

Oracle® Hyperion Financial Management, Fusion Edition

Administrator's Guide

RELEASE 11.1.2

ORACLE®

**ENTERPRISE PERFORMANCE
MANAGEMENT SYSTEM**

Financial Management Administrator's Guide, 11.1.2

Copyright © 2000, 2010, Oracle and/or its affiliates. All rights reserved.

Authors: EPM Information Development Team

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited. The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

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

U.S. GOVERNMENT RIGHTS:

Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, duplication, disclosure, modification, and adaptation shall be subject to the restrictions and license terms set forth in the applicable Government contract, and, to the extent applicable by the terms of the Government contract, the additional rights set forth in FAR 52.227-19, Commercial Computer Software License (December 2007). Oracle USA, Inc., 500 Oracle Parkway, Redwood City, CA 94065.

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

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

This software and documentation may provide access to or information on content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services.

Contents

Documentation Accessibility	15
Chapter 1. About Financial Management	17
Financial Management Features	17
Architecture	18
Performance Management Architect	18
EPM Workspace	18
Related Products	18
Financial Management Dimensions	18
Scenario Dimension	19
Year Dimension	19
Period Dimension	19
Entity Dimension	19
Value Dimension	20
Account Dimension	20
Intercompany Dimension	20
View Dimension	20
Custom Dimensions	21
User-Defined Elements	21
Chapter 2. Managing Applications	25
Application Administration	26
Creating Application Profiles	27
Entering Languages	27
Defining Calendars	28
Editing Frequencies	29
Editing Periods	29
Saving Application Profiles	32
Creating Applications	32
Using the Financial Management Configuration Utility	35
Registering Servers and Clusters	35
Enabling DCOM	36

Specifying Database Information for Extended Analytics	36
Opening Applications	37
Closing Applications	38
Deleting Applications	39
Copying Classic Administration Applications	40
Copying Performance Management Architect Applications	42
Copy as Classic Application	42
Copy as Performance Management Architect When Source Application Exists at Target	44
Using Sample Applications	46
Working with System Messages	47
Retrieving System Messages	47
Viewing and Printing System Messages	48
Deleting System Messages	48
System Message Detail Strings	48
Managing System Users	49
Viewing Users	49
Logging Off Users	50
Managing Servers and Applications	51
Disabling and Enabling Connections	51
Viewing Disabled Components	52
Synchronizing Servers	52
Auditing Tasks	53
Auditing Data	55
Monitoring Running Tasks	57
Scanning For and Deleting Invalid Records	58
Changing Settings for Multiple Server Tasks	59
Maintaining Server Logon Information	60
Chapter 3. Managing Application Security	61
Application Security Considerations	62
Launching the Shared Services Console from Financial Management	62
Assigning Users and Groups to Financial Management Applications	63
Setting Up Security Classes for Financial Management Applications	63
Assigning User Access to Security Classes	65
Setting Up E-mail Alerting	66
Running Security Reports for Financial Management Applications	68
Loading Application Security	68
Clearing and Loading Security Information	70
Before Clearing Security Information	71

After Clearing Security Information	72
Extracting Application Security	73
Chapter 4. Managing Metadata	75
Defining Accounts	76
Account Type Behavior	78
Defining Dynamic Accounts	79
Defining Custom Members	80
Defining Entity Members	81
Defining Scenario Members	82
Defining Application Settings	84
Organization by Period	86
Defining Consolidation Methods	87
Using Consolidation Methods	88
Using POWN and POWNMIN Methods	88
Assigning Consolidation Methods Manually	90
Defining Currencies	91
System-Generated Accounts	92
Consolidation Accounts	92
Ownership Accounts	93
Editing System-Generated Accounts	93
Setting Up Intercompany Partners	94
Editing System-Generated ICP Members	95
Editing System-Generated Value Members	95
Metadata Filtering Based on Security	96
Creating Metadata Files of the APP Format	96
File Format	97
Version	98
Application Settings	98
Currencies	98
Dimension	99
Members	99
Consolidation Methods	102
Hierarchies	102
Dimensions Not Included in Metadata Files	104
Using Metadata Manager Views	105
Changing the Metadata File Format	106
Tree View Tasks	106
List View Tasks	111

Sorting Metadata in List View	113
Creating Aliases for Custom Dimensions	113
Creating Metadata Reports in File Properties	114
Metadata Referential Integrity	114
Metadata Referential Integrity Checks	115
Metadata Log File Referential Integrity Errors	115
Loading Metadata	115
Load Options	116
Extracting Metadata	119
Chapter 5. Managing Member Lists	121
Creating Member List Files	121
EnumMemberLists	122
EnumMembersInList	122
Dynamic Member Lists	124
Dynamic POV Member List	124
Loading Member Lists	126
Extracting Member Lists	127
System Lists by Dimension	128
Chapter 6. Managing Journals	129
Creating Journal Files	129
File Format Section	130
Version Section	130
Journal Group Section	131
Standard Section	131
Recurring Section	131
Header Section	131
Loading Journals	133
Extracting Journals	135
Chapter 7. Managing Data Forms	139
Creating Data Forms in the Form Builder	139
Properties	140
POV	141
Columns	142
Rows	142
Details	143
Headers	143
Script	143

Data Form Script Syntax	144
AddMember	146
BackgroundPOV	147
Blank	148
Cn	149
Calc1	150
CalcByRow	150
Cell_Link	150
CellText	151
CustomHeader	151
CustomHeaderStyle	152
DynamicPOV	152
FormInputBoxLength	152
FormNumDecimals	153
FormRowHeight	153
FormScale	154
HeaderOption	154
Instructions	155
LineItemDetailSinglePeriod	155
Link	155
MaxCells	156
MaxColsForSparseRetrievalMethod	156
NoSuppress	156
NumDecimals	157
Override	157
PrintNumDataColsPerPage	158
PrintNumRowsPerPage	159
PrintRepeatHeadersonAllPages	159
Rn	159
ReadOnly	160
ReportDescription	161
ReportLabel	161
ReportSecurityClass	162
ReportType	162
SCalc	162
Scale	164
SelectablePOVList	164
ShowDescriptions	165
ShowLabels	165

String	166
Style	166
SuppressColHeaderRepeats	168
SuppressInvalidColumns	168
SuppressInvalidRows	169
SuppressNoDataColumns	169
SuppressNoDataRows	169
SuppressRowHeaderRepeats	170
SuppressZeroColumns	170
SuppressZeroRows	170
Using Relative Periods and Years	171
Order of Precedence for Conflicting Attributes	171
Editing Data Forms	172
Creating Data Form Folders	173
Viewing and Hiding Private Data Forms	174
Loading Data Forms	174
Extracting Data Forms	175
Deleting Data Forms	175
Chapter 8. Using Extended Analytics	177
Creating a Data Link (UDL) File for Extended Analytics	177
Configuring the Data Link for Oracle	178
Configuring the Data Link for Microsoft SQL Server	179
Configuring the Data Link for IBM DB2	179
Encrypting UDL Files	180
Extended Analytics Registry Settings	181
Star Schema	181
Star Schema Formats	183
Prefix Tables	183
Flat File Extract Options	184
Creating and Exporting Data to a Star Schema	185
Updating a Star Schema	187
Deleting a Star Schema	188
Creating a Star Schema Template	188
Deleting a Star Schema Template	189
Chapter 9. Defining Reports	191
Defining Reports	191
Explore Data Report Example	192
Creating Journal Reports	193

Journal Report Example	193
Creating Intercompany Partner Matching Reports	194
Selecting Member Lists for Intercompany Matching Reports	194
Selecting Accounts for Intercompany Matching Reports	194
Specifying Decimal Places in Intercompany Matching Reports	195
Selecting Style Sheets for Intercompany Matching Reports	195
Specifying Currencies in Intercompany Matching Reports	195
Suppression Options for Intercompany Matching Reports	195
Defining Intercompany Partner Matching Report Scripts	197
Creating Intercompany Transaction Reports	203
IC Transaction Report Syntax	204
Chapter 10. Managing Rules	207
Rule Types	208
Rule Considerations	209
Calculation Commands	210
Current Dimension Members	214
Account Expressions	215
Functions Automatically Clear Data	216
Error Messages	217
Rule Execution During Consolidation	217
Default Translation	218
Financial Management Objects	219
HS Object	219
Commonly Used Rules	220
Reusing Data	220
Setting Accounts by Calculating Amounts	220
Conditional Rules	221
Setting Opening Balances of All Accounts	223
Creating Rules in the Rules Editor	223
Configuring the Rules Editor	223
Using the Function Wizard	224
Loading Rules from the Rules Editor	225
Extracting Rules into the Rules Editor	225
Creating Rules Files	225
Equity PickUp Rules Example	226
Loading Rules	227
Extracting Rules	228

Chapter 11. Rule Functions	231
Functions Overview	234
ABSExp	239
AccountType	240
AccountTypeID	241
AddEntityToList	242
AddEntityToListUsingIDs	243
AddMemberToList	244
AddMemberToListUsingIDs	244
Alloc	245
AllowAdjFromChildren	246
AllowAdjs	247
CalculateExchangeRate	247
CalculateRate	248
CalcStatus	249
Clear	250
Con	251
Consol1, Consol2, Consol3	252
ConsolidateYTD	253
ContainsCellText	253
CreateInputCache	254
Currency	254
Custom1Top, Custom2Top, Custom3Top, and Custom4Top	255
Decimal	256
DefaultFreq	257
DefaultTranslate	257
DefaultView	258
DefCurrency	259
DOWn	260
Dynamic	261
Exp	262
Dimension Intersection Considerations	263
Period and Year Keywords	263
Mathematical Calculations	264
Placing Other Functions Within Exp	265
Simultaneously Setting Several Accounts	265
Exp and Dimension Intersection Considerations	265
GetCell	268
GetCellNoData	269

GetCellRealData	270
GetCellText	271
GetCellType	271
GetItem	272
GetItemIDs2	273
GetNumItems	274
GetNumLID	274
GetRate	275
GetSubmissionGroup	275
GetSubmissionPhase	276
Holding	276
ICPTopMember	277
ICPWeight	278
IDFromMember	278
ImpactStatus	279
Input	280
IsAlmostEqual	281
IsBase	281
IsCalculated	283
IsChild	283
IsConsolidated	285
IsDescendant	285
IsFirst	287
IsICP	288
IsLast	289
IsTransCur	289
IsTransCurAdj	290
IsValidDest	290
IsZero	291
List	292
Member	293
MemberFromID	293
Method	295
NoInput	295
NoRound	296
NumBase	296
Number	298
NumChild	298
NumDescendant	300

NumPerInGen	302
NumPeriods	302
OpenDataUnit	303
OpenDataUnitSorted	304
Owned	304
Owner	305
PCon	305
PEPU	306
PeriodNumber	306
PlugAcct	307
POwn	308
PVAForBalance	309
PVAForFlow	309
RateForBalance	310
RateForFlow	311
ReviewStatus	311
ReviewStatusUsingPhaseID	312
Round	313
Scale	314
SecurityAsPartner	315
SecurityClass	316
SetData	317
SetDataWithPOV	317
SubmissionGroup	318
SupportsProcessManagement	319
SupportsTran	319
SwitchSign	320
SwitchType	321
Trans	322
TransPeriodic	323
UD1...3	324
ValidationAccount	325
ValidationAccountEx	325
XBRLTags	326
Chapter 12. Creating Rules Using Calculation Manager	327
Calculation Manager Security Roles	327
Working with Applications in Calculation Manager	328
Migrating Rules to Calculation Manager	328

VB Function Support in Function Selector	329
Special VB Script Functions for Financial Management	330
Chapter 13. Managing Intercompany Transactions	333
Managing Intercompany Transaction Periods	333
Opening Periods	333
Setting Matching Tolerances	334
Setting Match/Validate Before Post Option	335
Closing Periods	335
Locking and Unlocking Entities	336
Managing Reason Codes	337
Adding Reason Codes	337
Editing Reason Codes	338
Removing Reason Codes	338
Monitoring Intercompany Transactions	338
Viewing the Intercompany Transactions Summary	340
Chapter 14. Managing Process Management Submission Phases	341
Defining Submission Phases by Scenario and Period	341
Submission Group Assignments by Scenario and Period	343
Assigning Submission Groups to Phases	345
Viewing Unassigned Submission Groups	346
Chapter 15. Managing E-mail Alerting	347
Setting Up Process Management Alerting	347
Setting Up Intercompany Transaction Alerting	348
Appendix A. Accessibility	351
Enabling Accessibility for Financial Management	351
Using Keyboard Equivalents	351
File Menu	351
View Menu	352
Administration Menu	352
Navigating to the Application	353
Financial Management Preferences	353
Shortcut Keys by Module	354
Glossary	387
Index	391

Documentation Accessibility

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

Accessibility of Code Examples in Documentation

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

Access to Oracle Support for Hearing-Impaired Customers

Oracle customers have access to electronic support through My Oracle Support or by calling Oracle Support at 1.800.223.1711. Hearing-impaired customers in the U.S. who wish to speak to an Oracle Support representative may use a telecommunications relay service (TRS). Information about the TRS is available at <http://www.fcc.gov/cgb/consumerfacts/trs.html/>, and a list of telephone numbers is available at <http://www.fcc.gov/cgb/dro/trsphonebk.html>. International hearing-impaired customers should use the TRS at +1.605.224.1837. An Oracle Support engineer will respond to technical issues according to the standard service request process.

In This Chapter

Financial Management Features.....	17
Architecture.....	18
Performance Management Architect	18
EPM Workspace	18
Related Products.....	18
Financial Management Dimensions.....	18
User-Defined Elements	21

Financial Management Features

Oracle Hyperion Financial Management, Fusion Edition supports these features:

- A single, unified view of enterprise financial information consolidates key performance and operating metrics from global sources in a scalable, Web-based application.
- “Virtual close” features trim days and weeks off your close cycle using Web-based intercompany reconciliations and a consistent set of data and business measures.
- Powerful multidimensional analysis identifies and reports new sources of profitability and cash flow at corporate, cost center, product, brand, customer, and channel levels.
- Flexible “what if” scenario management feature dynamically consolidates and reports all financial budgets, forecasts and plans, producing new statements as assumptions and facts change.
- High-volume, preformatted reports deliver timely, accurate financial information for internal management as well as for external regulatory and government bodies, from the same application.
- Prepackaged features are deployed out-of-the-box, quickly and cost-effectively, including features such as world-class allocations, multicurrency translations, and robust data integration with legacy applications, ERP, and CRM systems.
- Customizable and extensible application solves your specific issues quickly and cost-effectively, using industry standard tools.
- Architected for the Web so users can easily and securely access global financial information from any location, using a standard Web browser. Relational data storage ensures mission critical data is available to users 24x7x365.

Architecture

Financial Management is designed to operate as a multitier system.

- The client tier contains the user interface and the ability to communicate with the application tier. You can display data and metadata, enter data, and maintain metadata in this tier.
- On the Web server tier, you access the Web-enabled portions of Financial Management.
- The middle tier contains the domain intelligence and connections to the relational database.
- The data tier contains the relational database and all Financial Management data and metadata.

Performance Management Architect

Oracle Hyperion EPM Architect, Fusion Edition is an optional component of Financial Management installation and configuration. You use it to create and work with applications and dimensions, synchronize data, and work with Oracle Hyperion Planning, Fusion Edition. In addition, you can use it to create and work with rules using Hyperion Calculation Manager.

For help on tasks performed in Performance Management Architect, see the *Oracle Hyperion Enterprise Performance Management Architect Administrator's Guide*.

EPM Workspace

Financial Management is available within Oracle Enterprise Performance Management Workspace, Fusion Edition. For information on tasks performed in EPM Workspace, such as Workspace preferences or features in the Navigate, Favorites, Manage, or Tools menu, see the *Oracle Enterprise Performance Management Workspace User's Guide*, *Administrator's Guide* and Web help.

Related Products

You can use the Extended Analytics module to send data to a database in Oracle Essbase. You can also drill down from data grids and data forms to view data details in Oracle Hyperion Financial Data Quality Management, Fusion Edition. In addition, you can work with data and data forms in Oracle Hyperion Smart View for Office, Fusion Edition.

Financial Management Dimensions

Dimensions describe an organization's data and usually contain groups of related members. Examples of dimensions are Account, Entity, and Period. Financial Management supplies eight system-defined dimensions and enables you to populate up to four custom dimensions that you can apply to accounts.

Dimension members are arranged in hierarchies. Upper-level members are called parent members, and a member immediately below a parent member is referred to as the child of a parent member. All members below a parent are referred to as descendants. The bottom-level hierarchy members are called base-level members.

The following sections describe the system-defined dimensions. For information on setting dimension attributes, see the *Oracle Hyperion Enterprise Performance Management Architect Administrator's Guide* if you are using Performance Management Architect, or [Chapter 4, “Managing Metadata”](#) in this guide if you are using Classic Application Administration.

Scenario Dimension

The Scenario dimension represents a set of data, such as Budget, Actual, or Forecast. For example, the Actual scenario can contain data from a general ledger, reflecting past and current business operations. The Budget scenario can contain data that reflects the targeted business operations. The Forecast scenario typically contains data that corresponds to predictions for upcoming periods. A Legal scenario can contain data calculated according to legal GAAP format and rules.

You can define any number of scenarios for an application and define attributes for Scenario dimension members, such as the default frequency, the default view, and zero data settings.

Year Dimension

The Year dimension represents the fiscal or calendar year for data. An application can contain data for more than one year. You specify a year range when you create the application and select a year from the Year dimension to process data.

Period Dimension

The Period dimension represents time periods, such as quarters and months. It contains time periods and frequencies by displaying the time periods in a hierarchy. For example, if the Actual scenario maintains data on a monthly basis, generally 12 periods of data are available for this scenario in a year. Financial Management supports years, months, and weeks for the period dimension. It does not support days for the dimension.

Entity Dimension

The Entity dimension represents the organizational structure of the company, such as the management and legal reporting structures. Entities can represent divisions, subsidiaries, plants, regions, countries, legal entities, business units, departments, or any organizational unit. You can define any number of entities.

The Entity dimension is the consolidation dimension of the system. Hierarchies in the Entity dimension reflect various consolidated views of the data. Various hierarchies can correspond to geographic consolidation, legal consolidation, or consolidation by activity. All relationships among individual member components that exist in an organization are stored and maintained

in this dimension. Entities in an organization are dependent, base, or parent entities. Dependent entities are owned by other entities in the organization. Base entities are at the bottom of the organization structure and do not own other entities. Parent entities contain one or more dependents that report directly to them.

You define attributes for Entity dimension members, such as the default currency and security class, and to specify whether the entity allows adjustments and stores intercompany detail.

Value Dimension

The Value dimension represents the different types of values stored in your application, and can include the input currency, parent currency, adjustments, and consolidation detail such as proportion, elimination, and contribution detail. For example, the Entity Currency member stores the value for an entity in its local currency. The Parent Currency member stores the value for an entity translated to the currency of its parent entity. The Value dimension is useful for providing an audit trail of the transactions applied to data.

Account Dimension

The Account dimension represents a hierarchy of natural accounts. Accounts store financial data for entities and scenarios in an application. Each account has a type, such as Revenue or Expense, that defines its accounting behavior.

You define attributes for Account dimension members, such as the account type, the number of decimal places to display, and whether the account is a calculated, consolidated, or intercompany partner account.

Intercompany Dimension

The Intercompany dimension represents all intercompany balances that exist for an account. This is a reserved dimension that is used in combination with the Account dimension and any custom dimension. Financial Management can track and eliminate intercompany transaction details across accounts and entities. You can also run Intercompany Matching reports to view intercompany transactions.

View Dimension

The View dimension represents various modes of calendar intelligence; for example, Periodic, Year-to-Date, and Quarter-to-Date frequencies. If you set the view to Periodic, the values for each month are displayed. If you set the view to Year-to-Date or Quarter-to-Date, the cumulative values for the year or quarter are displayed.

Custom Dimensions

Four custom dimensions are available for analysis of detailed data. You can use custom dimensions to store additional details associated with accounts, such as products, markets, channels, balance sheet movement, or types of elimination. For example, custom dimensions could include Product Line, Region, Channel, or Customers.

User-Defined Elements

Many elements in Financial Management are user-defined. For example, when you create a journal, you give the journal a label and a description.

User-defined elements, the minimum and maximum length for each element, and additional restrictions are listed below. The table groups the elements by the modules in which they are found.

Table 1 Requirements for User-Defined Elements

Element	Min. length	Max. length	Restrictions
Application Profile			
Language	1	20	None
Period label	1	80	● Must contain only alphanumeric characters. ● Cannot start with a number. ● Cannot contain spaces, symbols, or diacritical marks such as umlauts.
View label	1	10	● Must contain only alphanumeric characters. ● Cannot start with a number. ● Cannot contain spaces, symbols, or diacritical marks such as umlauts.
View description	0	40	Cannot contain an ampersand (&).
Period description	0	40	Cannot contain an ampersand (&).
Create Application			
Application label	1	10	● Must contain only alphanumeric characters. ● Cannot start with a number. ● Cannot contain spaces, symbols, or diacritical marks such as umlauts. Note: Application labels are not case-sensitive. For example, App1 and APP1 are considered the same application label.
Application description	1	255	Cannot contain an ampersand (&).

Element	Min. length	Max. length	Restrictions
Metadata Manager			
Member label	1	80	Must be unique. The label can contain up to 80 characters including spaces, but cannot start with a space. Note: If you are using an Oracle database, member labels cannot contain spaces. You cannot use the following characters in the member name: ● period (.) ● plus sign (+) ● minus sign (-) ● asterisk (*) ● forward slash (/) ● number sign (#) ● comma (,) ● semicolon (;) ● at symbol (@) ● double quotation mark (") ● curly brackets ({ }) Note: You cannot use ALL as the name of an entity.
Member description	0	40	Cannot contain an ampersand (&).
Alias label	0	80	Cannot contain an ampersand (&).
Security			
Security class	1	80	None
Journals			
Journal label	1	80	Cannot include the following characters: ● period (.) ● plus sign (+) ● minus sign (-) ● asterisk (*) ● forward slash (/) ● number sign (#) ● comma (,) ● semicolon (;) ● at symbol (@) ● double quotation mark (") ● curly brackets ({ }) Note: If you are using an Oracle database, journal labels cannot contain spaces.
Journal description	0	255	None

Element	Min. length	Max. length	Restrictions
Journal group	0	30	None
Journal line item description	0	50	None
Load/Extract			
Delimiter character	1	1	Must be one of the following characters and cannot be used in the file or in the file name: ● comma (,) ● tilde (~) ● at sign (@) ● dollar sign (\$) ● percent sign (%) ● ampersand (&) ● carat (^) ● line () ● colon (:) ● semicolon (;) ● question mark (?) ● back slash (\) Note: The ampersand (&) is not a valid delimiter for metadata .app files. You must use the same delimiter character throughout the file. Using different delimiter characters within the same file causes an error when you load the file.
Data grids			
Cell description	1	1900	None
Line item detail	1	80	None
Annotation	0	255	None
Decimal character	1	1	The following characters are invalid decimal characters for data grids: ● back slash (\) ● forward slash (/) ● plus sign (+) ● minus sign (-)

Element	Min. length	Max. length	Restrictions
Documents			
Document names (including folder and report names)	1	16	The following characters are invalid characters for document names: ● plus sign (+) ● question mark (?) ● asterisk (*) ● back slash (\) ● forward slash (/) ● number sign (#) ● comma (,) ● semicolon (;) ● colon (:) ● at sign (@) ● double quotation mark (") ● curly brackets ({ }) ● line () ● less than sign (<) ● greater than sign (>) ● period (.) at the end of a document name Note: Document names also cannot contain trailing or leading white space.

In This Chapter

Application Administration	26
Creating Application Profiles	27
Creating Applications	32
Using the Financial Management Configuration Utility	35
Opening Applications	37
Closing Applications	38
Deleting Applications	39
Copying Classic Administration Applications	40
Copying Performance Management Architect Applications	42
Using Sample Applications	46
Working with System Messages	47
Managing System Users	49
Managing Servers and Applications	51
Auditing Tasks	53
Auditing Data	55
Monitoring Running Tasks	57
Scanning For and Deleting Invalid Records	58
Changing Settings for Multiple Server Tasks	59
Maintaining Server Logon Information	60

Caution! The information in this chapter about application administration is provided for use with Classic Financial Management applications only. If you upgrade an application created in Classic Application Administration to Performance Management Architect, you cannot return to working with that application in Classic Application Administration. For information on application administration using Performance Management Architect, see the *Oracle Hyperion Enterprise Performance Management Architect Administrator's Guide*.

For application administration, these security roles are required: Dimension Editor and Application Creator/Financial Management Application Creator. For information on roles, see the *Oracle Hyperion Enterprise Performance Management System User and Role Security Guide*.

An application consists of a set of entities, accounts, scenarios, and other dimensions that you use together. You can create as many applications as you need. For example, you can set up one application to report on tax data for several organizations and another application to report on Security and Exchange Commission data for other organizations.

Applications run on application servers. You can set up clusters of application servers to balance the load on multiple servers. For instructions, see the *Oracle Hyperion Enterprise Performance Management System Installation and Configuration Guide*.

Note: If you are using the Financial Management Win32 client, or if you selected the ADM Client component during installation to work with Oracle Hyperion Financial Reporting, Fusion Edition, Oracle's Hyperion® Web Analysis, and Oracle's Hyperion® Application Builder for J2EE, you must register servers and clusters using the Financial Management Configuration Utility. For instructions, see [“Using the Financial Management Configuration Utility” on page 35](#).

Only members of the Administrator group that you specify when you configure the application server can perform these administrative tasks:

Users on System

- List users
- Log out users

Manage Servers and Applications

- View disabled components
- Enable or disable connections
- Log out users

System Messages

- View - must have Application Administrator role
- Delete - must be member of Administrator group

To assign users to the Administrator Group, see the *Oracle Hyperion Enterprise Performance Management System User and Role Security Guide*.

Application Administration

You define an application in five steps:

1. Create an application profile, which contains calendar, language, and frequency definitions for the application. See [“Creating Application Profiles” on page 27](#).
2. Create an application shell, in which you define the application server where the application resides, an application label and description, an application profile, and a location for a log file. See [“Creating Applications” on page 32](#).

3. Define security for the application including which users have access to the application and what type of access each user has. See [Chapter 3, “Managing Application Security.”](#)
4. Define metadata for the application, including accounts, entities, scenarios, and custom dimensions, as well as application settings, consolidation methods, and currencies. See [Chapter 4, “Managing Metadata.”](#)
5. Load data, data forms, member lists, rules, and journals to the application.

You can have only one application open at a time, however, you can have several instances of Financial Management open with different applications on each instance.

Creating Application Profiles

An application profile contains language, calendar, frequency, and period information for an application.

You can use a profile for more than one application. For each application that you create, you must specify an application profile.

Note: You create application profiles in the Financial Management Desktop in Windows.

See these topics:

- [Entering Languages](#)
- [Defining Calendars](#)
- [Editing Frequencies](#)
- [Editing Periods](#)
- [Saving Application Profiles](#)

► To create an application profile:

- 1 From the Financial Management Windows desktop, select **Define Application Profile**.
- 2 Select an option:
 - Create a New Application Profile
 - Select an Existing Application Profile and click  to locate the file.

Note: By default, application profile files use the PER file extension.

- 3 Click **Next**.

Entering Languages

You can define up to 10 languages for labels that are used throughout an application. You can use different languages to create descriptions for items in your application.

► To set up languages:

1 In the grid, enter a language, and press the **Tab** key to move down to the next line.

Note: You can enter up to 10 languages, and each language can contain a maximum of 20 characters. Note that a space is counted as a character.

2 Click **Next**.

Defining Calendars

When you select the type of calendar and the time periods for the application profile, default frequencies are created for the application profile. For example, if you select standard calendar and include half-years, quarters, and months as the time periods, the system creates these frequencies: yearly, half-yearly, quarterly, and monthly.

You can also select a custom calendar or manual calendar. If you choose to define a custom calendar, you must define the number of periods and the period label prefix.

A flat list is created and you can modify the hierarchy later. If you choose to define a manual calendar, the frequencies and periods are empty. You can then enter the frequencies and periods that you need.

► To set up a calendar:

1 Select a type of calendar:

- Standard Calendar
- Custom Calendar
- Manually Defined Calendar

2 Do one of these actions:

- If you selected **Standard Calendar**:
 - Select the time periods to include (half-years, quarters, trimesters, or months).
 - If you included months in the calendar, from the **Start Month** list, select the first month in the calendar.
- If you selected **Custom Calendar**:
 - For **Number of Base Periods**, enter the number of periods in the year.
 - For **Period Label Prefix**, enter a prefix for the periods to include.

