values are the same as those for Account Table Code. They are free to diverge so that how you report is not tied to how you store your data, however. These values are defined in PFT_ACCOUNT_GROUP_DSC. You can add values or change existing values of this code. There is no user interface for modifying this table (MLS-enabled view), however. If you need to do so, you must do it directly. Note that the RPT_DISPLAY_NAME_SUFFIX column is currently not used.

Note: Each value of this code summarizes into one value of Account Category Code.

Note: The Customer Value Index and the Customer Household Index calculations use the CHECKING_FLG column to identify code values that represent checking accounts.

ACCOUNT_ GROUP_CD	ACCOUNT_ GROUP	ACCOUNT CATEGORY CD	CHECKING FLAG	RPT_FLEX_ FIELD_ NUMBER	RPT_ DISPLAY_ NAME_ SUFFIX
10	Checking Accounts	10	1	1	Checking
20	Commercial Loans	20	0	2	Commercial Loans
30	Consumer Loans	20	0	3	Consumer Loans
40	Credit Cards	20	0	4	Credit Cards
50	Credit Lines	20	0	5	Credit Lines
60	Investments	30	0	6	Investments
70	Leases	20	0	7	Leases
80	Merchant Cards	20	0	8	Merchant Cards
90	Mortgage-Backed Securities	20	0	9	Mortgage-Backed Securities
100	Mortgage Loans	20	0	10	Mortgage Loans

Account Table Codes

These values are maintained in the PFT_ACCOUNT_TABLE_DSC table. They define a unique code for each Party Profitability account table. This code is used in FEM_PARTY_ACCT_REL and PFT_PARTY_PROFIT_DETAIL to ensure the uniqueness of rows in the table and to identify the account table to which the full account definition belongs.

PFT_ACCOUNT_TABLE_DSC is initially seeded with one value for each of the new 4.5 Release 3 account table definitions. If you use the 4.5 Release *instrument* or *services* tables, or other user-defined tables, you need to add an entry into this table for each account table that you use with Party Profitability. There is no user interface for maintaining this table. You must do it directly. Note that the ID_ NUMBER_MERGE_ADD column is not currently used.

ACCOUNT_TABLE_CD	ACCOUNT_TABLE_NAME
10	FEM_CHECKING
20	FEM_COMMERCIAL_LOANS
30	FEM_CONSUMER_LOANS
40	FEM_CREDIT_CARDS
50	FEM_CREDIT_LINES
60	FFEM_INVESTMENTS
70	FEM_LEASES
80	FEM_MERCHANT_CARDS
90	FEM_MORTGAGE_BACK_SEC
100	FEM_MORTGAGES
110	FEM_OTHER_SERVICES
120	FEM_RETIREMENT_ACCOUNTS
130	FEM_SAVINGS
140	FEM_TERM_DEPOSITS
150	FEM_TRUSTS
160	FEM_WHOLESALE_FUNDING

Party Profitability Processing Tables

The Party Profitability processing tables include:

- PFT_PARTY_PROFIT_DETAIL
- PFT_CUST_PROFIT
- PFT_HH_PROFIT
- PFT_VALUE_INDEX_REGIONS
- FEM_PARTY_PROFITABILITY

The PFT_PARTY_PROFIT_DETAIL table stores all of the customer account records that are initially stored in disparate account tables. This table also joins the account relationship (primary or secondary) data with the customer account data. Ultimately, this table maintains the necessary profitability information for each account that is to be aggregated for customer and household profitability.

The PFT_CUST_PROFIT and PFT_HH_PROFIT tables store the aggregate profitability balances of each individual customer or household. It is on these tables that profit percentile and value index calculations occur. These tables maintain additional columns for users needing additional profit components for developing customer/household profitability. In PFT_CUST_PROFIT, the ACCOUNT_CONTRIB_TOTAL and NET_PROFIT_CONTRIB columns receive the aggregate ACCOUNT_CONTRIB value from the PFT_PARTY_PROFIT_DETAIL table for each individual customer. In PFT_HH_PROFIT, the ACCOUNT_CONTRIB_TOTAL and NET_PROFIT_CONTRIB columns are the household-level aggregate of the same columns in the PFT_CUST_PROFIT table. If users require additional calculations for developing customer/household profitability, the final account contribution balance must be stored in the NET_PROFIT_CONTRIB column of both tables. The Value_Index2 and Value_Index3 columns are for user usage. The Value_Index1 column stores the value index that is calculated by the system.

The PFT_VALUE_INDEX_REGIONS table stores the results from the Region Counting Process Step. This data is used for calculating value index for both customer and households.

The FEM_PARTY_PROFITABILITY table is the party profitability reporting table that maintains customers and households and their profitability measures and summary information about each of the types of accounts owned by a customer or household.

Enabling Party Profitability Processing Tables for Performance Analyzer

The following Party Profitability processing tables can be enabled as *Instrument Profitability* tables so that you can run Performance Analyzer allocations on them:

- PFT_PARTY_PROFIT_DETAIL
- PFT_CUST_PROFIT
- PFT_HH_PROFIT

PFT_PARTY_PROFIT_DETAIL

Because PFT_PARTY_PROFIT_DETAIL contains a row for every account in every account table, you can use it to perform allocations that work the same way for all types of accounts. Consequently, you need to maintain only one allocation rule and not one for each account table.

Because PFT_PARTY_PROFIT_DETAIL potentially contains secondary account relationships, however, an account can have more than one row in this table. Performance Analyzer requires the rows to be unique by ID_NUMBER and IDENTITY_CODE. The PFT_ACCOUNT_PROFIT_DETAIL_V view filters out the secondary relationships and omits the PARTY_ID column, returning only one row per account, as required by Performance Analyzer. This assumes that either IDENTITY_CODE has a distinct value for each account table that feeds PFT_PARTY_PROFIT_DETAIL, or that ID_NUMBER is unique across all accounts in all feeder account tables.

To enable the view PFT_ACCOUNT_PROFIT_DETAIL_V view for Performance Analyzer, you must:

1. Add all of your Oracle Financial Services Applications (OFSA) user-defined leaf columns to PFT_PARTY_PROFIT_DETAIL as NOT NULL columns.
2. Add other balance or rate columns necessary for your allocation processing.
3. Re-create the PFT_ACCOUNT_PROFIT_DETAIL_V view to include the columns that you have added to PFT_PARTY_PROFIT_DETAIL. (See its definition in the pprof_views.sql file in *dbs/453000001/views* directory of Performance Analyzer server installation.)
4. Register the new columns in FDM Administration for the PFT_ACCOUNT_PROFIT_DETAIL_V view.
5. Register the view for the *Instrument Profitability* table classification.

6. Map all of the user-defined columns that you added to PFT_PARTY_PROFIT_DETAIL to a source column in each accounts table. See "Column Mappings for Account Relationship Consolidation".

PFT_CUST_PROFIT and PFT_HH_PROFIT

You can stop the Party Profitability process after the Customer Profit Aggregation Process Step or the Household Profit Aggregation Process Step to allocate customer or household relationship expenses directly to the customer or household level. To enable these tables for Performance Analyzer processing, you must:

1. Add all OFSA user-defined leaf columns as NOT NULL columns, with a column DEFAULT of value of 0.
2. Register them in FDM Administration.
3. Register the tables for the *Instrument Profitability* table classification.

The user-defined leaf columns in PRT_CUST_PROFIT and PFT_HH_PROFIT are not actually used but are required as NOT NUL columns for compatibility with Performance Analyzer. The column default is required because these columns are not explicitly populated by the aggregation steps.

The Customer Profit Aggregation Process Step populates the PARTY_ID value into ID_NUMBER and a numeric date representation into IDENTITY_CODE so that each row in each of these tables is unique by IDENTITY_CODE and ID_NUMBER, as required by Performance Analyzer.

The Customer Profit Aggregation Process Step also populates the NET_PROFIT_CONTRIB column with the same value as the ACCOUNT_CONTRIB_TOTAL column. This way, if you have no customer or household-level relationship expenses that you want to allocate, NET_PROFIT_CONTRIB is ready-populated and ready for the Customer and Household Report Table Preparation Process Steps. If you do want to allocate relationship expenses, however, here is the suggested approach:

1. Run one or more allocations to allocate the customer and household relationship expenses to the RELATIONSHIP_EXP column.
2. Run another allocation to add RELATIONSHIP_EXP into NET_PROFIT_CONTRIB.

You can add *working* columns to these tables to support more complex allocation processing. Only the pre-defined columns from these tables are copied to the reporting table (FEM_PARTY_PROFITABLITY), however.

Extra Columns in PFT_CUST_PROFIT and PFT_HH_PROFIT

These tables contain extra columns defined for user-defined processes. VALUE_INDEX1 is used for the Oracle-generated Customer Value Index and Household Value Index calculations. VALUE_INDEX2 and VALUE_INDEX3 are available for user-defined index calculations. Likewise, the SEGMENT1…SEGMENT5 columns are available for any user-defined values populated by user-defined processes. The Customer and Household Report Table Preparation Process Steps copy all of these columns to corresponding columns in FEM_PARTY_PROFITABILITY. Using FDM Administration, you can set the display names for these columns in FEM_PARTY_PROFITABILITY to match their usage.

Column Mappings for Account Relationship Consolidation

The Account Relationship Consolidation Process Step populates the following columns in the PFT_PARTY_PROFIT_DETAIL table from columns in each account table according to column mapping definitions in OFSA_COLUMN_PROPERTIES:

- ACCOUNT_CONTRIB
- ALLOCATED_EQUITY
- PRIMARY_BAL
- CONTRIB_AFTER_CAPITAL_CHG
- CUR_NET_BOOK_BAL_C
- EQUITY_CREDIT
- INTEREST_CHARGE_CREDIT
- INTEREST_INC_EXP
- LOAN_LOSS_PROVISION
- NET_FEE_INCOME
- NET_INT_MARGIN
- OPEN_ACCOUNT_FLG
- ORIGINATION_DATE
- RETURN_ON_EQUITY
- TOTAL_ACCOUNT_EXP
- TOTAL_TRANSACTIONS

All of these columns are required by the seeded Party Profitability reports and by the Customer and Household Report Table Preparation Process Steps. The ACCOUNT_CONTRIB column is also required by the Customer and Household Profit Aggregation Process Steps.

Although you can use the 4.5 Release *services* tables or the 4.5 Release *instrument* tables as source account tables for this process, Oracle recommends that you use the 4.5 Release 3 account tables. These new account table definitions already contain corresponding columns for each of these five columns in PFT_PARTY_PROFIT_DETAIL. Mapping entries are automatically created for them when you run the scripts that create these new account tables.

If you choose to use the 4.5 Release *instrument* tables, *services* tables, or other user-defined account tables, you may need to add columns to those tables that can be mapped to these columns in PFT_PARTY_PROFIT_DETAIL, and you have to create mappings for each of these five columns for each instrument or services table. No mappings are seeded for these tables.

If you want to make additional columns available for user-defined reports, you can add columns to PFT_PARTY_PROFIT_DETAIL. These columns must then be mapped to source columns in each account table. This is also true for any user-defined leaf columns that you add to PFT_PARTY_PROFIT_DETAIL.

Because there is currently no user interface that works for populating these mappings in OFSA_COLUMN_PROPERTIES, you must insert mapping rows directly into this table. The mappings for the new FEM_CHECKING account table to the five columns in PFT_PARTY_PROFIT_DETAIL that require mappings are shown in the table below. You should follow this pattern when creating mappings for any other columns that you add to PFT_PARTY_PROFIT_DETAIL, for each account table.

The following table shows some of the mappings for the FEM_CHECKING table.

TABLE_NAME	COLUMN_NAME	PROPERTY_COLUMN
The Account table containing the source column	The source column in the account table	The TARGET column in PFT_CUST_PROFIT_ DETAIL
FEM_CHECKING	ACCOUNT_CONTRIB	ACCOUNT_CONTRIB
FEM_CHECKING	ALLOCATED_EQUITY	ALLOCATED_EQUITY
FEM_CHECKING	AVG_NET_BOOK_BAL_C	PRIMARY_BAL
FEM_CHECKING	CONTRIB_AFTER_ CAPITAL_CHG	CONTRIB_AFTER_ CAPITAL_CHG
FEM_CHECKING	TOTAL_TRANSACTIONS	TOTAL_TRANSACTIONS

The remaining columns in OFSA_COLUMN_PROPERTIES should be populated as follows (the values are the same for every PFT_PARTY_PROFIT_DETAIL mapping row):

Column	Definition
OWNER	The OFSA database owner, that is, the username of the schema that owns all of the objects in the FDM database.
COLUMN_PROPERTY_ CD	40 (defines that this is a mapping to PFT_PARTY_PROFIT_ DETAIL)
PROTECTED_FLG	0 (indicating that this is a user-defined / user-modifiable mapping)

These PFT_PARTY_PROFIT_DETAIL columns are *hard-code mapped* for the Account Relationship Consolidation Process Step, that is, they will be copied without any entry in OFSA_COLUMN_PROPERTIES for Column Property Code 40. An entry in OFSA_COLUMN_PROPERTIES for Column Property Code 40 will cause a fatal error in the Account Relationship Consolidation Process Step.

- AS_OF_DATE
- ACCOUNT_TABLE_CD
- PARTY_ID
- ID_NUMBER
- IDENTITY_CODE
- GL_ACCOUNT_ID
- ORG_UNIT_ID
- COMMON_COA_ID
- ISO_CURRENCY_CD
- PARTY_TYPE_CD
- PARTY_NUMBER
- ACCOUNT_NUMBER
- ACCOUNT_CATEGORY_CD
- ACCOUNT_GROUP_CD
- PARENT_PARTY_NUMBER
- IDENTITY_CODE_CHG
- RECORD_COUNT
- PRIMARY_REL
- SECONDARY_REL

Column Mappings for Customer and Household Report Table Preparation

Most of the pre-defined columns in PFT_PARTY_PROFIT_DETAIL are already included in the Customer and Household Report Table Preparation Process Steps to make them available for reporting. Any columns that you add to PFT_PARTY_PROFIT_DETAIL for reporting must be mapped to columns in FEM_PARTY_PROFITABILITY so they can be populated by this step. In aggregating the account-level rows in PFT_PARTY_PROFIT_DETAIL to customer and household-level rows in FEM_PARTY_PROFITABILITY, the SUM function is used for all reporting columns.

There are two ways that reporting columns from PFT_PARTY_PROFIT_DETAIL can be populated into FEM_PARTY_PROFITABILITY:

- Single-sum columns are summed and copied into a single column.
- Group-sum columns are summed for each value of Account Category Code or for each value of Account Group Code.

Each sum is copied to one column in a group of columns.

Single-Sum Reporting Columns

Single-sum reporting columns are columns in PFT_PARTY_PROFIT_DETAIL that are mapped to a single column in FEM_PARTY_PROFITABILITY. These mappings are also defined in OFSA_COLUMN_PROPERTIES, with PFT_PARTY_PROFIT_DETAIL as the source table and FEM_PARTY_PROFITABILITY implicitly defined as the target table by using COLUMN_PROPERTY_CD = 42. Suppose you added the column NET_FEE_INCOME to PFT_PARTY_PROFIT_DETAIL and mapped it from each of your account tables, as described in the previous section. To make this column available for customer and household-level reporting, you must create a mapping for it from PFT_PARTY_PROFIT_DETAIL to some column, say REVENUE1, in FEM_PARTY_PROFITABILITY. Here is how to map it in OFSA_COLUMN_PROPERTIES:

TABLE_NAME	COLUMN_NAME	PROPERTY_COLUMN
Always PFT_PARTY_PROFIT_DETAIL	The source column in PFT_PARTY_PROFIT_DETAIL	The TARGET column in FEM_PARTY_PROFITABILITY
PFT_PARTY_PROFIT_DETAIL	NET_FEE_INCOME	REVENUE1

The remaining columns in OFSA_COLUMN_PROPERTIES should be populated as follows (the values are the same for every PFT_PARTY_PROFIT_DETAIL mapping row):

Column	Definition
OWNER	The OFSA database owner, that is, the username of the schema that owns all of the objects in the FDM database
COLUMN_ PROPERTY_CD	42 (defines that this is a mapping to FEM_PARTY_PROFIT-ABILITY)
PROTECTED_FLG	0 (indicating that this is a user-defined/user- modifiable map-ping)

Using FDM Administration for the REVENUE1 column in FEM_PARTY_ PROFITABILITY, you can change its Display Name to *Net Fee Income*, which represents the true use of this column as you have defined it. You can add single-sum columns to FEM_PARTY_PROFITABILTIY and map them to source columns that you have added to PFT_PARTY_PROFIT_DETAIL.

Group-Sum Reporting Columns

Group-sum columns are columns in PFT_PARTY_PROFIT_DETAIL that are mapped to a group of columns in FEM_PARTY_PROFITABILITY. Each column in that target group represents the sum of the source column for one value of Account Category Code or Account Group Code (or some other user-specified code that has been populated for each row in PFT_PARTY_PROFIT_DETAIL). One source column can be mapped to more than one column set, enabling summarization by both of these codes. These mappings are maintained in the PFT_RPT_FLEX_FIELDS_MAP table.Because there is currently no user interface for maintaining this table, you must create any additional group-sum mappings by inserting rows directly into this table.

The following table shows the seeded entries for the pre-defined group-sum reporting columns in PFT_PARTY_PROFIT_DETAIL. You should follow this pattern for any columns that you have added to PFT_PARTY_PROFIT_DETAIL that you want to define as group-sum reporting columns. Some of the source columns are summarized by both Account Category Code and Account Group Code into two separate column sets. The remaining columns in PFT_RPT_FLEX_FIELDS_MAP, not shown here, are not currently used.

FLEX_FIELD_ GROUP_PREFIX	LOV_TABLE	LOV_COLUMN	SOURCE_ COLUMN_NAME
Identifies the group of columns in FEM_ PARTY_ PROFITABLITY	List of Values table defining the code to summarize by	The Code column in the List of Values table	The source column in PFT_PARTY_ PROFIT_DETAIL
ALLOC_EQUITY_ CAT	PFT_ACCOUNT_ CATEGORY_DSC	ACCOUNT_ CATEGORY_CD	ALLOCATED_ EQUITY
BALANCE_CAT	PFT_ACCOUNT_ CATEGORY_DSC	ACCOUNT_ CATEGORY_CD	PRIMARY_BAL
BALANCE_GROUP	PFT_ACCOUNT_ GROUP_DSC	ACCOUNT_ GROUP_CD	PRIMARY_BAL
CACC_CAT	PFT_ACCOUNT_ CATEGORY_DSC	ACCOUNT_ CATEGORY_CD	CONTRIB_AFTER_ CAPITAL_CHG
PRI_ACCOUNTS_ CAT	PFT_ACCOUNT_ CATEGORY_DSC	ACCOUNT_ CATEGORY_CD	PRIMARY_REL

FLEX_FIELD_ GROUP_PREFIX	LOV_TABLE	LOV_COLUMN	SOURCE_ COLUMN_NAME
PRI_ACCOUNTS_ GROUP	PFT_ACCOUNT_ GROUP_DSC	ACCOUNT_ GROUP_CD	PRIMARY_REL
PROFIT_CONTRIB_ CAT	PFT_ACCOUNT_ CATEGORY_DSC	ACCOUNT_ CATEGORY_CD	ACCOUNT_ CONTRIB
PROFIT_CONTRIB_ GROUP	PFT_ACCOUNT_ GROUP_DSC	ACCOUNT_ GROUP_CD	ACCOUNT_ CONTRIB
ROE_CAT	PFT_ACCOUNT_ CATEGORY_DSC	ACCOUNT_ CATEGORY_CD	See Note.
SEC_ACCOUNTS_ CAT	PFT_ACCOUNT_ CATEGORY_DSC	ACCOUNT_ CATEGORY_CD	SECONDARY_REL
SEC_ACCOUNTS_ GROUP	PFT_ACCOUNT_ GROUP_DSC	ACCOUNT_ GROUP_CD	SECONDARY_REL
TRANSACTIONS_ CAT	PFT_ACCOUNT_ CATEGORY_DSC	ACCOUNT_ CATEGORY_CD	TOTAL_ TRANSACTIONS

Note: The computation for the ROE_CAT columns is not computed as a simple sum. It is defined in the Customer and Household Report Table Preparation Process Steps as SUM (ACCOUNT_CONTRIB) / SUM (ALLOCATED_EQUITY) * 12, where accounts in PFT_PARTY_PROFIT_DETAIL are grouped by Account Category Codes.

Each value of the code used for summarization maps to one column in the assigned column set in FEM_PARTY_PROFITABILITY. Code values are mapped to the columns in the column set according to the value of the RPT_FLEX_FIELD_NUMBER column in the List of Values table. For example, ALLOCATED_EQUITY maps to ALLOC_EQUITY_CAT1, ALLOC_EQUITY_CAT2, and ALLOC_EQUITY_CAT3, whose display names are *Equity – Deposits*, *Equity – Loans*, and *Equity – Other*, corresponding to the code values of Account Category Code. Note also that ALLOC_EQUITY_CAT4 and ALLOC_EQUITY_CAT5 are available for new code values of Account Category Code. If you add new values for Account Category code, you should then use FDM Administration to set the display names for the corresponding columns in each column set, according to the meaning of those new codes. Viewing all of the columns and their display names in FDM Administration can give you a better understanding of how these columns are mapped.