The label can contain a maximum of 10 characters and can include spaces.

Tip: If you enter 10 for the number of periods and NewPeriod as the label prefix, these periods are added to the hierarchy: NewPeriod1 through NewPeriod10.

- If you selected **Manually Defined Calendar**, continue with the next step.

3 For **Start Year**, enter the start year for the calendar.

4 For **Number of Years**, enter the total number of years to include.

5 Click Next.

Editing Frequencies

Frequencies and their corresponding views are created based on the time periods that you selected when defining the calendar. You can add, modify, and delete frequencies. You can also enter a descriptive label for each frequency and view in each language that you previously defined.

When editing frequencies, you cannot change the label of the YTD frequency, which is contained in the first column of Frequency 1 row. However, you can enter a description of the YTD frequency for each language that you define.

Note: In addition to the frequencies that you can define, each application contains two system-defined frequencies and corresponding views, Scenario View and Periodic.

If you selected Manually Defined Calendar as the time period for the calendar, the Frequencies grid is empty, and you must enter the necessary frequency views and their descriptions. You should enter one frequency for each level of the Period dimension.

Caution! After you create an application, you cannot change the frequency descriptions or labels in the application.

► To set up frequencies:

- 1 From the **Language** list, select a language.
- 2 Enter or edit the values in the **View** column and enter a **View description**.

You can enter up to six frequencies.

Note: The View label can contain a maximum of 80 characters. The View description can contain a maximum of 40 characters. Labels cannot include these characters:

. + - * / # { } ; , @ "

Tip: Because you cannot modify frequencies after an application is created, make sure to include a view description for each frequency in each language.

3 Click Next.

Editing Periods

The period hierarchy is a combination of the time period and frequencies that you define. You can make changes to this hierarchy by adding or deleting periods. For example, you may want to add another month to the fourth quarter to achieve a 13-month year.

For instructions on adding sibling and child periods to the hierarchy, see [“Adding Sibling and Child Periods” on page 30](#).

When you select a period, you can view the period frequency and change the period label or description.

Note: You cannot edit period labels based on language. In other words, if you change a period label in one language, the change is carried over into all other languages defined in the application profile. However, you can have a unique period description for each language.

► To enter and edit periods:

- 1 From the period hierarchy, select a period, and, for **Description**, enter or edit the period description.

The period description can contain a maximum of 40 characters. Note that a space is counted as a character. You cannot edit the description for the Year period.

Tip: Make sure to include a description for each period when you are creating an application profile. If you do not include a description and choose to add one later, you will need to modify the application profile and re-create all applications that use the profile.

- 2 For **Label**, enter or edit the period label.

The period label can contain a maximum of 80 characters. Note that a space is counted as a character. You cannot edit the label for the Year period. Changes that you make to the period label are reflected in all languages defined in the application profile.

Tip: You can use  and  to expand or collapse the levels in the period hierarchy.

- 3 From the **Language** list, select a language.

You can have a unique period description for each language. You cannot have a unique period label for each language.

- 4 Click **Next**.

Adding Sibling and Child Periods

You can add single or multiple sibling and child periods to the period hierarchy.

► To add a single child or sibling period:

- 1 Highlight the period to which to add a child or sibling period.
- 2 Right-click the period, and select **Insert Sibling**, or **Insert Child**.

Tip: You can also use the Insert Child toolbar button  and the Insert Sibling toolbar button .

- 3 Enter a name for the new sibling or child period.

➤ To add multiple child or sibling periods:

- 1 Highlight the period to which to add multiple child or sibling periods.
- 2 Right-click the period, and select **Insert Multiple**.
- 3 Select an option:

- Multiple Sibling Periods  to add multiple sibling periods to the selected period
- Multiple Child Periods  to add multiple child periods to the selected period

- 4 Enter the number of periods to add to the hierarchy.
- 5 Enter a label prefix for the new periods.

Tip: If you enter 10 for the number of periods and NewPeriod as the label prefix, these periods are added to the hierarchy: NewPeriod1 through NewPeriod10.

Moving Periods

You can use the mouse to drag and drop periods. You can move periods to sibling or child positions in the hierarchy.

➤ To move periods:

- 1 Highlight a period to move.
- 2 Click and drag the period to the new location in the hierarchy.
- 3 Select an option:
 - Move Here (Sibling) to move the period and make it a sibling of the highlighted period
 - Move Here (Child) to move the period and make it a child of the highlighted period

Note: You can also copy the period as a sibling or child of the selected period. When you release the mouse button, select the Copy Here (Sibling) or Copy Here (Child) option.

Copying Periods

You can copy single and multiple periods in the period hierarchy. You can copy a selected period and all child periods of the selected period. You can paste a copied period as a sibling or child of the selected period.

➤ To copy periods:

- 1 Highlight a period to copy.
- 2 Right-click the highlighted period and select an option:
 - Copy Period to copy only the highlighted period
 - Copy All to copy the highlighted period and all child periods of the highlighted period

3 Right-click the period to which to paste the copied period and select an option:

- **Paste as Sibling** to paste the copied period as a sibling of the selected period
- **Paste as Child** to paste the copied period as a child of the selected period

4 For **Label, make changes to the period label.**

The label can contain a maximum of 80 characters. Note that a space is counted as a character.

5 For **Description, enter a description for the period.**

The description can contain a maximum of 40 characters. Note that a space is counted as a character.

Deleting Periods

When you delete periods from the hierarchy, all children of the period are also deleted.

Note: You cannot delete the Year period.

➤ To delete a period, take one of these actions:

- Right-click the period and select **Delete**.
- Highlight the period and click .

Saving Application Profiles

By default, application profile files use the PER file extension. If you are creating an application profile, you must complete all steps in the Design Application Profile module.

➤ To save an application profile:

- 1 For **File Name**, enter a name for the file, or click  to browse for a profile to overwrite.**

Note: By default, application profile files use the PER extension.

- 2 Click **Finish**.**

Creating Applications

Caution! This information is provided for use with Classic Financial Management applications only. For information on application administration using Performance Management Architect, see the *Oracle Hyperion Enterprise Performance Management Architect Administrator's Guide*.

You create an application by specifying this information:

- The application server cluster on which to run the new application (The cluster name must be a valid server name.)

Note: If you are using the Financial Management Win32 client, or if you selected the ADM Client component during installation to work with Financial Reporting, Web Analysis, and Oracle's Hyperion® Application Builder for J2EE, you must register servers and clusters using the Financial Management Configuration Utility. For instructions, see [“Using the Financial Management Configuration Utility” on page 35](#).

- The application label
- The application description
- The application profile, which contains calendar, language, and frequency definitions for the application
- The local storage folder, which stores some application information on your workstation
- The Oracle's Hyperion® Shared Services project to which to add the application.

See the *Oracle Hyperion Enterprise Performance Management System User and Role Security Guide*.

- The Web server URL for security administration

Note: After you create an application, you must set up security for it. See [Chapter 3, “Managing Application Security.”](#)

Windows Procedure

➤ To create an application:

- 1 From the desktop navigation frame, select **Create Application**.
- 2 From the **Server** list, select the application server cluster on which to run the application.

If the server cluster is not listed, you may need to register the server cluster. See [“Registering Servers and Clusters” on page 35](#).

- 3 For **Application Label**, enter a name for the application.

The application label can have a maximum of 10 alphanumeric characters but cannot start with a number or contain spaces or special characters. Application labels are not case-sensitive. For example, App1 and APP1 are considered the same application. HFM, HSV, and HSX are reserved names and cannot be used for application labels.

Caution! Do not create applications with the same name even if they are on different Financial Management application servers. Applications of the same name but from different Financial Management application servers cannot coexist on one Shared Services server.

- 4 For **Application Description**, enter a description for the application.

The application description can have a maximum of 255 characters, and can include spaces, but cannot contain an ampersand (&) character. HFM, HSV, and HSX are reserved names and cannot be used for application descriptions.

- 5 For **Application Profile**, enter the path and file name of the application profile, or click  to browse for the application profile to use.

By default, application profiles use the PER file extension.

- 6 For **Local Storage Folder**, enter the path and the folder name where downloaded application files will be stored on the workstation, or click  to browse to the local storage folder.

- 7 From the **Project** list, select the Shared Services project to which to add the application.

Note: Each application must belong to a Shared Services project. See the *Oracle Hyperion Enterprise Performance Management System User and Role Security Guide*.

- 8 For **Financial Management Web Server URL for Security Administration**, enter the URL.

- 9 Click **Create**.

- 10 Click **Close**.

Web Procedure

- To create an application:

- 1 Select **Navigate**, then **Administer**, then **Classic Application Administration**, and then **Consolidation Administration**.

- 2 Select **Create Application**.

- 3 From the **Server** list, select the application server cluster on which to run the new application.

If the server cluster is not listed, you may need to register it. See the *Oracle Hyperion Enterprise Performance Management System Installation and Configuration Guide*.

- 4 For **Application Name**, enter a name for the new application.

The application label can have a maximum of either 10 alphanumeric characters or 12 bytes. It cannot start with a number or contain spaces, or special characters. Application labels are not case-sensitive. For example, App1 and APP1 are considered the same application. HFM, HSV, and HSX are reserved names and cannot be used for application labels or descriptions.

Caution! Do not create applications with the same name even if they are on different Financial Management application servers. Applications of the same name but from different Financial Management application servers cannot coexist on one Shared Services server.

- 5 For **Application Description**, enter a description for the application.

The application description can have a maximum of 255 characters, and can include spaces. HFM, HSV, and HSX are reserved names and cannot be used for application descriptions.

- 6 Click **Browse** next to the **Application Profile** text box, and locate the application profile to use.

By default, application profiles use the PER file extension.

- 7 From the **Project** list, select the **Shared Services** project to which to add the application.

Note: Each application must belong to a project. See the *Oracle Hyperion Enterprise Performance Management System User and Role Security Guide*.

- 8 For **Financial Management Web Server URL for Security Administration**, enter the URL.
- 9 Click **Create**.

Using the Financial Management Configuration Utility

When you install Financial Management, you configure it using the Oracle's Hyperion Enterprise Performance Management System Configurator. However, If you use the Financial Management Win32 client, or if you selected the ADM Client component during installation to work with Financial Reporting or Web Analysis, you must register servers and clusters and enable DCOM using the Financial Management Configuration utility.

If you are using the Extended Analytics feature, you use the Financial Management Configuration Utility to specify database information for Extended Analytics.

Registering Servers and Clusters

You must register your server or cluster before you can access applications.

If you register one of the servers in a cluster, the system registers all of the servers in that cluster. For example, you create an application server cluster named `Marketing_Servers` with application servers `Mkt1`, `Mkt2`, `Mkt3`, and `Mkt4`. You can enter `Mkt1` as the server to be registered, and the system registers all of the servers in that cluster.

If you run the configuration utility from an admin account, the configuration utility creates the `HKeyLocalMachine/Software/Hyperion Solutions` registry key so that you can register server clusters. If you are not running the configuration utility from an admin account, you might encounter problems when registering servers. In this case, in the registry, you should manually assign Read/Write access rights to `HKeyLocalMachine/Software` for the user who is registering the servers.

When you specify an individual server name from the servers in the cluster, the entire cluster is registered. If there is only one application server in the server cluster, you must enter the server name.

- To register an application server cluster:

- 1 Start the utility.
- 2 In the **Server/Cluster Registration** box, enter the server name that you are registering.
The application server name is used to obtain application server cluster information.
- 3 Select one of these options:

- Use **Automatic Load Balancing** - to register the server cluster that was created in the Application Server Clusters tab.
- Use **One Server Only** - to specify that only one server be used instead of a cluster

4 Click Add.

Tip: If you need to unregister a server or cluster, select it from the list and click **Remove**.

Enabling DCOM

From the configuration utility, you can enable DCOM for the entire computer. Enabling DCOM makes possible the launching of servers and connecting to objects by remote clients for the machine. It also sets the DCOM Default Authentication level to None for the computer. The Enable DCOM step is required for Financial Management client components to communicate with Financial Management application server components when the application server is on a different computer. It also makes it possible for the Financial Management client and application server computers to be on different domains.

➤ To enable DCOM:

1 Click Enable DCOM.

Note: This button is grayed out if previously configured.

2 Click OK.

Specifying Database Information for Extended Analytics

The Extended Analytics feature enables you to use Essbase to analyze data and produce reports. You use a data source name (DSN) to specify the relational database destination for Extended Analytics. You must specify the data source name and path. You can add, remove, or modify data source names as needed.

Note: This tab is only available if you installed the Financial Management Server components during the installation process.

Note: When you use multiple servers with the Extended Analytics feature, you must enable multiple server support on each application server, and you must set up each application server so that these elements are the same on all servers: Clock setting, Data source, and UDL file.

For information on using Extended Analytics, see [Chapter 8, “Using Extended Analytics”](#).

➤ To add a data source name:

- 1 From the Financial Management Configuration Utility, select the **Extended Analytics DSNs** tab.

Note: This tab is available only if you installed the Financial Management Server components during the installation process.

- 2 Click **Add DSN**.

- 3 In **DSN Name**, enter the data source name.

- 4 In **DSN Path**, enter or browse for the data source path.

- 5 **Optional:** To specify the database tablespaces in which the Financial Management data tables and indexes are created:

- a. Click  (next to the **Data Tablespace** field), and select the data table location.
- b. Click **OK**.
- c. Click  (next to the **Index Tablespace** field), and select the index location.

Note: For Oracle and IBM DB2 SMS, you can specify only a data tablespace; index tablespaces are not used.

- d. Click **OK**.

- 6 Click **OK**.

➤ To modify a data source name:

- 1 From the Financial Management Configuration Utility, select **Extended Analytics DSNs**.

- 2 In **DSN Info**, select the DSN.

- 3 Click **Modify DSN** and change the name.

- 4 Click **OK**.

➤ To remove a data source name:

- 1 From the Financial Management Configuration Utility, select **Extended Analytics DSNs**.

- 2 In **DSN Info**, select the DSN.

- 3 Click **Remove DSN** and click **OK**.

Opening Applications

You can open one application for each Financial Management session that you are running. However, you can have several Financial Management sessions open, with different applications open on each session.

You must specify a local working folder for each application. You specify the working folder for an application when the application is created. If the application was created on a computer

other than the one that you are using, you need to specify a local working folder the first time you open the application.

Note: To access an application, you must be assigned as a user of the application. See “[Assigning Users and Groups to Financial Management Applications](#)” on page 63.

Windows Procedure

► To open an application:

- 1 From the desktop navigation frame, select **Open Application**.
- 2 Click **Connect**.

When you connect to an application server or cluster, the system authenticates your user name and password. If you enter an invalid username-password combination, you cannot access or create applications.

- 3 From the **Cluster** list, select the application server or cluster on which the application is running.
- 4 Highlight the application to open, and click **Open Application**.

Web Procedure

► To open an application:

- 1 Select **Navigate**, then **Applications**, then **Consolidation**, and *AppName*, where *AppName* is the application name.
- 2 From the **Select Cluster** list, select the application server on which the application is running.

When you connect to an application server or cluster, the system authenticates your username and password. If you enter an invalid username-password combination, you cannot access or create applications.

If the server cluster is not listed, you may need to register it. See the *Oracle Hyperion Enterprise Performance Management System Installation and Configuration Guide*.

- 3 Select the application to open.

Closing Applications

When you close an application, the system prompts you to save the changes that you made.

Note: If you close an application and then want to re-open it, wait approximately 10 seconds before reopening the application to provide time for the Financial Management services to shut down properly.

Windows Procedure

- To close an application:
 - 1 From the desktop navigation frame, select **Close Application**.
 - 2 Click **Yes** to close the application.

Web Procedure

- To close an application:
 - 1 Select **File**, then **Close Application**.
 - 2 Click **Yes** to close the application.

Deleting Applications

Caution! This information is provided for use with Classic Financial Management applications only. For information on application administration using Performance Management Architect, see the *Oracle Hyperion Enterprise Performance Management Architect Administrator's Guide*.

Before deleting an application, be sure that no other user is currently using the application.

Note: To delete an application, the user must be the Application Administrator for that application. Additionally, for the Web, the user also needs the global roles of Dimension Editor and Application Creator.

Windows Procedure

- To delete an application:
 - 1 From the desktop navigation frame, select **Delete Application**.
 - 2 From the **Cluster** list, select the application server cluster on which the application is running.

If the server cluster is not listed, you may need to register the server cluster. See the *Oracle Hyperion Enterprise Performance Management System Installation and Configuration Guide*.
 - 3 Click **Connect**.
 - 4 Select an application, and click **Delete Application**.
 - 5 Click **Yes** to delete the application.
 - 6 Click **Close**.

Web Procedure

► To delete an application:

1 Select **Navigate**, then **Administer**, then **Classic Application Administration**, and then **Consolidation Administration**.

2 Select **Delete Application**.

If you have the application open, you must close it before you can delete it.

3 Select the application server on which the application is running.

If the server cluster is not listed, you may need to register it. See the *Oracle Hyperion Enterprise Performance Management System Installation and Configuration Guide*.

4 Select the application to delete.

5 Click **Yes** to delete the application.

Copying Classic Administration Applications

Financial Management provides a utility to copy and move an application from one application server to the same server or another application server. An application can be copied across databases. For example, an application created in a SQL database can be copied to an Oracle database.

The copy application utility, `HFMCopyApplication.exe`, is installed to this default location:

`%Oracle_Home%\FinancialManagement\Server`

Note: Before copying an application, make sure that all users are disconnected from the application that is being copied and that all processing is stopped. Otherwise, the copied application may contain incorrect data.

After you copy an application, you must register the new application with Shared Services and then set up a user with Application Administrator role for the new application. See the *Oracle Hyperion Enterprise Performance Management System User and Role Security Guide*.

► To copy an application:

1 In Windows Explorer, navigate to the **Server** directory of your Financial Management installation, and double-click `HFMCopyApplication.exe`.

2 At the Welcome screen, click **Next**.

3 Click  to locate the source database connection file (UDL) for the application you want to copy and click **Next**.

4 From the list of applications, select the application to copy and click **Next**.

5 Click  to locate the destination database connection file (UDL) and click **Next**.

- 6 From the application drop-down list, select the application name to which you want to copy or enter a new application name and click **Next**.
- 7 On the **Options** panel, select any or all of these actions and click **Next**.
 - **Copy Application Data** to copy data from the source application to the destination application. This generates an identical copy of the source application.
 - **Copy Audit Data** to copy task and data audit data from the source application to the destination application.
 - **Copy Cluster Settings** to copy cluster information from the source application to the destination application.
 - **Overwrite Existing Application (if it exists)** to overwrite an application of the same name as the destination application.
 - Drop All application tables prior to copy
 - Only drop tables that are being copied
 - **Optional:** Click **Advanced Options** for additional options.
 - a. Modify these values as required:
 - Use Client-side Cursor or Use Server-Side Cursor
 - Use SQL Binding
 - Use default thread count or Use custom thread count
 - Log SQL errors
 - Number of task retries
 - b. Click **OK** to close the Advanced Options box.
- 8 Confirm the settings and click **Next**.
- 9 When processing completes, click **OK**.
- 10 To view a task detail either in Completed tasks or Failed tasks (if any), double-click the task, or select the task and click the question mark (?) icon.
- 11 Click **Close**.
- 12 Click **Finish** to exit, or **View Log File** to view the log file and see any errors that occurred during the copy operation.

Note: An improperly sized Oracle instance can cause the utility to fail to copy all table rows. If the error log displays an Oracle issue, or table rows are missing, you may need to increase the size of the REDO logs.

- 13 After you copy an application, register the new application with Oracle's Hyperion® Shared Services and set up a user with administrative rights for the new application.

Copying Performance Management Architect Applications

You can copy Performance Management Architect applications from one application server to another. The copy process migrates Performance Management Architect metadata from the Source to the Target environment and makes it available in the Performance Management Architect Dimension Library.

You can create a Target application or replace a Target application. You can copy a deployed or undeployed application and, based on the Source application status, the Target copied application is created with the same status. After the copy process is complete, you register the application with Shared Services in the new environment.

The copy application utility, `HFMCopyApplication.exe`, is installed by default to the Server directory of your Financial Management installation.

To copy applications, you must have the Application Administrator security role.

The copy process depends on the following possible scenarios, based on the application copy method and the Source and Target environments.

- Copy as Classic application. The Source application may or may not already exist at the Target destination. See [“Copy as Classic Application” on page 42](#).
- Copy as Performance Management Architect application. The Source application exists at the Target destination. See [“Copy as Performance Management Architect When Source Application Exists at Target” on page 44](#).

Copy as Classic Application

► To copy an application:

- 1 If the Source application exists at the Target destination, log in to the target environment and delete the application from within Performance Management Architect.

If the Source application does not exist at the Target destination, proceed to step 2.

- 2 Ensure that the Source and Destination Financial Management servers are not running.
- 3 Navigate to the directory where the Copy Application utility is installed and double-click `HFMCopyApplication.exe`.

Note: The default location is the Server directory of your Financial Management installation.

- 4 At the Welcome screen, click **Next**.

- 5 Click  to locate the Source database connection file for the application that you want to copy and click **Next**.

After successful validation of the Source database connection, the utility displays the Source applications (Classic and Performance Management Architect).

- 6 From the application list, select an application to copy and click **Next**.
- 7 At the prompt “This is an EPM Architect application. Do you want to convert it to a Classic Administration application?”, click **Yes**.
- 8 A warning message is displayed, advising that if the selected application currently exists in the target destination, it must be deleted prior to copying it. Perform an action:
 - If you have not deleted the existing application, click **No** and then delete it.
 - If the application does not exist in the target environment, or you have deleted it, click **Yes** to continue.
- 9 Click  to locate the destination database connection file (UDL) and click **Next**.
- 10 From the application list, select the application to which you want to copy or enter a new application name and click **Next**.
- 11 From **Options**, select options and click **Next**.
 - **Copy Application Data** to copy data from the source application to the destination application. This selection generates an identical copy of the source application.
 - **Copy Audit Data** to copy task and data audit data from the source application to the destination application.
 - **Copy Cluster Settings** to copy cluster information from the source application to the destination application.
 - **Overwrite Existing Application** (if it exists) to overwrite an application of the same name as the destination application.
 - Drop All application tables prior to copy
 - Only drop tables that are being copied
 - **Optional:** Click **Advanced Options** for additional options.
 - a. Modify these values as required:
 - Use Client-side Cursor or Use Server-Side Cursor
 - Use SQL Binding
 - Use default thread count or Use custom thread count
 - Log SQL errors
 - Number of task retries
 - b. Click **OK** to close the Advanced Options box.
- 12 Confirm the settings and click **Next**.
- 13 When the copy process completes, click **OK**.
- 14 To view a task detail in Completed tasks or Failed tasks (if any), double-click the task, or select the task and click the question mark (?) icon.
- 15 Click **Close**.
- 16 Click **Finish** to exit, or **View Log File** to view the log file and see errors that occurred during the copy operation.

Note: An improperly sized Oracle instance can cause the utility to fail to copy all table rows. If the error log displays an Oracle issue, or table rows are missing, you may need to increase the size of the REDO logs.

- 17 Log in to the target Performance Management Architect Application Library.
- 18 Select the newly copied application and select **Application Upgrade** to upgrade the application.

Copy as Performance Management Architect When Source Application Exists at Target

When you copy Performance Management Architect applications, you use Lifecycle Management (LCM) to migrate Financial Management artifacts such as dimensions, security, rules, documents, Web forms, Web grids, journals, rules, and member lists.

To migrate data across Performance Management Architect applications, you must manually move the data. See the Financial Management Data Migration procedure on the Oracle Technology Network (OTN) Web site (<http://www.oracle.com/technology/index.html>).

► To copy an application:

- 1 From the Shared Services console, navigate to the LCM Artifacts List and select **Export to File System**.
- 2 Using the LCM Migration Wizard, select the options and export the file to the file system. The Migration Definition XML with all selected artifacts is extracted to the specified file system.
- 3 Import the Migration Definition file from the file system to the appropriate application, selecting the **Merge or Replace Artifacts** option (applicable only for dimensions).

The Merge option allows any artifacts that exist in the target destination, but not in the source, to be included in the target application.

The Replace option creates the target application as a copy of the source but may affect other applications on the target Performance Management Architect system that use those shared dimensions.

- 4 After the Import process ends successfully, the Target application is updated with the artifacts from the Source application. You can now start the copy application process.
- 5 Ensure that the Source and Destination Financial Management servers are not running.
- 6 Navigate to the directory where the Copy Application utility is installed and double-click `HFMCopyApplication.exe`.

Note: The default location is the Server directory of your Financial Management installation.

- 7 At the Welcome screen, click **Next**.
- 8 Click  to locate the Source database connection file for the application that you want to copy and click **Next**.

After successful validation of the Source database connection, the utility displays the Source applications (Classic and Performance Management Architect).

- 9 From the list of applications, select an application to copy and click **Next**.
- 10 At the prompt “This is an EPM Architect application. Do you want to convert it to a Classic Administration application?”, click **No**.
- 11 A warning message is displayed, advising that prior to copying the application, you must perform an LCM migration of the artifacts related to the application. Perform an action:
 - If you have not performed the LCM migration, click **No** and complete that procedure first.
 - If you have performed the LCM migration, click **Yes** to continue.
- 12 Click  to locate the destination database connection file (UDL) and click **Next**.
- 13 From the application list, select the application to which you want to copy or enter a new application name and click **Next**.
- 14 On the **Options** panel, select options and click **Next**.
 - **Copy Application Data** to copy data from the source application to the destination application. This generates an identical copy of the source application.
 - **Copy Audit Data** to copy task and data audit data from the source application to the destination application.
 - **Copy Cluster Settings** to copy cluster information from the source application to the destination application.
 - **Overwrite Existing Application (if it exists)** to overwrite an application of the same name as the destination application.
 - Drop All application tables prior to copy
 - Only drop tables that are being copied
 - **Optional: Click **Advanced Options** for additional options.**
 - a. Modify these values as required:
 - Use Client-side Cursor or Use Server-Side Cursor
 - Use SQL Binding
 - Use default thread count or Use custom thread count
 - Log SQL errors
 - Number of task retries
 - b. Click **OK** to close the Advanced Options box.
- 15 Confirm the settings and click **Next**.
- 16 When the copy process completes, click **OK**.
- 17 To view a task detail in Completed tasks or Failed tasks (if any), double-click the task or select the task and click the question mark (?) icon.
- 18 Click **Close**.

19 Click **Finish** to exit, or **View Log File** to view the log file and see errors that occurred during the copy operation.

Note: An improperly sized Oracle instance can cause the utility to fail to copy all table rows. If the error log displays an Oracle issue, or table rows are missing, you may need to increase the size of the REDO logs.

20 Log in to the target Performance Management Architect Application Library.

21 The newly updated application has a status of “Out of Sync with deployed application.” Redeploy the application.

If the copy process was successful, the Target application is a copy of the Source application with all relevant artifacts and application data. If you selected the Merge option during the LCM migration, new metadata may be included in the application. If you selected the Replace option, the target application is identical to the source.

Using Sample Applications

Financial Management provides sample application files that you can use to populate a test application.

If you installed the Sample Applications component during the installation process, the files are in the Sample Apps folder in the directory to which you installed Financial Management.

If you did not install the sample files, you can obtain them by reinstalling Financial Management and selecting the **Sample Applications** component.

When you create a test application, you can load files from the Sample Apps directory. The directory includes sample security, metadata, data, rules, and journal files; report definitions, data grids, and Web Data Entry form scripts. Instructions for loading the sample files are included in the Documentation folder of each sample application, for example, \FinancialManagement\SampleApps\Comma\Documentation.

Table 2 Application File Types

Sample File	Contents
Member List (.lst)	Dimension member lists
Metadata (.ads) (.xml for Classic Application Administration)	Metadata
Data (.dat)	Applicable scenarios and years with data
Rules (.rle)	Rules used to run logic on the data in the application
Journals (.jlf)	Sample journal and template file formats
Data Explorer (.hde)	Explore Data grids
System Report (.rpt)	System reports for Explore Data, Journals, or Intercompany Reports

Sample File	Contents
Web Data Grid files (* .xml)	Web data grids
Web Data Entry Forms (.wdf)	Web Data Entry Form scripts
Task List (.xml)	Task List

Working with System Messages

You can retrieve, view, print, and delete Financial Management system messages . You can retrieve system messages by date, server name, and application name. System messages remain in the log until you delete them. The system messages module is available only in the Web user interface.

To view system messages, you must have the Application Administrator security role. Only members of the Administrator group that you specify when you configure the application server can delete system messages. To assign users to the Administrator Group, see the *Oracle Hyperion Enterprise Performance Management System User and Role Security Guide*.