Each column set for Account Category Code has five columns. Each column set for Account Group Code has 20 columns. You can add more columns to each column set in FEM_PARTY_PROFITABILITY if necessary to support additional user-defined code values. You can set the display names for these columns according to the meanings of the code values that you map to them. You can also add column sets for group-sum columns that you have added to PFT_PARTY_PROFIT_DETAIL, and map them according to the pattern shown in this section.

The following columns in FEM_PARTY_PROFITABILITY are *hard-code mapped* in the Report Table Preparation Process Steps, without any mappings defined in OFSA_COLUMN_PROPERTIES or PFT_RPT_FLEX_FIELDS_MAP. If you create mappings in either of the mapping tables for these columns, you will cause the Report Table Preparation Process Steps to fail with the error "duplicate column name."

FEM_PARTY_PROFITABILITY Column	Source Table
AS_OF_DATE	PARTY_PROFIT_DETAIL
PARTY_ID	PARTY_PROFIT_DETAIL for customers, FEM_PARTIES for households
PARTY_TYPE_CD	PARTY_PROFIT_DETAIL for customers, ’HH’ for households
PARTY_NUMBER	PARTY_PROFIT_DETAIL for customers, FEM_PARTIES for households
PARENT_PARTY_NUMBER	PARTY_PROFIT_DETAIL for customers, FEM_PARTIES for households

ISO_CURRENCY_CD	PARTY_PROFIT_DETAIL
CREATED_BY	null
CREATION_DATE	null
LAST_UPDATED_BY	null
LAST_UPDATE_DATE	SYSDATE
LAST_UPDATE_LOGIN	null
ALLOC_EQUITY_TOTAL	PARTY_PROFIT_DETAIL
CACC_TOTAL	PARTY_PROFIT_DETAIL
NET_PROFIT_CONTRIB	PFT_CUST_PROFIT or PFT_HH_PROFIT
PRI_ACCOUNTS_TOTAL	PARTY_PROFIT_DETAIL
PROFIT_CONTRIB_TOTAL	PARTY_PROFIT_DETAIL
PROFIT_DECILE	PFT_CUST_PROFIT or PFT_HH_PROFIT
PROFIT_PERCENTILE	PFT_CUST_PROFIT or PFT_HH_PROFIT
RELATIONSHIP_EXP	PFT_CUST_PROFIT or PFT_HH_PROFIT
ROE_TOTAL	PARTY_PROFIT_DETAIL
SEC_ACCOUNTS_TOTAL	PARTY_PROFIT_DETAIL
SEGMENT1	PFT_CUST_PROFIT or PFT_HH_PROFIT
SEGMENT2	PFT_CUST_PROFIT or PFT_HH_PROFIT
SEGMENT3	PFT_CUST_PROFIT or PFT_HH_PROFIT
SEGMENT4	PFT_CUST_PROFIT or PFT_HH_PROFIT
SEGMENT5	PFT_CUST_PROFIT or PFT_HH_PROFIT
TRANSACTIONS_TOTAL	PARTY_PROFIT_DETAIL
VALUE_INDEX1	PFT_CUST_PROFIT or PFT_HH_PROFIT
VALUE_INDEX2	PFT_CUST_PROFIT or PFT_HH_PROFIT
VALUE_INDEX3	PFT_CUST_PROFIT or PFT_HH_PROFIT

Global Options

There are three global options for the Party Profitability process (all are numeric):

Option Code	Option Name	Description	Default Value
100	Parent Job Sleep Interval	The interval, in seconds, that a Party Profitability job (parent job) sleeps between status checks while waiting for its child jobs to complete	30

The values for these options are stored in the PFT_CPP_GLOBAL_OPTIONS table. There is currently no user interface for changing the values of these parameters. To change the values for any of these parameters, you must update the VALUE column in PFT_CPP_GLOBAL_OPTIONS for the option you want to change.

PFT_CPP_GLOBAL_OPTIONS:

OPTION_CD	VALUE_NUM	VALUE_CHAR
100	30	

Specific Party-Data Requirements

Party ID

A unique number assigned to each customer or household, unique across all party types (for example, where PARTY_TYPE_CD is 'HH' or 'IC', and so on). It may be system-generated. Ideally, it is the same from one period to the next, but this is not a requirement.

PARTY_NUMBER

A unique character (VARCHAR2) value assigned to each customer or household, unique across all party types (for example, where PARTY_TYPE_CD is 'HH' or 'IC'). Represents the bank's *real* customer number. It should be the same from one period to the next.

> **Note:** VARCHAR2 column matches require exact matching. No leading or trailing spaces are allowed.

PARENT_PARTY_NUMBER

A character (VARCHAR2) value assigned to each customer and to each account to indicate the PARTY_NUMBER of the household to which the customer or the account belongs. Households may also have a PARENT_PARTY_NUMBER. The PARENT_PARTY_NUMBER should be the same from one period to the next.

> **Note:** VARCHAR2 column matches require exact matching. No leading or trailing spaces are allowed.

FEM_PARTIES Data Object

The FEM_PARTIES data object stores all customer and household demographic information. FEM_PARTIES is used to determine if a customer or household represented elsewhere in the system is valid. A customer or household that cannot be found in FEM_PARTIES (either by PARTY_ID or by PARTY_NUMBER) is not a valid party.

Each Household should have at least one customer. Otherwise, it is not used by the Party Profitability process. If a household does not have any customers, it is simply wasting space in the FEM_PARTIES table.

Conversely, each customer is a member of only one Household.

Every customer row in FEM_PARTIES should have a corresponding household row (PARTY_TYPE_CD = 'HH') also in FEM_PARTIES, where the PARENT_PARTY_NUMBER of the customer is equal to the PARTY_NUMBER of the Household.

A Household row in FEM_PARTIES must be designated with a PARTY_TYPE_CD equal to 'HH' and must have either a NULL in its PARENT_PARTY_NUMBER column or refer to another valid parent row in FEM_PARTIES having a user-defined value for PARTY_TYPE_CD.

Row-Count Comparisons

When a processing cycle is complete, it should not be possible to have any customers in the following tables (for the current period) that do not exist as customers in FEM_PARTIES:

PFT_PARTY_PROFIT_DETAIL
PFT_CUST_PROFIT
FEM_PARTY_PROFITABILITY

Also, it should not be possible to have any households in these tables (for the current period) that do not exist as households in FEM_PARTIES:

PFT_PARTY_PROFIT_DETAIL
PFT_HH_PROFIT
FEM_PARTY_PROFITABILITY.

Account-Level Issues

Accounts belonging to one customer should all link to the same household. Another way to look at this is that all accounts that belong to a household must be owned by a customer who belongs to that household.

Here are two different ways to ask essentially the same question:

Number of customers with primary account relationships that are assigned to more than one household:

```
select count(*)
from (select party_id
      from fem_party_acct_rel
      where as_of_date = current_as_of_date
        and primary_rel_flg = 1
      group by party_id
      having count(distinct parent_party_number) > 1);
```

Number of accounts that are pointing to a different household than the one that the customer that owns the account belongs to:

```
select count(*)
from fem_parties cust,
     fem_party_acct_rel acct
where acct.as_of_date = current_as_of_date
  and acct.party_id = cust.party_id
  and acct.parent_party_number!= cust.parent_party_number;
```

Other FEM_PARTIES Column Requirements

The HEAD_OF_HH_PARTY_NUMBER value is the PARTY_NUMBER of the customer who is designated as the head of the household. It is used only for household rows. It is not used for customer rows.

The POSTAL_CODE column must be populated at least for customers to get Region counting to work. For households, either the POSTAL_COE column must be populated, or the HEAD_OF_HH_PARTY_NUMBER must be populated and the POSTAL_CODE must be populated for the customer who is the head of that household.

All households referenced in account relationship rows in the PFT_PARTY_PROFIT_DETAIL PARENT_PARTY_NUMBER column for the current period must exist as household rows in FEM_PARTIES. Otherwise, they are excluded in the Household Aggregation Process Step. After the Account Relationship Consolidation Process Step is complete, you can run this SQL to check for missing households in FEM_PARTIES:

```
select distinct Parent_Party_Number
from PFT_PARTY_PROFIT_DETAIL
where as_of_date = current_as_of_date
MINUS
select Party_Number
from FEM_PARTIES
where Party_Type_Cd = 'HH';
```

Account Table and Relation Table Requirements

The FEM_PARTY_ACCT_REL table joins Party information with Account information. Care must be taken to properly match party information (for example, PARENT_PARTY_NUMBER, PARTY_NUMBER) with account information (for example, ID_NUMBER, IDENTITY_CODE).

ID_NUMBER and IDENTITY_CODE combinations must be unique across all account tables. You can append the ACCOUNT_TABLE_CODE for each account table to the IDENTITY_CODE values in that table to make a unique IDENTITY_CODE for each table. You can then use the same ID_NUMBER in two or more different tables.

The FEM_PARTY_ACCT_REL table entry for a secondary account relationship uses the same ID_NUMBER and IDENTITY_CODE as for the primary account relationship. The differences are the PARTY_ID and the PARTY_NUMBER of the row, which point to a different customer. Also, the PRIMARY_REL_FLG of a secondary account is set to 0. Primary accounts have a 1 in PRIMARY_REL_FLG.

During the Account Relationship Consolidation Process Step, in PFT_PARTY_PROFIT_DETAIL, the column PRIMARY_REL is set to 1 for primary accounts and SECONDARY_REL is set to 0. For secondary accounts it is reversed: PRIMARY_REL is set to 0 and SECONDARY_REL is set to 1.

Every account relationship that is the *only* account relationship for that account *must* be marked as a Primary relationship (PRIMARY_REL_FLG = 1 in FEM_PARTY_ACCT_REL). This is because account profitability information is copied to PFT_PARTY_PROFIT_DETAIL only for primary account relationships, and only primary account relationships are aggregated to PFT_CUST_PROFIT. Also, missing customers in PFT_CUST_PROFIT may result in missing households. It certainly results in incompletely aggregated households in PFT_HH_PROFIT. This, in turn, results in missing or incorrect customer and household profit information for these customers and households in FEM_PARTY_PROFITABILITY.

For every customer or household in FEM_PARTY_PROFITABILITY that has at least one primary account relationship, there should be a matching row in PFT_CUST_PROFIT or PFT_HH_PROFIT. When including secondary relationships in the process, every customer and household that have only secondary account relationships are represented in FEM_PARTY_PROFITABILITY, but there is not any profitability information for these rows and they are not represented in PFT_CUST_PROFIT or PFT_HH_PROFIT.

The following two queries for valid primary customers should return the same row counts:

```
SELECT COUNT(*)
FROM FEM_PARTY_PROFITABILITY
WHERE party_type_cd in ('IC','BC')
    AND pri_accounts_total > 0
    AND as_of_date = currrent_as_of_date;

SELECT COUNT(*)
FROM pft_cust_profit
WHERE as_of_date = current_as_of_date;
```

The following two queries for households should return the same row counts:

```
SELECT COUNT(*)
FROM FEM_PARTY_PROFITABILITY
WHERE party_type_cd = 'HH'
    AND pri_accounts_total > 0
    AND as_of_date = currrent_as_of_date;

SELECT COUNT(*)
FROM pft_hh_profit
WHERE as_of_date = current_as_of_date;
```

If the primary and secondary relationships are improperly set (for example, a secondary account relationship that is the only relationship for the account), the customer and the household rows for this account are represented in FEM_PARTY_PROFITABILITY but without any profitability information. Also, they are not represented in PFT_CUST_PROFIT or PFT_HH_PROFIT.

To check for secondary account relationships in FEM_PARTY_ACCT_REL that should be primary because they are the only relationship for that account:

```
SELECT COUNT(*)
FROM (SELECT account_table_cd, identity_code, id_number
      FROM fem_party_account_rel
      WHERE as_of_date = current_as_of_date
        AND primary_rel_flg = 0
      MINUS
      SELECT account_table_cd, identity_code, id_number
      FROM fem_party_account_rel
      WHERE as_of_date = current_as_of_date
        AND primary_rel_flg = 1);
```

Step-Specific Requirements

For the Account Relationship Consolidation Process Step, no two rows in separate account tables can have the same ID_NUMBER and IDENTITY_CODE combination. You can differentiate between these by appending the Account Table Code for each table to the IDENTITY_CODE for that table.

Orphans and Corrections

If an account table row does not match a row in FEM_PARTY_ACCOUNT_REL, the row does not accumulate during processing of the Account Relationship Consolidation Process Step. You can view the message log and see the number of rows processed on each step. You should see your orphan account in PFT_ORPHAN_ACCOUNTS.

If an account table row matches a row in FEM_PARTY_ACCOUNT_REL but its PARTY_ID does not match a row in FEM_PARTIES, that row is inserted into FEM_ORPHAN_ACCT_REL.

A row in FEM_PARTY_ACCOUNT_REL that does not match an account table row is totally ignored. It is not an orphan relationship.

For every customer party in FEM_PARTIES that is not represented in the reporting table (FEM_PARTY_PROFITABILITY), it must be the case that it did not have any valid account relationships represented for it in FEM_PARTY_ACCT_REL. These should be marked as a potential customer or a former customer during the Prospect Identification Process Step.

For every household row in FEM_PARTIES that is not represented in the reporting table, it must be the case that there were no accounts assigned to that household in FEM_PARTY_ACCT_REL. Nothing is done in FEM_PARTIES to identify these households.

To check for households in FEM_PARTIES without any accounts assigned to them in FEM_PARTY_ACCT_REL, including potentially "not used" household assignments on account orphans or account relationship orphans:

```
SELECT COUNT(*)
FROM (SELECT party_number
      FROM fem_parties
      WHERE party_type_cd = 'HH'
      MINUS
      SELECT distinct parent_party_number
      FROM fem_party_acct_rel
      WHERE as_of_date = current_as_of_date);
```

To check for households in FEM_PARTIES without any accounts assigned to them in FEM_PARTY_ACCT_REL, excluding the household assignments on account orphans or account relationship orphans:

```
SELECT COUNT(*)
FROM (SELECT party_number
      FROM fem_parties
      WHERE party_type_cd = 'HH'
      MINUS
      SELECT distinct parent_party_number
      FROM pft_party_profit_detail
      WHERE as_of_date = current_as_of_date);
```

It may be legitimately possible for customers or households in FEM_PARTY_PROFITABILITY to be missing from PFT_CUST_PROFIT or PFT_HH_PROFIT, respectively, because of the way secondary account relationships are handled.

Working with the Party Profitability Process ID

The Party Profitability Process ID processes and transforms data using bulk SQL statements. This processing mode is extremely efficient but requires that the base data set must be in exact order prior to processing. If data is not exactly in order, errors may occur that lead to absolutely no results. Similarly, correcting a single error can be all it takes to get results.

To aid in diagnosing problems, the following table contains the most common issues that may arise during implementation of the Party Profitability solution.

If...	Then...
If you want to see more detailed logging information,	Implement full debug logging. You set it up by inserting rows into the OFSA_DEBUG_MODE table for each step you want, as well as for the step master. INSERT INTO OFSA_DEBUG_MODE (PROCESS_CD , USERNAME) VALUES (2000, <YOUR LOGIN>); INSERT INTO OFSA_DEBUG_MODE (PROCESS_CD , USERNAME) VALUES (2002, <YOUR LOGIN>); INSERT INTO OFSA_DEBUG_MODE (PROCESS_CD , USERNAME) VALUES (2010, <YOUR LOGIN>); INSERT INTO OFSA_DEBUG_MODE (PROCESS_CD , USERNAME) VALUES (2020, <YOUR LOGIN>); INSERT INTO OFSA_DEBUG_MODE (PROCESS_CD , USERNAME) VALUES (2030, <YOUR LOGIN>); INSERT INTO OFSA_DEBUG_MODE (PROCESS_CD , USERNAME) VALUES (2040, <YOUR LOGIN>); INSERT INTO OFSA_DEBUG_MODE (PROCESS_CD , USERNAME) VALUES (2050, <YOUR LOGIN>); INSERT INTO OFSA_DEBUG_MODE (PROCESS_CD , USERNAME) VALUES (2060, <YOUR LOGIN>); INSERT INTO OFSA_DEBUG_MODE (PROCESS_CD , USERNAME) VALUES (2070, <YOUR LOGIN>); INSERT INTO OFSA_DEBUG_MODE (PROCESS_CD , USERNAME) VALUES (2080, <YOUR LOGIN>); INSERT INTO OFSA_DEBUG_MODE (PROCESS_CD , USERNAME) VALUES (2090, <YOUR LOGIN>); INSERT INTO OFSA_DEBUG_MODE (PROCESS_CD , USERNAME) VALUES (2091, <YOUR LOGIN>); INSERT INTO OFSA_DEBUG_MODE (PROCESS_CD , USERNAME) VALUES (2100, <YOUR LOGIN>); You can delete these rows when you are ready for production. You can select from the OFSA_PROCESSES table in the FDM database to see the definition for each of these Process Codes.

If...	Then...
If you want to monitor your Party Profitability jobs as they execute,	See the Discoverer report included in the installation to address Party Profitability job status monitoring. This report, installed into the OFSA45 Disco41 directory, is called `chjobstatus.dis`. This workbook contains three work sheets. The first sheet is called "Job Status Monitor Report." This report is useful to run during Party Profitability processing because it displays the steps that are running and the job status of each step. Alternatively, you can periodically query directly from the PFT_MONITOR_JOB_V view, upon which this report is based. You may want to restrict your queries by JOB_NUMBER, SYS_ID_NUM, or JOB_START_TIMESTAMP.
If you want to view all logged errors and messages for your Party Profitability jobs,	See the second sheet in the Discoverer report mentioned above, called the "Error Message Report." This report displays all status, warning, and error messages logged by each Party Profitability job. It includes messages for substep jobs for the Account Relationship Consolidation step but not for the Report Table Preparation subprocess jobs. The third sheet is called the "Detailed Error Report." This report displays warning and error messages logged during the Report Table Preparation subprocess jobs. There are potentially a larger number of these subprocess jobs, so their messages are excluded from the regular Error Messages Report, and only warning and error messages (no status messages) are logged for these subprocess jobs. Use this report to determine which substep failed (and why it failed) if the reporting steps do not complete successfully. Alternatively, you can query directly from the views PFT_CPP_FULL_STEP_STATUS_V and PFT_CPP_RANGE_SUBSTEP_STATUS_V. You may want to restrict your queries by AS_OF_DATE, JOB_NUMBER, SYS_ID_NUM, or JOB_START_TIMESTAMP.

If...	Then...
If you need to break up your Party Profitability job into separate steps,	You need to understand some details about the ID prerequisite checking rules. Dependency checking is performed for prerequisite steps. If you uncheck the Household Profit Aggregation Process Step, you cannot have any other HH steps checked in the ID (for example, Household Profit Percentile or Household Value Index). If a step has options and an option disables another step, for example, Report Table Preparation: Households (the Include Household Profitability - No step), then there is no prerequisite checking of the earlier, dependent steps (Household Profit Percentile, in this example). Also, no checking is done of any required steps that precede the first checked check box. This enables you to break up the process into separate IDs and run allocations in between, all in a Batch ID.
If your customer account relationship data does not match your account data,	Party Profitability will probably not run. If the account table rows do not match a row in FEM_PARTY_ACCOUNT_REL, no data accumulates during processing of the first step. When you view the message log, you can see the number of rows processed on each step. With no matches at all, you also get no rows in PFT_ORPHAN_ACCT_REL. All steps run but you get no data. You should see your account table rows in PFT_ORPHAN_ACCOUNTS.
If after running a seemingly successful Account Relationship Consolidation step you have no rows for the current period in PFT_PARTY_PROFIT_DETAIL,	Your customer account relationships in FEM_PARTY_ACCT_REL do not match the accounts in your account tables. The probable cause of the mismatches is that the IDENTITY_CODE values do not match, or you are trying to process for the wrong AS_OF_DATE. The identifying columns for each unmatched account are written to the PFT_ORPHAN_ACCOUNTS table. The FEM_PARTIES and FEM_PARTY_ACCT_REL tables must be populated completely and strictly according to the directions outlined in "Specific Party-Data Requirements".

If...	Then...
If you are having trouble with Secondary Account relationships,	Remember, secondary account relationships are, in reality, a result of a second party accessing an account. As such, they present no new data as far as account balances. They do influence the number of accounts reported in the secondary account relationship columns in the report table. The FEM_PARTY_ACCT_REL table entry for a secondary account uses the same ID_NUMBER and IDENTITY_CODE as the primary account holder. The differences are in the PARTY_ID and PARTY_NUMBER of the row and the PRIMARY_REL_FLG is set to 0. During the Customer Profit Aggregation, the DETAIL row column PRIMARY_REL is set to 1 for primary accounts and SECONDARY_REL is set to 0. For secondary accounts, the settings are reversed: PRIMARY_REL is set to 0 and SECONDARY_REL is set to 1.
If one or more of the substeps for the Account relationship Consolidation Process Step fail,	The status for this step is set to *Failed*. Once you have corrected the problems, you can re-run the ID. The checkpoint-restart logic in the Party Profitability Process ID determines that this step has not yet completed successfully and executes it again. Furthermore, it skips each substep that completed successfully on the first run and reprocesses those substeps that failed initially. When all substeps for this step have completed successfully, then the step is marked as *Completed*.
To totally redo a step for a specific AS_OF_DATE,	Perform the following procedure: **1.** To avoid a unique constraint error on insert, delete all rows from the table or tables that were populated by the step for that AS_OF_DATE. **2.** Delete rows from the two status tables for that AS_OF_DATE for the STEPS_SEQ for the step. The PFT_CPP_STEP_STATUS table contains status of each step. The PFT_CPP_SUBSTEP_STATUS table contains processing status for each substep. This process does not eliminate messages in the message log that are associated with previous runs. Once status is deleted, however, messages for the *undone* step may not appear correctly in the PFT_CPP_FULL_STEP_STATUS_V view.