Security is not implemented on the system messages log; therefore, it can be accessed and cleared by any user.

Note: The system messages date format is always displayed as mm/dd/yyyy, regardless of your user preferences.

See these procedures:

- [“Retrieving System Messages” on page 47](#)
- [“Viewing and Printing System Messages” on page 48](#)
- [“Deleting System Messages” on page 48](#)

Retrieving System Messages

► To retrieve system messages:

- 1 **Select *Administration*, then *System Messages*.**
- 2 **Optional:** You can filter messages using these criteria:
 - Start date
 - End date
 - Server name
 - Application name

Note: You can leave the server name and application name blank to view system messages for all servers and applications.

3 Click **View**.

Viewing and Printing System Messages

► To view and print system messages:

1 In the **System Messages** log, select the check box next to a message to view details.

Note: To view detail for a system message, click the relevant cell in the System Message Summary column.

2 Click **Details**.

3 Click **Print** to print the messages displayed.

4 When you finish printing message details, click **Close**.

Deleting System Messages

► To delete system messages:

- To delete individual messages, select the check box next to each message to delete, and click **Delete**.
- To delete all system messages, click **Delete All**.

System Message Detail Strings

Some methods return strings of technical information for system messages such as errors. The strings contain a uniquely identifying error reference number, followed by various fields of information. The fields are delimited by semicolons, and each field has a label that is followed by a colon, as in this example:

```
Error Reference Number: {219EB33B-BF50-11D6-A43E-0000863DCCF1}  
Num: 0x800415c6; Type: 1; DTime: 9/3/02 12:20:10 PM; Svr: GSZAB01; File:  
CHsxServerImpl.cpp; Line: 1842; Ver: 3.0.0.196;
```

The following table describes the system message fields:

Table 3 Fields in System Message Detail Strings

Field	Description
Num	The error number in hexadecimal form
Type	<i>For internal use only</i>

Field	Description
DTime	The error TimeStamp.
Svr	The machine name of the computer on which the error occurred
File	The name of the source code file to which the error applies
Line	The line number in the source code file to which the error applies
Ver	The version number of the DLL to which the error applies

Managing System Users

The Users on System feature enables you to view the users on the system and log off users of an application or server. You can view what module the user is working in and what activities are being performed by the user. See [“Viewing Users” on page 49](#) and [“Logging Off Users” on page 50](#).

Note: Users on System is only available in the Financial Management Web user interface.

To view and log off users by application or server, you need not be logged on to the application. However, to log off individual users, you must be logged on to the application.

Only members of the Administrator group that you specify when you configure the application server can list users and log out users. To assign users to the Administrator Group, see the *Oracle Hyperion Enterprise Performance Management System User and Role Security Guide*.

Viewing Users

You can view logged-on users for all applications and all servers, logged-on users of a specific application on a specific server, logged-on users of a specific application on all servers, and logged-on users of all applications running on a specific server.

This information is available for each logged-on user:

- Username
- Current module in which the user is working
- Current activity of user
- Time the activity was started
- Server name
- Application name

➤ To view logged-on users:

- 1 Select **Administration**, then **Users on System**.

2 Specify the users to view by completing one of these actions:

- To view all logged-on users for all applications and all servers, leave **Server Selection** and **Application Selection** blank.
- To view all logged-on users for all applications on a server, enter the server name in the **Server Selection** text box and leave the **Application Selection** text box blank.
- To view all logged-on users of an application, enter the application name in the **Application Selection** text box and leave the **Server Selection** text box blank.
- To view all logged-on users of an application on a server, enter the server name in the **Server Selection** text box and the application name in the **Application Selection** text box.

3 Click **View Logged-In Users**.

Logging Off Users

To perform a system-wide process, such as a backup and restore, you can log users off an application or an application server. For example, you can log off users logged on to the server and disable future logging on to the server. See [“Disabling and Enabling Connections” on page 51](#). When you log off users, the system does not disconnect them immediately — there can be a five minute potential delay while the processes that the user is executing finish before the user is logged off.

You can control the user session timeout by changing the timeout setting in IIS, or by changing the Web Session Timeout setting using the EPM Configurator. The default timeout is 20 minutes.

When you log a user off the system, the system displays a Stopped status for the user. In addition, the system notifies the user who has been logged off immediately following the first user action after logout.

➤ To log users off an application or server:

1 Select **Administration**, then **Users on System**.

2 Set options:

- Enter the server name from which to log off users.
- Enter the application name from which to log off users.

Note: If you leave these text boxes blank, the system assumes ALL for the text box. For example, if you enter a server name but leave the application text box blank, all users of all applications on the specified server are logged off.

3 Click **Logout Users**.

➤ To log individual users off an application or server:

1 Open the application from which to log off users.

2 Select **Administration**, then **Users on System**.

3 Specify the users to view by completing one of these actions:

- To view all logged-on users for all applications and all servers, leave **Server Selection** and **Application Selection** blank.
- To view all logged-on users of all application on a server, enter the server name in the **Server Selection** text box and leave the **Application Selection** text box blank.
- To view all logged-on users of an application, enter the application name in the **Application Selection** text box and leave the **Server Selection** text box blank.
- To view all logged-on users of an application on a server, enter the server name in the **Server Selection** text box and the application name in the **Application Selection** text box.

4 Select View Logged-In Users.

5 In the Users on System list, click in the Logout column to log an individual user off the application.

6 Click OK to confirm.

Managing Servers and Applications

You can use the Manage Servers and Applications feature to disable user connections to an application or server and to log off users.

See [“Disabling and Enabling Connections” on page 51](#) and [“Logging Off Users” on page 50](#).

Only members of the Administrator group that you specify when you configure the application server can view disabled components, enable or disable connections, or log out users. To assign users to the Administrator Group, see the *Oracle Hyperion Enterprise Performance Management System User and Role Security Guide*.

Note: The Manage Servers and Applications feature is available only in the Financial Management Web user interface.

Disabling and Enabling Connections

When you disable connections, the system prevents new users from logging on to the specified server or application. You can use the disable connections feature with the log off user feature. For example, you can disable logging on to an application, log off users logged on to the application, load metadata, and then enable connections to the application. See [“Logging Off Users” on page 50](#).

Note: Administrators can perform all administrative tasks, such as loading metadata and deleting applications, when connections are disabled.

► To disable or enable user connections to an application or server:

1 Select Administration, then Manage Servers and Applications.

2 You can perform these actions:

- To disable or enable user connections for all applications and all servers, leave **Server Selection** and **Application Selection** blank.

Note: This also disables connections for users in the Administrator group.

- To disable or enable user connections for all applications on a server, enter the server name in the **Server Selection** text box and leave the **Application Selection** text box blank.

Note: This also disables connections for users in the Administrator group.

- To disable or enable user connections for an application, enter the application name in the **Application Selection** text box and leave the **Server Selection** text box blank.
- To disable or enable user connections for an application on a server, enter the server name in the **Server Selection** text box and the application name in the **Application Selection** text box.

3 Take one of these actions:

- Select **Disable Connections**, and click **OK** to confirm.
- Select **Enable Connections**, and click **OK** to confirm.

Note: You can also enable connections by clicking  in the Enable column of the Disabled Components list.

Viewing Disabled Components

From the Manage Servers and Applications window, you can view a list of disabled components and enable them.

► To view disabled components:

- 1 Select **Administration**, then **Manage Servers and Applications**.
- 2 Click **View Disabled Components**.
- 3 To enable a component, from the Disabled Components list, select a component and click  in the **Enable** column.

Synchronizing Servers

The synchronization between Financial Management application servers is based on system time. Changing the clock can affect this synchronization. For the time change to and from Daylight Savings Time, Oracle recommends that you stop the servers before the time change and restart them afterward.

Auditing Tasks

You can use the Task Audit feature to view the tasks performed by users. You can filter audited tasks by date range, application server, user, and task performed.

Note: Only users assigned the Administrator role can perform a task audit.

These user activities are logged in the task audit:

- Idle
- Rules Load
- Rules Scan
- Rules Extract
- Consolidation
- Chart Logic
- Translation
- Custom Logic
- Allocate
- Data Load
- Data Extract
- Data Extract via HAL
- Data Entry
- Data Retrieval
- Data Clear
- Data Copy
- Journal Entry
- Journal Retrieval
- Journal Posting
- Journal Unposting
- Journal Template Entry
- Metadata Load
- Metadata Extract
- Member List Load
- Member List Scan
- Member List Extract
- Security Load
- Security Scan

- Security Extract
- Logon
- Logon Failure
- Logoff
- External
- Metadata Scan
- Data Scan
- Extended Analytics Export
- Extended Analytics Schema Delete
- Transactions Load
- Transactions Extract
- Document Attachments
- Document Detachments
- Create Transactions
- Edit Transactions
- Delete Transactions
- Post Transactions
- Unpost Transactions
- Delete Invalid Records
- Data Audit Purged
- Task Audit Purged
- Post All Transactions
- Unpost All Transactions
- Delete All Transactions
- Unmatch All Transactions
- AutoMatch by ID
- AutoMatch by Account
- IC Matching Report by ID
- IC Matching Report by Acct
- IC Transaction Report

The task audit log includes this information:

- Username
- Activity performed
- Activity start time

- Activity end time
- Server name
- Description
- Current module

Note: The Task Audit feature is available only in the Financial Management Web user interface.

The task audit log information is stored in a table (*APPNAME_TASK_AUDIT*). You can back up or extract the information in the table to a file, then clear the table. You should monitor the log size and clear it on a regular basis.

► To perform a task audit:

1 Select Administration, then Task Audit.

Note: You must have an application open for the Task Audit link to be available.

2 Optional: You can filter the audit by selecting criteria:

- Enter a start date and an end date.

Tip: Click  to select dates from a pop-up calendar.

- From the **Server Selection** list, select a server, or select **All**.
- From the **User Selection** list, select the user to audit, or select **All** to audit all users.
- From the **Task Filter** list, select the task to audit, or select **All** to audit all tasks.

3 Click View.

The tasks that meet the filtering criteria are displayed. Tasks are sorted from most recent to least recent, as shown in this example.

4 Optional: To export the audit information to a CSV file, click Export, and follow the download instructions.

5 Optional: To delete the entries from the log, click Clear Log to delete the entries from the log.

Note: When you clear the log, a record of the clear remains in the log and cannot be erased.

Auditing Data

You can use the Data Audit feature to view data changes performed by users. You can filter the data changes by date range, application server, user, and the dimension members of cells.

You can audit Scenario and Account members by setting the EnableDataAudit metadata attribute to Y for the scenarios and accounts to audit. To turn off auditing of Scenario and Account members, change the EnableDataAudit attribute to N.

Note: Only users assigned to the Administrator role can perform a data audit.

These user activities are logged in the data audit:

- Data Entry
- Data Clear
- Data Copy
- Data Load
- Journal Entry

The data audit log includes this information:

- Username
- Activity performed
- Time modified
- Server name
- Point of view
- New value for point of view

Note: The Data Audit feature is available only in the Financial Management Web user interface.

The data audit log information is stored in a table (*APPNAME_DATA_AUDIT*). You can back up or extract the information in the table. You should monitor the log size and clear it on a regular basis.

➤ To perform a data audit:

1 Select Administration, then Data Audit.

You must have an application open for the Data Audit link to be available.

2 Optional: Set filter options:

- Enter a **start date** and an **end date** to filter by date range.

Tip: Click  to select dates from a pop-up calendar.

- From the **Server Selection** list, select the server to audit, or select **All** to audit all servers.
- From the **User Selection** list, select the user to audit, or select **All** to audit all users.
- Enter member names in the text boxes for the applicable dimensions to filter by dimension members.

Note: The system filters only by the dimensions for which you specify members. For example, if you specify a member in the Period text box and leave the other text boxes blank, the system filters only by the Period dimension.

3 Click View.

The data changes that meet the filtering criteria are displayed.

4 Optional: In the **Point of View** column, click a link to view the history for the corresponding cell.

5 Optional: Click **Export** to export the audit information to a CSV file.

6 Optional: Click **Clear Log** to delete all entries from the log.

Note: When you clear the log, a record of the clear remains in the task audit log and cannot be erased.

Monitoring Running Tasks

You can use the Running Tasks feature to view and terminate running tasks. You can filter running tasks by application server, user, task performed, and task status.

Note: Only the user who starts a task or a user assigned to the Administrator role can terminate a task that is running.

You can view and terminate these tasks:

- Consolidation
- Data Load
- Extended Analytics Export
- Post All Transactions
- Unpost All Transactions
- Delete All Transactions
- UnMatch All Transactions
- AutoMatch by ID
- AutoMatch by Account
- IC Matching Report by ID
- IC Matching Report by Acct
- IC Transaction Report

You can filter the running tasks by these task status types:

- Initializing
- Running

- Paused
- Uninitializing
- Stopped
- Aborted
- Completed
- Not Responding
- Scheduled Stop
- Scheduled Start

➤ To monitor and stop running tasks:

1 Select Administration, then Running Tasks.

You must have an application open for the Running Tasks link to be available.

2 Optional: For Server Selection, enter the server name to filter by application server.

If you leave the Server Selection box blank, tasks running on all servers are listed.

3 Optional: Set filter options:

- From the **User Selection** list, select the user to monitor, or select **All** to view running tasks for all users.
- From the **Task Filter** list, select the task to monitor, or select **All** to view all running tasks.
- From the **Status Filter** list, select the status type to monitor, or select **All** to monitor all status types.

4 Click View.

The tasks that meet the filtering criteria are displayed.

5 In the Stop Task column, click next to the task to stop.

Note: Click  in the Log column of the list to view the log file for a completed task.

By default, running tasks remain in the database for 900 seconds (15 minutes). You can change the default by modifying the `AutoClearDeadTasksAfterSeconds` registry key in the `HKEY_LOCAL_MACHINE\SOFTWARE\Hyperion Solutions\Hyperion Financial Management\Server\Running Tasks` registry location. The minimum value for the registry key is 720 seconds (12 minutes); the maximum value is 864,000 seconds (10 days).

Scanning For and Deleting Invalid Records

You can use the Delete Invalid Records feature to scan an application for invalid records and to remove them. Administrator access is required to use this feature.

Running the Delete Invalid Records process impacts the database, network, and Financial Management environment and can cause performance issues. This process must be run in a maintenance window where users are not accessing the Financial Management environment.

Windows Procedure

- To scan for and delete invalid records:
 - 1 Open the application from which to delete invalid records.
 - 2 From the navigation frame, select **Database Management**.
 - 3 Click the **Delete Invalid Records** tab.
 - 4 Browse to or enter the log file name and path.
 - 5 Perform a task:
 - Click Scan.
 - Click Delete.
 - 6 **Optional:** To view the log file, click **View**.

Web Procedure

- To scan for and delete invalid records:
 - 1 Open the application from which to delete invalid records.
 - 2 In **Browser View**, expand **Tasks**, and select **Data Tasks**.
 - 3 Select **Database Management**.
 - 4 Click **Delete Invalid Records**.

Note: This tab is available only to users with Administrator access.

 - 5 Perform a task:
 - Click Scan Only.
 - Click Delete Invalid Records.

Changing Settings for Multiple Server Tasks

There are several registry settings that control the amount of consolidations, data loads, and Extended Analytics exports that can run at any one time in a multi-server environment. If you need to run several concurrent consolidations, for example, you can change the values in these registry settings.

Registry Setting	Description
MaxNumConcurrentConsolidations	Controls the number of concurrent consolidations allowed per application server. Any consolidations executed above the value are queued as Scheduled Consolidations. Minimum value is 1; maximum value is 8.
NumConsolidationThreads	Controls the multi-threading of consolidations per application server. Lowering the value limits the system's utilization of system resources, resulting in slower consolidation performance. Minimum value is 1; maximum value is 8.
NumConsolidationsAllowed	Controls the number of consolidations allowed per application across all the application servers. Default value is 8 and the range is 1-20.
NumDataLoadsAllowed	Controls the number of data loads allowed per application across all the application servers. Default value is 8 and the range is 1-20.
NumEAExportsAllowed	Controls the number of Extended Analytic exports allowed per application across all the application servers. Default value is 8 and the range is 1-20.

► To change a server registry setting:

- 1 **Select **Start**, then **Run**.**
- 2 **In the **Open** box, type **regedit32**, and click **OK**.**
- 3 **Select `HKEY_LOCAL_MACHINE\SOFTWARE\Hyperion Solutions\Hyperion Financial Management\Server\RunningTasks`.**
- 4 **Double-click the registry setting, change the value, and click **OK**.**

Maintaining Server Logon Information

When you use application server clusters, by default, users are kept on the same application server for subsequent logons. For example, a user logged on to Financial Management who runs reports with Financial Reporting will always use the same application server, instead of potentially logging on to different application servers.

You can create the UseStickyServer cluster level registry setting on the client to disable Sticky Server support. You must create the registry setting for each Web server and Win32 client.

► To disable UseStickyServer support:

- 1 **Navigate to `HKEY_LOCAL_MACHINE\SOFTWARE\Hyperion Solutions\Hyperion Financial Management\Client\Clusters\machinename`.**
- 2 **Create a `REG_DWORD` value named `UseStickyServer`.**
- 3 **Enter the value `"0"`.**

Note: To enable it, change the value to 1, or delete the key. By default, a "1" is assumed if the key is missing.

3

Managing Application Security

In This Chapter

Application Security Considerations	62
Launching the Shared Services Console from Financial Management.....	62
Loading Application Security.....	68
Extracting Application Security.....	73

Security and access rights enable you to control access to Financial Management applications and application elements. Setting up security enables you to protect data and prevent unauthorized users from changing data. For example, you can restrict access to certain data elements or forms within an application.

Security exists at two levels:

- Authentication by an external provider
- Financial Management security, in which users and groups are assigned to applications and application elements are assigned to security classes

There are two ways to set up security for Financial Management applications:

- Load a security file into an application. See [“Loading Application Security” on page 68](#).
- Use the Oracle's Hyperion® Shared Services Console to set up security information. See [“Launching the Shared Services Console from Financial Management” on page 62](#).

These security roles are required for application administration. For information on assigning roles, see the *Oracle Hyperion Enterprise Performance Management System User and Role Security Guide*.

Security Role	Description
Dimension Editor	Creates and manages import profiles for dimension creation, as well as creating and managing dimensions manually within the Performance Management Architect user interface or the Classic Application Administration option. Required to access Classic Application Administration options for Financial Management and Planning using Web navigation.

Security Role	Description
Application Creator/Financial Management Application Creator	Creates and deploys Performance Management Architect applications. Users with this role can create applications, but can change only the dimensions to which they have access permissions. Required in addition to the Dimension Editor role for Financial Management and Oracle Hyperion Planning, Fusion Edition users to be able to navigate to their product's Classic Application Administration options. When a user with Application Creator role deploys an application from Performance Management Architect, that user automatically becomes the application administrator and provisioning manager for that application. The Application Creator can create all applications. The Financial Management Application Creator can create Consolidation applications and Generic applications. To create applications, the user must also be a member of the Application Creators group specified in the Configuration Utility.

Application Security Considerations

Financial Management security offers flexibility in securing application elements and tasks. Because security classes are assigned to application elements as they are created, you should design your security system before you set up your applications.

After you design a security system for one application, you can extract the security elements for backup or loading into another application. See [“Loading Application Security” on page 68](#) and [“Extracting Application Security” on page 73](#).

Before setting up security in Financial Management, you should consider these questions:

- How do you want to group and classify Financial Management tasks and application elements?
- How do you want to group users?
- What level of access right should be assigned for your users and groups?
- What security classes do you want to assign to application elements as they are created?

Launching the Shared Services Console from Financial Management

Before you can set up security for Financial Management applications, you must do these tasks:

1. Create Financial Management applications. For Performance Management Architect applications, see the *Oracle Hyperion Enterprise Performance Management Architect Administrator's Guide*, or for Classic Financial Management applications, see [“Creating Applications” on page 32](#).
2. Provision users by assigning users and groups to applications and assigning user roles. See the *Oracle Hyperion Enterprise Performance Management System User and Role Security Guide*.

You can then use the Shared Services Console to set up security for Financial Management applications. In the console, you can do these application tasks:

- Assign users and groups
 - Assign user permissions to security classes
 - Run security reports
- To launch the Shared Services Console from Financial Management, select **Administration**, then **Shared Services Console**.

Assigning Users and Groups to Financial Management Applications

Only a user assigned to the Provisioning Manager role can define users and groups. See the *Oracle Hyperion Enterprise Performance Management System User and Role Security Guide*.

- To select users and groups for an application:
- 1 From the Shared Services Console, expand **Application Groups**, right-click the application name, select **Assign Access Control**, and then **Select Users and Groups**.
 - 2 Select an option:
 - **Show All** to show all users that are provisioned
 - **Users or Groups**, and in Search Criteria, enter search criteria, and click **Search**.
 - 3 From **Available Users and Groups**, select users and groups to assign to the application and select roles, and use the arrow keys to move them to the **Selected Users** column.
 - 4 Click **Next**.

Setting Up Security Classes for Financial Management Applications

Security classes determine the access that users have to application elements.

Caution! The information in this section is provided for use with Classic Financial Management applications only. For information on setting up security using Performance Management Architect, see the *Oracle Hyperion Enterprise Performance Management Architect Administrator's Guide*.

In the Select Classes module, you can perform these procedures:

- [“Creating Security Classes” on page 64](#)
- [“Deleting Security Classes” on page 64](#)
- [“Selecting Security Classes” on page 64](#)

Note: Only users assigned to the Provisioning Manager role can define security classes for applications.

After you define security classes for an application, you can assign the security classes to application elements such as accounts and entities.

A user's or group's ability to access application elements depends on the security classes to which the user or group belongs and on the security class associated with the application elements.

Creating Security Classes

► To create security classes:

- 1 From the Shared Services Console, expand **Application Groups**, right-click the application name, select **Assign Access Control**, and then **Select Classes**.
- 2 For **Class Name**, enter a name for the security class.

Note: The name can contain up to 80 characters.

- 3 Click **Add**.

Deleting Security Classes

Before you delete a security class from an application, you must disassociate it from the application elements to which it is assigned.

You can disassociate an entity, account, or scenario from a security class by modifying the security class in the metadata file. You can disassociate a journal from a security class by modifying the journal file or by updating the security class for the journal in the Process Journals module.

► To delete security classes:

- 1 From the Shared Services Console, expand **Application Groups**, right-click the application name, select **Assign Access Control**, and then **Select Classes**.
- 2 From **Available Classes**, select the security classes to delete.
- 3 Click **Delete Classes**.
- 4 Click **Yes** to confirm deletion.

Selecting Security Classes

► To select security classes for an application:

- 1 From the Shared Services Console, expand **Application Groups**, right-click the application name, select **Assign Access Control**, and then **Select Classes**.

- 2 From **Available Classes**, select the security classes to assign to the application, and click **Add Single User or Group**.
- 3 Click **Next** or **Assign Access**.

Assigning User Access to Security Classes

After you define users and groups and security classes, you can specify the level of access each user and group has to each security class in the application and set up e-mail alerts.

Table 4 User Access Level

Access Level	Description
None	No access to elements assigned to the security class.
Metadata	View a specified member in a list but cannot view or modify data for the member.
Read	View data for elements assigned to the security class but cannot promote or reject.
Promote	View data for elements assigned to the security class and can promote or reject.
All	Modify data for elements assigned to the security class and can promote and reject.

You can use the Pivot Table feature to toggle between two views for assigning access. For example, if you have users and groups on rows and security classes on columns and click Pivot Table, users and groups will be on columns and security classes on rows.

Note: A user assigned to the Application Administrator role for an application has access to all information in the application.

➤ To assign user access to security classes:

- 1 From the **Shared Services Console**, expand **Application Groups**, right-click the application name, select **Assign Access Control**, and then **Assign Access**.
- 2 Select cells for which to assign access rights.

Tip: Use the Shift and Ctrl keys to select multiple cells. Select a column or row by clicking in the column or row header.

- 3 From **Access Rights**, select the access level to assign.

Note: See [Table 4, “User Access Level,”](#) on page 65.

- 4 Click  to apply the level to the selected cells.
- 5 **Optional:** To add an e-mail alert, select cells in the table and click **Add Alert**.

Caution! The alerting process uses the e-mail addresses stored in the authentication files, such as MSAD, LDAP, or Native Directory. See [“Setting Up E-mail Alerting” on page 66](#).

Note: To remove e-mail alerts, select the cell and click Remove Alert.

- 6 Click **Save**.
- 7 Click **Next** or **Security Report**.

Setting Up E-mail Alerting

You can use e-mail alerting for intercompany transactions and during the process management review process. E-mail alerts help highlight a key event or data change in the system. For example, you can send an e-mail alert that an intercompany transaction is mismatched and needs to be matched, or that a process unit is ready for the next promotion level.

Note: The alerting process uses the e-mail addresses that are stored in the authentication files, such as LDAP, MSAD, or Native Directory.

Process Management Alerting

► To set up process management e-mail alerts:

- 1 For the scenario in the process unit, set the Enable Process Management metadata attribute to “A” to allow alerts.
- 2 Assign the user to the Receive E-mail Alerts for Process Management role.

Note: For information on assigning users to roles, see the *Oracle Hyperion Enterprise Performance Management System User and Role Security Guide*.

- 3 Assign the user to Process Management notifiable roles as defined in [Table 5, “Process Management User Roles and Alert Notification”](#), on page 66.
- 4 Assign the user ALL or PROMOTE access to the security classes assigned to the scenario and entity in the process unit and add an alert for each security class.

Users who meet all criteria receive e-mail alerts.

Table 5 Process Management User Roles and Alert Notification

Process Unit Level Before or After Action	Process Management User Roles Notified
First Pass	Users with ALL or PROMOTE access to the entity are notified.
Review Level 1	Reviewer 1 and Submitter roles are notified.
Review Level 2	Reviewer 2 and Submitter roles are notified.

Process Unit Level Before or After Action	Process Management User Roles Notified
Review Level 3	Reviewer 3 and Submitter roles are notified.
Review Level 4	Reviewer 4 and Submitter roles are notified.
Review Level 5	Reviewer 5 and Submitter roles are notified.
Review Level 6	Reviewer 6 and Submitter roles are notified.
Review Level 7	Reviewer 7 and Submitter roles are notified.
Review Level 8	Reviewer 8 and Submitter roles are notified.
Review Level 9	Reviewer 9 and Submitter roles are notified.
Review Level 10	Reviewer 10 and Submitter roles are notified.
Submitted	Review Supervisor role is notified. Only users with this role can approve the submitted process unit.
Approved	Reviewer 1 to Reviewer 10 and Submitter roles are notified.
Published	Users with ALL, READ, or PROMOTE access to the entity are notified.

Note: E-mail alerts are not generated when the process unit is at the Not Started level or for the Sign Off action.

Users with the Application Administrator role do not receive e-mail alerts. For a user with the Application Administrator role to receive e-mail alerts, set up as a separate user and assign the role to receive alerts. The user that performed the action to the process unit is also notified with an e-mail confirmation log stating to whom e-mails were sent. For information on generating e-mail alerts in process management, see the *Oracle Hyperion Financial Management User's Guide*.

Intercompany Transaction Alerting

► To set up intercompany transaction e-mail alerts:

- 1 Assign the user to the Receive E-mail Alerts for IC Transactions role.

Note: For information on assigning users to roles, see the *Oracle Hyperion Enterprise Performance Management System User and Role Security Guide*.

- 2 Assign the user to the Inter-Company Transaction Admin or Inter-Company Transaction User role.
- 3 Assign the user ALL, READ, or PROMOTE access to the security classes that are assigned to the scenario and entity in the transaction and add an alert for each security class. See [“Assigning User Access to Security Classes” on page 65](#).

Users who meet all criteria receive e-mail alerts from the Intercompany Transactions or Intercompany Partner Matching Report modules.

For information on generating e-mail alerts in intercompany transactions, see the *Oracle Hyperion Financial Management User's Guide*.

Running Security Reports for Financial Management Applications

You can run security reports on the information that you selected while setting up security for the application. You can run reports for classes by user, roles by user, classes and roles by user, and users by group. You can view the report online or you can export it to a CSV file.

► To create a security report:

- 1 From the Oracle's Hyperion® Shared Services Console, expand **Application Groups**, right-click the application name, select **Assign Access Control**, and then **Security Reports**.
- 2 Select a report option:
 - **Rights and select options:**
 - Classes by User
 - Roles by User
 - **Users by Group**
- 3 Select an option:
 - **Launch Report** to open the report in a new window
 - **Export to File** to save the report as a CSV file.

Loading Application Security

Caution! You can only load security classes for Classic Financial Management applications.

You must load application security before you can load other information to an application.