<table>
<tr><th>If...</th><th>Then...</th></tr>
<tr><td>If you receive the message "cannot insert null" during population of the PFT_CUST_ PROFIT or PFT_HH_ RROFIT tables,</td><td>Apply a column default value for all user-defined leaf columns.</td></tr>
<tr><td>If you have a unique constraint violation on the Account Relationship Consolidation step,</td><td>Two rows have the same key. This can mean that:
Two rows have the same ID_NUMBER in two different account tables and the IDENTITY_CODE is identical for each table
You are re-running the step and did not delete the Detail rows generated from the previous run</td></tr>
</table>

A

Fiscal Year Information

Oracle Financial Services (OFS) applications support either a calendar or fiscal year configuration. If your organization has implemented a fiscal year configuration this appendix provides technical information on how a fiscal year configuration affects the LEDGER_STAT table and as-of-date and year-to-date calculations.

Note that, in the context of this appendix, fiscal year denotes a non-calendar financial year.

This appendix presents the following topics:

FISCAL_YEAR_INFO Configuration for a Fiscal Year

In the FISCAL_YEAR_INFO table, the values in the Start_Month and Fiscal_Period columns set the beginning month and duration of your financial year, respectively, for the LEDGER_STAT table. The values in these two columns convert the Month_XX and YTD_XX columns in the LEDGER_STAT table from a calendar year to a fiscal year configuration.

Start_Month Column

This column holds the value that represents the first month of your financial year. For a calendar year, the value in this column is always 1, which equates to January. A fiscal period generally has a value between 2 and 12. For example, this column would have a value of 3 if your fiscal year began in March and a value of 7 if your fiscal year began in July.

For OFS applications, the default setting is a calendar year and the value in this column is 1.

Note that a value of less than 1 or greater than 12 in this column generates an error message.

Fiscal_Period Column

This column holds the value that represents the duration of the financial period. For a calendar year, this value is always 12 (for 12 months) and, for a fiscal year, this value is typically 12.

The fiscal year configuration also gives you the flexibility to set shorter financial periods within a 12-month period. For example, if your organization has two 6-month fiscal periods over a 12-month span, then the value in this column would be 6. The only restriction on using a period shorter than a year is that the period must be a factor of 12 (1, 2, 3, 4, or 6).

If you set your start month at 1 and the fiscal period at 6, the system automatically treats this setting as a fiscal year configuration.

For OFS applications, the default setting is a calendar year and the value in this column is 12.

Note that a value of less than 1 or greater than 12 in this column generates an error message.

Year-To-Date Calculations Affected by a Fiscal Year Configuration

Year-to-date calculations begin at the start date of the fiscal period rather than January of the year in which the calculations are performed. For example, if July is the start date for your organization’s fiscal period and you process year-to-date calculations through September, the system returns data from the three months from July through September and not from January through September.

This is an important consideration for the following calculations:

- Updating the LEDGER_STAT table when either processing or undoing an Allocation ID
- Performing calculations when ledger data is loaded through the Oracle Financial Data Manager (FDM) (see the Import Ledger subsection in the Overview of IDs chapter in this guide for load options with a fiscal year configuration)
- Performing calculations when transfer rates are migrated from instrument tables to the LEDGER_STAT table through Oracle Transfer Pricing

Viewing the Start_Month and Fiscal_Period Values in the FISCAL_YEAR_INFO Table

To view the Start_Month and Fiscal_Period values, open a SQL Talk window and complete the following steps:

1. From the menu bar, select Process and SQL Talk to display the SQL Talk window.
2. Type the following script in the top pane of the window:

   ```
   Select * from fiscal_year_info;
   ```

3. Press Ctrl-Enter to run the script. The FISCAL_YEAR_INFO table appears in the bottom pane of the window.

   If your organization is using a calendar year configuration the Start_Month column displays a value of 1 and the Fiscal_Period column displays a value of 12. For a fiscal period the values in one or both columns will be different from the calendar year configuration.

4. Close SQL Talk.

Note: Oracle Portfolio Analyzer does not have SQL Talk. If you are using this product, ask your DBA or System Administrator to access the table for you.

Examples of Calendar and Fiscal Year Configurations

The following examples show different configurations in the FISCAL_YEAR_INFO table and how these configurations affect the LEDGER_STAT table.

Using a Calendar Year Configuration

In this example, the FISCAL_YEAR_INFO table holds the following values:

Column	Value
Start_Month	1
Fiscal_Period	12

This is the default setting for OFS applications.

In the LEDGER_STAT table, the calendar months correspond to the monthly buckets, as shown in the following table:

year_s	month_01	month_02	month_03	month_04	month_05	month_06	month_07	month_08	month_09	month_10	month_11	month_12
1999	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEPT	OCT	NOV	DEC

If you want to process a current month Balance Sheet report and you set the as-of-date in your Configuration ID to June 30, 1999, the data for the report comes from the Month_06 bucket.

Similarly, if you want to process an Allocation ID, undo an Allocation ID, or migrate transfer rates to the LEDGER_STAT table, the system updates year-to-date buckets that are affected by the Allocation ID and the Month_06 bucket.

The YTD_XX buckets in the LEDGER_STAT table correspond to the calendar months in the same way as the monthly buckets. For example, YTD_01 contains year-to-date values for January and YTD_06 contains year-to-date values from January through June.

Using a Fiscal Year Configuration with a 12-Month Duration

In this example, the FISCAL_YEAR_INFO table holds the following values:

Column	Value
Start_Month	7
Fiscal_Period	12

In the LEDGER_STAT table, the calendar months correspond to the monthly buckets, as shown in the following table:

year_s	month_01	month_02	month_03	month_04	month_05	month_06	month_07	month_08	month_09	month_10	month_11	month_12
1999	JUL	AUG	SEP	OCT	NOV	DEC	JAN	FEB	MAR	APR	MAY	JUN

In this example, the first month of the fiscal period is July, 1999, and January through June are in the year 2000. The OFS system determines the year of the fiscal period record by the year in which the first month occurs. Therefore, the system treats the months January through June as part of the 1999 record.

If you want to process a current month Balance Sheet report and you set the as-of-date in your Configuration ID to January 31, 2000, the data for the report comes from the Month_07 bucket.

Similarly, if you want to process an Allocation ID, undo an Allocation ID, or migrate transfer rates to the LEDGER_STAT table the system updates the year-to-date buckets that are affected by the Allocation ID and the Month_07 bucket.

Using a Fiscal Year Configuration with a 6-Month Duration

In this example, the FISCAL_YEAR_INFO table holds the following values:

Column	Value
Start_Month	4
Fiscal_Period	6

In the LEDGER_STAT table, the calendar months correspond to the monthly buckets, as shown in the following table:

year_s	month_01	month_02	month_03	month_04	month_05	month_06	month_07	month_08	month_09	month_10	month_11	month_12
1999	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DEC	JAN	FEB	MAR

In this example, the first month of the fiscal period is April, 1999, and January through March are in the year 2000. The OFS system determines the year of the fiscal period record by the year in which the first month occurs. Therefore, the months January through March are treated as part of the 1999 record.

If you want to process a current month Balance Sheet report and you set the as-of-date in your Configuration ID to January 31, 2000, the data for the report comes from the Month_10 bucket.

Year-to-Date Calculations with a 6-Month Duration

When performing year-to-date calculations and the fiscal period value is 6, the OFS system segments the financial year into two 6-month periods. Therefore, if you select August, 1999, for year-to-date calculations your data comes from the months of April, 1999, through August, 1999 (five months). If you select December for year-to-date calculations then your data comes from October, 1999 through December, 1999 (three months).

Using the Undo Function with a Fiscal Year Configuration

The undo function backs out records one period at a time. This is important if you have chosen a fiscal period duration that is less than 12. For a calendar year or 12-month fiscal year you can set the as-of-date anywhere within the 12-month span and undo the allocation for the entire fiscal period.

If your organization uses a fiscal period shorter than 12-months, however, then you need to set the as-of-date within the period for which you want to undo an allocation. For example, if you have set your Fiscal_Period to 6 and you want to undo an allocation that you posted in the first six-month period then you must set the Configuration ID as-of-date to a date within that period.

B

Transformation ID Error Messages

This appendix contains:

- Procedure to determine why a Process ID has failed
- Transformation ID error information

Determining Why a Process ID Has Failed

To determine why a Process ID has failed, consult the Process_Errors table. You can obtain Transformation ID error information after you have identified the Process_Sys_ID number.

To determine the Process_Sys_ID number, complete the following steps:

1. Select the Transformation ID icon. The Select Transformation ID dialog appears.
2. From the Transformation ID drop-down menu, select the Transformation ID you are working on. **Do not select OK**.
3. Right mouse click inside the Select Transformation ID dialog. The Transformation ID Information dialog appears. Make a note of the Sys_id_num (for example, 105877).
4. Select the SQL icon and execute the following SQL statement:

```
Select * From OFSA_Process_Errors
Where Process_Sys_ID = xxxxxx;
```

Press Ctrl+Enter to execute the statement and then look at the results. Each error row includes an error number, as well as other context information that can help you diagnose your process error.

Transformation ID Error Information

The remainder of this appendix contains error code descriptions.

Error Code	Error Message	Error Condition	Effect on Processing
20702	Error: Failed to properly receive params.	The parameters passed to the engine are invalid. This is either an internal error or bad command-line usage.	Processing stops.
20704	Process ID or Configuration ID failed to convert ID where <xxxx>, <yyyy>.	Process ID or Configure ID is not recognized by OFSA (xxxx = Process ID System ID number, and yyyy = Configuration ID System number).	Processing stops.
20712	Critical Memory Allocation Failure.	Internal Application Logic Error.	Processing stops.
20713	Unable to Process.	The transformation engine was unable to perform the requested processing due to an error that is either listed in the Process Errors table or is unknown.	This message appears anytime processing stops for any error, known or unknown.
20714	LoadStart Failed During Read.	Internal Application Logic Error. OR Database Access Layer failure.	Processing stops.
20715	IoBase: Memory Allocation Failure.	Internal Application Logic Error.	Processing stops.
20716	IoMulti: Memory Allocation Failure.	Internal Application Logic Error.	Processing stops.
20717	Bad key allocation.	Internal Application Logic Error.	Processing stops.
20725	LedgerTransform: Fatal Error During Parse: Prepare	Internal logic error.	Processing stops.
20726	LedgerTransform: Fatal Error During Parse: Pivot	Internal logic error.	Processing stops.
20727	LedgerTransform: Fatal Error During Parse: Time	Internal logic error.	Processing stops.

Error Code	Error Message	Error Condition	Effect on Processing
20728	LedgerTransform: Input date is not valid.	The Transformation ID has Ledger Stat as the source type, Processing Option is Selective and either Start date or End date < 01 JAN,1950. OR The Transformation ID has Ledger Stat as the source type, Processing Options is Selective and the Start Date is greater than the End Date. OR The Transformation ID has Ledger Stat as the source type, the Processing Options is Current Month and the activated Configuration ID has an As Of Date < 01 JAN,1950.	Processing stops.
20730	LedgerTransform: Fatal Error During Parse: Output	Internal logic error.	Processing stops.
20731	LedgerTransform: Fatal Error During Parse: Bind Output table	Internal logic error.	Processing stops.
20732	LedgerTransform: Fatal Error During Parse: Writer Prepare	Internal logic error.	Processing stops.
20734	LedgerTransform: Fatal Error. FE Not in DETAIL_ELEM.	Transformation ID has Ledger Stat as the source type and the pivoted group has at least one Financial Element ID that is not being defined in the DETAIL_ELEM table.	Processing stops.
20735	LedgerTransform: Fatal Error. FE No column name defined in DETAIL_ELEM.	The Transformation ID has Ledger Stat as the source type, and at least one Financial Element ID that is defined in DETAIL_ELEM table does not have a corresponding COLUMN_NAME defined.	Processing stops.
20736	LedgerTransform: Error- All dimensions are filtered out.	Transformation ID has Ledger Stat as the source type and all the Dimensions are being filtered out.	Processing stops.

Error Code	Error Message	Error Condition	Effect on Processing
20737	Transformation ID does not exist in table.	A bad Sys_ID_Num is assigned to the Transformation ID.	Processing stops.
20738	Configuration ID does not exist in table.	Bad Sys_ID_Num is assigned to the Configuration ID.	Processing stops.
20739	Processing ID is bad.	There is an error in the format of the Processing ID.	Processing stops.
20741	Transformation ID: Unsupported source table type.	Transformation ID has an Undefined SOURCE_TBL_TYPE_CD saved in IDT_TRANSFORM table.	Processing stops.
20742	Ledger Transform Error: Transformation would cause table overwrite.	Transformation ID has Ledger Stat as the source type, Processing Options is Entire Table, Drop Existing Table is unchecked, and the Transform Into table had already existed in the database.	Processing stops.
20743	Filter is filtering out a financial element that already exists in the table.	The Transformation ID has Ledger Stat as the source type, and the Data Filter has filtered out at least one of its Financial_elem_ID which already exists in the Transform Into table.	Processing stops.
20746	A column for financial element is being added to the table where historic data is plugged with a zero, which causes inconsistent data.	A Ledger Stat Transformation ID is adding data to an existing output table. The added data causes a column to be added to the output table for a new financial element. However, the values for the new financial element column for records that already exist in the output table are plugged with a zero. This is in error since valid values for this financial element for the existing records exist in the input table.	Processing stops.
20755	Fatal Error occurred loading Transformation ID.	Internal logic error.	Processing stops.
20756	Fatal Error occurred loading dimension filter.	Failure in Database Layer.	Processing stops.

Error Code	Error Message	Error Condition	Effect on Processing
20757	Ledger Transform: No Records returned.	Either there are no records in the LEDGER_STAT table, or the filter being used excludes all records.	No records are processed.
20764	RollupTransform: Transformation would cause table overwrite.	The Transformation ID has Tree Rollup as the source type, the Drop Existing Table Flag is unchecked and the Transform Into table already exists in the database.	Processing stops.
20765	RollupTransform: Rollup Sys_ID not found in Level_Desc table.	For the Tree Rollup ID being transformed, no records exist in the Level _Description table.	Processing stops.
20766	RollupTransform: Too many levels found in Level_Desc table.	The value in the LEVEL_NUM COLUMN in the Level Description table exceeds the Maximum number of levels, which is 15.	Processing stops.
20767	RollupTransform: Rollup Sys_ID not found in Catalog of IDs table.	The Tree Rollup ID (SYS_ID_NUM) cannot be found in the CATALOG_OF_IDS table.	Processing stops.
20768	RollupTransform: Leaf not found in Catalog of Leaves table.	The Leaf Type for the Tree Rollup being transformed does not exist in the Catalog_of_Leaves table.	Processing stops.
20769	RMTransform: Source table does not exist.	The Transformation ID has Risk Manager Result Detail as the source type and the table for the Process_ID does not exists in the database.	Processing stops.
20770	RMTransform: Error-Unable to drop existing table(s).	Missing Privileges: The DBA should run Grants in the OFSA System Administration.	Processing stops.
20771	RMTransform: Error: No data was found to transform.	The Transformation ID has Risk Manager Result Detail as the source type, the source table exists in the database but no data was found in the source table.	No records are processed.
20772	RMTransform: Error: No information on product or organization leaf was found.	The Transformation ID has Risk Manager Result Detail as the source type and either Product Leaf or Organization Unit ID in the source table is not defined in The CATALOG_OF_LEAVES table.	Processing stops.

Error Code	Error Message	Error Condition	Effect on Processing
20773	RMTransform: Error- All dimensions are filtered out.	The Transformation ID has Risk Manager Result Detail as the source type and all the Dimensions are being filtered out.	Processing stops.
20774	RMTransform: Inapplicable dimensions found in dimension filter.	The Transformation ID has Risk Manager Result Detail as the source type and the Dimension Filter excludes the Leaf type which is neither the Product Leaf nor the Organization Leaf Type that is being used by the input table.	Processing stops.
20775	RMTransform: Error-Required bucket information not found in RESULT_BUCKET table.	Transformation ID has Risk Manager Result Detail as the source type and at least one Scenario_Num in the source table cannot be found in the RESULT_BUCKET table.	Processing stops.
20776	RMTransform: RESULT_BUCKET has more start date indexes than result table.	The RESULT_BUCKET table expects more scenarios than are contained in the result detail table. Data may be missing from the result detail table.	Note: This is only a warning message.
20777	RMTransform: Error: FE in source table is not in DETAIL_ELEM.	The Transformation ID has Risk Manager Result as the source type, and has at least one Financial Element ID not defined in the DETAIL_ELEM table.	Processing stops.
20778	RMTransform: Error: Prepare Writer failed.	Internal logic error.	Processing stops.
20780	LedgerTransform:Transformation data inconsistent with existing table.	Pivoting Data causes the existing Transform Into table data to become inconsistent.	Processing stops.
20782	Transform: Duplicate Column Name found in DETAIL_ELEM.	Pivoted Data has at least two Financial Elem IDs having the same COLUMN_NAME defined in the DETAIL_ELEM table.	Processing stops.
20783	LedgerTransform: Dimension filter contains a column which is used in Data Filter.	A leaf column is being excluded that is being used by the data filter.	Processing stops.
20784	RMTransform: Error-Unable to create output table.	The Transformation ID was unable to create the Transform Into table or Index.	Processing stops.

Error Code	Error Message	Error Condition	Effect on Processing
20789	Transform: Failed to drop table.	Missing database privilege.	Processing stops.
20790	Transform: Failed to create rollup table.	Run Grants in OFSA System Administration, or see your DBA.	Processing stops.
20791	Transform: Failed to create table.	Run Grants in OFSA System Administration, or see your DBA.	Processing stops.
20792	Transform: Failed to append columns to table.	Run Grants in OFSA System Administration, or see your DBA.	Processing stops.
20793	Transform: Failed to reserve table.	Run Grants in OFSA System Administration, or see your DBA. OR Transform Into table has already been reserved by other User.	Processing stops.

C

ABM/Performance Analyzer Integration

The Oracle financial services vertical provides the most comprehensive solution for the industry. A key to providing a comprehensive footprint for the financial services industry is delivering packaged integration between multiple, specialized software products. The major objective of the Oracle Activity Based Management (ABM) and Oracle Performance Analyzer integration is transferring unit cost data from ABM to Performance Analyzer. ABM transfers unit costs into Performance Analyzer for use in cost allocations. ABM unit costs are primarily used in Performance Analyzer for pricing transaction table data but could prove useful elsewhere as well. Integration of ABM unit costs with Performance Analyzer requires sharing dimensions between the two products.

This appendix supports only ABM version 11i. The ABM / Performance Analyzer Integration Process requires Oracle Financial Services Applications (OFSA) release 4.5.33. Earlier versions of OFSA will not work.

This appendix describes the routines necessary to integrate ABM Activity Rate Sets with Performance Analyzer Table IDs. It divided into the following sections:

- Integration Steps
- New Database Objects
- Mapping Design
- Running Procedures Without Scripting
- Integration Data Examples

Integration Steps

Database Administer Steps

Before running the ABM / Performance Analyzer Integration Process, the administrator has to set up the ABM objects in Oracle Financial Database (FDM) schema, so that the integration scripts can access ABM objects. The following are the ABM tables used by the integration scripts:

- ABM_MLS
- ABM_MDLS
- ABM_MLS_RESOURCES
- ABM_PDS
- ABM_RESOURCES
- ABM_ACT_TAGS
- ABM_TAGS
- ABM_TAG_SETS
- ABM_MLS_RES
- ABM_RE_ACT_RATE_DAT

These ABM tables often reside in a remote instance or in the same instance but different schema. In order for the integration scripts to run, the administrator creates a database link called *abm_dblink*. In order to run the ABM Integration Process, the administrator runs a SQL package creating new objects and registering new views in the FDM schema. After registering the views, the administrator runs grant all in FDM Administration. Finally, the administrator assigns the users who want to run the integration process to the OFDM_R_PA user group.

Creating the ABM_DBLINK

Before running the setup script, the administrator has to create a database link called *abm_dblink*. The following is the syntax to create the database link:

CREATE DATABASE LINK *abm_dblink* CONNECT TO <abm_user> IDENTIFIED BY <abm_password> USING '*<abm_database_connect_string>*'

Summary of the Steps

1. Log in to SQL*Plus as FDM schema.

> **Note:** You must start SQL*Plus at OFSA_INSTALL/dbs/<OFSA_release>/utilities/abm_integration directory.

2. Create abm_dblink, the Database Link to the ABM schema.

> **Note:** This is a one-time-only step. The FDM database TNSNAMES.ORA file must have ABM database access information.

3. Exit SQL*Plus.

Creating ABM/Performance Analyzer Integration Objects

A script creates and registers all of the objects needed for running the ABM Integration process. A prerequisite for running this script is the creation of a database link named *abm_dblink*. This database link must be created, regardless of whether the ABM objects reside in the same instance or in a remote database. This script also registers and assigns table classifications for all of the ABM Integration views. The OFDM_R_PA role has access to all of these views. All of the views are assigned table_classification_cd =510. The OFSA_ABM_DIM_TO_LEAF_MAPPING, ABM_VAL_TO_NODE_MAPPING_V, and ABM_VAL_TO_LEVEL_MAPPING_V objects are also assigned to table_classification_cd=480. This enables access using the Data Verification ID.