To remove a role from a user or group, you must modify the role in the Shared Services Console. See the *Oracle Hyperion Enterprise Performance Management System User and Role Security Guide*.

Windows Procedure

► To load application security:

- 1 Open the application.
- 2 From the navigation frame, select **Load Security**.

- 3 For **Security Filename**, enter the file name to load, or click  to find the file.

Note: You can click View to display the contents of the application security file that you specified. By default, security information files use the SEC file extension. The load process accepts other file extensions such as TXT or CSV, however, Oracle recommends that you use the SEC file extension.

- 4 For **Log Filename**, enter a log file name, or click  to find the file.

- 5 For **Delimiter Character**, enter the character used to separate information in the file.

These characters are valid:

, ~ @ \$ % ^ & | : ; ? \

Note: You must use a character that is not used in the file name or in any other way in the file. For example, if you use the comma in an entity description, you cannot use the comma as the delimiter.

- 6 **Optional:** Select **Clear All Security Info Before Loading** to clear security information for the application before loading new security information.

Caution! You can use the Clear All option only if you have been assigned the Application Administrator and Provisioning Manager roles. Also, if you use this option, you will have to reprovision users, as all users (including the user doing the clear) will be removed in this process. For information on provisioning users, see the *Oracle Hyperion Enterprise Performance Management System User and Role Security Guide*. Before selecting the clear option, review [“Clearing and Loading Security Information” on page 70](#).

- 7 Select the types of security information to load.

Tip: Use the Select All and De-Select All buttons to quickly select or de-select the security types.

- 8 Click **Load**.

Note: After you load the security file, you can click the View button next to Log Filename to display the log file.

Web Procedure

Note: Oracle recommends that you add Financial Management to the exceptions for your Web pop-up blocker. When you perform some Financial Management tasks, such as loading data on the Web, a status window pops up showing the task status. If you have a pop-up blocker enabled on your computer, the status window is not displayed.

► To load application security:

- 1 Open the application.
- 2 In **Browser View**, expand **Tasks**, and select **Load Tasks**.
- 3 Select **Load Security**.
- 4 For **Security File**, enter the file name to load, or click **Browse** to find the file.

Note: By default, application security information files use the SEC file extension. The load process accepts other file extensions such as TXT or CSV, however, Oracle recommends that you use the SEC file extension.

- 5 For **Delimiter Character**, enter the character used to separate information in the file. These characters are valid:

, ~ @ \$ % ^ & | : ; ? \

Note: You must use a character that is not used in the file name or in any other way in the file. For example, if you use the comma in an entity description, you cannot use the comma as the delimiter.

- 6 **Optional:** Select **Clear All Security Info Before Loading** to clear security information for the application before loading the new security information.

Caution! You can use the Clear All option only if you have been assigned the Application Administrator and Provisioning Manager roles. Also, if you use this option, you will have to reprovision users, as all users (including the user doing the clear) will be removed in this process. For information on provisioning users, see the *Oracle Hyperion Enterprise Performance Management System User and Role Security Guide*. Before selecting the clear option, review [“Clearing and Loading Security Information” on page 70](#).

- 7 From **Security Options**, select the types of security information to load.
- 8 Click **Load**.

Clearing and Loading Security Information

You can clear security information from an application and then load new security information. For example, if you plan to change security class Class1 to Class2 during the load of security, you need to make the change to all application elements that reference the security class Class1.

However, because the system generates new security references for application elements that use security class information, you must perform prerequisite steps before you load the new security information, and you must perform follow-up steps after you load the new security information.

► To clear security information and load a new security file:

- 1 Extract application elements from the application. See [“Before Clearing Security Information” on page 71](#).

- 2 Select to clear existing security information and load a new security file.
- 3 Load application elements to the application. See [“After Clearing Security Information” on page 72](#).

Note: You must be assigned to the Application Administrator security role to be able to perform these procedures.

Before Clearing Security Information

Before you clear security information and load a security file, you must perform these tasks for the specified application elements that utilize security class information.

Metadata

- To update metadata before clearing and loading security information:
 - 1 Extract all application metadata elements in the application.
 - 2 Make changes to the security class information of the metadata elements as necessary.

Journals

- To update journals before clearing and loading security information:
 - 1 Unpost posted journals in the application.
 - 2 Reject approved journals so that the journal status reverts to Working.
 - 3 Extract all journals.
 - 4 Make changes to the journal security class information as necessary.

Grids

- To update grids before clearing and loading security information:
 - 1 Extract all grids that have a security class assigned.
 - 2 Make changes to the grid security class information as necessary.

Data Forms

- To update data forms before clearing and loading security information:
 - 1 Extract all data forms that have a security class assigned.
 - 2 Make changes to the data forms security class information as necessary.

After Clearing Security Information

After you clear security information and load a security file, you must perform these tasks for the specified application elements that utilize security class information.

Metadata

- To update metadata after clearing and loading security information, load the updated metadata file to the application.

Journals

- To update journals after clearing and loading security information:
 - 1 Load the updated journal file.
 - 2 Post journals that you unposted before clearing and loading security information.
 - 3 Approve journals that you rejected before clearing and loading security information.

Grids

- To update grids after clearing and loading security information:
 - 1 Load the updated grid files.
 - 2 Select the option to overwrite existing documents.

Data Forms

- To update data forms after clearing and loading security information:
 - 1 Load the updated data form file.
 - 2 Select the option to overwrite existing documents.

Folders

- To update folders after clearing and loading security information:
 - 1 Delete folders that may have an incorrect security class assigned.
 - 2 Add new folders back to the application.

Reports

► To update reports after clearing and loading security information:

- 1 Reload all reports with an assigned security class.
- 2 Provide the new security class assignment if applicable.

Task Lists

► To update task lists after clearing and loading security information:

- 1 Reload all task lists with an assigned security class.
- 2 Provide the new security class assignment if applicable.

Extracting Application Security

When you extract application security from an application, save the file in a format that supports multibyte character sets (MBCS). By default, application security files use the SEC file extension. You can extract application security to view or modify it in a text editor.

You can extract these types of security information:

- Users and groups
- Security classes
- Role access
- Security class access

Note: Oracle recommends that you periodically extract security to a backup file. For information on backing up security information, see the *Oracle Hyperion Enterprise Performance Management System Installation and Configuration Guide*.

Windows Procedure

► To extract application security:

- 1 Open the application.
- 2 From the navigation frame, select **Extract Security**.
- 3 For **Security Filename**, enter a file name to extract, or click  to find the file.

Note: You can click the View button next to the security filename to display the contents of the application security file that you specified. By default, application security information files use the SEC file extension.

- 4 For **Log Filename**, enter a log file name, or click  to find the file.

5 For Delimiter Character, enter the character used to separate information in the file.

These characters are valid:

, ~ @ \$ % ^ & | : ; ? \

Note: You must use a character that is not used in the file name or in any other way in the file. For example, if you use the comma in an entity description, you cannot use the comma as the delimiter.

6 Select the types of security to extract.

Tip: Click Select All and De-select All to quickly select or de-select all security types.

7 Click Extract.

Note: After you click Extract, you can click the View button next to Log Filename to display the log file.

Web Procedure

► To extract application security:

1 Open the application.

2 In Browser View, expand Tasks, and select Extract Tasks.

3 Select Extract Security.

4 Select the types of security to extract.

5 For Delimiter Character, enter the character used to separate information in the file.

These characters are valid:

, ~ @ \$ % ^ & | : ; ? \

Note: You must use a character that is not used in the file name or in any other way in the file. For example, if you use the comma in an entity description, you cannot use the comma as the delimiter.

6 Click Extract.

7 Follow the download instructions displayed in the browser.

The instructions vary depending on the Web browser that you are using. Make sure to save the file in the Web directory that you set up.

4

Managing Metadata

In This Chapter

Defining Accounts.....	76
Defining Custom Members	80
Defining Entity Members	81
Defining Scenario Members.....	82
Defining Application Settings	84
Organization by Period.....	86
Defining Consolidation Methods.....	87
Defining Currencies	91
System-Generated Accounts.....	92
Setting Up Intercompany Partners.....	94
Metadata Filtering Based on Security.....	96
Creating Metadata Files of the APP Format	96
Using Metadata Manager Views	105
Metadata Referential Integrity	114
Loading Metadata.....	115
Extracting Metadata	119

Caution! The information in this chapter is provided for use with Classic Financial Management applications only. For information on managing metadata for applications created using Performance Management Architect, see the *Oracle Hyperion Enterprise Performance Management Architect Administrator's Guide*.

You can define metadata for Classic Financial Management applications in two ways:

- Create an XML or APP metadata file and load it to an application.
- Use Metadata Manager to create an XML or APP metadata file and load it to an application.

Note: You must set up security for an application before you can load metadata. See [Chapter 3, “Managing Application Security.”](#)

Sample metadata files are included when you install Sample Applications for Financial Management. The files are located in the Sample Apps folder in the directory to which you installed Financial Management.

Defining Accounts

You define accounts with the attributes described in [Table 6](#).

Table 6 Account Member Attributes

Attribute	Description
AccountType	(Required) One of these values: ● ASSET—Store values that represent the assets of a company ● LIABILITY—Store point-in-time balances that represent the liabilities of a company ● REVENUE—Store periodic and year-to-date values that increase net worth if the value is positive Note: In Financial Management releases prior to 4.1, this account type was called Income. ● EXPENSE—Store periodic and year-to-date values that decrease net worth if the value is positive ● FLOW—Store periodic and year-to-date values ● BALANCE—Store unsigned values that relate to a particular point in time ● BALANCERECURRING—Store unsigned values that relate to a particular point in time and that re-occur in future periods ● CURRENCYRATE—Store currency rate information ● GROUPLABEL—Use the account for grouping purposes ● DYNAMIC—Indicates that the account value is calculated dynamically from the data that you are viewing See “Account Type Behavior” on page 78.
CalcAttribute	Description of the calculations in the rules file that are done for this account This information is displayed as part of cell info in data forms and data grids. It can contain up to 80 characters including spaces.
Custom1TopMember, Custom2TopMember, Custom3TopMember, Custom4TopMember	Which top member in the hierarchy of a Custom dimension is valid for the account Only the specified member, including all descendants, is valid for the account.
DefaultParent	The default parent for the account
Description	The account description The description can contain up to 80 characters, including spaces, and cannot use an ampersand (&) or slash (\).
EnableCustom1Aggr, EnableCustom2Aggr, EnableCustom3Aggr, EnableCustom4Aggr	Whether aggregation is enabled for intersections of the Account and Custom dimensions This attribute is used for special totals, not summing. Specify Y if the account can aggregate with Custom dimensions or N if it cannot.
EnableDataAudit	Whether the account can be audited Specify Y to enable account auditing or N to disable auditing. The default is N. This attribute, when applied to an account or scenario, determines what can be audited.
ICPTopMember	The ICP top member for the account The specified member, including all descendants, is valid for the account.

Attribute	Description
IsCalculated	Whether the account is calculated Only base-level accounts can be calculated. If a base-level account is calculated, you cannot manually enter values. Specify Y if the account is to be calculated; otherwise, specify N.
IsConsolidated	Whether the account is consolidated in the entity hierarchy. If the account is not consolidated, it is ignored during consolidation. Specify Y if the account is to be consolidated and N if the account is not to be consolidated.
IsICP	Specifies whether the account is an intercompany account. If the account is an intercompany account, you must also specify a plug account. Specify one of these values: ● Y if ICP transactions, including self-ICP transactions, are enabled for the account ● N if ICP transactions are not enabled for the account ● R if ICP transactions are enabled for the account, but the account cannot have ICP transactions with itself
Member	Specifies the name for the account. This attribute is required. The name must be unique. It can contain up to 80 characters, including spaces, but cannot start with a space. Note: Account names cannot contain spaces if you are using an Oracle database. Do not use these characters in an account name: ● Period (.) ● Plus sign (+) ● Minus sign (-) ● Asterisk (*) ● Slash mark (/) ● Number sign (#) ● Comma (,) ● Semicolon (;) ● At sign (@) ● Double quote (") ● Curly brackets ({ })
NumDecimalPlaces	Specifies the number of digits to be displayed to the right of the decimal point for account values. This attribute is required. Specify a value from 0 to 9 for this attribute.
PlugAcct	Specifies the account name used for identifying discrepancies in intercompany transactions. The PlugAcct attribute is required when the IsICP attribute for the account is selected.
SecurityClass	Specifies the security class that defines the users who can access the account data. Security class names can contain up to 80 characters. Security access applies only to account data.
Submission Group	Specifies the submission group. The value can be a number from 0 to 99. The default is blank. A blank value defaults to the value of 1. If you set the submission group to zero (0), the account is not included in the review process.
UserDefined1, UserDefined2, UserDefined3	Stores custom information for the account. You can enter a maximum of 80 characters. The UserDefined1, UserDefined2, and UserDefined3 functions retrieve the text stored in this attribute.

Attribute	Description
UsesLineItems	Specifies whether an account can have line items. Specify Y if the account uses line items and N if the account does not use line items. Caution! If you change this attribute after line-item detail is entered, the stored line-item detail may no longer be valid for the account. These behaviors occur: ● If the account accepted line items and now it cannot, the line-item detail stored in the database is no longer valid. Only the total is displayed. ● If the account did not accept line items and now it can, there is a total amount but no corresponding line-item detail information for the account. You can extract the total and then load it as line-item detail data so that the total matches the line-item detail information.
XBRL Tags	Specifies XBRL tags for the account. You can enter a maximum of 225 characters.

Account Type Behavior

Table 7 describes how account types behave in the system. For example, ASSET accounts do not total across periods, while a REVENUE account provides a year to date total. If you debit an ASSET account, the value that you enter is added to the account. If you credit an ASSET account, the value that you enter is subtracted from the account. All account types, except for GROUPLABEL, contain data.

Table 7 Account Type Behaviors

Type	YTD Total	Debit	Credit	Default Translation
ASSET	No	Add	Sub	DefaultRateForBalance Accounts
LIABILITY	No	Sub	Add	DefaultRateForBalance Accounts
REVENUE	Yes	Sub	Add	DefaultRateForFlow Accounts
EXPENSE	Yes	Add	Sub	DefaultRateForFlow Accounts
FLOW	Yes	Add	Sub	None
BALANCE	No	Add	Sub	None
BALANCE RECURRING	No	Add	Sub	None
CURRENCYRATE	No	N/A	N/A	N/A
GROUPLABEL	N/A	N/A	N/A	N/A
DYNAMIC	N/A	N/A	N/A	N/A

Table 13 indicates how an account type behaves when totaled into a specific type of parent account. For example, when aggregated, ASSET account values are added into parent ASSET and EXPENSE accounts and subtracted from parent LIABILITY and REVENUE accounts.

Note: The account types across the top of the table are identified by the first one or two letters of the account type.

Table 8 Account Type Behaviors During Aggregation into Parent Accounts

Account Type	Parent Account									
	A	L	R	E	F	B	BR	C	G	D
ASSET	Add	Sub	Sub	Add	Add	Add	Add	No	No	No
LIABILITY	Sub	Add	Add	Sub	Add	Add	Add	No	No	No
REVENUE	Sub	Add	Add	Sub	Add	Add	Add	No	No	No
EXPENSE	Add	Sub	Sub	Add	Add	Add	Add	No	No	No
FLOW	Add	Add	Add	Add	Add	Add	Add	No	No	No
BALANCE	Add	Add	Add	Add	Add	Add	Add	No	No	No
BALANCE RECURRING	Add	Add	Add	Add	Add	Add	Add	No	No	No
CURRENCYRATE	No	No	No	No	No	No	No	No	No	No
GROUPLABEL	No	No	No	No	No	No	No	No	No	No
DYNAMIC	No	No	No	No	No	No	No	No	No	No

Note: In Table 8, No indicates that the account type is not aggregated into the parent account.

Example

This example illustrates how account types are aggregated into parent accounts:

```

Total Assets 80
├ Fixed Assets 100
└ Amortization 20

```

In this example, Total Assets is an ASSET account and the parent of Fixed Assets (an ASSET account) and Amortization (a LIABILITY account). When the accounts are aggregated into the parent account, the Fixed Assets value of 100 is added, the Amortization value of 20 is subtracted, and the resulting value for Total Assets is 80.

Defining Dynamic Accounts

Dynamic accounts are accounts with values that are dynamically calculated when the data is requested. The values for dynamic accounts are not stored. The most common type of dynamic calculation is ratio calculation.

► To define a dynamic account and calculation:

1 Set up an account that uses the Dynamic account type.

Only base accounts can be dynamic.

Note: These account attributes are ignored for dynamic accounts: IsCalculated, IsConsolidated, EnableCustom1Aggr, EnableCustom2Aggr, EnableCustom3Aggr, EnableCustom4Aggr, UsesLineItems.

2 In a rules file, create a Sub Dynamic () section.

3 In the rules file, define a calculation.

For more information on writing calculations, use the guidelines for creating rules.

Defining Custom Members

You define Custom members by using the attributes in [Table 9](#).

Table 9 Custom Member Attributes

Attribute	Description
DefaultParent	Specifies the default parent for the Custom dimension member.
Description	Specifies the description for the custom member. The description can contain up to 80 characters, including spaces.
IsCalculated	Specifies whether the base-level Custom account is calculated. If a base-level Custom account is calculated, you cannot manually enter values. Specify Y if the Custom account is to be calculated and N if the Custom account is not to be calculated.
Member	Specifies the name for the custom member. This attribute is required. The name must be unique and can contain up to 80 characters, including spaces, but cannot start with a space. Note: Custom member names cannot contain spaces if you are using an Oracle database. Note: The name of a Custom1 dimension member cannot duplicate the name of a consolidation method. Do not use these characters in the custom member name: ● Period (.) ● Plus sign (+) ● Minus sign (-) ● Asterisk (*) ● Slash mark (/) ● Number sign (#) ● Comma (,) ● Semicolon (;) ● At sign (@) ● Double quote (") ● Curly brackets ({ })

Attribute	Description
SecurityClass	Specifies the security class name that defines the users who can access the Custom dimension data. Security class names can contain up to 80 characters. Security access applies only to data.
Submission Group	Specifies the submission group. The value can be a number from 0 to 99. The default is blank. A blank value defaults to the value of 1. If you set the submission group to zero (0), the account is not included in the review process.
SwitchSignForFlow	Specifies sign change (Debit or Credit) for FLOW accounts that use these rules: ● ASSET to LIABILITY ● LIABILITY to ASSET ● EXPENSE to REVENUE ● REVENUE to EXPENSE ● BALANCE to FLOW ● FLOW to BALANCE Specify Y if the sign for the account is switched and N if the sign for the account is not switched.
SwitchTypeForFlow	Specifies the account type change for FLOW accounts that use these rules: ● ASSET to EXPENSE ● EXPENSE to ASSET ● LIABILITY to REVENUE ● REVENUE to LIABILITY ● BALANCE to FLOW ● FLOW to BALANCE Specify Y if the account type for the account is switched and N if the account type for the account is not switched.
UserDefined1, UserDefined2, UserDefined3	Stores custom information for the custom dimension member. You can enter a maximum of 80 characters. The UserDefined1, UserDefined2, and UserDefined3 functions retrieve the text stored in this attribute.

Defining Entity Members

You define entity members by using the attributes in [Table 10](#).

Table 10 Entity Member Attributes

Attribute	Description
AllowAdjFromChildren	Specifies whether journal postings from children are permitted for the parent entity. Specify Y if journal postings from children are permitted and N if journal postings from children are not permitted.
AllowAdjs	Specifies whether journal postings are permitted for this entity. Specify Y if journal postings are permitted for the entity and N if journal postings are not permitted for the entity.
DefaultParent	Specifies the default parent for the entity.

Attribute	Description
DefCurrency	Specifies the default currency for the entity. This attribute is required.
Description	Specifies the description for the entity. The description can contain up to 80 characters, including spaces.
HoldingCompany	Specifies the holding company for the entity. Can be the name of an entity or blank.
IsICP	Specifies whether the entity is an intercompany entity. Specify Y if the entity is an intercompany entity and N if the entity is not an intercompany entity. If the entity is an intercompany entity, it is displayed in the POV in the ICP dimension under ICP Entities.
Member	Specifies the name for the entity. This attribute is required. The name must be unique and can contain up to 80 characters including spaces but cannot start with a space. Caution! Entity names cannot contain spaces if you are using an Oracle database. Do not use these characters in the entity name: ● Period (.) ● Plus sign (+) ● Minus sign (-) ● Asterisk (*) ● Slash mark (/) ● Number sign (#) ● Comma (,) ● Semicolon (;) ● At sign (@) ● Double quote (") ● Curly brackets ({ }) Note: You cannot use ALL as the name of an entity.
SecurityAsPartner	Specifies the name of a valid security class for the ICP entity. This attribute enables you to secure the entity in an ICP dimension.
SecurityClass	Specifies the name of a valid security class of users who can access the data of the entity. Security class names can contain up to 80 characters.
UserDefined1, UserDefined2, UserDefined3	Stores custom information for the entity. You can enter a maximum of 80 characters. The UserDefined1, UserDefined2, and UserDefined3 functions retrieve the text stored in this attribute.

Defining Scenario Members

You define scenario members by using the attributes in [Table 11](#).

Table 11 Scenario Member Attributes

Attribute	Description
ConsolidateYTD	Specifies the view for consolidation. This attribute is required. Specify Y for YTD or N for periodic.

Attribute	Description
DefaultFreq	Specifies the types of periods for which data input is valid for the scenario. This attribute is required. For example, a value of Monthly indicates that you can extract input data only in month-based periods, not in quarter-based or year-based periods. The frequency must be defined in the application profile.
DefaultParent	Specifies the default parent for the scenario.
DefaultView	Specifies the view to use when Scenario View is selected in the point-of-view bar. This attribute is required. Specify YTD or Periodic. If you change the default view for a scenario and line-item detail has been entered, you should first extract the line-item detail and save it. Then delete the line-item detail from the scenario before changing the view. You must change the extracted line-item detail to match the new default view before loading it.
DefFreqForICTrans	Specifies the default frequency for intercompany transactions. This attribute must be a valid frequency and can contain a maximum of 80 characters. The default for this attribute is blank.
Description	Specifies the description for the scenario. The description can contain up to 80 characters, including spaces.
EnableDataAudit	Specifies whether the scenario is audited. This attribute for an account or a scenario determines what can be audited. Specify one of these values: ● Y to automatically audit all accounts. Even accounts that have EnableDataAudit set to False will be audited. ● O to audit only those accounts that have EnableDataAudit set to True. ● N to disable auditing for all accounts.
MaximumReviewLevel	Specifies the maximum Process Management review level for the scenario. Specify a review level from 1 to 10. This attribute is required.
Member	Specifies the name for the scenario. This attribute is required. The name must be unique and can contain up to 80 characters, including spaces, but cannot start with a space. Note: Scenario names cannot contain spaces if you are using an Oracle database. Do not use these characters in the scenario name: ● Period (.) ● Plus sign (+) ● Minus sign (-) ● Asterisk (*) ● Slash mark (/) ● Number sign (#) ● Comma (,) ● Semicolon (;) ● At sign (@) ● Double quote (") ● Curly brackets ({ })
PhasedSubmissionStartYear	Specifies the start year for phased submissions in process management.

Attribute	Description
SecurityClass	Specifies the name of a valid security class that defines users who can access the data for the scenario. Security class names can contain up to 80 characters. For example, a user with None access rights to a scenario can open journal periods for the scenario.
SupportsProcessManagement	Specifies whether Process Management functionality is enabled. Specify one of these values: ● Y to enable the Process Management ● N to disable the Process Management option ● A to enable Process Management and E-mail alerting.
UserDefined1, UserDefined2, UserDefined3	Stores custom information for the scenario. You can enter a maximum of 80 characters. The UserDefined1, UserDefined2, and UserDefined3 functions retrieve the text stored in this attribute.
UsesLineItems	Specifies whether the scenario can have line items. Specify Y if the scenario can accept line items and N if the scenario cannot accept line items. Note: If you change this attribute after line-item detail is entered, the stored line item detail may no longer be valid for the scenario. These behaviors occur: ● If the scenario accepted line items and now it cannot, the line-item detail stored in the database is no longer valid. Only the total is displayed. ● If the scenario did not accept line items and now it can, there is a total amount but no corresponding line-item detail information for the scenario. You can extract the total and then load it as line-item detail data so that the total matches the line-item detail information.
ZeroViewForAdj	Specifies how to interpret missing, adjusted data values for the period. This attribute is required. Specify YTD or Periodic.
ZeroViewForNonadj	Specifies how to interpret missing, nonadjusted data values for the period. This attribute is required. Specify YTD or Periodic.

Defining Application Settings

Application settings apply to an entire Financial Management application. Application settings determine the following information for the application:

- Is the organization dynamic, using organization by period?
- Which dimensions are secured?
- What default translation rates and methods are used?
- What is the ICP weight?
- Are consolidation rules applied?
- What is the default currency?

You define application settings by using the attributes in [Table 12](#).

Table 12 Application Settings Attributes

Attribute	Description
ConsolidationRules	Specifies whether consolidation rules are supported. Specify one of these values: Y to use the rules written in the Consolidate() routine in a user-defined rule. R to derive the proportional value in the Value dimension. Note that the proportional data is not stored. N to use the default consolidation and eliminations.
DefaultCurrency	Specifies the default currency for the application. This attribute is required.
DefaultRateForBalanceAccounts	The account that contains the translation rate to use for ASSET or LIABILITY accounts. This attribute is required.
DefaultRateForFlowAccounts	The account that contains the translation rate to use for REVENUE or EXPENSE accounts. This attribute is required.
DefaultValueForActive	Specifies the default value for the Active account. This attribute is required. Specify 0 if the child is considered inactive and is not consolidated into the parent. Specify 1 if the child is considered active and is consolidated into the parent.
EnableMetadataSecurityFiltering	Specifies whether users of an application see all dimension members or only the members to which they have access. The system filters these dimension members: ● Scenario ● Entity ● Intercompany Partner (ICP) ● Account ● Custom1, Custom2, Custom3, Custom4 Specify Y to filter out the dimension members to which the user does not have access. The default for this attribute is N.
FDMApplName	Name of the Oracle Hyperion Financial Data Quality Management, Fusion Edition application
ICPEntitiesAggregationWeight	Specifies the percentage of intercompany partner entity [ICP Entities] amounts that aggregate to the [ICP Top] member of the Value dimension. This attribute is required. The percentage is scaled to hundreds, with 1.0 equalling 100 percent.
MaxCellTextSize	Specifies the maximum number of characters that can be used for cell text. Specify -1 for no limit or a positive number up to 2,147,483,646. The default value is 8000.
MaxDocAttachmentSize	Specifies the maximum number of bytes for the size of document attachments. Specify -1 for no limit or a positive number up to 2,147,483,646. The default value is -1.
MaxNumDocAttachments	Specifies the maximum number of document attachments per user. Specify -1 for no limit or a positive number up to 2,147,483,647. The default value is -1.
NodeSecurity	Specifies the type of security access for nodes. This attribute is required. Specify Entity to check node data based on security access for the entity and Parent to check node data based on security access for the parent.
OrgByPeriodApplication	Specifies whether new consolidation structures can coexist with past consolidation structures in the application. Specify Y to allow new organizational structures or N to allow only current organizational structures.

Attribute	Description
SupportSubmissionPhaseforAccounts	Specifies whether phased submissions in process management are supported for accounts in the application. Valid values are Y or N. Default is N.
SupportSubmissionPhaseforCustom1, SupportSubmissionPhaseforCustom2, SupportSubmissionPhaseforCustom3, SupportSubmissionPhaseforCustom4	Specifies whether phased submissions in process management are supported for the Custom members in the application. Valid values are Y or N. Default is N.
SupportSubmissionPhaseforICP	Specifies whether phased submissions in process management are supported for ICP members in the application. Valid values are Y or N. Default is N.
UsePVAForBalanceAccounts	Specifies the default translation method for BALANCE accounts. Specify Y to use the periodic value (PVA) translation method and N to use the value at exchange rate (VAL) translation method.
UsePVAForFlowAccounts	Specifies the default translation method for FLOW accounts. Specify Y to use the periodic value (PVA) translation method and N to use the value at exchange rate (VAL) translation method.
UseSecurityForAccounts	Specifies whether accounts in the application are protected by security. Specify Y for security on accounts and N for no security.
UseSecurityForCustom1, UseSecurityForCustom2, UseSecurityForCustom3, UseSecurityForCustom4	Specifies whether custom dimensions in the application are protected by security. Specify Y for security on custom dimensions and N for no security on custom dimensions.
UseSecurityForEntities	Specifies whether entities in the application are protected by security. Specify Y for security on entities and N for no security on entities.
UseSecurityForICP	Specifies whether ICP members in the application are protected by security. Specify Y for security on ICP members and N for no security on ICP members.
UseSecurityForScenarios	Specifies whether scenarios are protected by security. Specify Y for security on scenarios and N for no security on scenarios.
UseSubmissionPhase	Specifies whether phased submissions in process management are used in the application. Valid values are Y or N. Default is N.
ValidationAccount	Specifies the account name to use for validation. The account used for validation must be an existing account and must have a valid intersection with [ICPTop] in the Custom dimensions. In process management, validation accounts are used to ensure that the value equals zero before a process unit can be promoted to the next review level. Validation account is used for Submission Phase 1, and Validation Accounts 2 to 9 are used for Submission Phases 2 to 9.

Organization by Period

The organization by period functionality enables the most recent consolidation structure to coexist with past structures in the same application.

Organizational structures can change for many reasons, including acquisitions, disposals, mergers, and reorganizations. To support organizational changes, Financial Management uses a system account, Active, to reflect the active or inactive consolidation status of a child into its parent. The Active account acts as a filter of the entity hierarchy. The Active account is an intercompany account that stores data at the parent level and uses the ICP dimension to store information about children.

For an ICP member that corresponds to a child of a parent, the Active account indicates to the system whether the child should be considered as an active consolidation member for the current year, scenario, and time period. Children that correspond to ICP members for which the Active account is equal to 0 are considered to be inactive children and are not consolidated. Children that correspond to ICP members for which the Active account is equal to 1 are considered to be active children and are consolidated. Active account values can be viewed or changed in data grids. Similarly, changes to active child data affect the parent, while changes to inactive child data does not affect the parent.

The DefaultValueForActive attribute controls the status of children for which the Active account is blank. So, every parent-child intersection does not have to be flagged as active or inactive. By default, every child is active in relation to its parent unless otherwise specified.

Defining Consolidation Methods

You define consolidation methods for an application by using the attributes in [Table 13](#).

Table 13 Consolidation Methods Attributes

Attribute	Description
ConsolMethod	Specifies the name for the consolidation method. This attribute is required. The name must be unique and can contain up to 80 characters, including spaces. You cannot use these characters in the name: ● Period (.) ● Plus sign (+) ● Minus sign (-) ● Asterisk (*) ● Slash mark (/) ● Number sign (#) ● Comma (,) ● Semicolon (;) ● At sign (@) ● Double quote (") ● Curly brackets ({ })

Attribute	Description
Control	Specifies the threshold that corresponds to the type of control to be used by the calculation routine. Specify one of these values for this attribute: ● Blank ● No ● Limited ● Full
Description	Specifies the description for the consolidation method. The description can contain up to 80 characters, including spaces.
IsHoldingMethod	Specifies whether the consolidation method is used for the holding company. This attribute is optional. Specify Y to use the method for the holding company and N to use a different method for the holding company.
PercentConsol	Specifies the consolidation percentage applied by the ownership calculation routine. Specify a value for the percent (such as 100) or one of these keywords: ● POWN ● POWNMIN Note: For information on POWN and POWNMIN, see “Using POWN and POWNMIN Methods” on page 88.
ToPercentControl	Specifies the upper boundary of the range for PercentControl. Used for the ownership calculation routine. Specify a value between 0 and 100. Note: One of the method records must have a value of 100.
ToPercentControlComp	Specifies whether the upper boundary of the range of percent control is included in the range. Used for the ownership calculation routine together with the ToPercentControl attribute. This attribute is optional if the UsedByCalcRoutine attribute is N. Specify < or <= for this attribute.
UsedByCalcRoutine	Specifies whether the method is used for the automatic ownership calculation routine. Specify Y to use the method for ownership calculations and N not to use the method for ownership calculations.

Using Consolidation Methods

Consolidation methods are defined in the consolidation methods section of the metadata. The methods are used during the consolidation and calculate ownership processes.

When you define consolidation methods in the metadata, the system automatically generates the [ConsolMethod] system list for the Custom1 dimension. The system list consists of all methods defined in the consolidation methods section of the metadata.

There are two ways to assign the consolidation method to an entity for use during consolidation. The method can be assigned manually through data load or data entry. The method can also be assigned by the calculate ownership routine, which is based on the ultimate percent control assigned to the entity.

Using POWN and POWNMIN Methods

POWNMIN is a keyword used in the consolidation method table. You use it for the method corresponding to the EQUITY method. The settings in this table are used by the Ownership

Calculation routine to compute the percentages of control, the ultimate percentages of ownership, and to automatically assign the percentages of consolidation and the consolidation methods for legal consolidation. Using POWNMIN, the percentage of consolidation that is assigned for the EQUITY company corresponds to the percentage used in a staged consolidation

POWNMIN Calculation

$$\text{POWNMIN} = \text{POWN} + \text{Sum of (Percent Minority of Entity Owners * Direct Percentage of Ownership in the Entity)}$$

Where:

Percent Minority = Percent Consolidation – Percent Ownership

Entity Owners means any entity (within the descendants of the current parent) owning shares of the entity processed.

Direct Percentage of Ownership in the Entity can be retrieved from the account [shares%owned].

Example:

- B is owned by A: 80%
- C is owned by A: 70%
- D is owned by B: 20%
- D is owned by C: 20%

The Parent GROUP has A, B, C, and D as dependents (A is the holding company). The system calculates percent ownership as follows:

- A: 100%
- B: 80%
- C: 70%
- D: 30%

Suppose the Percent Consolidation of D (from the consolidation method table) is POWNMIN.

In this case, the POWNMIN calculation process would be:

1. Consolidation of D into B using direct ownership percentage: 20%
2. Consolidation of D into C using direct ownership percentage: 20%
3. Consolidation of B and C into A using their respective percentages: (80% and 70%)

It is calculated as follows:

$$\text{Entity D's Percent consolidation} = 30\% + (100\% - 80\%) * 20\% + (100\% - 70\%) * 20\% = 40\%$$

Entity D is consolidated using a total percentage of 40%.

The POWNMIN calculation also could require iterations. The system stops when two subsequent iterations have the same result.

When the subholdings B and C are consolidated into A, some minority interests corresponding to 10% are calculated on the Equity from Entity D.

POWN Calculation

However, if the consolidation is done using a flat hierarchy, the process typically uses the ultimate percentage of ownership (POWN) as percentage of consolidation for the Equity company. In this case, the percentage of consolidation for D into the Group would be 30%. No minority interests would be calculated on the Equity from Entity D.

In summary:

- Using POWN, the percentage of consolidation assigned to Entity D would be 30% (ultimate percentage of ownership)
- Using POWNMIN, the percentage of consolidation assigned to Entity D would be 40% (percentage of consolidation)

For information on managing ownership, see the *Oracle Hyperion Financial Management User's Guide*.

Assigning Consolidation Methods Manually

To enter consolidation method information manually, you can create a data grid with this information:

POV: Scenario, Year, Period, View, Entity, Value, Account, C2, C3, C4

Scenario: Applicable scenario

Year: Applicable year

Period: Applicable period

Entity: A parent entity

Value: [None]

Account: [Method], a system generated account called Method

C2: [None]

C3: [None]

C4: [None]

Row: ICP entities (For parents, user can use the system list [ICP Entities] or the user-defined list of selected ICP entities.)

Column: Custom1 (The user should use the system-generated list [ConsolMethods].)

Method assignment information is stored in the account method of the data file of the parent entity. For each child of a parent, the system stores the consolidation method assignment in the ICP dimension. The assigned method is used when the children are consolidated to the parent.

For an intersection of the grid, use 1 to indicate the method assignment to the ICP entity. For example, if a parent group has two children, A and B, and you assign the Global method to A and the Equity method to B, enter 1 in the intersection for the Global method and entity A and 1 in the intersection for the Equity method and entity B.

Defining Currencies

You define currencies for an application by using the attributes in [Table 14](#).

Table 14 Currency Attributes

Attribute	Description
Currency	Specifies the name for the currency. This attribute is required. The name must be unique and can contain up to 80 characters, including spaces. Do not use these characters in the currency name: ● Period (.) ● Plus sign (+) ● Minus sign (-) ● Asterisk (*) ● Slash mark (/) ● Number sign (#) ● Comma (,) ● Semicolon (;) ● At sign (@) ● Double quote (“) ● Curly brackets ({ })
Description	Specifies the currency description. The description can contain up to 80 characters, including spaces.
DisplayInICT	Specifies if currencies display in the drop-down list in the intercompany transactions module. Specify Y to display currencies and N to not display currencies. The default is Y.
Scale	Specifies the unit in which amounts are displayed and stored for the currency by identifying where the decimal point is placed. This attribute is required. Determines how the exchange rate must be entered. Scale is a currency attribute, not an entity attribute. Specify one of these values for this attribute: ● Blank = None ● 0 = Units ● 1 = Tens ● 2 = Hundreds ● 3 = Thousands ● 4 = Ten thousands ● 5 = Hundred thousands ● 6 = Millions ● 7 = Ten millions ● 8 = Hundred millions ● 9 = Billions

Attribute	Description
TranslationOperator	Specifies the conversion calculation for intercompany transactions. The default is blank. Specify one of these values: ● D to calculate the local currency by dividing the transaction currency by the rate ● M to calculate the local currency by multiplying the transaction currency by the rate ● blank to default the value to D

The system member list [Currencies] is available for the Custom1 and Custom2 dimensions. Currencies that you add to the application are added to the [Currencies] member list. The [Currencies] list enables the entry of currency translation rates for pairs of currencies and provides a way of filtering out non-currency members.

System-Generated Accounts

When you create an application, system accounts for consolidation and ownership are automatically created for the application.

Note: You can change only the description, security class, and decimal location for system accounts. All other attributes for system accounts are predefined and cannot be modified.

Consolidation Accounts

The system accounts described in [Table 15](#) are required for each parent in the Entity dimension and are used in the consolidation process.

Note: All system accounts that are used for consolidation, except for the Active account, are BALANCE accounts. The Active account is a BALANCERECURRING account.

Table 15 System Accounts for Consolidation

Account	Description
Active	Consolidation status of a child into its parent. Yes if the child is consolidated into its parent. No if the child is not consolidated into its parent.
[PCON]	Percent consolidation. The percentage of the value of an entity that consolidates to the parent of the entity. Positive or negative numbers between -100 and 100, including 0. Default value is 100. Note: For subsequent periods, derived as 0. Therefore, you must enter the percentage in all subsequent periods.
[POWN]	Percent ownership based on the shares of the entity that are owned by other entities. A positive number between 0 and 100. Default value is 100.
[DOWN]	Percent of direct ownership. A positive number between 0 and 100. Default value is 100.
[PCTRL]	Percent control based on the voting shares of the entity that are owned by other entities. A positive number between 0 and 100. Default value is 100.

Account	Description
Method	Consolidation method assigned to the entity. None or a selection from the list of available methods.
Consol1, Consol2, Consol3	Consolidation methods. A number between 0 and 255.

Ownership Accounts

The system accounts described in [Table 16](#) are used for ownership calculations.

Note: All system accounts that are used for ownership calculations are BALANCE accounts.

Table 16 System Accounts for Ownership

Account	Description
SharesOwned	Total number of shares owned. Positive number or 0. Default is 0. Note: Total shares owned must be less than or equal to the total shares outstanding.
VotingOwned	Number of voting shares owned. Positive number or 0. Default value is 0. Note: Total voting shares owned must be less than or equal to the total voting shares outstanding.
SharesOutstanding	Total number of shares outstanding or the percentage of shares outstanding. Positive number or 0. Default value is 0. Note: Enter the number of shares outstanding, or enter shares outstanding as a percentage. Enter 100 for percentage.
VotingOutstanding	Number of voting shares outstanding. A positive number or 0. Default value is 0. Note: Enter the number of voting shares outstanding, or enter voting shares outstanding as a percentage. Enter 100 for percentage.
Shares%Owned	Calculated by system.
Voting%Owned	Calculated by system.

Editing System-Generated Accounts

When you create an application, system account members are automatically created for the application.

Note: You can modify only the description, security class, and the decimal location for system account members. All other attributes are predefined and cannot be modified.

➤ To modify system accounts:

1 In Metadata Manager, open the file that contains the system-generated account members.

- 2 Select the **List View** tab.
- 3 From the list, select a system account, and modify the description, security class, and decimal location as needed.
- 4 As desired, repeat step 3 to modify other system account members.
- 5 Click **Save File**.

Note: You must load the updated metadata file into your application for your changes to take effect.

Setting Up Intercompany Partners

Intercompany transactions are managed across the Intercompany Partner (ICP) dimension. The ICP dimension contains all intercompany balances that exist for an account. ICP is a reserved dimension used in combination with the Account dimension and custom dimensions to track and eliminate intercompany transaction details.

To set up an application for intercompany transactions, you must perform these actions:

- Indicate the accounts that perform intercompany transactions and indicate a plug account for each intercompany account (IsICP and PlugAcct attributes in account metadata)
- Indicate the entities that perform intercompany transactions (IsICP attribute in entity metadata)

When you create intercompany transactions, each group must have at least one intercompany account and one plug account. You designate an account as intercompany by selecting the IsICP attribute for the account in Metadata Manager. When an account is designated as intercompany and intercompany transactions are entered, eliminating or reversing entries are generated in the [Elimination] Value dimension member through the consolidation process.

A plug account is an account that, when eliminations are completed, stores the difference between two intercompany accounts in the Elimination Value dimension. A plug account can be set up as an ICP account. For a plug account to be detailed by ICP, set the IsICP attribute to Y or R so that the system writes eliminations to the corresponding ICP member. If you do not want a plug account to be detailed by ICP, set the IsICP attribute to N so that the system writes eliminations to [ICP None]. During consolidation, transactions between valid intercompany entities are eliminated. See [“Defining Entity Members” on page 81](#).

Following are the system-generated ICP elements.

Table 17 System-Generated ICP Elements

ICP Element	Description
[ICP Top]	Specifies the top intercompany member
[ICP None]	Specifies that no intercompany member is used
[ICP Entities]	Specifies the entities that are designated for intercompany transactions

Editing System-Generated ICP Members

When you create an application, Intercompany Partner members are automatically created for the application. An ICP member is created for each Entity member for which the IsICP attribute is selected.

Note: You can modify only the description and security class for ICP members. All other attributes are predefined and cannot be modified.

➤ To modify intercompany members:

- 1 In **Metadata Manager**, open the file that contains the system-generated intercompany partner members.
- 2 From the list, select an ICP member and, as needed, modify the description and security class.
- 3 As desired, repeat step 2 to modify other ICP members.
- 4 Click **Save File**.

Note: You must load the updated metadata file into your application for your changes to take effect.

Editing System-Generated Value Members

When you create an application, Value members are automatically created for the application.

Note: You can modify only the description for Value members. All other attributes are predefined and cannot be modified.

After you load metadata, Financial Management automatically creates three Value dimension members for each currency in your application: *CurrencyName*, *CurrencyName* Adjs, and *CurrencyName* Total, where *CurrencyName* is the currency label. For example, for a currency of USD, Financial Management creates these Value dimension members: USD, USD Adjs, and USD Total.

Note: The metadata file must have a description specified for the Value member <Currency>. If descriptions for currencies have not been specified in the metadata file, when you load metadata, the currency descriptions will not be displayed.

➤ To modify a Value member description:

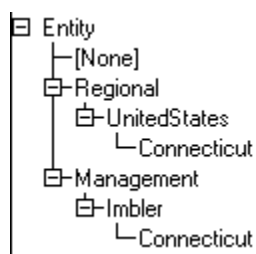
- 1 In **Metadata Manager**, open the file that contains the system-generated Value members.
- 2 On the **Member Attributes** tab, in the hierarchy, select a member, and modify its description attribute.
- 3 As desired, repeat step 2 to add descriptions for other Value members.
- 4 Click **Save File**.

Note: You must load the updated metadata file into your application for your changes to take effect.

Metadata Filtering Based on Security

When you filter metadata based on security, users see only the elements of the Scenario, Entity, ICP, Account, and Custom1 through Custom4 dimensions to which they have access. You set up metadata filtering at the application level by setting the AppSettings EnableMetadataSecurityFiltering metadata attribute to Y. For elements a user can view in a hierarchy, assign a security class and give the user Metadata access to the security class.

Users have implied access to the parents and ancestors of members to which they have access. With implied access, users see ancestors and parents in a hierarchical tree structure but cannot access them. For example, in the following tree structure, the user has access to only Connecticut even though the parents (UnitedStates and Imbler) and the ancestors (Management and Regional) are displayed in the tree.



Creating Metadata Files of the APP Format

You can use a metadata file of the APP format to add metadata to a Financial Management application. The metadata file sections can be arranged in any order; however, Financial Management automatically processes sections in this order:

- Currencies
- Entity dimension
- Scenario dimension
- Custom dimensions
- Account dimension
- Value dimension
- Intercompany Partner dimension
- Application settings
- Consolidation methods

For each type of dimension-related metadata, sections are processed in this order:

- Dimension

- Members
- Hierarchies

Metadata for the Entity, Scenario, Account, and Custom dimensions is placed in the members and hierarchies sections. Custom dimensions may include the dimension section. The sections for each type of metadata can appear only once in a metadata file.

You can use these characters as a delimiter:

, ~ @ \$ % ^ | : ; ? \

Note: You must use a character that is not used in the file name or in any other way in the file. Delimiters are necessary only for ASCII files with the APP file extension. Delimiters are not necessary for extensible markup language (XML) files.

A line starting with an exclamation point (!) indicates the beginning of a new section in the metadata file and must be followed by a valid section name, for example, currencies, members, or hierarchies. True or false values are represented as Y for true and N for false. A line starting with an apostrophe (') is considered a comment line and is ignored by the system.

You can use these sections in a metadata file:

- File format
- Version
- Application settings
- Currencies
- Dimension
- Members
- Hierarchies
- Consolidation methods

File Format

This section of a metadata file indicates the file version number. The version number changes only when changes are made to the file format. The file format is automatically generated when you extract metadata; if you are defining a file to load, you must include a valid file format. This syntax specifies the file format:

```
!FILE_FORMAT = majorNumber.minorNumber
```

majorNumber and *minorNumber* consist of one or two digits. *majorNumber* can contain a leading zero, and *minorNumber* can contain a trailing zero. You must include *majorNumber* and *minorNumber* and use only a period (.) as the decimal separator. These examples represent valid file format values:

```
!FILE_FORMAT = 3.4
!FILE_FORMAT = 3.40
```

```
!FILE_FORMAT = 03.40
!FILE_FORMAT = 03.4
```

Version

This section of a metadata file indicates the version of Financial Management that was used to extract metadata. The version number is automatically generated when you extract metadata; if you are creating a metadata file for loading, you do not need to specify a version. This syntax represents the version:

```
!VERSION = major version.minor version.build version
```

This example represents a valid version value:

```
!VERSION = 3.5.365
```

Application Settings

This section of a metadata file defines settings that apply to the entire Financial Management application. For information on application settings attributes, see [“Defining Application Settings” on page 84](#).

This example specifies application settings attributes:

```
!APPLICATION_SETTINGS
DefaultCurrency=USD
DefaultRateForBalanceAccounts=Rate1
DefaultRateForFlowAccounts=Rate2
UsePVAForBalanceAccounts=Y
UsePVAForFlowAccounts=Y
ICPEntitiesAggregationWeight=1
DefaultValueForActive=1
ConsolidationRules=N
OrgByPeriodApplication=N
NodeSecurity=Entity
UseSecurityForAccounts=N
UseSecurityForEntities=Y
UseSecurityForScenarios=Y
UseSecurityForCustom1=N
UseSecurityForCustom2=N
UseSecurityForCustom3=N
UseSecurityForCustom4=N
UseSecurityForICP=N
EnableMetadataSecurityFiltering=N
```

Currencies

This section of a metadata file defines currencies. This syntax specifies a currency:

```
Label; Scale; Descriptions
```

See [“Defining Currencies” on page 91](#).

This example specifies currency attributes:

```
!CURRENCIES
EURO;0;English=European Euro
GBR;0;English=Great Britain Pounds
USD;0;English=United States Dollars
```

Dimension

This section of a metadata file defines aliases for Custom dimensions. Create one dimension section for each Custom dimension. Use this syntax to create the dimension section for a Custom dimension, where N is the number of the Custom dimension:

```
!DIMENSION=CustomN
```

This section contains only the Alias attribute. An alias is an alternate name for a Custom dimension and can have a maximum of 80 characters. Custom dimension aliases are displayed on the Metadata Item Properties tab of the Manage Metadata module.

Enter an alias by using this syntax, replacing the <> characters with the alias name:

Alias=<>

This example specifies a dimension and an alias:

```
!DIMENSION=Custom2
Alias=Customers
```

Members

This section of a metadata file defines the members of a dimension. You can use delimiters to represent missing values as empty. Enter dimension members by using this syntax:

```
!MEMBERS=Custom2  
'Label;IsCalculated;SwitchSignForFlow;SwitchTypeForFlow;  
UserDefined1;UserDefined2;UserDefined3;SecurityClass;  
DefaultParent;Descriptions  
[None];N;N;N;;;;DefaultParent=#root  
AllCustomers;Y;N;N;;;;DefaultParent=#root  
Customer2;N;N;N;;;;DefaultParent=AllCustomers  
Customer3;N;N;N;;;;DefaultParent=AllCustomers  
Customer4;N;N;N;;;;DefaultParent=AllCustomers  
Customer5;N;N;N;;;;DefaultParent=AllCustomers
```

These topics list the formats for the members sections of the Account, Scenario, Entity, Custom, Value, and ICP dimensions.

Account

What follows is the syntax for Account dimension members:

'Label, AccountType, IsCalculated, IsConsolidated, IsICP, PlugAcct, Custom1TopMember, Custom2TopMember, Custom3TopMember, Custom4TopMember, NumDecimalPlaces, UsesLineItems, EnableCustom1Aggr, EnableCustom2Aggr, EnableCustom3Aggr, EnableCustom4Aggr, UserDefined1, UserDefined2, UserDefined3, XBRLTags, SecurityClass, ICPTopMember, EnableDataAudit, DefaultParent, Descriptions

See [“Defining Accounts” on page 76](#).

The following example specifies attributes for two accounts, AdminExpenses and CapitalStock:

```
!MEMBERS=Account
AdminExpenses;EXPENSE;N;Y;Y; ;Golf;ALL;ALL;ALL;2;N;Y;Y;Y;Y; ;
; ; ; ;N;DefaultParent=NetIncome
CapitalStock;LIABILITY;N;Y;N; ; ; ;ClosingBalance;6;N;Y;Y;Y;Y;
; ; ; ;N;DefaultParent=TotalEquity;English=Capital Stock
```

Scenario

What follows is the syntax for Scenario dimension members:

```
'Label, DefaultFreq, DefaultView, ZeroViewForNonadj, ZeroViewForAdj, ConsolidateYTD,
UserDefined1, UserDefined2, UserDefined3, SupportsProcessManagement, SecurityClass,
MaximumReviewLevel, UsesLineItems, EnableDataAudit, EnableJournalsAutoLabel,
DefFreqForPostingFlowTrans, DefaultParent, Descriptions
```

See [“Defining Scenario Members” on page 82](#).

The following example specifies attributes for two scenarios, Actual and Budget:

```
!MEMBERS=Scenario
Actual;MTD;Periodic;Periodic;Periodic;N; ; ;N; ;10;Y;N;N;MTD ;DefaultParent=#root
Budget;MTD;Periodic;Periodic;Periodic;N; ; ;Y; ;10;Y;N;N;MTD ;DefaultParent=#root
```

Entity

What follows is the syntax for Entity dimension members:

```
'Label, DefCurrency, AllowAdjs, IsICP, AllowAdjFromChildren, SecurityClass,
UserDefined1, UserDefined2, UserDefined3, HoldingCompany, SecurityAsPartner,
DefaultParent, Descriptions
```

See [“Defining Entity Members” on page 81](#).

The following example specifies attributes for three entities, California, Canada, and Connecticut:

```
!MEMBERS=Entity
California;USD;Y;Y;Y;US;;;;;DefaultParent=Imbler;English=State of
California;French=California
Canada;USD;Y;N;N;;;;;DefaultParent=Regional
Connecticut;USD;Y;Y;N;US;Northeast;;;;;DefaultParent=Imbler
```

Custom 1 Through Custom4

Following is the syntax for Custom1 through Custom4 dimension members:

```
'Label, IsCalculated, SwitchSignForFlow, SwitchTypeForFlow, UserDefined1, UserDefined2,
UserDefined3, SecurityClass, DefaultParent, Descriptions
```

See [“Defining Custom Members” on page 80](#).

The following example specifies attributes for members of the Custom2 dimension:

```

!MEMBERS=Custom2
[None];N;N;N; ; ; ;DefaultParent=#root
AllCustomers;Y;N;N; ; ; ;DefaultParent=#root
Customer2;N;N;N; ; ; ;DefaultParent=AllCustomers
Customer3;N;N;N; ; ; ;DefaultParent=AllCustomers
Customer4;N;N;N; ; ; ;DefaultParent=AllCustomers
Customer5;N;N;N; ; ; ;DefaultParent=AllCustomers

```

Value

You can use the members section to define descriptions for system-defined members of the Value dimension. In addition, for Value dimension members that the system creates for currencies, you can define descriptions that will be appended to the currency descriptions. Following is the syntax for Value members:

Label;Descriptions

You can specify the label of a system-defined Value member. You also can use these labels to create descriptions that are appended to the descriptions for the corresponding Value members that the system creates for user-defined currencies:

```

<Currency Total>
<Currency Adjs>
<Currency>

```

For example, suppose that you define the currencies USD and EUR with descriptions of “US Dollars” and “Euro,” respectively. In addition, suppose you define these Value member descriptions in a loaded metadata file:

```

[None];English=ValueNone
<Currency Total>;English=Total
<Currency Adjs>;English=Adjs
<Currency>;English=Base

```

[Table 18](#) describes the Value dimension member triplets that the system creates for the USD and Euro currencies.

Table 18 Value Dimension Descriptions

Value Member	Description
USD Total	US Dollars Total
USD Adjs	US Dollars Adjs
USD	US Dollars Base
EUR Total	Euro Total
EUR Adjs	Euro Adjs
EUR	Euro Base

Note: The metadata file must have a description specified for the Value member <Currency>. If descriptions for currencies have not been specified in the metadata file, when you load metadata, the currency descriptions will not be displayed.

Intercompany Partner

You can use the members section to define security classes and descriptions for these system-defined members of the Intercompany Partner dimension:

- [ICP Top]
- [ICP None]
- [ICP Entities]

Following is the syntax for Intercompany Partner members:

Label;SecurityClass;Descriptions

This example shows how to define descriptions for [ICP Top], [ICP None], and [ICP Entities] without specifying security classes:

```
[ICP Top];;English=Top ICP
[ICP None];;English=No ICP
[ICP Entities];;English=Entities ICP
```

Consolidation Methods

This section of a metadata file defines the consolidation methods.

Following is the syntax for consolidation methods:

*Label;UsedByCalcRoutine;IsHoldingMethod;ToPercentControlComp;
ToPercentControl;PercentConsol;Control;Descriptions*

See [“Defining Consolidation Methods” on page 87](#).

This example specifies attributes for consolidation methods:

```
!CONSOLIDATION_METHODS
M2;Y;N;<=;20;0;No
M3;Y;N;<=;50;POWN;Limited
M4;Y;N;<=;50;50;Limited
M1;Y;Y;<=;100;100;Full
M5;Y;N;<=;100;100;Full
```

Hierarchies

This metadata file section defines parent-child relationships. A parent-child relationship is referred to as a node. A node can have its own set of attribute values.

A node record is a delimited list. The first two items of each line of the list identify a parent and child. You can use delimiters to represent missing attribute values as empty. All top-level members in a hierarchy should be represented as children of an empty parent.

Tip: Node records for Custom dimensions contain a third attribute. See [“Custom1 Through Custom4 Hierarchies” on page 104.](#)

To begin a hierarchies section, enter this line, replacing the <> characters with the dimension name:

```
!HIERARCHIES=<>
```

Do not include spaces when starting sections for Custom dimensions. For example, begin the Hierarchies section for the Custom2 dimension with this line:

```
!HIERARCHIES=Custom2
; [None] ; 1
; AllCustomers ; 0
AllCustomers ; Customer2 ; 1
AllCustomers ; Customer3 ; 1
AllCustomers ; Customer4 ; 1
AllCustomers ; Customer5 ; 1
```

These topics list the formats for the Hierarchies sections of the Account, Scenario, Entity, and Custom dimensions.