Summary of the Steps

1. Log in to SQL*Plus as FDM schema. Note: You must start the SQL*Plus at OFSA_INSTALL / dbs / <OFSA_release> / utilities / abm_integration directory.
2. Execute setup_abm_integration.sql to create the views.
3. SQL>@ setup_abm_integration.

Note: This is a one-time-only step. This script needs to run only once.

4. Exit SQL*Plus.
5. Go to FDM Administration and run grant all.

Overview of User Integration Steps

Before integrating the activity rate set unit cost data from Activity Based Management (ABM) to a Performance Analyzer Table ID, the user needs to set up the mapping between ABM Dimensions (Attributes & Attribute Values) and OFSA Leaf or OFSA Rollup Level Nodes. To set up these mappings the user runs the following scripts:

1. RUN_ABM_DIMENSIONS.SQL
2. RUN_ABM_DIMENSION_VALUES.SQL
3. RUN_CREATE_OFSA_LEAVES.SQL

> **Note:** For a new Model ID, you must run the scripts in the order shown above.

After setting up the Attribute and Attribute Value mapping, users can run the following script to integrate the activity rate set unit cost data from Activity Based Management to a Performance Analyzer Table ID:

RUN_ABM_MAP_TABLE_ID.SQL

> **Note:** Administrator must do the Database Administer Step before running any of the User integration steps. Refer to Database Administer Step section for more details.

If users want to re-map or change one of the mappings, they can run the following script:

RUN_ABM_CLEANUP.SQL.

The scripts keep track of the selections that the user makes and stores the appropriate data in tables. Users can access these tables directly in the database.

The ABM integration run parameters are stored uniquely for each user. Several users can run the integration process, with each user having unique dimension to leaf mappings. There can be only one set of parameters for each unique Master List Set ID and Model ID pair, however. If each different user wants his or her own run parameters, each needs to run different Master List Set ID and Model ID combinations.

All scripts track errors in a log file. The log files are called <script name>#.LOG and are written to the local directory that SQL*PLUS is started from.

Note: Users wanting to run the integration process must have CREATE TABLE and CREATE PROCEDURE privileges.

User Integration Script Details

RUN_ABM_DIMENSIONS.SQL

This script brings the new dimensions for a Master List Set and Model ID combination into the OFSA_ABM_DIM_TO_LEAF_MAPPING table. This enables the user to map ABM (Attributes) to the OFSA Leaf Number ID. The script sets up the ABM Attribute data. Nothing is mapped to OFSA. The user is responsible for editing the values for OFSA leaves in the OFSA_ABM_DIM_TO_LEAF_MAPPING table.

Go to OFSA_INSTALL / dbs / <OFSA_release> / utilities / abm_integration directory. Then, log in to SQL*PLUS and type in the following command to run the script:

```
SQL > @run_abm_dimensions
```

The script prompts for the following parameters:

- Master List Set ID
- Model ID
- Activity Rate Set Name

These parameters are described below:

Master List Set ID The script shows the user a list of available ABM Master List Set IDs. The user selects the Master List Set ID to set up dimension mapping. After the user selects a Master List Set ID, the script does the following check:

Checks that the user entered a valid Master List Set ID. The Master List Set ID must exist in ABM_OBJECT_SET_V.

Note: Master List Set ID is case sensitive.

If the user entered an invalid Master List Set ID, the script terminates and lists the error in the SQL*PLUS window.

Model ID The script shows a list of Model IDs for the chosen Master List Set ID. After the user enters the Model ID for setup, the script performs the following checks:

Make sure the user entered a valid Model ID. The Model ID must exist in ABM_MODEL_V.

Note: Model ID is case sensitive. Also, any errors occurring at this point are logged in setup_abm_val1.log in the log subdirectory and exit out of SQL*PLUS.

Activity Rate Set Name

The script shows the user a list of available ABM Activity Rate Sets for the previously selected Master List Set and Model ID. The user selects the Activity Rate Set serving as the data input for rates. After the user selects a Activity Rate Set, the script does the following check:

Checks that the Activity Rate Set name entered belongs to the previously specified Master List Set and Model ID combination

---

Note: Activity Rate Set name is case sensitive.

If no errors occur, the script queries the ABM (Attributes) that are not in OFSA_ABM_DIM_TO_LEAF_MAPPING table and inserts them into the OFSA_ABM_DIM_TO_LEAF_MAPPING table. Any existing ABM (Attributes) in OFSA_ABM_DIM_TO_LEAF_MAPPING remain untouched. After running the script, the user can use the Data Verification ID to set up the mapping between ABM (Attributes) and OFSA Leaf Number IDs.

The following list shows the requirements to set up mapping OFSA leaves to ABM (Attributes):

- Mapped OFSA Leaf Number ID must exist in OFSA_CATALOG_OF_LEAVES and it cannot be a virtual leaf.
- The relationship between OFSA Leaf Number ID and ABM Attribute ID is one to one. This means that one OFSA Leaf Number ID must map to one and only one ABM dimension or ABM dimension attribute.
- Users cannot change the mapped OFSA Leaf Number ID after running run_abm_dimensions.sql. They should use run_abm_cleanup.sql to clear the dimension value setup before changing the mapped OFSA Leaf Number ID.

The script does not support the following conditions:

- The script does not detect the removal of one of the mapped Attributes from ABM.
- The script does not delete the missing dimension in the OFSA_ABM_DIM_TO_LEAF_MAPPING table.

(Users can use run_abm_cleanup.sql to remove the Attribute, however.)

All script errors write to a log file. The log file is called *RUN_ABM_DIMENSIONS.LOG* and is written to the local directory that SQL*PLUS is started from.

RUN_ABM_DIMENSION_VALUES.SQL

This step brings the ABM Attribute values for a designated Master List Set ID and Model ID combination into the FDM database. Attribute values are inserted into either OFSA_ABM_VAL_TO_NODE_MAPPING table or the OFSA_ABM_VAL_TO_LEVEL_MAPPING table, depending upon whether the user is mapping by Leaf values, or by Rollup level. OFSA_ABM_VAL_TO_NODE_MAPPING is used to set up the mapping of OFSA leaf values to ABM Attribute Values, while OFSA_ABM_VAL_TO_NODE_MAPPING maps OFSA Rollup Nodes to ABM Attribute Values. This script sets up only the ABM Attribute values in the Master List Set. Nothing is mapped to OFSA yet.

This script really performs two roles:

- It enables the users to select the ABM Attribute used in the construct of a Performance Analyzer Table ID. The user also indicates whether each ABM Attribute maps to a leaf or rollup and on which tab of the Table ID the Attribute appears.
- It seeds the ABM Attribute values into the appropriate leaf or rollup mapping tables.

Go to OFSA_INSTALL / dbs / <OFSA_release> / utilities / abm_integration directory. Then, log in to SQL*PLUS and type in the following command to run the script:

```
SQL > @run_abm_dimension_values
```

The script prompts you for the following parameters:

- Dimensions to be used in the Table ID
- Mapping Options (L = Leaf Values or R = Rollup Level Nodes)
- Tab number for dimension
- Target or non-target leaf
- Rollup System ID
- Rollup Level Number

These parameters are described below:

The script shows the Master List Set ID and Model ID that is set up in the OFSA_ABM_DIM_TO_LEAF_MAPPING table from the *run_abm_dimensions* script. A list of all available Attributes for the Master List Set ID and Model ID is displayed and the user is allowed to chose as many as four of those Attributes for inclusion in the Table ID.Mapping Options.

For each new Dimension ID chosen in the Model ID of a Master List Set ID, the script prompts the user for the dimension value mapping options. The mapping options are:

1. Leaf or Rollup mapping
2. Table ID destination tab
3. Target or non-target leaf
4. If rollup, sys_id_num and node level of rollup

Rollup System ID This prompt appears if the user picks a rollup level mapping option for the Attribute ID. After the user enters the Rollup System ID, the script checks that the user entered a valid Rollup System ID.

Rollup Level Number This prompt appears if the user picks rollup level mapping option for the Attribute ID. After the user enters the Rollup level, the script checks that the Rollup level exist in the Rollup Sys ID.

Note: Any error occurring at this point are logged in setup_abm_val2_2.log in the log directory and exit out SQL*PLUS.

After all the information is gathered, the script inserts the mapping options, the Rollup System ID, and the Rollup level into the OFSA_ABM_RUN_DIM_DTL table. Then it inserts all the new Attribute values in either OFSA_ABM_VAL_TO_NODE_MAPPING or OFSA_ABM_VAL_TO_LEVEL_MAPPING depending on the mapping options. Next, the user can use the Data Verification ID to set up the mapping. (Use the views to set up the mapping).

There are three ways the user can set up the mapping between ABM Attribute values and OFSA leaf values in OFSA_ABM_VAL_TO_NODE_MAPPING:

1. There is a match (existing) OFSA leaf value. The user finds the matching OFSA leaf value and then manually enters the OFSA leaf number in OFSA_LEAF_NODE column in OFSA_ABM_VAL_TO_NODE_MAPPING table.
2. There is no matching (existing) OFSA leaf value and the user wants the integration process to create it. The user cannot find any matching OFSA leaf values and would like to have the integration process create the new leaf values. The user does not need to enter any value in the OFSA_ABM_VAL_TO_NODE_MAPPING table. The user runs RUN_CREATE_OFSA_LEAVES.SQL to create new leaf values.
3. There is no matching (existing) OFSA leaf value and the user wants to create his or her own leaf numbers. The user cannot find any matching OFSA leaf values and wants to create their own leaf node number in the OFSA_ABM_VAL_TO_NODE_MAPPING table. The user enters values in the OFSA_LEAF_NODE column then runs RUN_CREATE_OFSA_LEAVES.SQL creating the new leaf values.

Note: If you select option 2, make sure you do not have one OFSA leaf node mapped to more than one ABM dimension value because run_create_ofsa_leaves.sql does not know which dimension name to use to create the new leaf node.

Unsupported Condition:

1. If one of the mapped ABM Attribute values has been removed in ABM, this script does not delete the missing Attribute value in the OFSA_ABM_VAL_TO_NODE_MAPPING or OFSA_ABM_VAL_TO_LEVEL_MAPPING tables.
2. When the user changes the mapping between an ABM Attribute value and OFSA leaf value in OFSA_ABM_VAL_TO_NODE_MAPPING or OFSA_ABM_VAL_TO_LEVEL_MAPPING tables, the script does not update the mapped Table ID.

All script errors write to a log file. The log files are called *RUN_ABM_DIMENSION_VALUES#.LOG* and are written to the local directory that SQL*PLUS is started from. As many as three different log files are recorded during this script.

RUN_CREATE_OFSA_LEAVES.SQL

This script creates new OFSA Leaf values for any ABM Attribute Values in OFSA_ABM_VAL_TO_NODE_MAPPING that are not currently mapped to an existing OFSA Leaf value.

Running this script is required only when there is OFSA_LEAF_NODE values' setup in OFSA_ABM_VAL_TO_NODE_MAPPING that do not exist in the OFSA_LEAF_DESC table.

The script checks for the following:

1. The leaf value is not a reserved leaf value.(OFSA_LEAF_NODE >10000 for Financial Element ID)
2. One leaf value is mapped to one Attribute value in each Master List Set ID. This script does not know which Attribute name to use when creating new leaf values if the new leaf value is mapped to more than one Attribute value.

If there are new leaf values created in OFSA, the script shows a list of Attributes and the number of new leaf values created per Attribute.

If there are no leaf values created in OFSA, this script tells the user and exits.

This script scans through all values in OFSA_ABM_VAL_TO_NODE_MAPPING table. Any OFSA_LEAF_NODE columns with values that do not exist in OFSA_LEAF_DESC are created in OFSA.

All script errors write to a log file. The log file is called *RUN_CREATE_OFSA_LEAVES.LOG* and is written to the local directory that SQL*PLUS is started from.

RUN_ABM_MAP_TABLE_ID.SQL

This script brings over the ABM activity cost information into FDM database and creates a Table ID to store it. The script enables the user to choose creating a new Table ID or overwriting an existing Table ID. When creating a new Table ID, the script prompts for the Table ID name and Folder Name. When overwriting an existing Table ID, the script prompts for the sys_id_num of the existing Table ID. The Attributes previously specified in RUN_ABM_DIMENSION_VALUES.SQL above are used to appropriately populate the Table ID.

Go to OFSA_INSTALL/dbs/<OFSA_release>/utilities/abm_integration directory. Then, log in to SQL*PLUS and type in the following command to run the script:

```
SQL > @run_abm_map_table_id
```

The script prompts you for the following parameters:

- Periodic Data Set ID
- Mapping Table ID Options and New or existing Table ID

These parameters are described below:

Periodic Data Set ID The script shows a list of Periodic Data Set IDs available for the previously specified Model ID. After the user enters the Periodic Data Set ID, the script checks for the following:

1. The user entered an valid Periodic Set Data ID.
2. All the ABM Attribute values in this period are set up in OFSA_ABM_VAL_TO_NODE_MAPPING.
3. All the ABM Attribute values in this period are set up in OFSA_ABM_VAL_TO_LEVEL_MAPPING.
4. No more than one driver is assigning to one resource ID (combination of Resource/Department/Activity).

> **Note:** Periodic Data Set ID is case sensitive.

Mapping Table ID Options and New or existing Table ID The script asks the user whether he or she wants to map ABM Unit Costs to a new Table ID or an existing Table ID.

If the user wants to create a new Table ID, then the script prompts for the Table ID name and the folder name for the new ID. The script only shows the list of folder names that the user has access to. After the user enters the Table ID name and the folder name, the script makes sure that:

1. The user entered a valid folder name
2. The Table ID name does not already exist in OFSA

Note: Folder Name is not case sensitive.

If the user wants to replace a Table ID, then the script prompts the Table System ID Number to replace. After the user enters the Table System ID Number, the script checks that the Table ID Sys ID Number is valid.

If no errors occur, the script transforms the ABM Activity Unit Cost row data to column data in OFSA_ABM_TEMP_MAPPING table.

If no error, the script starts to integrate ABM Activity Unit Cost to OFSA Table ID.

The user can go to OFSA_ABM_MAP_TABLE_ID to check which Table ID the Activity Rate Set map to.

Unsupported Condition: The Table ID creation script is a one-to-one mapping of activity cost in ABM to Table ID row in Performance Analyzer. The process does not summarize multiple activity rates into a single row in a Table ID. The process also cannot process data from activity rate sets that have multiple drivers associated to a single activity. When multiple drivers are associated with a single activity, ABM creates multiple rows in the storage table. The script that creates the Table IDs cannot determine which row to use and fails.

All script errors write to a log file. The log files are called *RUN_ABM_MAP_TABLE_ID#.LOG* and are written to the local directory that SQL*PLUS is started from. As many as three different log files are recorded during this script.

RUN_ABM_CLEANUP.SQL

This script enables the user to remove previously created ABM Integration information. The user can remove information by Master List Set ID, by Model ID, or for a specific Dimension ID.

This script provides the following cleanup operations:

- Remove all setup information for a Master List Set ID (This operation cleanups the entire dimension mapping, mapping options and dimension value mappings that belong to a Master List Set ID.)
- Remove one Dimension ID set up in a Master List Set ID (This operation cleanups one of the dimensions in a Master List Set ID. It removes the dimension mapping, mapping options and dimension value mapping for the specified Dimension ID.)
- Remove a dimension value mapping option in a Model ID (This operation cleanups one Dimension ID's options. It does not delete the Dimension ID mapping. This step enables deletion of mapped dimension values.)

Go to OFSA_INSTALL / dbs / <OFSA_release> / utilities / abm_integration directory. Then, log in to SQL*PLUS and type in the following command to run the script:

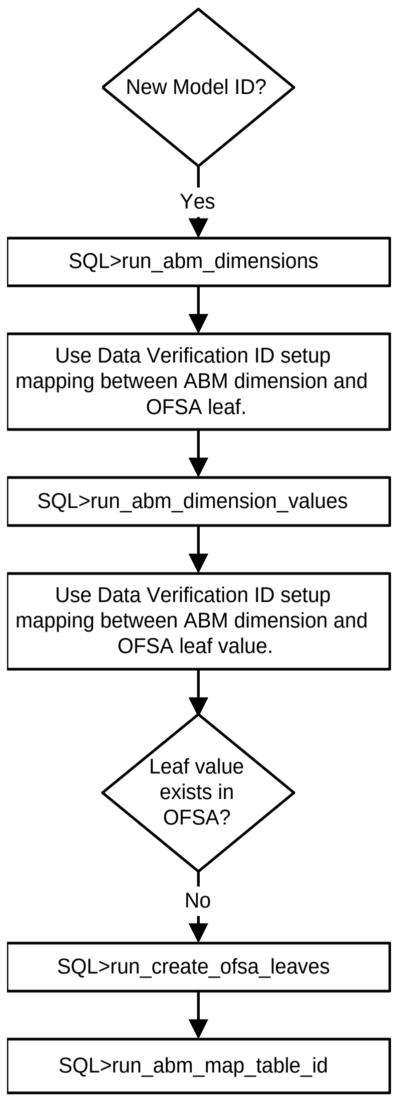
```
SQL > @run_abm_cleanup
```

The script prompts the user for the Master List Set ID and the Model ID. From the entered Master List Set ID and Model ID a list of dimensions is displayed. The user chooses a dimension and the desired clean up option.

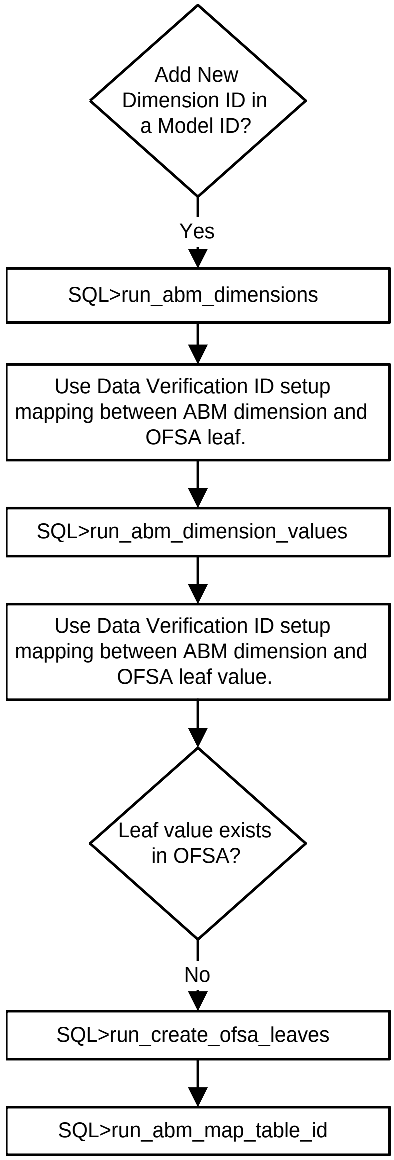
All script errors write to a log file. The log files are called *RUN_ABM_CLEANUP#.LOG* and are written to the local directory that SQL*PLUS is started from. As many as three different log files are recorded during this script.