Account Hierarchies

Following is the syntax for Account dimension hierarchies:

parentmemberlabel;childmemberlabel

This example specifies Account dimension hierarchies:

```
!HIERARCHIES=Account
; [None]
; ExchangeRates
ExchangeRates ; Rate1
ExchangeRates ; Rate2
; Plug
; NetProfit
NetProfit ; NetIncome
NetIncome ; GrossMargin
GrossMargin ; Sales
GrossMargin ; TotalCosts
TotalCosts ; Purchases
TotalCosts ; Salaries
TotalCosts ; OtherCosts
NetIncome ; AdminExpenses
NetIncome ; InterestCharges
NetProfit ; Taxes
```

Scenario Hierarchies

Following is the syntax for Scenario dimension hierarchies:

parentmemberlabel;childmemberlabel

This example specifies Scenario dimension hierarchies:

```
!HIERARCHIES=Scenario
;Actual
;Budget
```

Entity Hierarchies

Following is the syntax for Entity dimension hierarchies:

```
parentmemberlabel;childmemberlabel
```

This example specifies Entity dimension hierarchies:

```
!HIERARCHIES=Entity
;[None]
;Regional
Regional;UnitedStates
UnitedStates;California
California;Sunnyvale
California;FosterCity
```

Custom1 Through Custom4 Hierarchies

Following is the syntax for Custom1 through Custom4 dimension hierarchies:

```
parentmemberlabel;childmemberlabel;AggregationWeight
```

This example specifies Custom1 through Custom4 dimension hierarchies:

```
!HIERARCHIES=Custom1
;[None];1
;AllProducts;0
AllProducts;Golf;1
Golf;GolfBalls;1
Golf;GolfShoes;1
Golf;GolfTees;1
Golf;GolfClubs;1
```

Dimensions Not Included in Metadata Files

The Year, Period, and View dimensions are not included in metadata files. You define these dimensions in the application profile that you specify when you define an application.

Value and Intercompany Partner dimension members are mostly system-defined. However, you can define descriptions for Value members, and security classes and descriptions for some members of the Intercompany Partner dimension.

- Value — Standard members are automatically generated. In addition, after you load metadata, Financial Management automatically creates a triplet of Value dimension members for each currency that you loaded: *CurrencyName*, *CurrencyName* Adjs, and *CurrencyName* Total, where *CurrencyName* is the currency label. For example, for a currency of USD, Financial Management creates this triplet of Value dimension members: USD, USD Adjs, and USD Total. You can define descriptions of the system-generated members, as well

as descriptions that will be appended to the Value members that the system creates for user-defined currencies.

- **Intercompany Partner** — This dimension is automatically generated. An Intercompany Partner dimension member is generated for each Entity dimension member whose IsICP attribute is set to TRUE. You can define security classes and descriptions for some Intercompany Partner members as described in [“Intercompany Partner” on page 102](#).

Using Metadata Manager Views

You use Metadata Manager to edit and create metadata files. Metadata includes information relating to dimension member attributes and currencies. For example, you can use Metadata Manager to add accounts to an application.

Note: Metadata Manager is available only in the Financial Management Desktop in the Windows client.

Metadata Manager can open files in extensible mark-up language (XML) and APP file formats. You can use Metadata Manager to create an XML or APP file or to edit an XML or APP file that was extracted from a Financial Management application. Metadata files created in Metadata Manager are automatically encoded with the Unicode format, using Little Endian byte ordering. After you create or edit a metadata file, you can load the metadata contained in the file into an application.

Note: All procedures in this topic assume that you have the Metadata Manager workspace open. To open the workspace, select Manage Metadata from the desktop navigation frame.

You cannot edit metadata directly in an application. When you make changes to metadata in Metadata Manager, you are making changes only to the file, not to the metadata in the application. After you extract and modify the metadata in the file, you must load the metadata back into the application for your changes to take effect.

[Table 19](#) describes the views in Metadata Manager.

Table 19 Metadata Manager Views

Tab	Description
Tree View	Use a hierarchical view to add or modify members. Note: You must use List View for AppSettings, ConsolMethod, and Currencies.
List View	Use a flat list format to add or modify members.
Metadata Item Properties	Use to create an alias for a Custom dimension.
File Properties	Use to create a metadata report.

For most metadata, you can add or modify members in Tree View or List View.

Note: You must use List View for AppSettings, ConsolMethods, and Currencies metadata.

In Tree View, you can add or modify members in a hierarchical view. You add each member as a child or sibling of a member. You enter attributes for each member on the Member Attributes tab. To modify data, select a member and modify it or its attributes. See [“Tree View Tasks” on page 106](#).

In List View, you add and modify members and member attributes in a flat list. To arrange members in a hierarchy, switch to Tree View and drag the members into the hierarchy. See [“List View Tasks” on page 111](#).

Note: Changes made in Tree View are carried over to List View, and vice versa.

Changing the Metadata File Format

You can convert an APP file to an XML file and vice versa in Metadata Manager by saving the file with the desired file extension.

➤ To save an XML file as an APP file:

- 1 Click **Open File**, and select the XML file to convert.
- 2 Click **Save File**.
- 3 From the **Save as Type** drop-down list, select **APP files (*.app)**.
- 4 Modify the file name, and change the file extension to `APP`.
- 5 Click **Save**.

Note: When you open a newly-converted APP file in Metadata Manager, you are prompted to enter the delimiter character. Delimiters are required only for APP files.

Tree View Tasks

All procedures in this topic assume that you have a metadata file open in Metadata Manager. After you make changes to a metadata file, make sure to save the file. See these procedures:

[“Adding and Modifying Members” on page 107](#)

[“Modifying Node Attributes for Custom Dimensions” on page 108](#)

[“Adding Members from the Member List” on page 108](#)

[“Moving Members” on page 108](#)

[“Promoting Members” on page 109](#)

[“Repositioning Members” on page 109](#)

[“Deleting and Removing Members” on page 109](#)

[“Adding Orphaned Members” on page 110](#)

[“Removing Orphaned Members” on page 110](#)

[“Expanding and Collapsing the Hierarchy” on page 111](#)

Adding and Modifying Members

When you add a member to a new hierarchy, you add it as a child of the top member. For example, if you are creating a metadata file and want to set up accounts, the top member, by default, is named Account. You can add child accounts only to Account. Note that you can rename the top member by right-clicking it.

Note: If a new member is a child and the parent exists more than once in the tree, the new child is added as a child of all instances of the parent.

When you add a member to or select a member from Tree View, the member attributes are displayed in the right side of the Metadata Manager workspace.

The right side of Tree View contains the additional tabs described in [Table 20](#).

Table 20 Tree View Tabs

Tab	Description
Member Attributes	Use to display, edit, and enter attributes for the selected member.
Member List	Use to drag members from the list to the hierarchy.
Node Attributes	Use for Custom dimensions. This tab contains the attributes applicable to the relationship between the currently selected Custom dimension member and its parent.

➤ To add or modify members:

1 Make sure that the **Tree View** tab is selected and, from the **Metadata Item** drop-down list, select a dimension.

2 Do one of these tasks:

- To add members, select a member, and click  to add a child to the selected member, or click  to add a sibling to the selected member.
- To modify members, select a member.

Note: You can add children only to the top member in the hierarchy.

3 On the **Member Attributes** tab, enter or modify attributes for the member.

4 Repeat steps 2 and 3 until you have added or updated all members.

Modifying Node Attributes for Custom Dimensions

By changing the aggregation weight attribute on the Node Attributes tab, you can modify the relationship between the selected Custom member and its parent. See [“Defining Custom Members” on page 80](#).

The aggregation weight for the Custom dimension can be any value (positive or negative), and fractions are allowed (for example, 1.5 is valid). The default values are 0 (no aggregation) and 1 (to aggregate). If the value is not 0 or 1, then the child member is aggregated to the parent using that multiplier.

► To modify node attributes:

- 1 Make sure that the **Tree View** tab is selected and, from the **Metadata Item** drop-down list, select a Custom dimension.
- 2 In the hierarchy, highlight a Custom member.
- 3 On the right side of the Metadata Manager workspace, select the **Node Attributes** tab.
- 4 Modify the AggrWeight attribute.

Adding Members from the Member List

You can add members to the hierarchy by dragging members from the Member List tab and dropping them into the hierarchy structure. If you add a child member to a parent member and the parent exists more than once in the hierarchy, the child member is added as a child member for all instances of the parent member. You can add multiple members simultaneously.

► To add members from the Member List tab:

- 1 Make sure that the **Tree View** tab is selected and, from the **Metadata Item** drop-down list, select a dimension.
- 2 From the **Member List** tab, select the members to add.

Note: You can select multiple members by holding down the Ctrl key as you select each member.

- 3 Drag the selected members to the hierarchy.

Note: The members are not removed from the Member List tab; they are only copied into the hierarchy.

Moving Members

You can move members in the hierarchy by dragging the members. You can move multiple members simultaneously.

► To move members:

- 1 Make sure that the **Tree View** tab is selected, from the **Metadata Item** drop-down list, select a dimension.

- 2 In the hierarchy, select a member.

Note: You can select multiple members by holding down the Ctrl key as you select each member.

- 3 Drag the selected members to the new location.

Promoting Members

You can promote members in the hierarchy by using the Promote toolbar button . When you promote a member, it moves it up one level in the hierarchy.

► To promote members:

- 1 Make sure that the **Tree View** tab is selected and, from the **Metadata Item** drop-down list, select a dimension.
- 2 In the hierarchy, select a member to promote.
- 3 Click .

Repositioning Members

You can reposition members in the hierarchy by using the Move Up  and Move Down  toolbar buttons. Repositioning does not promote members to a new level but moves them up or down within their current level in the hierarchy.

► To reposition members in relation to siblings:

- 1 Make sure that the **Tree View** tab is selected, and, from the **Metadata Item** drop-down list, select a dimension.
- 2 In the hierarchy, select a member to move.
- 3 Click  or  to reposition the selected member.

Deleting and Removing Members

You can delete members completely from the metadata, or you can remove members from parents and retain the members in the Member List tab. If you remove a member from its parent and the member has no other parent, it is considered an orphaned member in the Member List tab. Deleting a member completely from the hierarchy also deletes it from List View.

► To delete or remove members:

- 1 Make sure that the **Tree View** tab is selected, and, from the **Metadata Item** drop-down list, select a dimension.
- 2 In the hierarchy, select a member.

3 Right-click and select one of these menu options:

- Remove from parent to remove the member from its parent but retain the member in the Member List tab
- Delete from dimension to delete the member from the dimension

Adding Orphaned Members

An orphaned member is a member that is not part of the hierarchy and, therefore, does not have a parent or sibling member. When you add members in List View, the members are orphaned until you add them to the hierarchy in Tree View.

Caution! A metadata file that contains orphaned members cannot be scanned or loaded into an application.

► To add orphaned members to the hierarchy:

- 1 Make sure that the **Tree View** tab is selected, and, from the **Metadata Item** drop-down list, select a dimension.
- 2 On the right side of the Metadata Manager workspace, select the **Member List** tab.
- 3 Select **Only Show Orphaned Members** to view all orphaned members of the dimension.
- 4 Highlight members and drag them to a new position in the hierarchy.

Removing Orphaned Members

If you select to show only orphaned members, you can easily remove orphaned members from the member list.

Caution! A metadata file that contains orphaned members cannot be scanned or loaded into an application.

► To remove orphaned members:

- 1 Make sure that the **Tree View** tab is selected, and, from the **Metadata Item** drop-down list, select a dimension.
- 2 On the right side of the Metadata Manager workspace, select the **Member List** tab.
- 3 Select **Only Show Orphaned Members** to display the orphaned members of the selected dimension.
- 4 Highlight one or more members, right-click, and select **Remove Highlighted Members**.

Note: To remove all orphaned members, right-click, and select **Select All**.

Expanding and Collapsing the Hierarchy

You can expand or collapse the hierarchy by using the expand and collapse toolbar buttons. You can expand or collapse the entire hierarchy, or you can expand or collapse individual parent members.

► To expand or collapse the tree:

- 1 Make sure that the **Tree View** tab is selected, and, from the **Metadata Item** drop-down list, select a dimension.
- 2 In the hierarchy, select a member, and do one of these tasks:
 - Click  to expand the current member and all members below it in the hierarchy.
 - Click  to collapse the current member and all members below it in the hierarchy.

List View Tasks

All procedures in this topic assume that you have a metadata file open in Metadata Manager. After you make a change to a metadata file, make sure to save the file.

Note: After you use List View to add members, use Tree View to add the members to the hierarchy.

See these procedures:

[“Adding and Modifying Members” on page 111](#)

[“Copying, Cutting, and Pasting Members” on page 112](#)

[“Deleting Members” on page 112](#)

[“Validating Metadata” on page 112](#)

[“Showing or Hiding Columns” on page 113](#)

Adding and Modifying Members

When you use List View to add members, you can enter the member and its attributes in a flat list. The information that you add or modify is not validated until you change to a different view (for example, Tree View) or manually validate the data.

► To add or modify members:

- 1 Make sure that the **List View** tab is selected, and, from the **Metadata Item** drop-down list, select a dimension.
- 2 Take one of these actions:
 - To modify a member, select the member and modify it or its attributes.

- To add a member, click  and, in the new line, enter the member name and attributes.

3 Repeat step 2 until you have completed all member modifications and additions.

Copying, Cutting, and Pasting Members

You can cut, copy, and paste members from one cell to another cell or from multiple cells to multiple cells. You can also disable drop-down list cells. Disabling drop-down list cells makes it easier to copy, cut, and paste multiple cells. Changes made in List View are reflected in Tree View.

► To copy, cut, or paste members, application settings, or currencies:

- 1 Make sure that the **List View** tab is selected, and, from the **Metadata Item** drop-down list, select a dimension.
- 2 **Optional: Select Disable Combo Boxes.**
- 3 Select a cell or multiple, contiguous cells.

Note: To select an entire row or column, select the row number or the column header.

4 Do one of these tasks:

- Click  to cut the information in a cell and store it on the clipboard.
- Click  to copy the information in a cell and store it on the clipboard.
- Click  to paste information from the clipboard to the selected cell.

Deleting Members

When you delete a member you must also delete all attributes associated with the member. Therefore, to delete a member, you must select the entire member row. Changes made in List View are reflected in Tree View.

► To delete members:

- 1 Make sure that the **List View** tab is selected, and, from the **Metadata Item** drop-down list, select a dimension.
- 2 Highlight one or more rows by clicking the row numbers.
- 3 Click .

Validating Metadata

Changes that you make to members are not validated until you change to a different view (for example, Tree View) or until you select the validate toolbar button. If an error is found during the validation process, the system displays the row number where the error occurred.

Showing or Hiding Columns

You can specify the columns of information to be displayed by showing or hiding columns.

➤ To hide a single column:

- 1 Make sure that the **List View** tab is selected, and, from the **Metadata Item** drop-down list, select a dimension.
- 2 Right-click a column, and select **Hide Current Column**.

➤ To show or hide multiple columns:

- 1 Make sure that the **List View** tab is selected, and, from the **Metadata Item** drop-down list, select a dimension.
- 2 Right-click in the grid, and select **Show/Hide Columns**.
- 3 Select the columns to view.
- 4 Click **OK**.

Sorting Metadata in List View

You can sort metadata in List View by column. You can sort column information in ascending or descending order.

➤ To sort a column:

- 1 Make sure that the **List View** tab is selected, and, from the **Metadata Item** drop-down list, select a dimension.
- 2 Double-click a column header to sort the column information.

Tip: To sort the column in reverse order, double-click a column header again.

Creating Aliases for Custom Dimensions

All procedures in this topic assume that you have a metadata file open in Metadata Manager. After you make changes to a metadata file, make sure to save the file.

You use the Metadata Item Properties tab to enter aliases for Custom dimensions. An alias describes a Custom dimension.

Note: You can use aliases only for custom dimensions.

➤ To create a dimension alias:

- 1 Make sure that the **Metadata Item Properties** tab is selected, and, from the **Metadata Item** drop-down list, select a Custom dimension.

- 2 In the **Alias** text box, enter an alias for the dimension.

Creating Metadata Reports in File Properties

You can apply an XSL style sheet to transform metadata into HTML format for easier viewing on the Web.

Financial Management provides two default report XSL style sheets for metadata reports. For example, for the report to display metadata in hierarchies, use the `HFM_MetadataWithHierarchy.XSL` style sheet.

The style sheets are installed in the Server Working Folder during the installation process. They are located by default in the `Report Style Sheets\Metadata` folder in the directory to which you installed Financial Management. To create your own style sheets instead of using the ones provided, contact your IT administrator for support with XSL.

► To create a metadata report:

- 1 In Metadata Manager, select the **File Properties** tab, and open a metadata file.

Note: By default, metadata files use the XML or APP file extension.

- 2 **Optional:** Click **View XML** to view the metadata before generating the report.
- 3 In the **XSL Style Sheet Filename** text box, enter the style sheet name to apply to the metadata file; or click  to locate a file.

Note: By default, metadata report style sheets use the XSL file extension.

- 4 Click **View Metadata** to view the formatted report.

Metadata Referential Integrity

To prevent a referential integrity problem from occurring in the application, Financial Management verifies that metadata changes are valid to the application in its current state before accepting the changes.

When you load metadata, the system compares the metadata load file with the metadata elements in the application. All changes are recorded, and some changes are checked against existing data. Modifications that cause referential integrity problems are not allowed.

Note: When you load metadata, make sure that the Check Integrity option is selected.

Metadata Referential Integrity Checks

The log file provides information about specific changes in metadata attributes that require the system to check existing data in regard to the metadata file that you are loading. Only metadata attributes that have an effect on referential integrity are listed in the table.

The system also checks for invalid points of view between the load file and the metadata in the application. If a dimension member is not in the load file but exists in a journal in the application, the metadata load is prevented.

Metadata Log File Referential Integrity Errors

In the metadata log file, referential integrity errors are displayed under this section:

Metadata referential integrity check started at

Each line in the referential integrity check section refers to a metadata integrity error in the load file. Errors found during the integrity check are displayed in this format:

```
Journals::SINGLECA1 Scenario::Actual Year::2009
Value::[Contribution Adjs]
Period::January has 1 occurrences of
Changed::[SCENARIO::Actual::ZeroViewForAdj: Periodic to YTD]
```

This example shows that the metadata integrity error occurs in the SINGLECA1 journal with this point of view: Scenario Actual, Year 2009, Value [Contribution Adjs], Period January. The error is that the ZeroViewForAdj attribute for the Actual scenario was changed from Periodic to YTD. This change is not allowed because a journal exists for the Actual scenario.

Loading Metadata

When you load a metadata file, Financial Management replaces metadata with new metadata from the load file. Replacement is useful for making minor changes to the metadata, such as adding an account. For example, if your application includes a North America entity and you load entities from a metadata file, the attributes for the North America entity in the file replace the attributes for the North America entity in the application.

Note: Do not use the ampersand (&) character in a metadata file. If you do, an error occurs.

Loading large metadata files on the Web can result in a proxy timeout error. If this occurs, increase the Web proxy timeout setting or use the Win32 client to load the file.

When you load metadata files, the system waits for other tasks such as consolidation, data entry or other load processes to finish before proceeding to load the files. Oracle recommends that you load metadata during periods of light activity across the server cluster instead of, for example, during a long-running consolidation. You can check the Running Tasks page to see which consolidations or data loads, for example, are in progress.

After you load a metadata file to an application, users using the application are notified that the system has changed and that they must log off from the application and log back on.

Caution! You must delete orphaned members before loading; if orphaned members are not deleted, metadata is not updated.

Load Options

There are several options for how to load metadata to an application. You must select one type of load method: merge or replace. You have the option to clear all metadata before loading the new metadata and you have the option to check data integrity. [Table 21](#) describes how to use each type of load option.

Table 21 Metadata Load Options

Load Option	Description
Merge	If a dimension member exists in the load file and in the application database, then the member in the database is replaced with the member from the load file. If the database has other dimension members that are not referenced in the load file, the members in the database are unchanged. For example, a database contains entities CT, MA, and CA. You use the merge method to load a metadata file containing new information for CA only. In the database, CA is updated with the new information and MA and CT remain in the database and remain unchanged.
Replace	All dimension members in the application database are deleted and the members from the load file are put into the database. For example, a database contains entities CT, MA, and CA. You use the replace method to load a metadata file containing new information for CA only. In the database, CT and MA are deleted and the only entity will be CA with the new info from the load file.
Clear All Metadata Before Loading	All dimension members and corresponding data, journals, and intercompany transactions in the application database are deleted. Note: If this option is selected it overrides the function of the merge and replace methods.
Check Integrity	Checks the metadata against the data to ensure integrity. See “Metadata Referential Integrity” on page 114 .

Windows Procedure

► To load metadata:

- 1 Open the application.
- 2 From the navigation frame, select **Load Metadata**.
- 3 In the **Metadata Filename** text box, enter the file name to load; or click  to locate the file.

Note: By default, metadata files use the APP or XML file extension. The load process accepts other file extensions such as TXT or CSV, however, Oracle recommends that you use the APP or XML file extension.

You can click View (next to the Metadata Filename text box) to open the metadata file that you selected.

- 4 In the **Log Filename** text box, enter a name for the log file; or click  to find the file.
- 5 If you are loading an APP file, in the **Delimiter Character** text box, enter the character to be used to separate the metadata in the file.

Delimiter characters are necessary only for ASCII files that use the APP file extension. Delimiters are not necessary for XML files. These characters are valid:

, ~ @ \$ % ^ } | : ; \

Note: Use a character that is not used in the file name or in any other way in the file. For example, if you use the comma in an entity description, you cannot use the comma as the delimiter.

- 6 **Optional:** In the **Load Options** section, select **Clear All Metadata Before Loading**.

Note: If you select this option, you lose data or journals that are currently active in the application, and you cannot select elements in the Metadata Options section.

- 7 **Optional:** Select **Check Integrity** to check the metadata file against the metadata in the current application.

It is highly recommended that you select this option as it ensures that the application is not adversely affected by the metadata in the load file.

Note: If integrity errors occur, they are written to the metadata log file, and no portion of the file is loaded into the application.

You must fix the errors before you can continue with this procedure. For information on the metadata integrity errors in the log, see [“Metadata Log File Referential Integrity Errors” on page 115](#).

- 8 In the **Load Method** section, select one of these options:

- Merge
- Replace

- 9 In the **Metadata Options** section, select the types of metadata to load.

Tip: Use the Select All and De-select All buttons to quickly select or clear all metadata types.

- 10 **Optional:** Click **Scan** to verify that the file format is correct.

- 11 Click **Load**.

Note: After you load the metadata file, you can click View (next to the Log Filename text box) to display errors encountered during the load process.

Web Procedure

Note: Oracle recommends that you add Financial Management to the exceptions for your Web pop-up blocker. When you perform some Financial Management tasks on the Web such as loading data, a status window pops up showing the task status. If you have a pop-up blocker enabled on your computer, the status window is not displayed.

► To load metadata:

- 1 Open the application.
- 2 In **Browser View**, expand **Tasks**, and select **Load Tasks**.
- 3 Select **Load Metadata**.

Note: This option is only available if you open a Classic Financial Management application.

- 4 In the **Metadata File** text box, enter the file name to load; or click **Browse** to find the file.

Note: By default, metadata files use the XML or APP file extension. The load process accepts other file extensions such as TXT or CSV, however, Oracle recommends that you use the XML or APP file extension.

- 5 If you are extracting to an APP file, in the **Delimiter Character** text box, enter the character to be used to separate the metadata in the file.

Delimiter characters are necessary only for ASCII files with the APP file extension. Delimiter characters are not necessary for XML files. These characters are valid:

, ~ @ \$ % ^ | : ; ? \

Note: Use a character that is not used in the file name or in any other way in the file. For example, if you use the comma in an entity description, you cannot use the comma as the delimiter.

- 6 **Optional:** In the **Load Options** section, select **Clear All Metadata Before Loading**.

Note: If you select this option, you cannot select elements in the Metadata Options section.

- 7 **Optional:** Select **Check Integrity** to check the metadata file against the data in the current application.

It is highly recommended that you select this option as it ensures that the application is not adversely affected by the metadata in the load file.

Note: If integrity errors occur, they are written to the metadata log file and no portion of the file is loaded into the application.

You must fix the errors before you can continue with this procedure. See [“Metadata Log File Referential Integrity Errors” on page 115](#).

- 8 In the **Load Options** section, select one of these load methods:

- Merge
- Replace

9 In the **Metadata Options** section, select the types of metadata to load.

Tip: Use the Select All and De-select All buttons to quickly select or clear all metadata types.

10 **Optional:** Click **Scan** to verify that the file format is correct.

11 Click **Load**.

Extracting Metadata

When you extract metadata, you save the file as an XML or APP file and specify the file name and location. You can extract metadata to view or modify it in Metadata Manager. After you modify metadata, you must load the modified file into the application for the changes to take effect.

You cannot extract members of system-defined dimensions, such as the Value dimension. In addition, you cannot extract members of dimensions that are defined in application profiles, such as Year and Period.

Windows Procedure

► To extract metadata:

1 Open the application.

2 From the navigation frame, click **Extract Metadata**.

3 In the **Metadata Filename** text box, enter a name for the file to extract, or click  to find the file.

Note: By default, metadata files use the APP or XML file extension.

4 In the **Log Filename** text box, enter a name for the log file, or click  to find the file.

5 In the **Delimiter Character** text box, enter the character to be used to separate the metadata in the file.

Delimiter characters are necessary only for ASCII files using the APP file extension. Delimiter characters are not necessary for XML files. These characters are valid:

, ~ @ \$ % ^ | : ; ? \

Note: Use a character that is not used in the file name or in any other way in the file. For example, if you use the comma in an entity description, you cannot use the comma as the delimiter.

6 Select the types of metadata to extract.

Tip: Use the Select All and De-select All buttons to quickly select or clear all metadata types.

7 Click **Extract**.

Note: After you extract metadata, you can click the View button next to the Log Filename text box to display errors encountered during the process.

Web Procedure

► To extract metadata:

- 1 Open the application.
- 2 In **Browser View**, expand **Tasks**, and select **Extract Tasks**.
- 3 Select **Extract Metadata**.

Note: This option is only available if you open a Classic Financial Management application.

- 4 In the **Delimiter Character** text box, enter the character to be used to separate the metadata in the file.

Delimiter characters are necessary only for ASCII files with the APP file extension. Delimiter characters are not necessary for XML files. These characters are valid:

, ~ @ \$ % ^ | : ; ? \

Note: Use a character that is not used in the file name or in any other way in the file. For example, if you use the comma in an entity description, you cannot use the comma as the delimiter.

- 5 **Optional:** Select **HFM Application Format** to extract the file in the APP format.
- 6 Select the types of metadata to extract.

Tip: Use the Select All and De-select All buttons to quickly select or clear all metadata types.

- 7 Click **Extract**.
- 8 Follow the download instructions displayed in the browser.

The instructions vary depending on the Web browser that you are using. Make sure to save the file in the Web directory that you set up.

5

Managing Member Lists

In This Chapter

Creating Member List Files	121
Loading Member Lists	126
Extracting Member Lists	127
System Lists by Dimension	128

Member lists enable you to specify a segment of information within a dimension. You create member lists in a VBScript file and load the file into your application. Once loaded, the member lists are available to application users. For example, you can select member lists when setting the point of view in data grids or in journals or when copying or clearing data in Database Management. You can create member lists for all Financial Management dimensions.

You can create static and dynamic member lists. Static member lists contain specific dimension members. For example, you can create a static account member list called ProfitAndLoss that includes only these accounts: Sales, Purchases, Salaries, OtherCosts, and TotalCosts. To have additional accounts in the list, you must add the new accounts to the member list file.

Note: You can combine static and dynamic member lists in the same file.

A dynamic member list contains the members that, at runtime, meet the criteria that you specify in the member list file. For the Scenario, Year, Period, and Entity dimensions, you can use the current member in the POV as the starting point for the list. Because member lists are dynamic, changes to the metadata are automatically reflected in the list the next time that the list is called.

Sample member list files are included when you install Sample Applications for Financial Management. The files are located in the Sample Apps folder in the directory to which you installed Financial Management.

Creating Member List Files

You create member list files using VBScript in an ASCII format supporting multibyte character sets (MBCS), or a file encoded with the Unicode format, using Little Endian byte ordering. By default, member list files use the LST file extension.

Note: You cannot use quotation marks in user-defined member list names.

The file format for listing members within a member list file is the same for all dimensions with the exception of Entity. For member lists in the Entity dimension, you must specify the entity and its parent.

You use these subroutines to create member list files:

- Sub [EnumMemberLists](#) (): Specifies which dimensions have member lists, and defines the member lists for each dimension. You define the number of lists for each dimension and the name of each member list within that dimension.
- Sub [EnumMembersInList](#) (): Defines the members within each member list.

For information on adding dynamic member lists to the script, see [“Dynamic Member Lists” on page 124](#).

EnumMemberLists

Within the EnumMemberLists () subroutine, you use the syntax and functions in this table to define member lists:

Syntax	Description
Dim <i>Element</i> Lists(<i>n</i>)	Replace <i>Element</i> with the dimension for which you are creating member lists and replace <i>n</i> with the total number of member lists that you are defining for that dimension. The following syntax states that three member lists are defined for the Entity dimension: Dim EntityLists(3)
HS.Dimension = " <i>Element</i> "	Replace <i>Element</i> with the dimension. For example: If HS.Dimension = "Entity" Then
<i>Element</i> Lists(<i>n</i>) = " <i>ListName</i> "	Replace <i>Element</i> with the dimension, replace <i>n</i> with a unique identifier for the member list, and replace <i>ListName</i> with the member list name to use. You can use the @POV keyword to create a dynamic list based on the dimension member that is currently set in the POV. You can use the @POV keyword in entity member list names. The entities appearing in the entity list can be based on the Scenario, Year, Period and Entity selected in the POV of a report. Note: Dynamic POV member lists are only supported when used in Financial Reporting and Web Analysis For example: EntityLists(1) = "NewEngland" EntityLists(2) = "Alloc" EntityLists(3) = "AllEntities(@POV) "
HS.SetMemberLists <i>Element</i> Lists	Replace <i>Element</i> with the dimension. For example: HS.SetMemberLists EntityLists

EnumMembersInList

Within the EnumMembersInList () subroutine, you use the syntax and functions in this table to define the members of each member list:

Table 22 EnumMembersInList Syntax

Syntax	Description
HS.Dimension = "Element"	Replace <i>Element</i> with the dimension. For example: If HS.Dimension = "Entity" Then
HS.MemberListID	Specifies the member list unique identifier. HS.MemberListID = 1
HS.AddEntityToList HS.AddMemberToList	The member or entity to add to the member list. For Entity lists, use HS.AddEntityToList and list the entity's parent. For all other dimensions, use HS.AddMemberToList. HS.AddEntityToList "UnitedStates", "Maine" HS.AddMemberToList "July"
HS.Entity.List	Specifies an entity member list. HS.Entity.List(" ", "[Base]")
HS.MemberListEntity HS. MemberListScenario HS. MemberListYear HS. MemberListPeriod	Use to specify a dynamic memberlist.

Example

What follows is an example of the EnumMembersInList section of the file. In this example, the entities for three entity lists are defined. The members of the Account list are also defined.

```
Sub EnumMembersInList()
If HS.Dimension = "Entity" Then
    If HS.MemberListID = 1 Then
        HS.AddEntityToList "UnitedStates", "Connecticut"
        HS.AddEntityToList "UnitedStates", "Massachusetts"
        HS.AddEntityToList "UnitedStates", "RhodeIsland"
        HS.AddEntityToList "UnitedStates", "Maine"
    ElseIf HS.MemberListID = 2 Then
        HS.AddEntityToList "UnitedStates", "Connecticut"
    ElseIf HS.MemberListID = 3 Then
        HS.AddEntityToList "UnitedStates", "California"
    End If
ElseIf HS.Dimension = "Account" Then
    If HS.MemberListID = 1 Then
        HS.AddMemberToList "Sales"
        HS.AddMemberToList "Purchases"
        HS.AddMemberToList "Salaries"
        HS.AddMemberToList "OtherCosts"
        HS.AddMemberToList "TotalCosts"
        HS.AddMemberToList "GrossMargin"
        HS.AddMemberToList "HeadCount"
        HS.AddMemberToList "AdminExpenses"
        HS.AddMemberToList "InterestCharges"
        HS.AddMemberToList "NetIncome"
        HS.AddMemberToList "Taxes"
        HS.AddMemberToList "NetProfit"
    End If
End Sub
```

```

End If
End If
End Sub

```

Note: You can have as many member lists for each dimension as you need, and you do not need to create member lists for all dimensions.

Dynamic Member Lists

For dynamic member lists, instead of listing all members of the member list, you enter rules to pull the information from the system. You can use Financial Management functions and arguments to build member lists. Use the Rules Editor to write rules with proper syntax.

This syntax creates a dynamic member list to get all USD entities:

```

If HS.Dimension = "Entity" Then
  If HS.MemberListID=1 Then
    ELi=HS.Entity.List(" "," ")
    'Entities are read into an array.
    For i=Lbound(ELi) to Ubound(ELi)
    'Loops through all entities.
      If (StrComp(HS.Entity.DefCurrency(ELi(i)),
        "USD",vbTextCompare)=0) Then
        HS.AddEntityToList " ",ELi(i)
    'String compares default currency for entity to USD. If there is a match, the entity is
    added to the member list.
      End If
    Next
  End If
End If
End If

```

In this example, the system first gets the list of entities then compares the default currency for the entity value in the array with the desired value of USD. If the value is USD, the entity is added to the list. The system then processes the next entity in the array.

Dynamic POV Member List

Dynamic POV Member lists are created dynamically based on the current POV member of one or more dimensions. Note that these lists can only be used in Oracle Hyperion Financial Reporting, Fusion Edition and Oracle's Hyperion® Web Analysis.

Note: This feature is only applicable for Entity Member lists.

The bold sections of this sample member list file show the dynamic POV sections.

```

Sub EnumMemberLists()
Dim EntityLists(5)
If HS.Dimension = "Entity" Then
  EntityLists(1) = "AllEntities"
  EntityLists(2) = "AppCur"

```

```

EntityLists(3) = "NoAppCur"
EntityLists(4) = "Global(@POV)"
EntityLists(5) = "POWN(@POV)"
HS.SetMemberLists EntityLists
End If
End Sub
Sub EnumMembersInList()
If HS.Dimension = "Entity" Then
  If HS.MemberListID = 1 Then
    HS.AddEntityToList "", "Corp_Ops"
    HS.AddEntityToList "", "China"
    HS.AddEntityToList "", "Colombia"
    HS.AddEntityToList "", "Germany"
    HS.AddEntityToList "", "Spain"
    HS.AddEntityToList "", "UK"
  End If
  EntList=HS.Entity.List("", "")
  AppCur=HS.AppSettings.Currency

  For each Ent in EntList
    If HS.Entity.DefCurrency(Ent)=AppCur Then
      If HS.MemberListID = 2 Then HS.AddEntityToList "", Ent
      ElseIf Ent<>"[None]" Then
        If HS.MemberListID = 3 Then HS.AddEntityToList "", Ent
      End If
    End If
  Next
  ScenPOV=HS.MemberListScenario
  YearPOV=HS.MemberListYear
  PerPOV=HS.MemberListPeriod
  EntPOV=HS.MemberListEntity
  If HS.MemberListID = 4 Or HS.MemberListID = 5 Then
    If ( EntPOV <> "" ) Then
      EntList=HS.Node.List("E#" & EntPOV, "[Descendants]", "S#" & ScenPOV & ".Y#" &
YearPOV & ".P#" & PerPOV)
      If IsArray(EntList) Then
        For each Ent in EntList
          If Ent <> "[None]" Then
            If HS.Node.Method("S#" & ScenPOV & ".Y#" & YearPOV & ".P#" & PerPOV & ".E#"
& EntPOV & "." & Ent)="GLOBAL" Then
              If HS.MemberListID = 4 Then HS.AddEntityToList "", Ent
            End If
            If HS.Node.POwn("S#" & ScenPOV & ".Y#" & YearPOV & ".P#" & PerPOV & ".E#" &
EntPOV & "." & Ent) > 0.5 Then
              If HS.MemberListID = 5 Then HS.AddEntityToList "", Ent
            End If
          End If
        Next
      End If
    End If
  End If
End Sub

```

Loading Member Lists

When you load member list files, the system waits for other tasks such as consolidation, data entry or other load processes to finish before proceeding to load the files. Oracle recommends that you load member lists during periods of light activity across the server cluster instead of, for example, during a long-running consolidation. You can check the Running Tasks page to see which consolidations or data loads, for example, are in progress.

After you load a member list file into an application, users using that application are notified that the system has changed and that they must log off from the application and log back on.

Windows Procedure

► To load member lists:

- 1 Open the application.
- 2 From the navigation frame, select **Load Member Lists**.
- 3 For **Member List Filename**, enter the file name to load, or click  to find the file.

Note: By default, member list files use the LST file extension. The load process accepts other file extensions such as TXT or CSV, however, Oracle recommends that you use the LST file extension.

You can click View (next to Member List Filename) to open the file that you selected.

- 4 For **Log Filename**, enter a log file name, or click  to find the file.
- 5 **Optional:** Click **Scan** to verify that the file format is correct.
- 6 Click **Load**.

Note: After you load the file, you can click the View button next to Log Filename to display errors encountered during the load.

Web Procedure

Note: Oracle recommends that you add Financial Management to the exceptions for your Web pop-up blocker. When you perform some Financial Management tasks on the Web such as loading data, a status window pops up showing the task status. If you have a pop-up blocker enabled on your computer, the status window is not displayed.

► To load member lists:

- 1 Open the application.
- 2 In the **Browser View**, expand **Tasks** and select **Load Tasks**.
- 3 Select **Load Member Lists**.

- 4 For **Member Lists File** , enter the file name to load, or click **Browse** to find the file.

Note: By default, member list files use the LST file extension. The load process accepts other file extensions such as TXT or CSV, however, Oracle recommends that you use the LST file extension.

- 5 **Optional:** Click **Scan** to verify that the file format is correct.

- 6 Click **Load**.

Note: If an invalid member list is loaded, Web users may not be able to access the data grid. If the following error message is displayed, you may need to correct the member list and load the file:

```
"Type Mismatch /hfm/Data/ExploreData.asp. Error Number 13. Error Source: Microsoft  
VBScript runtime error."
```

Extracting Member Lists

Extracted member lists are saved to an ASCII file that supports multibyte character sets (MBCS). By default, member list files use the LST file extension. After you extract member lists, you can view and modify the member list information in a text editor.

Windows Procedure

- To extract member lists:

- 1 Open the application.
- 2 From the navigation frame, select **Extract Member Lists**.
- 3 For **Member List Filename**, enter a file name to extract, or click  to find the file.

Note: By default, member list files use the LST file extension.

- 4 For **Log Filename**, enter the log file name, or click  to find the file.
- 5 Click **Extract**.

Note: After you extract member lists, you can click the View button to display errors encountered during the extract.

Web Procedure

- To extract member lists:

- 1 Open the application.
- 2 In the **Browser View**, expand **Tasks** and select **Extract Tasks**.

3 Select **Extract Member Lists**.

4 Follow the download instructions displayed in the browser.

The instructions vary depending on the Web browser that you are using. Make sure to save the file in the Web directory that you set up.

System Lists by Dimension

The following table lists the name of the system-generated list and the dimensions in which it can be used.

Table 23 System Lists by Dimension

System List	Scenario	Entity	Account	ICP	Custom	Value	Year	Period
[Hierarchy]	X	X	X	X	X	X	X	X
[Descendants]	X	X	X	X	X	X	X	X
[Children]		X	X	X	X			X
[Base]		X	X	X	X			
[Parents]		X						
[Ancestors]		X						
[System]			X	X				
[Currencies]					C1, C2			
[ConsolMethod]					C1			
[Inputs]						X		
[Adjustments]						X		
[Totals]						X		
[Default Currencies]						X		
[First Generation]								X
[Second Generation]								X
[Third Generation]								X
[Fourth Generation]								X
[Fifth Generation]								X
[Sixth Generation]								X

6

Managing Journals

In This Chapter

Creating Journal Files	129
Loading Journals	133
Extracting Journals	135

Many external general ledger systems can generate ASCII text files containing journal information that you can load into a Financial Management application. If necessary, you can edit the ASCII file using a text editor before loading it.

Sample journal files are included when you install Sample Applications for Financial Management. The files are located in the Sample Apps folder in the directory to which you installed Financial Management.

Creating Journal Files

You can create journal files using an ASCII format supporting multibyte character sets (MBCS) or a file encoded with the Unicode format, using Little Endian byte ordering. By default, journal files use the JLF file extension.

A security information file can contain these sections:

- File Format
- Version
- Journal Group
- Standard
- Recurring
- Header - Scenario, Year, Period

A line starting with an exclamation point (!) indicates the beginning of a new section in the journal file, and must be followed by a valid section name (for example, Year). A line starting with an apostrophe (') is considered a comment line and is ignored by the system.

You can use these special characters to separate information within the file as long as the character is not used in the file in another way:

Character	Description
,	comma
~	tilde
@	at sign
#	number sign
\$	dollar sign
%	percent sign
&	ampersand
^	carat
	line
:	colon
;	semicolon
?	question mark
\	back slash

Note: You must use the same delimiter character throughout the file. Using different delimiter characters within the same file causes an error when you load the file.

File Format Section

This file section contains the file version number. This number only changes when changes are made to the file format. The file format is automatically generated when you extract journals.

Note: This section is not required.

This example specifies the file format:

```
!File_Format = 1.0
```

Version Section

This file section contains the Financial Management version that you used to extract journals. The version number is automatically generated when you extract journals.

Note: This section is not required.

This example specifies the version:

```
!Version = 1.1
```

Journal Group Section

This file section uses this syntax to define journal groups.

```
!GROUP=<journal group>;<journal group description>
```

For example, this example defines three journal groups:

```
!GROUP=Group1;This is the Group1 Journals Group
!GROUP=Group2;This is the Group2 Journals Group
!GROUP=Group3;This is the Group3 Journals Group
```

Standard Section

Standard templates apply to all scenarios, years, and periods in an application. They are not dependent on a specific combination of scenario, period, and year.

This syntax specifies a standard template:

```
!STANDARD = <label>, <balancing attribute>, <type>, <journal group>, <securityclass>,
<SingleParent.SingleEntity>
!DESC=<journal description>
<parent.entity>, <ICP>, <C1>, <C2>, <C3>, <C4>, <amount type>, <amount>, <line item desc>
```

Recurring Section

Recurring templates apply to all scenarios, years, and periods in an application. They are not dependent on a specific combination of scenario, period, and year, but are dependent on Value Adjs.

Note: You cannot create a recurring template for an auto reversing template. For the type attribute, the value must be R for regular.

This syntax specifies a recurring template:

```
!RECURRING = <label>, <balancing attribute>, <type>, <value>, <journal group>,
<securityclass>, <SingleParent.SingleEntity>
!DESC=<journal description>
<parent.entity>, <account>, <ICP>, <C1>, <C2>, <C3>, <C4>, <amount type>, <amount>,
<line item desc>
```

Header Section

This file section contains the scenario, year, and period information. The journal type header information and corresponding detail lines follow the Header section. This organizes journal

information according to a specific scenario, year, and period. You can specify multiple Header sections in the journal file.

This syntax specifies the scenario, year, and period:

```
!SCENARIO= Actual
!YEAR = 2009
!PERIOD = July
```

The Header section is followed by journal detail information for Actual, July, 2009.

Journal Subsection

This subsection of the Header section provides journal detail information for the Scenario, Period, and Year specified.

[Table 24](#) describes the attributes that are used in the Journal sub-section. These attributes are used for regular journals and recurring and standard templates.

Table 24 Journal Attribute Descriptions

Attribute	Value
<label>	User-defined label for journal with up to 80 characters
<balancing attribute>	● U = unbalanced ● B = balanced in total ● E = balanced by entity
<type>	● R = regular journals ● A = auto-reversing journals ● V = auto-reversal journal Note: You cannot load system-generated auto reversals, but you can extract them.
<status>	● W = Working ● S = Submitted ● A = Approved ● P = Posted ● R = Rejected
<value dimension>	● [Contribution Adjs] ● [Parent Adjs] ● <Entity Curr Adjs> ● <Parent Curr Adjs>
<journal group>	Optional: User-defined parameter with up to 20 characters Note: Groups need to be preloaded.

Attribute	Value
<security class>	Optional: Valid security class that is associated with the journal Note: If you do not assign a security class, the journal assumes the Default security class. Only users who have access rights to this security class can access the journal.
<singleparent. singleentity>	Optional: Valid parent/entity pair that is used by all the line items in the journal. When you specify a single parent/entity pair for the entire journal, the parent.entity attribute is not used.
<journal description>	Journal description, which can contain up to 255 characters. You can load descriptions with multiple lines, provided each line starts with this syntax: !DESC=
<parent.entity>	Valid member of the Entity dimension. The parent is required only for the Contribution Adjs, Parent Adjs, and ParentCurrency Adjs members of the Value dimension. This attribute is used only if the Single Parent.Single Entity attribute is not used.
<account>	Valid member of the Account dimension. For regular journals, this must be an input account and the account type must be REVENUE, EXPENSE, ASSET, LIABILITY, FLOW, or BALANCE.
<ICP>	Optional: Valid member of the Intercompany Partner dimension. This attribute is optional; however, you must at least specify ICP None.
<C1 - C4>	Optional: Valid members of the Custom1, Custom2, Custom3, and Custom4 dimensions
<amount type>	● D = debit ● C = credit
<amount>	Positive amount regardless if the amount type is debit or credit.
<line item description>	Optional: Description of the specific journal detail, which can contain up to 50 characters

This syntax specifies a regular journal:

```
!JOURNAL = <label>, <balancing attribute>, <type>, <status>, <value dimension>, <journal
group>, <SecurityClass>, <SingleParent.SingleEntity>
!DESC=<journal description>
<parent.entity>, <account>, <ICP>, <C1>, <C2>, <C3>, <C4>, <amount type>, <amount>,
<line item desc>
```

Loading Journals

Journals are loaded using the Replace mode, which clears all data for a particular journal label before loading the new journal data.

You can load working, rejected, submitted, approved, and posted journals as well as standard and recurring journal templates. You must load posted journals to periods that are open. If you load a posted auto-reversing journal, an approved reversal is automatically generated in the next period. You must manually post the generated reversal.

Note: You cannot load automated consolidation journals; they are created by the consolidation process.

If you load large journal files on the Web, you should modify IIS settings to increase the AspBuffering limit to avoid errors.

Windows Procedure

► To load journals:

1 Open the application.

Note: Before you can load journals, you must first open the periods to which to load journals. See “Managing Periods” in the *Oracle Hyperion Financial Management User's Guide*.

2 From the navigation frame, select **Load Journals**.

3 For **Journal Filename**, enter the file name to load, or click to find the file.

Note: By default, journal files use the JLF file extension. The load process accepts other file extensions such as TXT or CSV, however, Oracle recommends that you use the JLF file extension.

4 For **Log Filename**, enter a log file name, or click to find the file.

5 For **Delimiter Character**, enter the character used to separate the journal data in the file. These characters are valid:

, ~ @ \$ % & ^ | : ; ? \

Note: Use a character that is not used in the file name or in any other way in the file. For example, if you use the comma in an entity description, you cannot use the comma as the delimiter.

6 Click **Load**.

Note: After you load journals, you can click the View button next to Log Filename to display errors encountered during the load.

Web Procedure

Note: Oracle recommends that you add Financial Management to the exceptions for your Web pop-up blocker. When you perform some Financial Management tasks on the Web such as loading data, a status window pops up showing the task status. If you have a pop-up blocker enabled on your computer, the status window is not displayed.

► To load journals:

1 Open the application.

2 In the **Browser View**, expand **Tasks** and select **Load Tasks**.

3 Select Load Journals.

Note: Before you can load journals, you must first open the periods to which to load journals. See “Managing Periods” in the *Oracle Hyperion Financial Management User's Guide*.

4 For Journals File, enter the file name to load, or click **Browse** to find the file.

Note: By default, journal files use the JLF file extension. The load process accepts other file extensions such as TXT or CSV, however, Oracle recommends that you use the JLF file extension.

5 Click Load.

Note: The system displays errors encountered during the load.

Extracting Journals

You can extract journals, recurring journals, and journal templates from an application. You can select the Scenario, Year, Period, Entity, and Value dimensions for which to extract journals. You can select the journal status, journal type, and journal balance type to extract.

When you extract journals, they are saved to an ASCII file that supports multibyte character sets (MBCS). By default, journal files use the JLF file extension. After you extract journals, you can view and modify the journal information in a text editor.

You can extract automated consolidation journals to external systems, however you cannot re-import them into Financial Management. When you extract these journals, the Balance type is blank for automated consolidation journals.

Windows Procedure

► To extract journals:

1 Open the application.

2 From the navigation frame, select **Extract Journals**.

3 For **Journal Filename**, enter a file name to extract, or click to find the file.

Note: By default, journal files use the JLF file extension.

4 For **Log Filename**, enter a log file name, or click to find the file.

5 Select items to extract:

- Templates
- Recurring Templates
- Journals

Note: If you selected Extract Journals, you must specify the Scenario, Year, and Period from which to extract journals. You can extract journals from a specific period or from all periods.

- 6 For Delimiter Character, enter the character used to separate the journals data in the file. These characters are valid:**

, ~ @ \$ % & ^ | : ; ? \

Note: Use a character that is not used in the file name or in any other way in the file. For example, if you use the comma in an entity description, you cannot use the comma as the delimiter.

- 7 Click Extract.**

Note: After you extract journals, you can click the View button to display errors encountered during the extract.

Web Procedure

► To extract journals:

- 1 Open the application.**
- 2 In the Browser View, expand Tasks and select Extract Tasks.**
- 3 Select Extract Journals.**
- 4 Select items to extract:**
 - Templates
 - Recurring Templates
 - Journals
- 5 If you chose to extract Journals, you must specify a Scenario, Year, and one or more Periods from which to extract journals.**
- 6 Optional: Select one or more Entity and Value dimensions to extract.**

To extract automated consolidation journals, you must select [Proportion} and/or [Elimination] as the Value member.

Note: For the Period and Entity dimensions, if you do not select specific members, the system assumes that you want to extract all members for the dimension. However, if you specifically select members, the system displays a plus sign (+) next to the dimension to indicate multiple selections.

- 7 Optional: Enter the Label and Group for the journals to extract.**

You can use the percent sign (%) as a wildcard.

For automated consolidation journals, in the Label field, you can query on the Nature value of the audit transaction that you specified in the consolidation rule. You cannot use a number for the label.

8 Select the check box next to each journal status, journal type, and journal balance type to extract.

9 For **Delimiter Character**, enter the character used to separate the journals data in the file. These characters are valid:

, ~ @ \$ % & ^ | : ; ? \

Note: Use a character that is not used in the file name or in any other way in the file. For example, if you use the comma in an entity description, you cannot use the comma as the delimiter.

10 Click **Extract**.

11 Follow the download instructions displayed in the browser.

The instructions vary depending on the Web browser that you are using. Make sure to save the file in the Web directory that you set up.

7

Managing Data Forms

In This Chapter

Creating Data Forms in the Form Builder	139
Data Form Script Syntax	144
Using Relative Periods and Years	171
Order of Precedence for Conflicting Attributes	171
Editing Data Forms	172
Creating Data Form Folders	173
Viewing and Hiding Private Data Forms	174
Loading Data Forms	174
Extracting Data Forms	175
Deleting Data Forms	175

You create data forms for users who need to enter specific data. You can create data forms in two ways:

- Using the Form Builder integrated script editing tool
- Writing a script in a text file

When you create a data form, you are customizing the data that is displayed. By default, data forms use the WDF file extension.

Note: To load, extract, and delete Web forms, you must be assigned to the Manage Data Entry Forms role.

Creating Data Forms in the Form Builder

Financial Management provides a graphical user interface for creating and maintaining data forms.

For information on the tabs in the Form Builder, see these sections:

- “Properties” on page 140
- “POV” on page 141
- “Columns” on page 142

- “Rows” on page 142
- “Details” on page 143
- “Headers” on page 143
- “Script” on page 143

Some of these tabs provide links to the Member Selector, which you can use to select members and member lists for the form.

► To create a data form:

- 1 Open the application.
- 2 Select **Administration**, then **Manage Documents**.
- 3 Select the **Data Entry Forms** tab.
- 4 Click **New**.
- 5 Enter the form information on each tab of the Form Builder. You can move between tabs by clicking the tab or by clicking **Back** and **Next** at the bottom of the Form Builder. Each tab also contains these buttons:

- **Scan** - to scan the form for proper syntax.

After the system scans the form, it displays a page containing information such as the scan start and end times.

- If the form is valid, the page includes a **Preview** button, which you can use to view the form.
- If the form contains errors, the page lists the errors.

- **Update** - to post changes to the server.
- **Reset** - to reset form values. This button can be useful if you make a mistake.

- 6 Click **Save** at the bottom of the Form Builder, then select a file name and location.

After the system attempts to save the form, it displays a page containing information such as the start and end times of the save operation.

- If the form is valid, the system saves the form and the page includes a **Preview** button, which you can use to view the form.
- If the form contains errors, the form is not saved and the page lists the errors.

Note: If you attempt to close the form without saving the script, the system warns you and provides the option to save before continuing.

Properties

The Properties tab specifies general information for the data form.

Table 25 Properties Tab of Form Builder

Text Box	Description
Label	Form name. See ReportLabel .
Security Class	Form security class. See ReportSecurityClass .
Description	Form description. See ReportDescription .
Instructions	Instructions that administrators provide for form users. See Instructions .

If you hover over a label, a tooltip displays the script syntax.

POV

The POV tab specifies the Background and Selectable POV:

- A Background POV for the form specifies the initial members for dimensions. Use the Member text boxes to specify the Background POV members. Members specified in the Member text boxes are added to the `BackgroundPOV` keyword, which is described in [“BackgroundPOV” on page 147](#).
- A Selectable POV for the form consists of the dimensions for which users can select members. You use the Selectable text boxes to specify the member lists from which users can select members. Members specified in the Selectable text boxes are added to the `SelectablePOVList` keyword. For information on the keyword and the syntax used for member lists, see [“SelectablePOVList” on page 164](#).

You can specify a Background or Selectable POV for a dimension in two ways:

- Type a member label or the syntax for a member list in the applicable text box.
- Click  for the applicable text box to select members and member lists with the Member Selector.

These rules apply to using the POV tab:

- If Member and Selectable are blank and the dimension is not used in a row or column, the system uses the member from the User POV for the initial value for the dimension.
- If you specify values in a Background and Selectable POV for a dimension, and the member for the Background POV is not in the member list for the Selectable POV, the system uses the first member in the member list of the Selectable POV as the Background POV member.
- Users need full access to the member hierarchy to be able to work with all the members in the hierarchy. For example, if you want users to access all Custom 4 members, you must enable access to the parent entity, in this case, Custom4Top. For information on defining access, see [Chapter 4, “Managing Metadata”](#).

Columns

The Columns tab specifies the dimension members to display in the columns of the form. Each text box in the Col section represents a column. You can specify the members of a column in two ways:

- Type the member label in the text box for the column.
- Click  for the column to select members with the Member Selector.

If you have filled in all of the currently visible column text boxes, you can add a new text box by pressing the Tab key. You can define a maximum of 24 columns in the Columns tab.

Note: If a form requires more than 24 columns, you can use the Scripts tab to specify the additional columns. For information on the syntax to use, see [“Cn” on page 149](#).

Dimension elements specified in the column section override elements that are set in the Background POV or Selectable POV.

Note: You can specify one member list per column.

As you enter information in each column, the Options section for that column is available at the bottom of the Form Builder.

If you hover over a label, a tooltip displays the script syntax. For valid values for each option, see [“Data Form Script Syntax” on page 144](#).

Note: You can also use Other to specify syntax for column keywords not exposed in the Options section, such as Blank and ReadOnly. In addition, if the definition of a column in the script contains unexposed syntax, or deprecated or invalid syntax, that syntax is displayed in Other.

Rows

The Rows tab specifies the dimensions to display in the rows of a form. Each text box in the Row section represents a row. You can specify the members of a row in two ways:

- Type the member label or member list syntax in the text box for the row.
- Click  for the rows text box to select members and member lists with the Member Selector.

If you have filled in all of the currently visible Row text boxes, you can add a new text box by press the Tab key. You can define a maximum of 100 rows in the Rows tab.

Note: If a form requires more than 100 rows, you can use the Scripts tab to specify the additional rows. For information on the syntax to use, see [“Rn” on page 159](#).

Dimension elements specified in the row section override elements set in the Background POV or Selectable POV.

As you enter information in each row, the options section for that row is available at the bottom of the Form Builder.

If you hover over a label, a tooltip displays the script syntax. For information on the valid values for each option, see [“Data Form Script Syntax” on page 144](#).

Note: You can also use Other to specify syntax for row keywords not exposed in the Options section, such as Blank and ReadOnly. In addition, if the definition of a row in the script contains unexposed syntax, or deprecated or invalid syntax, that syntax is displayed in Other.

Details

The Details tab specifies grid, print, display, and suppression options for the Data Entry form.