Integration Setup Cases

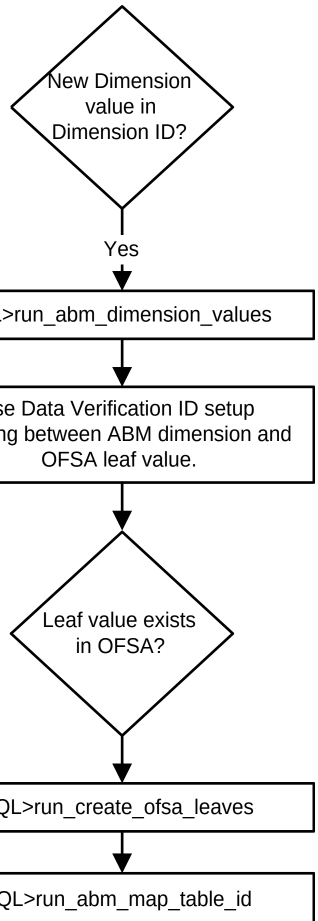
The following flow charts show examples of how the client would run the scripts:



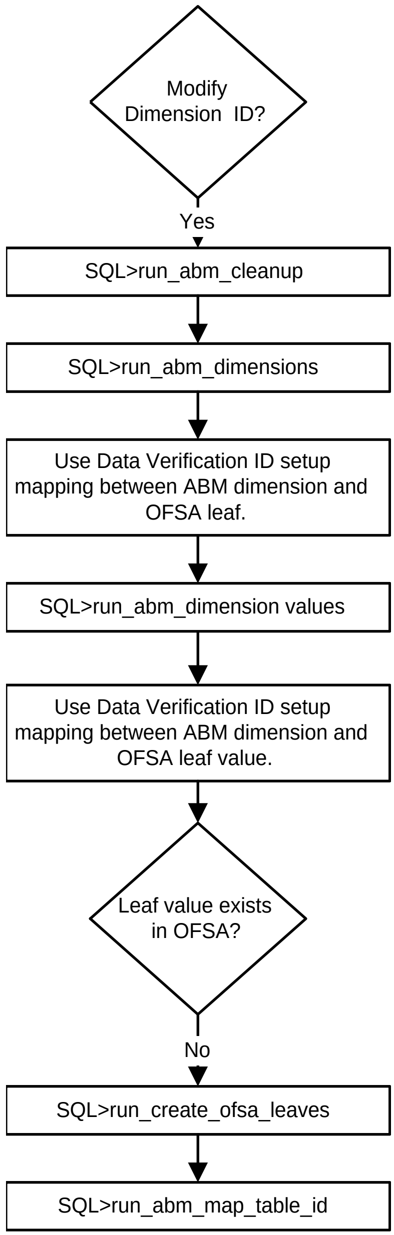
Case 1


Add New Dimension ID in a Model ID?
Yes
SQL>run_abm_dimensions
Use Data Verification ID setup mapping between ABM dimension and OFSA leaf.
SQL>run_abm_dimension_values
Use Data Verification ID setup mapping between ABM dimension and OFSA leaf value.
Leaf value exists in OFSA?
No
SQL>run_create_ofsa_leaves
SQL>run_abm_map_table_id

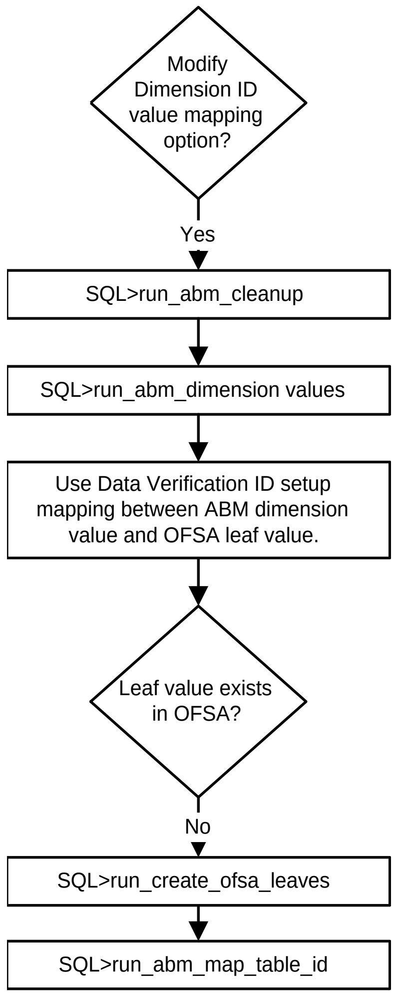
Case 2


New Dimension value in Dimension ID?
Yes
SQL>run_abm_dimension_values
Use Data Verification ID setup mapping between ABM dimension and OFSA leaf value.
Leaf value exists in OFSA?
SQL>run_create_ofsa_leaves
SQL>run_abm_map_table_id

Case 3


Modify Dimension ID?
Yes
SQL>run_abm_cleanup
SQL>run_abm_dimensions
Use Data Verification ID setup mapping between ABM dimension and OFSA leaf.
SQL>run_abm_dimension values
Use Data Verification ID setup mapping between ABM dimension and OFSA leaf value.
Leaf value exists in OFSA?
No
SQL>run_create_ofsa_leaves
SQL>run_abm_map_table_id

Case 4


Modify Dimension ID value mapping option?
Yes
SQL>run_abm_cleanup
SQL>run_abm_dimension values
Use Data Verification ID setup mapping between ABM dimension value and OFSA leaf value.
Leaf value exists in OFSA?
No
SQL>run_create_ofsa_leaves
SQL>run_abm_map_table_id

Case 5

New Database Objects

The integration process requires the creation of several tables and stored procedures to store the mapping information. These tables are used to set up the mapping before processing.

Table Detail

OFSA_ABM_DIM_TO_LEAF_MAPPING

This table stores the mapping between ABM dimensions and OFSA leaves. Users need to set up the mapping between ABM_DIMENSION_ID and OFSA_LEAF_NUM_ID values in this table. Users can use the Data Verification ID to fill in values for the OFSA_LEAF_NUM_ID columns. This table appears in Data Verification ID as *ABM Dimension To Leaf Mapping* (display name).

The following table lists the table structure:

COLUMN_NAME	NULL?	TYPE	DESCRIPTION
ABM_MASTER_LIST_SET_ID	NOT NULL	VARCHAR2(15)	Stores ABM master list sets id. Its dimensions are mapped to OFSA leaf. It must exist in ABM_OBJECT_SET_V.
ABM_DIMENSION_ID	NOT NULL	VARCHAR2(30)	Stores ABM Dimension ID that belongs to the Master List Set ID. This dimension can be either ABM Report Entity (RE) or ABM Activity's Attribute.
OFSA_LEAF_NUM_ID		NUMBER(5)	Mapping OFSA Leaf Number ID. This leaf number must exist in OFSA_CATALOG_OF_LEAVES. This column cannot be a virtual leaf.

Unique Constraints The primary key is ABM_MASTER_LIST_SET_ID, ABM_DIMENSION_ID.

OFSA_IDT_ABM_RUN_DIM

This table identifies the ABM Master List Set ID and Model ID used to run the *run_abm_dimensions* and *run_abm_dimension_values* steps of the ABM Integration process. The run parameters are specified by username. Users are allowed to run only a single set of parameters per username.

COLUMN_NAME	NULL?	TYPE	DESCRIPTION
ABM_MASTER_LIST_SET_ID	NOT NULL	VARCHAR2(15)	Stores the ABM master list set id that designates the ABM dimensions to be retrieved into the FDM database.
ABM_MODEL_ID	NOT NULL	VARCHAR2(15)	Stores the ABM Model ID that designates the ABM Dimension values to be retrieved into the FDM database.
USERNAME	NOT NULL	VARCHAR2(30)	Identifies the user for the run parameters.
ACTIVE_RATE_SET_ID	NOT NULL	VARCHAR2(30)	Stores the Active Rate Set name from which the rates are retrieved.

Unique Constraints

- The primary key is ABM_MASTER_LIST_SET_ID, ABM_MODEL_ID.
- An alternative unique key is USERNAME. This restricts each user to one Master List Set and Model ID combination at a time.

OFSA_ABM_RUN_DIM_DTL

The table identifies ABM Dimension IDs used for ABM Integration. The Dimension IDs entered in this table must exist in ABM for the Master List Set ID and Model ID designated in the OFSA_IDT_ABM_DIM table for the user running the procedure.

COLUMN_NAME	NULL?	TYPE	DESCRIPTION
ABM_MASTER_LIST_SET_ID	NOT NULL	VARCHAR2(15)	Stores the ABM master list set id that designates the ABM dimensions to be retrieved into the FDM database.
ABM_MODEL_ID	NOT NULL	VARCHAR2(15)	Stores the ABM Model ID that designates the ABM Dimension values to be retrieved into the FDM database.
ABM_DIMENSION_ID	NOT NULL	VARCHAR2(30)	Identifies the Dimension ID. Must exist in the Master List Set ID and Model ID specified in the OFSA_IDT_ABM_DIM table for the same username.
OFSA_TABLE_ID_LEVEL_ NUM	NOT NULL	VARCHAR2(1)	Designates the OFSA Table ID Tab# in which the Dimension will be placed. OFSA Table IDs have a maximum of four tabs, so this number must be between 1 and 4.
TARGET_FLG	NOT NULL	NUMBER(1)	Identifies whether the Dimension ID is a Target Leaf in the Table ID. A *1* indicates that the Dimension ID is a target leaf.
ROLLUP_LEVEL_FLG	NOT NULL	NUMBER(1)	Designates whether a Tree Rollup ID is to be employed in the Table ID for the designated Dimension ID. A *1* indicates that a Tree Rollup will be used.

OFSA_ROLLUP_SYS_ID		NUMBER(10)	Identifies the Tree Rollup used in the Table ID for the designated Dimension ID.
OFSA_ROLLUP_LEVEL_NUM		NUMBER(5)	Identifies the Level in the Tree Rollup used in the Table ID for the designated Dimension ID.

Unique Constraints The primary key is ABM_MASTER_LIST_SET_ID, ABM_MODEL_ID, ABM_DIMENSION_ID.

OFSA_IDT_ABM_MAP_TABLE_ID

This table identifies the run parameters creating the Table ID storing the ABM integration data. The Table ID parameters are specified by username. Users are allowed to run only a single set of Table ID parameters per username.

COLUMN_NAME	NULL?	TYPE	DESCRIPTION
USERNAME	NOT NULL	VARCHAR2(30)	Identifies the user for the Dimension parameters.
ABM_MASTER_LIST_SET_ID	NOT NULL	VARCHAR2(15)	Identifies the Master List Set ID from which the ABM Dimension data is retrieved into the FDM database.
ABM_MODEL_ID	NOT NULL	VARCHAR2(15)	Identifies the Model ID from which the ABM Dimension data is retrieved into the FDM database.
ABM_PERIODIC_DATA_SET_ID	NOT NULL	VARCHAR2(30)	Identifies the Periodic Data Set. Must exist in the Master List Set ID and Model ID specified above.
NEW_TABLE_ID_FLG	NOT NULL	NUMBER(1)	A *1* indicates that the run_map_table_id process will create a new Table ID in Performance Analyzer. Any other value designates that the process will overwrite the existing Table ID specified in existing_ table_id_sys_id.
NEW_TABLE_ID_DESC_ SHORT	NOT NULL	VARCHAR2(15)	Designates the name for the new Table ID to be created in the FDM database (only relevant if new_table_id_flg=1).

NEW_TABLE_ID_FOLDER_ NAME	NOT NULL	VARCHAR2(30)	Designates the Folder Name where the new Table ID will be created (only relevant if new_ table_id_flg=1).
EXISTING_TABLE_ID_SYS_ ID		NUMBER(10)	Designates the sys_id_ num in OFSA_ CATALOG_OF_IDS of the Table ID to be overwritten by the process (only relevant if new_table_id_flg<>1).

Unique Constraints The primary key is USERNAME.

OFSA_ABM_VAL_TO_NODE_MAPPING

This table stores the mapping between ABM dimension values and OFSA leaf values. It is the based table for the view OFSA_ABM_TO_NODE_MAPPING_V. Users set up the mapping of the OFSA_LEAF_NODE column using the view OFSA_ABM_VAL_TO_NODE_MAPPING_V (Refer to View Detail) that populates this table. Users can use Data Verification ID to set up OFSA_LEAF_NODE column.

Mapping between ABM dimension value and OFSA leaf value is global within a Master List Set ID. That means all Model IDs in a Master List Set ID are using the same dimension value mapping in this table.

The following table lists the table structure:

COLUMN_NAME	NULL?	TYPE	DESCRIPTION
ABM_MASTER_LIST_SET_ID	NOT NULL	VARCHAR2(15)	Stores ABM master list sets id where its dimensions and dimension values are mapped to OFSA leaf and OFSA leaf values. It must exist in OFSA_ ABM_DIM_TO_LEAF_ MAPPING table.
ABM_DIMENSION_ID	NOT NULL	VARCHAR2(30)	Stores ABM Dimension ID that belongs to the Master List Sets ID. This dimension ID must exist in OFSA_ABM_DIM_ TO_LEAF_MAPPING
ABM_DIMENSION_VALUE	NOT NULL	VARCHAR2(30)	Store the mapping ABM dimension value.
OFSA_LEAF_NODE		NUMBER(14)	Store the mapping OFSA Leaf Node value.

Unique Constraints The primary key is ABM_MASTER_LIST_SET_ID, ABM_ DIMENSION_ID, and ABM_DIMENSION_VALUE.

OFSA_ABM_VAL_TO_LEVEL_MAPPING

This table stores the mapping between ABM dimension value and OFSA rollup level nodes. It is the base table for OFSA_ABM_TO_LEVEL_MAP_V view. Users need to enter values in the OFSA_ROLLUP_LEVEL_NODE column using the view OFSA_ABM_VAL_TO_LEVEL_MAP_V (Refer to View Detail) that populates this table. Users can use Data Verification ID to set up the OFSA_ROLLUP_LEVEL_NODE column.

Mapping between ABM dimension value and OFSA rollup level node is Model ID specific, which means each Model ID in a Master List Set ID has its own mapping between ABM dimension values and OFSA rollup nodes.

The following table lists the table structure:

COLUMN_NAME	NULL?	TYPE	DESCRIPTION
ABM_MASTER_LIST_SETS_ID	NOT NULL	VARCHAR2(15)	Stores ABM master list sets id where its dimensions and dimension value are mapped to OFSA leaf and OFSA leaf value. It must exist in OFSA_ABM_VAL_MAPPING_OPTIONS table.
ABM_MODEL_ID	NOT NULL	VARCHAR2(15)	Stores ABM Model id where its dimension values are mapped to OFSA rollup level node. It must exist in OFSA_ABM_VAL_MAPPING_OPTIONS table.
ABM_DIMENSION_ID	NOT NULL	VARCHAR2(30)	Stores ABM Dimension ID that belongs to the Master List Sets ID. It must exist in OFSA_ABM_DIM_TO_LEAF_MAPPING table.
ABM_DIMENSION_VALUE	NOT NULL	VARCHAR2(30)	Stores the mapping ABM dimension value.

OFSA_ROLLUP_LEVEL_NODE		NUMBER(14)	Stores the mapping OFSA Rollup Level Node Value. This value must exist in the Rollup Sys ID and Level number specified in OFSA_ ABM_VAL_ MAPPING_OPTIONS.

Unique Constraints The primary key is ABM_MASTER_LIST_SET_ID, ABM_ MODEL_ID, ABM_DIMENSION_ID, ABM_DIMENSION_VALUE.

OFSA_ABM_CLEANUP

This table identifies the run parameters for the ABM Integration *cleanup* process The run parameters are specified by username. Users can run only a single set of run parameters per username.

COLUMN_NAME	NULL?	TYPE	DESCRIPTION
USERNAME	NOT NULL	VARCHAR2(30)	Identifies the user for the cleanup parameters.
CLEANUP_TYPE_CD	NOT NULL	NUMBER(5)	Designates how the cleanup process will remove data from the FDM database. The Cleanup Code values are described below.
ABM_MASTER_LIST_SET_ID	NOT NULL	VARCHAR2(15)	Identifies the Master List Set ID for which ABM Integration data will be removed from the FDM database.
ABM_MODEL_ID		VARCHAR2(15)	Identifies the Model ID for which ABM Integration data will be removed from the FDM database. This is populated only if the Cleanup_type_cd is 3.
ABM_DIMENSION_ID		VARCHAR2(30)	Identifies the Dimension ID for the cleanup. This is populated only if the Cleanup_type_cd is 2 or 3.
GLOBAL_FLG		NUMBER(1)	When cleaning up by Dimension ID, this designates that Dimension values will be removed globally across all Master List Set IDs and Model IDs from the ABM Integration data in the FDM database.

Cleanup Code Values

1. Remove ABM Integration for a Master List Set ID
2. Remove ABM Integration for one Dimension ID in Master List Set ID.
3. Remove ABM Integration for a dimension value map option in a Model ID (Enables deletion of mapped dimension values)

Unique Constraints The primary key is USERNAME.

View Detail

OFSA_ABM_VAL_TO_NODE_MAPPING_V

This view is derived from the following tables:

- OFSA_ABM_VAL_TO_NODE_MAPPING
- OFSA_ABM_DIM_TO_LEAF_MAPPING
- OFSA_CATALOG_OF_LEAVES.

It has all the columns in OFSA_ABM_VAL_TO_NODE_MAPPING and two extra columns: OFSA_LEAF_NUM_ID and OFSA_LEAF_DESCRIPTION. Users make use of this view in Data Verification IDs to set up the mapping between ABM dimension values and OFSA leaf values. This view appears in Data Verification ID pick list, not the base table. This view appears as *ABM Value To Leaf Node Mapping V* (display name) in the Data Verification ID.

The following table shows the structure of the view:

Column Name	Null?	Type
ABM_MASTER_LIST_SET_ID	NOT NULL	VARCHAR2(5)
ABM_DIMENSION_ID	NOT NULL	VARCHAR2(10)
ABM_DIMENSION_VALUE	NOT NULL	VARCHAR2(10)
OFSA_LEAF_NUM_ID		NUMBER(5)
OFSA_LEAF_DESCRIPTION		VARCHAR2(80)
OFSA_LEAF_NODE		NUMBER(14)

OFSA_ABM_VAL_TO_LEVEL_MAP_V

This view is derived from the following tables:

- OFSA_ABM_VAL_TO_LEVEL_MAPPING
- OFSA_ABM_DIM_TO_LEAF_MAPPING
- OFSA_ABM_VAL_MAPPING_OPTIONS
- OFSA_CATALOG_OF_LEAVES
- OFSA_LEVEL_DESC.

It has all the columns in OFSA_ABM_VAL_TO_LEVEL_MAPPING and six extra columns: OFSA_LEAF_NUM_ID, OFSA_LEAF_DESCRIPTION, OFSA_ROLLUP_SYS_ID, OFSA_ROLLUP_LEVEL_NUM, and OFSA_ROLLUP_LEVEL_DESC. Users make use of this view to set up the mapping between ABM dimension values and OFSA rollup level nodes. This view appears in Data Verification ID pick list, not the base table. This view appears as *ABM Value To Rollup Level Node Mapping V* (display name) in Data Verification IDs.

Column Name	Null?	Type
ABM_MASTER_LIST_SET_ID	NOT NULL	VARCHAR2(5)
ABM_MODEL_ID	NOT NULL	VARCHAR2(5)
ABM_DIMENSION_ID	NOT NULL	VARCHAR2(10)
ABM_DIMENSION_VALUE	NOT NULL	VARCHAR2(10)
OFSA_LEAF_NUM_ID		NUMBER(5)
OFSA_LEAF_DESCRIPTION		VARCHAR2(80)
OFSA_ROLLUP_SYS_ID		NUMBER(10)
OFSA_ROLLUP_LEVEL_NUM		NUMBER(5)
OFSA_ROLLUP_LEVEL_DESC		VARCHAR2(80)
OFSA_ROLLUP_LEVEL_NODE		NUMBER(14)

Metadata

Table Classification Code

The ABM/Performance Analyzer integration requires the addition of a new table classification code, 510, ABM Integration Objects. All ABM integration related tables or views have this table classification code assignment. In addition, tables or views used in the ABM/Performance Analyzer integration that require user edits or setup have a table classification code of 480. With table classification code 480, tables or views appear in the pick list in Data Verification IDs.

The following table shows the table classification assignments:

Table/View Name	Table Classification Code
OFSA_ABM_DIM_TO_LEAF_MAPPING	510,480
OFSA_ABM_VAL_MAPPING_OPTIONS	510,480
OFSA_ABM_VAL_TO_NODE_MAPPING	510
OFSA_ABM_VAL_TO_LEVEL_MAPPING	510
OFSA_ABM_VAL_TO_NODE_MAPPING_V	510,480
OFSA_ABM_VAL_TO_LEVEL_MAP_V	510,480
OFSA_ABM_TEMP_MAPPING	510

Database Privilege Assignment

OFDM_R_PA role has UPDATE, DELETE, INSERT, SELECT for all of above tables and views.

Mapping Design

There are two kinds of dimensions in Activity Based Management (ABM) that represent activity rate set data: Activity and Department.

Activity Dimension

One Activity can map to one or more OFSA Leaves. Activity, ACCN, for example, can mean COMMON_COA_ID + GL_ACCOUNT_ID+TRANS_ID in OFSA. In the Activity, there are Attributes associated with it and these Attributes become one of the integration process ABM Dimensions that is mapped to one of the OFSA Leaves. For each Attribute of an Activity, there is a list of values associated with it. This List of values becomes ABM Dimension Values in the integration process. This List of Values (Dimension values) is mapped to one or more OFSA leaf values or OFSA rollup level node.

The following lists the ABM tables that have Activity Dimension information:

Table	Content
ABM_TAG_SETS	Tag Set ID information
ABM_TAGS	Tag ID information
ABM_ACT_TAGS	Information about relationship among Activity, Tag Set ID, and Tag ID

Department Dimension

The Department Dimension can map to only one OFSA Leaf. There is no need to associate Attributes with this dimension, so a default dimension called *RE* is automatically created to represent this dimension. In ABM, there are Department values associated with the dimension and these Department values becomes ABM dimension values in the integration process. For each Attribute of an Activity, there is a list of values associated with it. This List of values becomes ABM Dimension Values in the integration process. Each attribute of Activity (Dimension value) is mapped to one OFSA leaf values or OFSA rollup level node.

ABM_MLS_RES is the only ABM table that contains Department information.

In ABM, the Activity must be associated with a Department. The combination of Activity and Department becomes a Bill Line ID. The following lists the tables that link the Activity, Department, and Bill Line ID:

Table	Content
ABM_MLS_RESOURCES	Bill Line IDs information (which Activity and Department become a Bill Line ID)
ABM_RESOURCES	Bill Line ID information that is used by which Model IDs

All the Model IDs and their dimension IDs and dimension values information are gathered in a view called *ABM_RESOURCES_V*.

Running Procedures Without Scripting

All of the five main scripts of the ABM/ Performance Analyzer Integration Process can run without the use of the scripting. Users can opt for running the procedures as executables. When running the procedures without the scripting, users are responsible for filling the necessary parameters into the appropriate tables before processing. The executables are named the same as the five main scripts described earlier in the User Integration Script Details section.

Go to OFSA_INSTALL/dbs/<OFSA_release>/utilities/abm_integration directory. Then, log in to SQL*PLUS and type in the following command to run the executable:

```
SQL > EXECUTE OFSA_ABM_INTEGRATION.run_abm_cleanup
```

The executable runs the same as the scripted process. All parameters that were prompted in the scripts are now pulled from the database tables.

Integration Data Examples

ABM_RESOURCES_V

MASTER_ LIST_SET_ID	MODEL_ ID	RESOURCE_ ID	ACTIVITY_ ID	DIMENSION _ID	DIMENSION_ VALUE	DIMENSION_ NAME
IBANK	UCOST	100000-ACCN	ACCN	CHANNEL_ ID	ALL	No channel
IBANK	UCOST	100000-ACCN	ACCN	PRODUCT_ ID	CREDITCAR D	Credit Cards
IBANK	UCOST	100000-ACCN	ACCN	TRAN_ID	OPENACCT	Open Account
IBANK	UCOST	100000-ACCN		RE	100000	**1Cost Accounting
IBANK	UCOST	100000-ACMC	ACMC	CHANNEL_ ID	ALL	No channel
IBANK	UCOST	100000-ACMC	ACMC	PRODUCT_ ID	COMMERCI AL	Commercial Loan
IBANK	UCOST	100000-ACMC	ACMC	TRAN_ID	CLOSEACCT	Close Account
IBANK	UCOST	100000-ACMC		RE	100000	**1Cost Accounting
IBANK	UCOST	100000-MALD	MALD	CHANNEL_ ID	ATM	ATM transactions
IBANK	UCOST	100000-MALD	MALD	PRODUCT_ ID	NONE	No Product
IBANK	UCOST	100000-MALD	MALD	TRAN_ID	DEPOSIT	Deposit to Account
IBANK	UCOST	100000-MALD		RE	100000	**1Cost Accounting

ABM_PDS_ACT_RATE_V

MASTER_ LIST_SET_ ID	MODEL _ID	MODEL _ NAME	PDS_ID	PDS_ NAME	RESOURCE _ID	ACTIVIT Y_ID	REPORTIN GENTITY_ ID	LOCAL_ TOTAL_ RATE
IBANK	UCOST	Unit Cost Creatio n	Dec-99	Dec-99	100000- ACCN	ACCN	100000	23.2573954
IBANK	UCOST	Unit Cost Creatio n	Dec-99	Dec-99	100000- ACMC	ACMC	100000	165.094988 2
IBANK	UCOST	Unit Cost Creatio n	Dec-99	Dec-99	100000- MALD	MALD	100000	1.15019648

OFSA_ABM_DIM_TO_LEAF_MAPPING

ABM_MASTER_LIST_ SET_ID	ABM_DIMENSION_ID	OFSA_LEAF_NUM_ID
IBANK	CHANNEL_ID	9
IBANK	PRODUCT_ID	3
IBANK	RE	1
IBANK	TRAN_ID	10

OFSA_ABM_RUN_DIM_DTL

ABM _MASTER_ LIST _SET_ID	ABM_ MODEL_ID	ABM_ DIMENSION _ID	OFSA _TABLE _ID _LEVEL _NUM	TARGET _FLAG	ROLLUP _LEVEL _FLG	OFSA_ ROLLUP_ SYS_ID	OFSA_ ROLLUP_ LEVEL_ NUM
IBANK	UCOST	CHANNEL_ ID	2	0	1	200023	2
IBANK	UCOST	PRODUCT _ID	3	0	0	0	0
IBANK	UCOST	RE	1	0	0	0	0
IBANK	UCOST	TRAN_ID	4	1	0	0	0

OFSA_ABM_VAL_TO_NODE_MAPPING_V

ABM_ MASTER_ LIST_SET_ ID	ABM_ DIMENSION_ ID	ABM_ DIMENSION_ VALUE	OFSA_ LEAF_NUM_ ID	OFSA_ LEAF_ DESCRIPTION	OFSA_LEAF_ NODE
IBANK	CHANNEL_ID	ALL	9	Actual Gl Account Id	20000000
IBANK	CHANNEL_ID	ATM	9	Actual Gl Account Id	10000
IBANK	RE	100000	1	Org Unit ID	4
IBANK	TRAN_ID	CLOSEACCT	10	Transaction Type Leaf	13
IBANK	TRAN_ID	DEPOSIT	10	Transaction Type Leaf	14
IBANK	TRAN_ID	OPENACCT	10	Transaction Type Leaf	16

OFSA_ABM_VAL_TO_LEVEL_MAPPING_V

ABM_ MASTER_ LIST_ SET_ID	ABM_ MODEL_ ID	ABM_ DIMEN-SION_ID	ABM_ DIMEN-SION_ VALUE	OFSA_ LEAF_ NUM_ID	OFSA_ LEAF_ DESCRIP-TION	OFSA_ ROLLUP_ SYS_ID	OFSA_ ROLLUP_ LEVEL_ NUM	OFSA_ ROLLUP_ LEVEL_ DESC	OFSA_ ROLLUP_ LEVEL_ NODE
IBANK	UCOST	PRODUCT_ ID	COMMER CIAL	3	Common COA ID	43903	3	Interest / N on-Interest	3912238
IBANK	UCOST	PRODUCT_ ID	CREDITCA RD	3	Common COA ID	43903	3	Interest / N on-Interest	3912233
IBANK	UCOST	PRODUCT_ ID	NONE	3	Common COA ID	43903	3	Interest / N on-Interest	3912238

OFSA_ABM_TEMP_MAPPING (User picks and Tab1=9, Tab2=1,Tab3=3,Target Tab = 10)

OFSA_ TABLE_ SYS_ID	ABM_ RESOURCE _ID	TAB1_ LEAF	TAB2_ LEAF	TAB3_ LEAF	TARGET_ LEAF	AMOUNT
9999943945	100000- MALD	10000	4	3912238	14	1.15019648
9999943945	100000- ACMC	20000000	4	3912238	13	165.0949882
9999943945	100000- ACCN	20000000	4	3912233	16	23.2573954

D

Party Profitability for Insurance Companies

The insurance industry is undergoing a dramatic upheaval that will ultimately change how financial services companies do business in the Twenty-First Century. This revolution is being brought about by two defining events.

The first major event has been the growth of the Internet as a way for financial services companies to communicate with their customers. The insurance buying public now has access to more information than ever before, including information about rates, products, financial ratings, and customer satisfaction. Insurance companies could once depend on a steady base of customers because shopping for insurance was just too difficult. As shopping for insurance becomes easier, companies need to identify the most profitable customer segments for preferred pricing or risk losing them.

The second major event is the passage of the Financial Services Reform Act of 1999 (also known as Gramm-Leach-Bliley, or GLB). GLB has removed the barriers that have prevented national banks from marketing and underwriting insurance products. Before long, we will see banks and insurance companies form strategic alliances in attempts to provide complete financial services solutions for their customers.

Oracle Financial Services Applications or OFSA is already well established among financial institutions as a provider of customer-level performance analysis tools. OFSA is now introducing an insurance software solution, Oracle Performance Analyzer for Insurance, that will do for insurance companies what Oracle Performance Analyzer has done for banks.

OFSA offers an insurance-specific data model designed to integrate with the existing Performance Analyzer application and serve as a data warehouse for management reporting purposes. The intended audience for this appendix is internal Oracle sales/consulting staff with minimal insurance knowledge and little or no familiarity with Performance Analyzer. Therefore, this appendix provides a cursory review of Performance Analyzer features that are relevant for insurance. Additional information on the functionality of Performance Analyzer is available online at http://products.us.oracle.com/ under the Vertical Solutions, Financial Services, Performance Analyzer tab).

This appendix is divided into the following sections:

- Types of Insurance Companies and Insurance Products
- Basic Insurance Terms
- Defining Insurance Company Profitability
- Data Organization
- The Insurance Data Model
- The Oracle Performance Analyzer Solution
- Management Reporting: Insurance Company Needs
- Insurance Implementation

Types of Insurance Companies and Insurance Products

The purpose of this section is to introduce and classify the different types of insurance companies and kinds of insurance products on the market.

Categorizing Insurance Companies

Insurance companies are grouped into two broad categories:

- Property and Casualty (also known as *P&C*) companies
- Life companies (includes Health insurance)

Although it is not uncommon for a financial institution to own both a P&C insurance company and a Life insurance company, the operations of each are usually kept separate. It is therefore not surprising that the charters, regulators, rating bureaus, and other service organizations in the insurance industry are also naturally divided along these same lines.

Property and Casualty Insurance

P&C insurance protects the insured from unexpected losses due to natural catastrophes, lawsuits, workplace accidents, or any number of other risks. P&C products can be further divided into personal lines and commercial lines. Personal lines or *consumer* insurance provides coverage for individual exposures, such as automobiles, homeowners, personal property, and recreational vehicles. Commercial lines insurance encompasses a large host of products including, but not limited to, workers compensation, medical malpractice, products liability, and commercial general liability (for example, small business owners, contractors, strip malls, and restaurants). Commercial business is typically divided into small and large accounts. Because exposures of large accounts tend to be heterogeneous, the risk selection methods and pricing process is considerably more sophisticated and requires individualized attention. Note, in countries other than the United States, the term *General Insurance* or *Non-Life Insurance* may be used instead of Property and Casualty.

Life Insurance

Life insurance company products can encompass much more than simply life insurance benefits. A consumer buys a life company product because it provides insurance or is an investment vehicle or for both reasons. Life *investment* products, such as annuities, are designed to provide funds for future retirement or education. The main Life *insurance* products include life, health / medical, disability, dental, and long-term care. The most popular forms of life insurance are term, whole life, and universal life. There are many variations of these policy forms, and these products are often sold under many different names.

Primary versus Reinsurance Companies

An insurance company that sells policies directly to consumers is referred to as a *primary* insurer. A primary insurer (Company A) can decide to retain only a portion of the insured risk and cede the unwanted balance to another insurer (Company B). Just as an individual may purchase an insurance policy from an insurer, an insurance company may purchase fairly comprehensive reinsurance from a single reinsurer or from a collection of reinsurers. Company B assumes the ceded risk of a primary insurer and is called a *reinsurance company.* (That is, reinsurers provide insurance for insurance companies.) This transaction serves to reduce an insurer's exposure to large losses, either individually or in the aggregate, in exchange for a reinsurance premium. The transfer of risk can be accomplished in a variety of ways. For example, the reinsurer assumes losses on a pro rata basis (taking 30 percent of every loss), or the reinsurer pays excess losses. If losses for a particular event exceed $50 million, for example, it pays the next $100 million.

Because reinsurance exposures arise from customers of primary insurers, reinsurers do not typically possess customer level data. Any customer data a reinsurer has is most likely a primary carrier's response to a data call (with the timing of this request usually coinciding with the contract negotiation process).

Applicability to OFSA Products

Oracle Performance Analyzer for Insurance has been designed especially for consumer P&C insurance (that is, personal lines such as automobile, homeowners, personal articles floaters, and umbrella), and life insurance (that is, term, whole, and universal life). The data model is designed to capture primary insurance company data.

OFSA expects that the initial demand for Performance Analyzer for Insurance will come from existing users of the Performance Analyzer application, primarily banks that want to expand their reach into insurance. Consumer insurance products represent the most easily accessible market for these clients. Banks are already writing auto and home loans for their customers, so it becomes a natural extension of their business relationship to offer insurance as well. These clients are also familiar with Performance Analyzer's unique approach to expense allocation and will naturally want to take a similar approach with their insurance operation.

Additionally, insurance companies that already use Oracle products are prime candidates for adopting the Performance Analyzer solution. Considering its long list of majors clients, Oracle has already established a presence in many large insurance companies in the United States.

Basic Insurance Terms

The insurance industry has developed its own particular way of collecting and reporting financial data, and it is important that the reader become acquainted with some basic terminology. The *premium*, *loss*, and *expense* terms defined here are integral pieces in the profitability picture. In fact, many canned management reports as well as ad hoc reports used by marketing, actuarial, and accounting personnel contain some variation of these statistics.

Premium Definitions

The amount an individual or entity pays for an insurance policy is called *premium*. For management and statutory reporting purposes, insurers must express this premium in several different ways. Written, unearned, earned, and in-force premiums are the most common. The following example highlights the differences among these definitions:

Example:

Policy # 1, a 6-month policy, is written on January 1, 2000 for $60

Policy # 2, a 6-month policy, is written on October 1, 2000 for $120

Written Premiums

Written premiums are the total premiums generated from all policies written over a period of time (a *cash flow number*). In the month of January 2000, the written premium is $60. For February through September, the written premium in each month is $0 because no activity occurred in these months. The premium written in October 2000 is $120. The full premium is booked in the month it was written regardless of the policy term (that is, Policy #1 has a written premium of $60 in January 2000 but is $0 for the remaining five months). The written premium in 2000 is simply the sum of the written premium for all the months in 2000: $60 + $120 = $180.

Unearned Premium

Unearned Premium is that portion of the written premium that has not been exposed to loss and is due to the policyholder if the policy is cancelled (because services have not been rendered yet). Unearned premium is a balance sheet item and calculated at a specific point in time. Because unearned premiums are considered liabilities, reserves (UEPR) are established. The unearned premium in the example above as of November 30, 2000, is $0 for Policy #1 (because it becomes fully earned six months after it is written) and $80 for Policy #2 (because four months of coverage remain to be provided, 4/6 of $120 is $80).

Earned Premium

Earned Premium is that portion of written premium that has been exposed to loss and is therefore earned by the insurer. Premium becomes *earned* when coverage has been provided. Earned premium is an income statement item and relates to a specific period of time (a *flow number*). To continue with the example above, the premium earned in calendar year 2000, as of November 30, 2000, is $60 + $40 = $100. (Only two months of premium for Policy #2 were earned as of November 2000.) In practice, earned premium is calculated based on the written premium and unearned premium reserves:

Earned Premium = Written Premium + Beginning UEPR - Ending UEPR.

In-Force Premium

In-Force Premium is the full premium for active policies at a specific point in time. The in-force premium on November 30, 2000, is $120 because Policy #1 is no longer in force.

Exposure Definitions

Insurance companies use the term *exposure* to refer to a measure of the number of risks insured. In most consumer insurance, it is common to simply track the number of policies over a specified time period. (For example, a typical auto exposure is a *car-year*, or one auto policy written for one year.) Exposures, like premiums, can be written, earned, or in-force.

Written Exposures

Written Exposures assign a value of 1 for a 12-month policy (or 0.5 for a six-month policy) in the month the policy is written as new business or the first renewal month. For the remaining months in the policy term, the written exposure is *0*.

Earned Exposures

Earned Exposures are the written exposure prorated over the policy term. For a 12-month policy, the earned exposure is 1/12 = .0833 each month the policy is active.

In-Force Exposures

In-Force Exposures are the number of active policies at a particular point in time. Each policy is given a count of *1* if it is currently in force.

Exposure counts are monitored over time to gauge how the book of business is changing (growing or contracting) in aggregate or for selected market segments. As shown below, exposures are the inputs used in computing several performance measurement statistics.

Loss Definitions

The largest expense item on an insurance company's financial statements is for insured losses. These losses can be divided into several components: paid losses, case reserves, and bulk reserves for supplemental case reserves, and incurred but not reported (IBNR) losses. Note that paid amounts and case reserves are losses for claims that have been reported to the insurance company. Furthermore, these two amounts can be associated with a specific policy.

When reviewing the definitions below, take note of which classes of loss are directly associated with a specific policy (customer) and which losses must be allocated by Performance Analyzer.

Paid Loss

Paid Loss is the partial or full amount paid to a claimant or beneficiary via a check or draft (usually excluding the expenses associated with the claim adjustment process).

Case or Policy Reserve

Case or Policy Reserve is an estimated amount set aside for the settlement of a claim that has been reported to the insurance company. The case reserve is typically selected by a claims adjuster and represents his best guess of what the claim will settle for based on existing information. For life insurance policies, the claim amount is fixed because it is simply the death benefit specified in the policy. Property and Casualty claim estimates, particularly for the casualty lines of business, are more uncertain because they consider not only the actual loss sustained but also non-economic damages such as pain and suffering, as well as environmental factors such as inflation. As new information surfaces during the claims investigation process, case reserves change up or down accordingly.

Bulk Reserves

Bulk Reserves are lump sum amounts set aside to cover the remainder of loss liabilities not accounted for by the paid losses and case reserves. The actuarial department typically has the responsibility of setting the bulk reserves, which are forecasted based on historical data and/or judgment. By definition, the bulk reserve is not specific to a particular claim and has four components:

Bulk Reserve	Components
Case Supplement	Bulk reserve supplements to the adjuster case reserves when they are insufficient
IBNR	Losses incurred but not yet reported to the insurance company
Pipeline	Losses that are reported but not yet recorded in the claims system
Reopens	Provision for claims that have closed but may reopen in the future

Bulk reserves are often a major component of an insurer’s financial statement. Using rules defined by the company, Performance Analyzer allocates these reserves to the customer level or customer segment level of detail. The resulting output provides a much clearer profitability picture than would be obtainable without Performance Analyzer.

Incurred Loss

Incurred Loss represents all losses experienced by the company during a particular time period, whether paid or reserved. The definition varies depending on the data organization.

- Accident Year Incurred Loss = Paid + All Loss Reserves
- Calendar Year Incurred Loss = Paid + Change in Reserves (that is, ending reserve - beginning reserve)

Other Expense Definitions

Insurance company expenses other than losses can be categorized as either claims related expenses or underwriting expenses.

Allocated Loss Adjustment Expenses (ALAE)

Allocated Loss Adjustment Expenses (ALAE) are claims handling expenses that can be attributable to a specific claim. Examples include attorneys fees, court costs, deposition fees, medical examination fees to determine the extent of injuries, and appraisal fees.

Unallocated Loss Adjustment Expenses (ULAE)

Unallocated Loss Adjustment Expenses (ULAE) are claim expenses that *cannot* be attributable to specific claims. These expenses tend to be fixed costs associated with the claims operations such as claim adjuster salaries, company car expenses, computer and other office supplies, and rents for branch claims offices.

The sum of all claim expenses, both allocated and unallocated, is called *loss adjustment expenses*, that is, LAE = ALAE + ULAE. As with losses, ALAE and ULAE may each have paid and reserve components. By definition, however, ULAE does not have a case reserve piece.

Non-Claim Related Expenses (Underwriting Expenses)

On the property and casualty side, these expenses are commonly grouped as commissions, other acquisition costs, general expenses, or taxes, license, and fees. Life expenses can include additional categories for medical inspections or maintenance costs. The sum of all non-claim related expenses can be divided by written premiums to compute an expense ratio. The expense ratio is a key piece of the profitability picture.

Performance Statistics

Loss Ratio

Loss Ratio is defined as losses divided by premium. The numerator can be paid or incurred losses, while the denominator is usually earned premium.

Combined Ratio

Combined Ratio is defined as the loss ratio plus expense ratio (that is, losses plus expenses divided by premiums). This ratio is a common measure of profitability. Obviously, it is desirable for the losses and expenses to be less than the premiums. A combined ratio of greater than 100 percent indicates that the underwriting operations are draining the company’s surplus. (This is before considering investment income.)

Frequency

Frequency is a measure of the number of claims per exposure unit. The numerator can be the number of paid claims (resulting in a paid claims frequency) or the number of reported claims (incurred claims frequency). The denominator can be written, earned, or in-force exposures.

Severity

Severity is the average cost per claim and is defined as losses divided by claim count. The severity can be computed using either paid or incurred loss data and can include or exclude loss adjustment expense.

Pure Premium

Pure Premium is the average cost per policy exposure and is defined as losses divided by the number of exposures. The pure premium is the average dollar amount needed from each policy exposure to pay for losses. This statistic can be compared to the average premium collected.

Defining Insurance Company Profitability

Insurers have two main sources of income: their underwriting operations and their investments. Investment income is earned primarily from policyholder surplus and insurance liabilities such as loss reserves, expense reserves, and unearned premium reserves. The loss and expense reserve accounts are established to pay future obligations, while the unearned premium reserves represent premiums that are not yet earned and may need to be refunded before the end of a policy's term. Depending on the line of insurance, investment income can be enormous, often large enough to offset underwriting losses resulting from the insurance operations. The investment income potential is greatest when losses are not expected to be paid until some time in the far future. These are called *long-tailed* losses. Most life insurance products are considered to be long-tailed because insureds buy policies when they are young but the death payment is not expected for many years to come. Not surprisingly, asset and liability management play a major role in the Life company being able to meet its future liabilities. Although property and casualty products have a shorter effective period (typical policy terms are six months or one year) than life insurance policies, investment income still plays an important role. This is due to claims taking an extended time to settle or claim payments being spread over many years. (This is particularly true for workers compensation insurance.)

The profitability of an insurance company's underwriting operation is measured in the same way as other industries:

Profit = Revenue - Expense

The revenue is the premium charged the insurance customer. For consumer insurance, premiums are generally fixed at the beginning of the policy term. Expenses include losses, claim adjustment expenses, and day-to-day operating costs. What distinguishes insurance from most other industries is that the amount and the timing of the expenses (loss amounts) are unknown at the time the policy is written and in fact, may still be unknown long after the policy expires. How, then, does an insurance company measure profitability if its largest expense, the losses, is not known with certainty? In practice, P&C companies estimate bulk reserves and add them to case reserves to arrive at an estimate of losses. Life insurance companies use mortality tables based on their policy exposures to predict the expected losses and reserves. The bottom line is that actual losses will not be known for a very long time. Therefore, insurance companies attempt to estimate them, and their estimates change over time as more information is revealed. As a result of this complexity, insurance companies look at losses on a few different bases: policy year, accident year, and, calendar year, as described in the following sections.

Users of Profitability Statistics

Several different departments in an insurance organization tend to use profitability statistics. The accounting department tends to view financial results at the highest level by company, state, product, and coverage type. Its external clients, such as the regulatory agencies for which it provides reports, are also interested in summaries at this level. Product Managers generally want results at a much finer segmentation because their objectives include uncovering groups that may deserve special prices or treatment. The actuarial department may analyze combined ratios for a particular market segment to determine which business tends to be the most profitable. The agencies department would want summaries by agent or district manager so it can determine the contribution of their sales force in terms of profitability and growth.

Data Organization

Because of the long-tailed nature of insurance losses, measuring profitability, particularly at the customer level, is very difficult. Companies need to match the premiums collected with the losses incurred. When those losses do not manifest themselves for months or even years after the policy is written, however, the company must make a compromise between accuracy and timeliness. To deal with these challenges, insurers have developed several standard ways of compiling their data to measure profitability.

Policy Year

Policy Year data matches the premiums collected for all policies effective in a particular year with the losses associated with those policies. The policy year approach is the theoretically correct way to measure profitability at the customer level. When losses are not immediately known, however, the company must make estimates of those losses. This is the approach taken in life insurance, with industry mortality tables used to project expected benefit payments. Mortality statistics are typically compiled and maintained by insurance service organizations. Therefore, this information may not necessarily be stored in an insurer's legacy system.

Accident Year

Accident Year data matches the premiums earned in a year with the losses that have occurred in that year. As with policy year data, estimating the losses that are expected to occur in the more recent years is often necessary. Most P&C companies use accident year loss experience to evaluate the performance of their customers.

Calendar Year

Calendar Year data compares the premiums earned in a particular calendar year with the losses incurred in that year. This method is typically how results are reported to rating agencies and stock analysts because it represents the true cash flow experience for the year. This method is inadequate, however, for measuring customer profitability because the losses incurred in the calendar year do not necessarily arise from the policies written in that year. A company is most interested in the experience of its current customers. For that purpose, the accident year or policy year method is more appropriate.

Following is an example that helps to clarify these differences:

Effective Period of Policy	Policy Year	Accident Date	Loss Paid	Date Paid
Jan 1, 1981 to Dec 31, 1981	1981	Dec. 1, 1981	$500	Feb 1, 1982
Oct 1, 1980 to Sep 30, 1981	1980	June 1, 1981	$300	Aug 1, 1981
Dec 31, 1981 to Dec 30, 1982	1981	March 1, 1982	$700	June 1, 1982

Accident Year 1981 Losses = $500 + $300 = $800

Accident Year 1982 Losses = $700

Policy Year 1981 Losses = $500 + $700 = $1,200

Policy Year 1982 Losses = $0 (no policy origination dates in 1982)

Calendar Year 1981 Losses = $300

Calendar Year 1982 Losses = $500 + $700 = $1,200

In this example, accident year 1981 includes all losses with an accident date in 1981 associated with policies written in 1980 and 1981. Policy year 1981 losses include some accidents that occurred in 1981 and some that occurred in 1982. The Calendar Year approach is insensitive to when accidents occurred or when the policy was written. All that matters is, “How much cash did we pay out in a particular year?”

Note that the losses for 1982 range from $0 to $1,200, depending on which basis we choose. Which is the correct loss amount? Was 1982 a profitable year? Answer: There is no one correct loss amount because all three amounts are correct within their respective data organization definitions. The profitability picture in 1982 depends on the point of view of the customer. The product manager sees that the policies written in 1982 generated $0 in losses, and may seek to reduce rates in response. The actuary, trying to project the losses expected next year uses the accident year losses as a starting point. The CEO of the company, however, has to explain to the stockholders why the company lost $1,200 in the calendar year. Fortunately, Performance Analyzer can quickly and easily produce reports using whichever data organization is appropriate. The key is a robust data model based on the ACORD standard (as defined in the following section) and the power of Oracle Discoverer.

The Insurance Data Model

The insurance data model consists of a few dozen database tables that are embedded in the existing Financial Data Manager (FDM) data structure designed for banking and securities products. Because every company manages data differently, there can be no such thing as a one-size-fits-all data model that meets each company's needs right out of the box. For this reason, the insurance tables are meant to be customized to the specific requirements of the individual company. The FDM tables are grouped into six product *stacks*: automobile, home, personal article floaters, umbrella, whole / universal life, and term life insurance.

Refer to Release 4.5 of the *Oracle Financial Services Technical Reference Manual* for P&C and life entity relation diagrams for the exact details of the data model. The following provides a high level review of the architecture and content contained in the database stacks.

Dimensions

The insurance model adopts the four mandatory dimensions in OFSA and creates an additional dimension called *Coverage*. These existing dimensions are applicable to insurance in the following manner:

Organization Unit ID

This column houses the assignments for each state / company combination and for the home office. For example, if Vision Insurance Group has two insurance company holdings (Vision Insurance Company and Focus Insurance Company) each writing in California, Nevada, and Texas, there will be six organization units excluding one for the home office. This is a natural way to assign levels because insurance companies are regulated on a state level (in the United States). By the click of the mouse, use of hierarchies provides financial results to be summarized for the state as whole or by company.

Common Chart of Accounts ID

This is the product type. If a company offers four different policy forms (ranging from ultra-preferred to substandard business) for auto insurance, each one is listed as a separate product. Again, hierarchies make it possible to view profitability results for user-defined product groupings.

Financial Element ID

You can use this column in the LEDGER_STAT table (the OFSA version of the summarized general ledger) to categorize the general ledger account entry. The OFSA default values that are relevant for insurance are:

- Beginning balance
- Ending balance
- Non-interest expense
- Non-interest income

Because reserve amounts are a point-in-time number, they would be categorized as either a beginning balance or ending balance. Premiums over a period of time (earned or written) should be categorized as non-interest income. Insurance operating expenses that have been paid would be assigned a financial element of non-interest expense.

General Ledger Account ID

This column identifies the asset, liability, or income statement items, as well as other insurance statistics used in a Performance Analyzer allocation rule. Examples of GL accounts are earned premium, written premium, case reserves, IBNR reserves, earned exposures, paid loss count, reported loss counts, advertising expenses, policy processing expenses, and agents' balances.

Coverage ID

An insurance policy (or account) consists of one or more coverages. Coverages can be modified by options or endorsements that are also attached to the policy. Coverage ID is a dimension that relates the multiple coverage, option, or endorsement records back to a specific policy record. In many cases, these ancillary records have separate premium and loss statistics that make it easy to perform profitability analysis at a granular level.

Property and Casualty Data Model

Each P&C stack has a policies table, a coverages table, and a claims table that are the main tables to be used for allocation and reporting purposes. As mentioned above, each insurance policy has one or more coverages. A coverage can have one or more claims associated with it. For P&C products, a policy may have multiple claims during the policy term. To accommodate the one-to-many record relationships, the coverages and claims information needs to be kept in separate tables.

The policies table is an instrument profitability table where entries are uniquely defined by the policy number and the monthly as-of date. The policy table contains summary premium and loss data for all the coverages combined. This table is Performance Analyzer enabled because some expense allocations are expected to be performed at this level.

The coverages table is an instrument transaction table where entries are uniquely defined by the policy number, coverage ID, and the monthly as-of date. The premium information and loss data, if any, is recorded for each coverage. Auto products tend to be sold *cafeteria style*, meaning that consumers pick and choose the coverages they desire. Each coverage is priced separately (stored in the coverages table) and then summed to compute the full policy premium (stored in the policies table). Home policies can be sold cafeteria style or on a package basis. If sold as a package, one premium is charged for a fixed set of coverages. Because the full premium is not easily divisible into coverages, all the premium information is contained in the policies table. The coverages table is also Performance Analyzer enabled because most allocations are performed at this level.

The claims table contains loss data such as paid losses or case reserves that are associated with a policy. Note that bulk reserves are not loaded in this table. By default, the claims table is not Performance Analyzer enabled. Therefore, allocations cannot be performed at this level. To perform allocations based on claims data, the user must summarize claim amounts and counts into the coverages table and perform the allocations there. As is, this configuration renders the claims table to be used for reporting purposes only.

The other tables in the model contain rich customer intelligence information: The auto stack has vehicle detail and vehicle operator tables that house all data on the driver and insured vehicle. For each driver, there are slots to record number of years of driving experience, age, credit score, and so on. The home stack has a table containing details such as the number of stories, type of chimney, and square footage of the insured dwelling. Other customer attributes, such as recent demographic data, can be easily incorporated into the model. Having this detailed information enables the user to allocate expenses to policies containing selected characteristics.

Life Data Model

For the most part, the life model has a similar structure to the P&C model except for two differences: the claims information is folded into the policies table and an additional table for coverage options is introduced. Unless a policy insures several lives, having multiple claims for one life product is impossible. Therefore, a separate claims table is unnecessary. Because an individual coverage can have multiple options associated with it, an additional table was created. The detail tables in this model include basic information on the insured, his or her lifestyle activities, medical conditions/history, as well as auxiliary data on loan and investment values. Several predefined database views in FDM join these details with the policies/coverages table and can be used to construct allocation rules. Most, if not all, of the expense allocations are expected to be performed at either the policies or coverage level.

How ACORD Fits In

Established in 1970, ACORD is a non-profit organization in the United States that promotes standards for the exchange of insurance information. Specifically, ACORD Standards provide a common *language* for communicating business information. Although the ACORD standards are developed by participating insurers in the US, they are recognized and used in several foreign countries. ACORD has a data model that facilitates the transmission of information between agents and the insurance company. The standards relieve the agent of maintaining multiple systems needed to relay information to each insurance company. Currently, the ACORD data model captures only exposure information and does not record claims data. At most, the agent receives the first notice of a claim. Beyond that, the claims operations are handled by the insurance company and the agent does not participate in the settlement process. (This explains why loss data is noticeably absent from the ACORD model.) The OFSA insurance model incorporates information contained in the ACORD model that is relevant for measuring profitability.

Because ACORD is a recognized leader in developing and maintaining standards for the insurance industry, Oracle's affiliation with ACORD will presumably lead to future business relationships with the 1,000 plus insurance companies that are ACORD members. ACORD supports standards for most major lines of business and continues to expand. (For example, a present initiative includes developing standards for reinsurance companies as well as participation in numerous groups whose objectives are to globalize insurance data.) By adopting the ACORD model, numerous fields can be directly mapped into the OFSA database. Additionally, the user inherits 200+ code tables complete with seeded data that are automatically mapped into the database.

The Oracle Performance Analyzer Solution

The insurance market is getting more competitive, with new entrants, including banks and Direct-via-Internet companies, threatening the establishment. Insurers are looking for that competitive advantage that will set them apart from a crowded field. Insurance companies are segmenting their customers and developing more refined pricing plans in an attempt to capture the most profitable customers. And whether the client is an established insurance company, an insurance start-up, or a bank with insurance sales aspirations, Performance Analyzer for Insurance has something unique and powerful to offer.

The strength of Performance Analyzer is its ability to allocate revenue and expenses to customer segments, providing for a more accurate and refined profitability assessment than has previously been possible. At present, very few, if any, insurance companies allocate expenses to the customer level of detail. Most expenses, as much as 90 percent in some companies, are already unique to the customer. These are the losses experienced by the insured policyholders.

Can allocating the remaining 10 percent of expenses to customers really make that much of a difference? Well, this is the same reaction that Performance Analyzer received when it was first introduced for banks. OFSA clients soon discovered that that 10 percent actually makes all the difference in the world. Price competition and new capital flowing into the insurance industry have reduced profit margins in recent years. Consider an insurance company that earns five cents of profit on each dollar of premium, with ten cents of overhead expense. Using Performance Analyzer, this company may learn that 80 percent of this overhead is required to service only 20 percent of its customers. The company would discover that this 20 percent of the business is actually *losing* 30 cents per dollar of premium. Armed with this knowledge, the company can exercise a strategy of repricing, selective marketing, or service differentiation to capture the more profitable customers.

Current users of Performance Analyzer in the banking industry will find that analyzing *insurance* customer profitability requires a whole new set of tools. Insurance losses are random and infrequent. The historical experience of one customer for one year is not a good indicator of that customer's profitability. A company must use several years of customer data, and it must pool customers with similar risk characteristics into unique segments. Insurance companies know that teenage drivers are bad risks, but the parents may not be. Customers that buy several products from the company are more profitable than those customers who buy only a single product. Performance Analyzer for Insurance is built with all of this in mind.

For more information, review the Insurance Demo Script for Performance Analyzer (available at http://sm-ofs-app-dev.us.oracle.com/fsdev/FSDEV.home). This document is replete with typical expense allocations that an insurance company would perform and emphasizes allocating expenses by distribution channel and policy age. Currently, insurance companies are not allocating expenses at a customer level, which implies that the expense ratio is the same for everyone. Is it really appropriate to spread expenses evenly by customer? Because new business policies are typically associated with higher acquisition costs, it would seem that expenses should be higher for these customers. But are insurance companies allocating a disproportionate amount of the policy processing costs to these new customers? Consumers who purchase insurance directly (without an agent) pay lower premiums because there is no commission provision. Do the combined ratio results for these insureds warrant the level of savings they receive? The Performance Analyzer solution is designed to provide answers to these questions.

Performance Analyzer offers many other advantages as well. The insurance model captures information at the lowest common denominator, which enables the user to easily roll-up or sub-group customers. Additionally, the data used in Performance Analyzer comes straight from the company's general ledger. No longer will companies have to struggle to reconcile different data sources. And because the entire system uses the Discoverer front-end, anybody in the organization can produce meaningful, presentation-quality reports without requiring a steep learning curve.

Finally, the Performance Analyzer/Discoverer suite is soon to be Internet-ready, enabling users throughout the organization to work from the same data without the overhead of an antiquated mainframe legacy system.

Management Reporting: Insurance Company Needs

Almost all management reports for insurance have some combination of premium, loss, exposure, and claim-count components. (Refer to the demo script for examples of basic reports and instructions on how to interpret them.) A typical profitability report has loss ratios or combined ratios by customer segment and product type. Other popular reports show average premium levels, frequency, severity, and pure premium over an extended period of time.

Banking institutions carry recent information on their customers (partly because they are not concerned with what loan or checking account balances were in the past). In contrast, insurance companies keep at least the most recent five years of data on their books. A long history is necessary to track trends, act as the basis for loss projections, and provide for accurate pricing.

As mentioned earlier, insurers do not look at profitability at an individual customer level. Instead, they group customers with similar risk characteristics together. The largest component of premium is the amount needed to pay for expected losses. If an insured has no claim during the policy term, he or she is profitable. On the other hand, once a claim occurs, he or she is extremely unprofitable. Due to the nature of insurance business (pooled losses and risks spread over a large group of people), viewing profitability at a customer segment level is more appropriate than viewing it individually.

Insurance companies may maintain dual accounting systems. Insurers are required to report financial results annually based on Statutory Accounting Principals (SAP). Publicly held insurance companies must also report financial results to the Securities Exchange Commission according to Generally Accepted Accounting Principles (GAAP).

SAP is concerned with liquidity or insurer solvency, so their guidelines tend to be conservative. GAAP is concerned with the ongoing viability of a company and therefore has less strict accounting standards. For example, SAP requires certain acquisition costs to be expensed at the time the policy is written. GAAP allows these same costs to be deferred and expensed during the course of the policy term. GAAP allows loss reserves to be discounted for the time value of money, while SAP requires the liabilities to be held at their full undiscounted value. Clearly, a product can be profitable under GAAP but not SAP. Does Performance Analyzer perform allocations on a GAAP or SAP basis? Actually Performance Analyzer and the data model are insensitive to whether the expense is on a SAP or GAAP basis. The distinction is made in the LEDGER_STAT table. Once the expense is loaded into the table, it can be allocated in a method that supports each standard.

Insurers today still tend to have hundreds of three-ring binders on the premises, binders containing monthly and quarterly production reports. No doubt the costs of storage and personnel required to maintain these systems are substantial. To the extent that OFSA can offer a management reporting system that is reliable and robust and that puts information at the user's fingertips, insurance companies will move toward being paperless and promoting operating efficiencies.

Lastly, the flexible data model of OFSA enables the client to add new data fields as the business evolves or to create new reports as they are needed. In the past, teams of mainframe programmers may have been required to upgrade systems and design reports. With Performance Analyzer and Discoverer, the power has shifted into the hands of the users.

Insurance Implementation

Although the most recent version of Oracle Performance Analyzer, Release 4.5.2, can perform a variety of insurance-type allocations, the processing engine has not been enhanced specifically for Performance Analyzer for Insurance. As a result, there are some limitations with the existing application, as designed for banks, that might be an issue for an insurance client. The purpose of this section is to document the known limitations and suggest workarounds if available. The discussion below assumes knowledge of Performance Analyzer and is particularly relevant for the consultant who may implement Performance Analyzer for Insurance.

Allocating Fixed Policy Expenses to Coverages

Business Case

Some expenses are fixed per policy (rather than varying by premium). Once allocated, the expense needs to be pushed down to the coverage level because profitability is typically viewed on a coverage, not policy level.

Two policies are written: Policy #001 for $100 and Policy #002 for $800. A percent distribution based on record count, from the LEDGER_STAT table to the policies instrument table, allocates equal amounts to each policy. Performance Analyzer works fine here.

Next, the fixed amounts that are already allocated need to be distributed to the coverages based on premium per coverage. Performance Analyzer is unable to perform this instrument-to-instrument transaction table allocation accurately because it requires doing a percent distribution by ID_NUMBER, which is not a leaf.

An example shows the desired results:

LEDGER_STAT table record is $100 (to be allocated to the General Expense column)

Policies Table

ID Code	ID Number	Premium	Record Count	General Expense
12312000	001	100	1	50
12312000	002	700	1	50

Coverages Table

ID Code	ID Number	Coverage ID	Premium	Record Count	General Expense
12312000	001	BI	75	1	37.50
12312000	001	PD	25	1	12.50
12312000	002	BI	400	1	25.00
12312000	002	PD	100	1	6.25
12312000	002	COLL	300	1	18.75

The desired result is that the general expenses for each policy sum to $50 in the coverages table. (37.5+12.5=50.0 for Policy #1 and 25+6.25+18.75=50 for Policy #2). The resulting output is a percent distribution based strictly on the premium in the coverages table (that is, no memory that $50 per policy has been allocated).

Group Expense Allocations

The Performance Analyzer engine encounters a similar problem when attempting to allocate Group Plan expenses down to the policy level for life insurance. The data model has two Group instrument tables (GROUP_TERM_LIFE_POLICIES and GROUP_WHOLE_UL_POLICIES) that contain summarized policy data by group plan. For example, Vision Insurance underwrites group life insurance for several large corporations (including Oracle). Each account has one record in the Group table. Oracle is allocated fixed administration expenses of $100,000 for its plan (supposing the plan insures 40,000 Oracle employees). The Performance Analyzer engine is unable to further allocate the $100,000 expense down to the 40,000 policies contained in the TERM_LIFE_POLICIES instrument table.

Workaround

None proposed.

Allocations Based on Attributes in the Policies Table

Business Case

A number of attributes may be stored only in the policies table and may be needed in allocations using the coverages table. For example, a column in the policies table called POLICY_STATUS_PC_CD indicates if the current policy is new business, renewal, cancelled, or expired. This attribute and several others are stored in the policies table because they are specific to the policy and not variable by coverage. Suppose one desires to allocate an expense from the LEDGER_STAT table to all new business records in the coverages table. A problem arises because new business policies cannot be identified in the coverages table.

An example shows the desired results:

LEDGER_STAT table record is $100 (to be allocated to the Advertising column)

Policies Table

ID Code	ID Number	Premium	Policy Status CD
12312000	001	100	NB
12312000	002	300	Ren
12312000	003	400	Ren
12312000	004	500	Ren
12312000	005	900	NB

Coverages Table

ID Code	ID Number	Coverage ID	Premium	Record Count	Advertising
12312000	001	BI	75	1	7.5
12312000	001	PD	25	1	2.5
12312000	002	BI	200	1	0
12312000	002	PD	100	1	0
12312000	003	BI	250	1	0
12312000	003	PD	150	1	0
12312000	004	BI	300	1	0
12312000	004	PD	200	1	0
12312000	005	BI	600	1	60
12312000	005	PD	300	1	30

Workaround

Duplicate the necessary columns, currently in the policies table, into the coverages table. Having to repeat the information (for multiple records) in the coverages table is certainly not very efficient, but it does enable the user to filter on the attribute.

Non Performance Analyzer-Enabled Insurance Tables

Business Case

Insurance coverages can have one or more discounts applied to the premium. Each discount associated with a coverage is recorded in the applicable discount table. (For auto policies, for example, this would be the PL_VEHICLE_DISCOUNTS table.) The coverages table does not contain the applicable discounts because this would necessitate a column for every possible discount. Because the discount table is not Performance Analyzer-enabled, it is used for management reporting purposes only. Making the table Performance Analyzer compliant requires the discount column to be defined as a leaf value. This violates the OFSA definition of how leaves were intended to be used, however.

The problem arises when a user wants to create an allocation rule using the discount type. Performance Analyzer cannot perform the allocation because the discount information is not accessible in Performance Analyzer.

For example, sample coverage and discount records are shown:

Coverages Table

ID Code	ID Number	Coverage ID	Premium
12312000	001	BI	75
12312000	001	PD	25
12312000	002	BI	200
12312000	002	PD	100
12312000	003	BI	250
12312000	003	PD	150
12312000	004	BI	300
12312000	004	PD	200

Discounts Table

ID Code	ID Number	Coverage ID	Discount CD
12312000	001	BI	Multi-Car
12312000	001	BI	Multi-Policy
12312000	002	BI	Good-Student
12312000	004	BI	Multi-Car
12312000	004	BI	Senior Discount
12312000	004	BI	Anti-Lock Brake

In the above example, Policy #001 has two discounts (that is, the premium of $75 in the coverages table is discounted for Multi-Car and Good-Student). Policy #002 and #004 have one and three discounts, respectively. Note that the discount table is populated only when the coverage record has an associated discount. Policy #003 does not qualify for any discounts and therefore has no entry in the discount table.

Suppose the user wants to allocate underwriting costs of $100 to coverages with the Good-Student discount. The $100 represents extra costs needed to verify that the insured still qualifies for the discount after the first renewal. Performance Analyzer cannot perform the allocation on the coverages table because the discount attribute resides in a non-enabled table.

Note that several other tables in the data model also have this same *one to many relationship*, where the *many* table is non-Performance Analyzer enabled. This structure precludes the user from referring to attributes in these tables to build allocations. In particular the tables affected include:

- All Discount tables
- All Surcharge tables
- All Deductible tables
- Accident History
- Citation History
- Umbrella Underlying Policies
- Criminal Conviction
- Driver Violations
- Family Illness
- Lifestyle Activities
- Medical Conditions
- Medical Prevention
- Medical Treatments
- Racing Competition
- Substance Usages

Workaround

Select columns that the user might need in an allocation and duplicate them in the appropriate Performance Analyzer-enabled instrument table. For example, if a company offers 30 discounts, the user might want to select the five most popular ones and create flag columns in the instrument table rather that duplicate all 30.

Performing Allocations Based on Claims Data

Business Case

Unlike life insurance, P&C coverages may have multiple claims during the policy term. Therefore, the P&C portion of the data model contains claims tables that are separate from the coverages table.

The following table shows four claims are currently pending as of 12/31/2000:

- Policy #001 has two separate accidents (one occurring on 9/15/1999 and the other occurring on 10/30/2000).
- Policy #002 and Policy #103 each have one open claim.

Claims Table

ID Code	ID Number	Claim Number	Coverage ID	Loss Date	Loss Case Reserve End	Loss Case Reserve End Bulk
12312000	001	111	BI	09/15/1999	5000	
12312000	001	222	BI	10/30/2000	10000	
12312000	002	333	BI	12/15/2000	12000	
12312000	503	444	BI	02/15/2000	60000	

The user wants to allocate case supplement reserves of $3000 in proportion to the current case reserve (that is, Loss Case Reserve End). Because the Claims table is not Performance Analyzer enabled, the allocation must be performed at the coverage level.

Workaround #1

As part of the data extract, the claims data can be summarized into the coverages table. This is in addition to populating the claims table. All numeric columns in the claims table are duplicated in the coverages table for this purpose.

Coverages Table

ID Code	ID Number	Expiration Date	Coverage ID	Premium	Loss Case Reserve End	Loss Case Reserve End Bulk
12312000	001	03/01/2001	BI	400	15000	
12312000	002	04/01/2001	BI	250	12000	
12312000	003	04/01/2001	BI	200	0	
12312000	004	04/01/2001	BI	300	0	

Note that the last claim record with a $60000 reserve did not summarize into the claims table. This happens when the coverage has expired and is no longer on the books. In this case, the claims (or loss) record does not have a matching exposure (premium) record as of December 2000. In most cases, a company keeps only in-force records in the coverages files. That is, when loading December 2000 records into the coverages table, the company does not include prior expired policies. Because the user wants all claims records to appear in the coverages table, he or she has two choices to handle these orphans:

1. Force the company to carry records in the policies and coverages tables for every policy that has an open claim, even if the policy has long expired. This ensures that every claims record has a matching exposure record.
2. Insert orphan records in the coverages table. These records do not contain the original coverage information, but they do have the claims data that the user needs for allocation purposes. The following sample SQL commands summarize selected columns from the claims table to the coverages table for the auto stack:

```
UPDATE pl_vehicle_coverages co
SET (loss_case_reserve_end, alae_case_reserve_end) =
    (SELECT SUM(loss_case_reserve_end), SUM(alae_case_reserve_end)
     FROM pl_vehicle_claims cl
     WHERE cl.identity_code = co.identity_code
       AND cl.id_number     = co.id_number
       AND cl.coverage_id   = co.coverage_id);

INSERT INTO pl_vehicle_coverages co
  (identity_code, id_number, coverage_id,
   org_unit_id, common_coa_id, gl_account_id,
   as_of_date, iso_currency_cd,
   vehicle_id_num, primary_operator_number, effective_date,
   loss_case_reserve_end, alae_case_reserve_end)
SELECT
   identity_code, id_number, coverage_id,
   MAX(org_unit_id), MAX(common_coa_id), 0,
   MAX(as_of_date), MAX(iso_currency_cd),
   0,0,max(effective_date),
   SUM(loss_case_reserve_end), SUM(alae_case_reserve_end)
FROM pl_vehicle_claims cl
WHERE NOT EXISTS
     (SELECT NULL
      FROM pl_vehicle_coverages co1
      WHERE co1.identity_code = cl.identity_code
        AND co1.id_number     = cl.id_number
        AND co1.coverage_id   = cl.coverage_id)
GROUP BY identity_code, id_number, coverage_id;
```

The resulting coverages table now contains the expired record with bogus information in the required columns.

Coverages Table

ID Code	ID Number	Expiration Date	Coverage ID	Premium	Loss Case Reserve End	Loss Case Reserve End Bulk
12312000	001	03/01/2001	BI	400	15000	
12312000	002	04/01/2001	BI	250	12000	
12312000	003	04/01/2001	BI	200	0	
12312000	004	04/01/2001	BI	300	0	
12312000	503	00/00/0000	BI	000	60000	

Once all claims records are in the coverages table, the user can proceed with building the allocation rule to include the Loss Case Reserve End column and write the results to the column called Loss Case Res End Bulk. By performing the allocation this way, the user cannot distribute his or her expense based on loss date. Because the loss date is contained in the claims table and is not carried to the coverages table, the user cannot allocate differing proportions for some loss dates versus others. For example, allocating 75 percent of the case supplement to older accident years and the remaining 25 percent to claims in last two years is impossible.

When the allocation is complete, and the Loss Case Res End Bulk column is populated, the user may want to write the result back to the claims table. In fact, the Performance Analyzer Standard Report called “pa_ins_pc_profit.dis” is dependent on this column in the claims table. The following is a sample SQL script:

```
UPDATE pl_vehicle_claims cl
SET loss_case_reserve_end_bulk =
   (SELECT co.loss_case_reserve_end_bulk *
          decode(co.loss_case_reserve_end,0,0,
          cl.loss_case_reserve_end / co.loss_case_reserve_end)
    FROM pl_vehicle_coverages co
    WHERE co.identity_code = cl.identity_code
      AND co.id_number     = cl.id_number
      AND co.coverage_id   = cl.coverage_id)
```

Workaround #2

Another way to allocate expenses based on claims data is to make the claims table Performance Analyzer enabled. This requires creating two additional leaves, CLAIM_NUMBER and CLAIM_OCCURRENCE_NUMBER. The claim number is unique for every loss date (that is, incident).One incident may have more than one claimant, however. Some insurance companies assign only one claim number per incident regardless of the number of claimants. In this case, the CLAIM_ OCCURRENCE_NUMBER identifies each individual claimant.

This workaround is certainly simpler than the first approach discussed. The disadvantage is requiring CLAIM_NUMBER and CLAIM_OCCURRENCE_ NUMBER to be leaves when they are not truly dimensions.

How to Handle Insurance Coverages that Are Packaged

Business Case

The purpose of this section is to alert the implementing consultant that insurance policies come in two common forms. Depending on the product, the consultant may need to customize the data model structure to accommodate the insurance client.

Policies are usually sold in one of two ways:

- The policyholder selects individual coverages and pays a separate premium for each piece. (This is similar to the *cafeteria style* insurance benefits Oracle offers its employees.)
- A policy is packaged with standard coverages and one premium is charged.

The auto stack of the data model is predicated on policyholders purchasing coverages cafeteria style. The coverages table contains the premiums charged for each coverage, and the resulting total premium is written to the policies table in the data extract.

Homeowner policies can be sold either cafeteria style or prepackaged. The data model has been designed to handle both types of policy structures. Package-specific information (such as the single premium charged) is intended to be stored in the policies table. The coverages table is populated only to the extent the package policyholder has purchased additional options that are not part of the standard package.

Regardless of the policy structure, the claims data is almost always recorded at the coverage level. For package policies, this implies the loss data is at the coverage level while premiums are not be divisible. Therefore, profitability would naturally be analyzed at the product and not coverage level.

One Policy Insures Multiple Objects

Business Case

Some insurance companies insure multiple objects under one policy number. For example:

Husband and wife own two autos. Each auto has unique coverage and a separate premium. Both autos are associated with the same policy number.

Husband and wife insure both their primary residence and their seasonal home with the same insurance carrier. Both are charged separate premiums but they are issued under one policy number.

Note that the insurance data model requires a unique policy number (ID_NUMBER) for each object that is separately insured. If two cars are insured under policy number 100005, the consultant needs to generate two separate records, such as 1000051 and 1000052, for each of the relevant tables. Each policy then has its unique characteristics stored as a separate row in the coverages and detail tables.

One Policy Insures Multiple Individuals

Business Case

In most cases, a vehicle policy (specifically, auto) has one assigned operator. However, one vehicle policy may be associated with multiple operator records. For example, husband and wife own one car. One policy is issued. Operator information is collected for both the husband and wife, resulting in two separate records in the operators table. Either the husband or wife must be categorized as the primary operator because the insurance data model contains views joining selected tables with the operator table based on the primary operator ID. This structure limits the user to primary operator characteristics when building an allocation rule.

The same phenomena also applies to several life insurance tables. One life insurance policy may insure multiple lives (husband, wife, dependent children). The insurance views require that one person be assigned as the primary participant and, if applicable, an optional secondary participant may be assigned as well. Information collected on either the primary or secondary participants can be referred to in an allocation. If the user needs to refer to characteristics of other participants in an allocation, he or she needs to add a new column (for example, *third participant*) in the policies and coverages table as well as update the views that reference the life participants table.

P&C Code Tables

Business Case

Seeded data is not provided for the following P&C code columns:

- FILING_JURISDICTION_CD
- LIC_PLATE_JURISDICTION_CD
- LIC_STATE_CD
- REGISTRATION_STATE_CD
- STATE_CD

These columns can take on the same set of code values, so creating five separate code tables (all five columns may map into one code table) may be unnecessary.

Home Details: Multiple Instances of Selected Columns

Business Case

You may need to repeat columns in the database tables to accommodate multiple instances:

- ATTACHED_STRUCTURES_CD
- EXTERIOR_BUILTINS_CD
- INTERIOR_BUILTINS_CD
- BATH_OTHER_CONSTR_QUALITY_CD
- BATH_OTHER_TYPE_CD
- DOG_TYPE_CD
- FIREPLACE_TYPE_CD
- ADDITION_TYPE_AMT
- BATH_OTHER_TYPE_CD
- DECK_TYPE_CD
- SPECIALTY_ROOM_CD
- WATERCRAFT_EQUIPMENT_TYPE_CD
- DETACHED_STRUCTURE_TYPE_CD

For example, an insured may have multiple interior builtins (such as dishwasher, elevator, hot tub, and wet bar), but the data model includes only one column to house this information. To capture all the builtins, the consultant needs to add extra columns to record this information. Because predicting the number of instances of each column is impossible, this exercise is left for the implementing consultant.

Life Coverages and Coverage Options Tables

Business Case

Some life insurance companies distinguish between coverages and options on coverages. The coverages table could indicate the type of record: base coverage or rider. The options hang off of the coverages. For instance, an accelerated benefit is typically sold as an option on a coverage and not as a coverage itself.

The data model contains two separate tables for coverage and option statistics. The leaf value, COVERAGE_ID, makes the life coverages tables Performance Analyzer enabled. By default, the options table is not Performance Analyzer enabled.

Workaround

If the user needs to perform an allocation with the data in the options table, he or she needs to make OPTION_ID (an existing column in the options table) a leaf column. The implementation is much simpler for the insurance company that does not differentiate between coverages and options. All the records can be combined into the coverages tables where COVERAGE_ID is used to identify the type of coverage or option.

Adding a Leaf to the Database

This is a reminder that when a new leaf is added to the database, the user must either

- Unregister all of the insurance views
- Re-register the views with the new leaf column added

Currently, the ADD_LEAF procedure (see Release 4.5, *Oracle Financial Services Installation and Configuration Guide*) is able to add only new leaf columns to instrument tables. It also registers such columns for the affected objects. But it currently does not support views for this process. Originally, the assumption was that most clients would have only the views that they created. Therefore, they would be responsible to upgrade the views for a new leaf. But because the insurance views are OFSA objects, clients do not really know about them.

ACORD Data Model

The OFSA model was designed specifically to run Performance Analyzer optimally and to do management reporting efficiently. Because the objectives of OFSA are different from those of ACORD (that is, the primary objective is to transmit information between agent and insurance company), the resulting data model contains much of the same information but is structurally different.

Mortality Tables and the Lookup/Table ID

Life insurance companies can use mortality tables to project benefits and compute policy reserves. Mortality tables commonly consist of factors provided by outside vendors such as insurance service organizations. The consultant should be aware of the Performance Analyzer's Lookup and Table ID. These two features are particularly well suited for managing the factors that can be retrieve for calculations.

Information Not Contained in the Legacy System

Not all columns in the data model are expected to be extracted from a company's legacy system. Columns that can be calculated following the data extract include:

- Earned Premiums and Unearned Premiums (which can be calculated from Written Premiums)
- Beginning Reserves (which are usually the ending reserve from the previous year end or month)
- Net Premiums (which can be computed from Directed and Ceded Premiums)

Glossary

As-of-Date

The date when the data is current.

Asset/Liability Management Committee

The Asset/Liability Management Committee (ALCO) is an organization within a financial institution whose charter is to manage interest rate risk.

At-Risk Period

The time period for value-at-risk. That is, the difference between the future date when portfolio loss is evaluated and the as-of-date.

Base Rates

A section in Oracle Risk Manager that stores the interest rates associated with the as-of-date of the data.

Basis Points

1/100th of a percent. Abbreviated as *bp*.

Breakage Charges

A charge assigned to an account whose actual cash flows are significantly different from the expected cash flows when the account funded.

Business Area

A conceptual grouping of tables or views represented as folders, items, conditions, filters, joins, and hierarchies. Each business area is grouped within the same End User Layer (EUL) that the current session is connected to.

Cash Flow Table

An instrument table that contains all of the Oracle Financial Services (OFS) cash flow columns.

Cash Flow Column

A column in an instrument table that is used by the OFS cash flow engine to perform cash flow analysis on a record.

Configuration ID

An ID that stores system default information that has been customized for the user and/or the institution.

CPR (Constant Prepayment Rate)

A method of measuring an annualized prepayment rate.

Credit Risk

The risk that a loan holder will not be able to repay some portion of a loan.

Current Rate Risk Profit

Component of funding center rate risk results attributed to current mismatches of assets and liabilities.

Data Correction Processing ID

An ID that enables you to simultaneously correct multiple columns in a database table in one pass through the database, using a previously defined Correction Rule ID.

DDA (Demand Deposit Account)

An account paying on demand, with no notice of withdrawal required, for example, a checking account.

De-annualize

To compute the monthly equivalent rate of an annual rate.

Duration

The rate of market value change with respect to discount rate changes. A measure of market value sensitivity: the lower the value, the less sensitive the market value to changes in interest rates.

Embedded Rate Risk Profit

A portion of funding center rate risk result attributed to prior rate bets.

End User Layer (EUL)

A metalayer that contains data about other tables in a database. Conceptually, the EUL shields the end user from the complexity of the database. The database tables are modified by the Oracle Discoverer Administration Edition and Oracle Financial Data Manager / Discoverer Integrator. Business areas are defined within the EUL and then used in the Oracle Discoverer End User Edition.

Floating Tree Bar

A window that enables you to choose the appropriate branch or leaf in the Tree ID that is currently active.

Funding Center

An area in a financial institution that receives the transfer pricing charge and credit for funds.

General Ledger (GL)

The main data source that defines the financial reality for an institution. The general ledger reflects all accounting entries.

Historical Rates Table

The OFS table that holds all historical actual interest rates.

Instrument

Legally enforceable agreements about types of financial security. See *Instrument Table.*

Instrument Table

A type of table in the Oracle Financial Data Manager (FDM) database that is used to store account-level information.

Instrument Records

Rows in the Oracle Financial Data Manager (FDM) database that contain account information for each customer. Examples are checking accounts, mortgage accounts, or installment debt accounts for a given customer.

Interest Rate Code

User-defined code to reference a yield curve or single rate index for historical analysis and transfer pricing and interest rate forecasting purposes.

Last Repricing Date

The date of the last rate change for an adjustable-rate instrument and the origination date for a fixed-rate instrument.

Leaf Fields

Oracle Financial Data Manager (FDM) database fields that are used to define hierarchical segmentations of data (trees). They also define the relationship between the instrument data and the general ledger data in the LEDGER_STAT table.

Leaf ID

A column in the Oracle Financial Data Manager (FDM) database that provides for the creation of hierarchical trees for use in assumptions and reporting. Leaf ID columns also define the relationship between instrument data and ledger data.

Leaf Values

Values that compose the leaf fields.

LEDGER_STAT

A table in the Oracle Financial Data Manager (FDM) database that stores all general ledger as well as statistical information for current and historical periods.

Liquidity Premiums

A charge levied on a long-term instrument to compensate for the illiquidity of the funds.

Long Run Rate

One of the user-defined parameters for the Vasicek (discrete-time) term structure model. It represents the equilibrium value of the one-month annually compounded rate.

Market Price of Risk

In financial economic theory, the market price of risk is a measure of intertemporal risk-aversion of the aggregate investor. For example, a high market price of risk during some future period means that investors will be more risk-averse, and that rates for that term should be higher to compensate for the risk. In practical terms, the market price of risk is the *plug* that makes the price of risk-free bonds correct.

Market Value

In Monte Carlo, the average of the (scenario specific) present values.

Matched Rate Transfer Pricing

A method of transfer pricing where all accounts have transfer rates that reflect their specific maturity and repricing characteristics.

Matched Spread

The interest profit margin for any account. For asset accounts, measured as the note rate minus the transfer rate and, for liability and equity accounts, the transfer rate minus the note rate.

Monthly Rate

The yield on a loan contracted at the beginning of the month for a period of one month, assuming a continuous compounding basis. The monthly rate is a function of time and scenario. The rate generator also computes rates for terms other than a month.

Next Repricing Term

The repricing frequency for an adjustable-rate instrument and the original term to maturity for a fixed-rate instrument

Operating Cost

The non-interest related cost of running a business.

Option Costs

The costs assigned to measure the value of any customer option on an instrument. For example, prepayments on a mortgage loan.

Portfolio Fields

Fields in the Oracle Financial Data Manager (FDM) database that are common to multiple instrument tables. Portfolio fields are determined by the OFS system administrator.

Prepayment

A reduction in the principal balance of a transaction record prior to the contracted schedule date.

Present Value

In Monte Carlo, the sum of cash flows paid by a security along a particular rate scenario, discounted by the stochastic discount factor.

Processing Filter ID

Filter ID used to define which data can go into a processing run.

PSA (Public Securities Association)

A prepayment specification method established by the Public Securities Association that relates a CPR to the age of an instrument.

Reconciliation

The process of comparing information in one data source to information in another data source.

Record

Usually a single account or transaction, or aggregation of accounts, stored in the database (also called a row).

Remote User

A user who is not connected to the client/server environment. A remote user needs to transmit information to the budget administrator using computer files.

Single Rate

An interest rate code with only one point defined. For example, prime rate or LIBOR.

Speed of Mean Reversion

One of the user-defined parameters for the Vasicek and Extended Vasicek (discrete-time) term structure models. Speed of mean reversion represents the long-run drift factor.

Spreadsheet Control Bar

A dialog box that facilitates the definition of a series of input values (dates or numbers) by providing several methods for defining structured patterns in the data.

SQL (Structured Query Language)

The standard method of accessing the Oracle Financial Data Manager (FDM) database.

Stochastic Discount Factor

Present value (along a rate scenario) of one dollar received at some future time. It is a function of future time, option adjusted spread (OAS), and scenario

Transfer Pricing

A method for valuing all sources and uses of funds for a balance sheet.

Transfer Pricing ID

An ID used to specify the method for transfer pricing all balance sheet accounts.

Transfer Pricing Table

An instrument table that contains all of the columns required to run transfer pricing.

Tree ID

Hierarchical structure for leaf fields in OFS products, for example, organization or product type.

UPR (Unscheduled Principal Runoff)

In addition to prepayments, a source of runoff in Risk Manager cash flow calculations. UPR is not applicable for Oracle Transfer Pricing.

Volatility

One of the user-defined term parameters of all (discrete-time) term structure models. It represents a standard deviation of the one-month annually compounded rate.

Yield Curve

The curve of annually compounded zero-coupon bond yield.

Index

A

B

C

D

E

F

G

Q

R

S

T

U

V

W

Y