Note: The items in the Details tab are only updated in the script if you change the default settings. If you leave the default settings unchanged, the keywords for these items do not display in the script.

If you hover over a label, a tooltip displays the script syntax. For valid values for each option, see [“Data Form Script Syntax” on page 144](#).

Headers

The Headers tab enables you to specify the maximum length of the labels or descriptions for members and member lists. You can specify different lengths for different dimensions.

The Headers tab also contains the Other text box, which has two uses:

- If syntax is added for the `HeaderOption` keyword but the syntax is not exposed through a new control in the Headers tab, you can specify that syntax in Other.
- If the form contains invalid syntax for the `HeaderOption` keyword for a dimension, the Other text box displays that syntax.

Script

The Script tab enables you to view, modify, and print the script. You enter the syntax in the text box on the Script tab. The Script tab also contains these buttons:

- **Print** - opens a new browser window that contains a printer-friendly version of the syntax.
- **More** - increases the height of the text box that contains the syntax. When you click More, the button is replaced by the Less button.

- **Less** - decreases the height of the text box that contains the syntax. When you click Less, the button is replaced by the More button.

Data Form Script Syntax

You can use three types of syntax elements when creating data form scripts: keywords, values, and options. Keywords are on a line of their own in the script and are placed to the left of the equal sign. Values are placed immediately after the equal sign to complete a line in the script. Options can be added to a line of script where each is delimited by a comma.

Note: When a value is required, it must come before options. Options are never required and can be in any order.

Sample data form scripts are installed with Sample Applications for Financial Management. The files are located in the Sample Apps folder in the directory to which you installed Financial Management.

Note: Data entry form script elements are not case-sensitive.

Table 26 Data Form Script Syntax

Script Syntax	Description
AddMember	Use in a row definition to allow the user to add data for a member that previously had no data or contained zeros and was suppressed from the form. The option adds an icon to the form that, when clicked, enables the user to select members to add to the form.
BackgroundPOV	Use to specify the background dimension members for the form.
Blank	Use to insert a blank row, column, or cell into the form.
Cn	Use to define each column in a form.
CalcByRow	Use to specify if the row calculation is used when a cell has an intersecting column calculation.
Cell_Link	Use with Link in a row definition to link to another data entry form.
CellText	Use to display cell text for a row, calculation, or cell.

Script Syntax	Description
CustomHeader	Use to specify custom headers for rows and columns. Use in a row or column definition. Note: You cannot use these keywords for Custom Headers: ● <code><pre></code> ● <code><textarea></code> ● <code><script></code> ● <code><javascript></code> ● <code><jscript></code> ● <code><vbs></code> ● <code><vbscript></code> ● strings such as <code><XonX=X></code>, where X = any string
CustomHeaderStyle	Use to assign custom style attributes to a row or column header.
DynamicPOV	Deprecated. Do not use.
FormInputBoxLength	Use to specify the input box width on the form.
FormNumDecimals	Use to specify the number of decimal spaces for the form. This keyword overrides the decimal settings for the cell currency. Use NumDecimals to override this setting for a row, column, or cell.
FormRowHeight	Use to specify the height of all rows in the form.
FormScale	Use to specify the scale for the form.
HeaderOption	Use to specify how dimension headers display in the form. Show labels and/or descriptions, set style attributes, set maximum or fixed width.
Instructions	Use to create instructions in HTML-formatted text and links.
LineItemDetailSinglePeriod	Use to specify if line item detail displays for the selected cell only or for all periods.
Link	Use with Cell_Link to link to another data entry form.
MaxCells	Use to specify the maximum number of cells for a data form.
MaxColsForSparseRetrievalMethod	Use to optimize performance of sparse forms. Use with forms containing more than 10 columns.
NoSuppress	Use to turn off suppression for one or more rows or columns. This setting overrides other suppression settings in the form: SuppressInvalidRows , SuppressNoDataRows , SuppressZeroRows , SuppressInvalidColumns , SuppressNoDataColumns , SuppressZeroColumns .
NumDecimals	Use to specify the number of decimal places for a row, column, or cell. This keyword overrides the decimal settings for the cell currency and the decimal setting for the form FormNumDecimals .
Override	Use to specify a different POV or calculation for one or more rows or columns or to add style attributes or set the scale. Use in a row or column definition.
PrintNumDataColsPerPage	Use to specify the number of columns to print on each page.
PrintNumRowsPerPage	Use to specify the number of rows to print on each page.
PrintRepeatHeadersonAllPages	Use to print headers on each page.

Script Syntax	Description
Rn	Use to define each row in a form.
ReadOnly	Use to specify read only rows, columns, or cells.
ReportDescription	Use to specify the description for the form. The description cannot contain an ampersand (&).
ReportLabel	Use to specify the label for the form. These characters are not supported for Data Form labels: Period (.), colon (:), semi-colon (;), question mark (?), forward slash (/), backward slash (\), pipe (), open and close curly brackets ({}), open and close parentheses (), double quotation marks ("), less than and greater than (< >), ampersand (&), plus (+), pound (#), asterisk (*), and trailing underscore (_).
ReportSecurityClass	Use to specify the security class for the form.
ReportType	Use to set the form type. The value must be set to WebForm.
SCalc	Use to specify server-side calculations for a row, column, or cell.
Scale	Use to specify scale for a row, column, or cell. This setting overrides the form scale setting. See FormScale .
SelectablePOVList	Use to specify the selectable dimension members in the form.
ShowDescriptions	Use to show descriptions for dimension members.
ShowLabels	Use to show labels for dimension members.
String	Use to add a text string to a column, row, or cell.
Style	Use to specify the style attributes for a row, column, cell, or dimension header.
SuppressColHeaderRepeats	Use to prevent repeated column headers from displaying.
SuppressInvalidColumns	Use to prevent invalid cells in from displaying in columns.
SuppressInvalidRows	Use to prevent invalid cells from displaying in rows.
SuppressNoDataColumns	Use to prevent columns with no data from displaying.
SuppressNoDataRows	Use to prevent rows with no data from displaying.
SuppressRowHeaderRepeats	Use to prevent repeated row headers from displaying.
SuppressZeroColumns	Use to prevent columns with zeros from displaying.
SuppressZeroRows	Use to prevent rows with zeros from displaying.

AddMember

Use this option within a Row definition to add an icon to the form which, when clicked, enables users to select members to add to the form. The new members are added to the form and the user can then enter data for these members.

You can use the AddMember option with the Account, ICP, and Custom1 through Custom4 dimensions.

Note: NoData cells can be suppressed by using the [SuppressNoDataRows](#) keyword or selecting the applicable check box on the form.

Syntax

`AddMember:MemberList`

Replace *MemberList* with the list name from which users will be able to add members to the form.

Caution! You can reference only one member list per AddMember line.

To use this option, two separate row definitions are needed:

- A summary row that displays totals for the member list and has the AddMember option.

Note: The summary row cannot be suppressed.

- A list row that has the same POV specified in the summary row, except that it has a list for the dimension to which members are being added. The list must contain at least the members from the summary row list.

Note: The list row can be placed before or after the summary row.

Example

```
R1=A#SalesInterco.I{[Base]}
R2=A#SalesInterco.I#[ICP Entities],
AddMember:I{[Base]}
```

In this example, Row 1 specifies the [Base] member list for the Intercompany Partner dimension. Row 2 specifies the total for the member list and the member list to be used with the AddMember icon.

Note: You can use the AddMember option in multiple rows in a form, but each summary row needs its own list row.

BackgroundPOV

Use this keyword to specify the background dimension members for the form. Dimensions not specified in the Background POV are considered dynamic and are taken from the user's point of view when the form is opened.

Users need full access to the member hierarchy to be able to work with all the members in the hierarchy. For example, if you want users to access all Custom 4 members, you must enable

access to the parent entity, in this case, Custom4Top. For information on defining access, see [Chapter 4, “Managing Metadata”](#).

Note: If a member in the Background POV is not valid based on the [SelectablePOVList](#) for the dimension, the system defaults to the first member of the list.

Syntax

BackgroundPOV=S#Scenario.W#View.E#Parent.Entity.V#Value.A#Account.I#ICP.
C1#Custom1.C2#Custom2.C3#Custom3.C4#Custom4

Table 27 Syntax for BackgroundPOV Keyword

Parameter	Description
Scenario	Name of a valid Scenario member.
View	A valid view.
Parent	Name of a valid Parent member. This parameter is optional.
Entity	Name of a valid Entity member.
Value	Name of a valid Value member.
Account	Name of a valid Account member.
ICP	Name of a valid ICP member.
Custom1, Custom2, Custom3, Custom4	Name of valid Custom1 through Custom4 members.

Note: You do not need to specify all dimension members in the Background POV. For dimensions for which you do not specify a member, the dimension member from the user’s point of view is used.

Example

BackgroundPOV=S#Actual.Y#2009.P#April.W#Periodic.V#<Entity
Currency>.A#Sales.I#[ICPNone].C4#[None]

Blank

Use this option to specify a blank row, column or cell in the form. Use this option within a row or column definition or within a cell override definition. The blank row, column, or cell is empty and is read only, and the context menu is disabled on the cell because it contains no data or POV. The Blank option is considered a server-side calculation, therefore it can be used anywhere that SCalc is used.

Example

C4=Blank

Cn

Use this keyword to define a column in the form. The keywords such as C1, C2, C3 provide the definition of each column in the specified order. The column identifier must begin with 1 and proceed in sequential order.

Dimension elements specified for the column override elements that are set in the [BackgroundPOV](#) or [SelectablePOVList](#). You can use member and system lists in column definitions.

Note: One list is allowed per column.

You can use these values and options within a column definition:

- [Blank](#)
- [Calc1](#)
- [CellText](#)
- [CustomHeader](#)
- [NumDecimals](#)
- [Override](#)
- [ReadOnly](#)
- [SCalc](#)
- [Scale](#)
- [String](#)
- [Style](#)

Syntax

Cn=CalcExpression
Cn=POVExpression

Table 28 Syntax for Columns Keyword

Parameter	Description
n	The column number.
CalcExpression	Use SCalc, Blank, or String. See “SCalc” on page 162 , “Blank” on page 148 , and “String” on page 166 .
POVExpression	A valid dimension intersection or member list.

Example

```
C1=S#Actual.P#July,CustomHeader:Actual_July
C2=S#Actual.P#August,(Override 2,3, P#July)
C3=S#Budget.P#September
C4=SCalc(col(1)+col(3)),numdecimals:4,scale:1,readonly
```

```
C5=SCalc(c1+c3),numdecimals:2,CustomHeader:SCalc(c1+c3)
C6=Blank
C7=S#Budget.P#October,Style:font-weight:bold
C8=C1{TotalProducts.[Hierarchy]}
```

Calc1

This value is deprecated. Use [SCalc](#).

CalcByRow

A keyword used to specify if row calculations are used when a cell has an intersecting column calculation. This keyword applies to the entire form.

Syntax

```
CalcByRow=Boolean
```

Where *Boolean* is True if row calculations are used; False if column calculations are used.

Example

```
ReportType=WebForm
ReportLabel=CalcByCol
ReportDescription=Demonstrate CalcByRow
BackgroundPOV=S#Actual.Y#2009.P#January.w#<Scenario
View>.E#UnitedStates.Connecticut.V#USD.A#Sales.I#[ICP
NONE].C1#Golfballs.C2#Customer2.C3#[None].C4#[None]
C1=S#Actual
C2=S#Budget
C3=SCalc(Col(2)-Col(1)),CustomHeader:Variance
C4=SCalc(Col(3)/Col(1)),CustomHeader:Variance %
R1=A#Sales
R2=A#Purchases
R3=A#OtherCosts
R4=SCalc(Row(1)+Row(2)+Row(3)),CustomHeader:Total
ShowLabels=True
CalcByRow=False
```

Cell_Link

Use this option to specify a link to another data form. Use with [Link](#). Links are references in row definitions. You can specify up to 64 links (Link1 - Link64). Links do not need to be numbered sequentially.

Note: Linked form names are case-sensitive.

Syntax

```
Cell_Linkx
```

`Linkx=FormName`

Replace *x* with the number to assign to the link and replace *FormName* with the form name to which to link.

Example

```
R1=A#Salaries, Cell_Link1
Link1=Dynamic
```

CellText

Use this option to display cell text for a row, column, or cell. The first 69 characters of the cell text entry are considered the title of the entry.

Note: When you extract a data form containing cell text, only the title (first 69 characters) of a cell text entry is extracted.

Syntax

```
CellText:1
```

Example

```
R1=A#Salaries, CellText:1
```

CustomHeader

Use this option to specify a custom header for a column or row. This is useful when you have a calculated column or row and want to hide the specific formula and replace it with a description, such as Variance. If you have nested dimensions in rows or columns, the custom header applies to the entire header in the row or column, not just to the dimension.

You can use a semicolon as a delimiter to specify custom headers for subsequent cells. For example, this syntax replaces the three dimension headers in the row with custom headers Scenario, Year, and Month:

```
R2=S#Actual.Y#2009.P#January,CustomHeader:Scenario;Year;Month
```

To replace some header cells but not others, use a period (.) to indicate that the original header should be displayed. You can also hide a header by leaving out the period. For example, the following syntax shows the original header for the first dimension, hides the header for the second dimension, and shows the original header for the third dimension.

```
R2=S#Actual.Y#2009.P#January,CustomHeader:.;.;.
```

Caution! When the CustomHeader option is used with calculated rows or columns, you can only set the text for the first cell of the header.

Syntax

`CustomHeader:HeaderName`

Replace *HeaderName* with the header to use.

Note: You cannot use a comma (,), colon (:), or ampersand (&) in the custom header.

Example

This example sets the custom header for column 3 to Variance.

```
C3=SCalc(C1-C2),customheader:Variance
```

CustomHeaderStyle

Use this option to assign custom style attributes to a row or column header. This is different from the Style option of the HeaderOption keyword in that it applies to a row or column header as a whole as opposed to a single dimension across all headers. When there is a conflict, the CustomHeaderStyle option will be used over HeaderOption:Style keyword. To mix the two styles instead of the CustomHeaderStyle being used, insert a semicolon before the CustomHeaderStyle option as in the example below. See “Style” on page 166.

Syntax

This option uses the standards supported by the W3C. See <http://www.w3.org/tr/rec-css2>.

`CustomHeaderStyle:Property:Value`

Example

```
C1=S#Actual.Y#2009,CustomHeaderStyle;;font:italic 12px arial;color:red
```

DynamicPOV

This keyword is deprecated.

If you have a form that contains the DynamicPOV keyword, take these steps to account for this deprecation:

1. Remove all dimensions specified with the DynamicPOV keyword from the BackgroundPOV keyword.
2. Delete the DynamicPOV keyword.

FormInputBoxLength

Use this keyword to specify the input box width . The default is 20 characters wide.

Note: This keyword does not determine the number of characters that can be entered into the box.

Syntax

`FormInputBoxLength=InputLength`

Replace *InputLength* with the number of characters for the input box width.

Example

`FormInputBoxLength=20`

FormNumDecimals

Use this keyword to specify the number of decimals for the entire form. If this keyword is specified, it overrides the number of decimals set for the cell. If this keyword is not specified, the number of decimals for the cell is used.

You can override a column, row, or cell decimal setting by using [NumDecimals](#).

Syntax

`FormNumDecimals=Decimals`

Replace *Decimals* with a value from 0 to 9.

Note: When you use a value of 9, maximum precision is used and therefore, up to 14 digits are actually included after the decimal.

Example

`FormNumDecimals=0`

FormRowHeight

Use this keyword to specify the height of all rows in the form. The default is 16px.

Syntax

`FormRowHeight=Pixels px`

Replace *Pixels* with the number of pixels for the row height.

Caution! You must include *px* after *Pixels*. If you omit *px*, rows and columns may not align correctly when the form is printed.

Example

```
FormRowHeight=16 px
```

FormScale

Use this keyword to specify the default scaling for the form.

If specified in a form, this keyword overrides the scale in the entity currency, otherwise the system uses the scaling defined for the currencies assigned to entities.

You can override the form scale setting by using the [Scale](#) option within row or column definitions or within a cell override.

Syntax

```
FormScale=n
```

Replace *n* with a value from -12 to 12.

Example

```
FormScale=0
```

HeaderOption

Use this keyword to specify options for how dimension headers are displayed. The options currently supported are:

- Length - for the maximum length of row headers. This can be a number or can be the word “Fixed”.

Note: Row headers are truncated with ellipses if they are longer than the specified length.

- ShowDescription
- ShowLabel
- Style

Syntax

```
HeaderOptionDimension=Length:n  
HeaderOptionDimension=ShowDescription  
HeaderOptionDimension=ShowLabel  
HeaderOptionDimension=Style:Property:Value;Property:Value...
```

Replace *Dimension* with the row dimension name, *n* with the length for the header or “Fixed”, *Property* with the style property and *Value* with the property value. See [Style](#).

Example

```
HeaderOptionPeriod=Length:4  
HeaderOptionScenario=ShowDescription
```

HeaderOptionsAccount=ShowLabel,Style:Font-style:italic

Instructions

Use this keyword to specify the instructions to give to form users. If you do not specify this keyword, the instructions window opens with this message: "There are no detailed instructions defined for this form".

Syntax

Instructions=*HTMLInstructions*

Replace *HTMLInstructions* with HTML-formatted text and links.

Example

Instructions=Please enter your cost center budgets for the year.Any questions, please contact the Budget Administrator.

LineItemDetailSinglePeriod

Use this keyword to specify whether line item details are displayed for just the selected cell or for all input periods. The default is True, which displays details for only the selected cell.

Syntax

LineItemDetailSinglePeriod=*Boolean*

Replace *Boolean* with True to display line item detail for the selected cell or False to show line item detail for all input periods.

Example

LineItemDetailSinglePeriod=True

Link

Use this keyword to specify a link to another data form. Use with [Cell_Link](#). Links are references in row definitions. You can specify up to 64 links (Link1 - Link64). Links do not need to be numbered sequentially.

Note: Linked form names are case-sensitive.

Syntax

Cell_Link*x*
Link*x*=*FormName*

Replace *x* with the number to assign to the link and replace *FormName* with the form name to which to link.

Example

```
R1=A#Salaries, Cell_Link1  
Link1=Dynamic
```

MaxCells

This keyword specifies the maximum number of cells allowed in the data form. If the form results in more than the MaxCells value, an error occurs. The default is 25000.

Syntax

```
MaxCells=n
```

Replace *n* with the number of cells for the form.

Note: The value is for the number of visible cells on the form including calculated cells. It does not include suppressed cells.

Example

```
MaxCells=500
```

MaxColsForSparseRetrievalMethod

Use this keyword to specify the number of columns in the form to optimize performance of sparse data forms. You specify this keyword for forms that have more than 10 columns. If your form has 10 or fewer columns, the optimization occurs automatically.

Syntax

```
MaxColsForSparseRetrievalMethod=n
```

Replace *n* with the number of columns in the form.

Example

```
MaxColsForSparseRetrievalMethod=11
```

NoSuppress

Use this option to turn off suppression for one or more rows or columns. Thus, regardless of the form suppression options, the row or column is displayed. Use this option within a row or column definition.

Example

```
R4=A#Inventory, NoSuppress
```

NumDecimals

Use this option to specify the number of decimal places to show for calculated or uncalculated rows or columns or in a cell override. If this keyword is specified, it overrides the number of decimals set for the cell or set by [FormNumDecimals](#).

Syntax

`NumDecimals:n`

Replace *n* with a value from 0 to 9.

Note: When you use a value of 9, maximum precision is used and therefore, up to 14 digits are actually included after the decimal.

Example

```
C4=A#Inventory,NumDecimals:1
```

Override

Use this option to specify different POV dimension members, formula calculations, or text for one or more consecutive columns or rows or to change a style. Use this option within a row or column definition.

Note: To override cells that are not consecutive, you can enter the override in the Other field of the form builder. You can enter multiple overrides by separating each override by a comma. The following example overrides three individual cells:

```
Override(1,1,string("455")),Override(3,3,string("23")),  
Override(5,5,string("2234"))
```

You cannot use member lists with the Override option. You can apply multiple overrides to a cell, with the last value being used when there is a conflict. In most cases, you can also mix the overrides on a cell. For example, if an override on a row specifies a value for NumDecimals while an override on a column specifies a value for Scale, there is no conflict except if the same dimension is used. If you use a leading semicolon, you can mix values for Style.

You can also mix overrides where they intersect by including a semicolon after the Override keyword. Note that without the semicolon, the style defined for the row is used because row values supersede column values when they conflict in the form. To mix the overrides from the row and column definitions, you must include the semicolon on the row keyword.

Syntax

```
Override(StartCell,EndCell,Override)
```

Parameter	Description
StartCell	An integer value representing the override starting point. If the override is defined for a row, this parameter indicates the starting column where the override is applied. In this example, the override starts with column 2: R2=A#Sales, Override(2,3,A#SalesTP)
EndCell	An integer value representing the override ending point. In the example above, the override ends with column 3.
Override	A POV. For example, to override with a different account, you specify A#newacct. To override with a different scenario and account, you specify A#newacct.S#newscenario. You can also use these values or options with a POV override or by themselves: • Blank • NumDecimals • ReadOnly • SCalc • Scale • String • Style

Example

In the following override example, the system overrides columns 2 and 3 for row 2 with December as the period and the PriorSales amount instead of the sales amount for the month:

```
R2=A#Sales, Override(2,3,Y#2009.P#December.A#PriorSales)
```

In the following override example, the system overrides columns 2 and 3 for row 2 with a formula calculation of the average of Sales1, Sales2, and Sales3:

```
R2=A#Sales, Override(2,3,SCalc((A#Sales1+A#Sales2+A#Sales3)/3))
```

In the following override example, the system overrides columns 2 and 3 for row 2 with the read only option.

```
R2=A#Sales, Override(2,3,readonly)
```

PrintNumDataColsPerPage

Use this keyword to specify the number of columns to print on each page. The default is 6.

Users can override this setting in the printer-friendly display of the form.

Note: When setting the value for this keyword, you should also take into account the specifics of the form, such as the row height, and the printer and printer settings used, including resolution and orientation.

Syntax

```
PrintNumDataColsPerPage=n
```

Replace *n* with the number of columns to print per page.

Example

```
PrintNumColsPerPage=6
```

PrintNumRowsPerPage

Use this keyword to specify the number of rows to print on each page. The default is 20.

Users can override this setting in the printer-friendly display of the form.

Note: When setting the value for this keyword, you should also take into account the specifics of the form, such as the row height, and the printer and printer settings used, including resolution and orientation.

Syntax

```
PrintNumRowsPerPage=n
```

Replace *n* with the number of rows to print per page.

Example

```
PrintNumRowsPerPage=20
```

PrintRepeatHeadersonAllPages

Use this keyword to specify whether to print headers on all pages. The default is True.

Users can override this setting in the printer-friendly display of the form.

Syntax

```
PrintRepeatHeadersonAllPages=Boolean
```

Replace *Boolean* with True or False.

Example

```
PrintRepeatHeadersonAllPages=True
```

Rn

Use this keyword to define a row in the form. The keywords such as R1, R2, R3 provide the definition of each row in the specified order. The row identifier must begin with 1 and proceed in sequential order. You can use member and system lists in row definitions.

Note: You can use multiple lists in a row.

Dimension elements specified for the row override elements that are set in the [BackgroundPOV](#) or [SelectablePOVList](#).

You can use these values and options within a row definition:

- [AddMember](#)
- [Blank](#)
- [Cell_Link](#)
- [CellText](#)
- [CustomHeader](#)
- [NoSuppress](#)
- [NumDecimals](#)
- [Override](#)
- [ReadOnly](#)
- [SCalc](#)
- [Scale](#)
- [String](#)
- [Style](#)

Syntax

Rn=CalcExpression
Rn=POVExpression

Table 29 Syntax for Rows Keyword

Parameter	Description
n	The row number.
CalcExpression	Use SCalc, Blank, or String. See “SCalc” on page 162 , “Blank” on page 148 , and “String” on page 166 .
POVExpression	A valid dimension intersection or member list.

Example

```
R1=A#Sales.I#[ICP Entities],AddMember:I{[Base]}
R2=A#Sales.I{[Base]}
R3=A#HeadCount.I#[ICP None],NoSuppress
R4=A#Purchases.I#[ICP None],CustomHeader:ABC
R5=SCalc(Row(2)*100),numdecimals:1,scale:0
R6=A{OperatingIncome.[Descendants]}
```

ReadOnly

Use this option to specify a read-only row, column, or cell in the form. Use this option within a row or column definition or within a cell override definition. The read-only cell looks like all

other cells but you cannot edit its contents. You can modify the style of a read-only row, column, or cell so that it looks different from editable rows, columns, and cells. See [Style](#).

Note: When you export to Excel, the read-only formatting is maintained.

Example

```
C4=S#Actual.Y#2009,ReadOnly
```

ReportDescription

Use this keyword to specify the form description.

Syntax

```
ReportDescription=Description
```

Replace *Description* with a description for the form. The description can contain a maximum of 255 characters.

Example

```
ReportDescription=Intercompany Detail
```

ReportLabel

Use this keyword specify the form name. This keyword is required.

Syntax

```
ReportLabel=Label
```

Replace *Label* with the form name. The name can contain a maximum of 20 characters. You cannot use these characters in the name:

- period (.)
- colon (:)
- comma (,)
- semicolon (;)
- question mark (?)
- forward slash (/)
- backslash (\)
- pipe (|)
- curly brackets ({ })
- parentheses (())

- quotation marks (")
- less-than and greater-than signs (< >)
- plus sign (+)
- at sign (@)
- number sign (#)
- asterisk (*)

Note: You can use an underscore (_) in the report label but it must be used between two characters. It cannot be used alone as the label name and it cannot be used at the end of a label name.

Example

ReportLabel=ICP Detail

ReportSecurityClass

Use this keyword to specify the security class assigned to the form. The default is [Default].

Syntax

ReportSecurityClass=*SecurityClass*

Replace *SecurityClass* with the name of a valid security class.

Example

ReportSecurityClass=Corporate

ReportType

Use this keyword to specify the report type. This keyword is required and the value must be set to WebForm for the file to be loaded as a data form script.

Syntax

ReportType=WebForm

SCalc

Use this value to specify calculations in rows, columns, or cells. Use this value within a row or column definition or within a cell override definition. These calculations are performed on the application server.

You can also use this value to create text within the form.

Syntax

`SCalc(<expression> [<operator> <expression>])`

Note: You can include multiple pairs of `[<operator> <expression>]` in an SCalc calculation.

Parameter	Description
operator	The mathematical operator for the calculation. These operators are supported: + - * /
expression	The values in the calculation. In addition to numeric values, you can include cell references, row references, column references, and various other types of items. For details and examples, see "Items Supported in SCalc Calculations" on page 163 .

Items Supported in SCalc Calculations

You can use these types of items in SCalc calculations:

- References to dimension members. This example references the Account dimension members Purchases and OtherCosts: `R6=SCalc((A#Purchases)-(A#OtherCosts))*100)`
- Cell references, using the syntax `Cell(rowIndex, columnIndex)`. This example refers to the cell in the fourth row of the second column in the form: `R1=SCalc(Cell(4,2))`
- Row references, using the syntax `Row(rowIndex)`. This example divides row 4 by row 2: `R3=SCalc(Row(4)/Row(2))`

Note: For rows or columns that contain member lists, the calculation occurs on the total for the members of the list.

- Column references, using the syntax `Col(columnIndex)`. This example adds column 1 and column 3: `C4=SCalc(Col(1)+Col(3))`
- Nested formulas, using parentheses to nest.
- Forward references to cells with SCalc calculations

Note: SCalc calculations are not performed until the data is saved and the calculated results are not displayed on the form until the form is refreshed.

An SCalc row or column can reference another SCalc row or column in its calculation, however you cannot forward reference in a SCalc row or column for another SCalc row or column. For example, this SCalc forward reference is allowed:

```
C1=A#Sales
C2=A#COGS
C3=SCalc(Col(1)-Col(2))
C4=SCalc(Col(3)/Col(1)*100)
```

Scale

Use this option to specify the scale for uncalculated columns, rows, or cells.

Syntax

`Scale:n`

Replace *n* with a value from -12 to 12.

Example

`C4=A#Inventory,Scale:2`

SelectablePOVList

Use this keyword to specify the members of a dimension that can be selected by users.

The initial value for a selectable dimension comes from the Background POV if one is specified for the dimension; otherwise, the initial value comes from the user's point of view. If the initial value is invalid for the list, the system defaults to the first member of the list.

When the user selects a new member, the selected member becomes part of the user's point of view.

Syntax

```
SelectablePOVList=Dimension{MemberList}  
SelectablePOVList=Dimension{[SystemList]}  
SelectablePOVList=Dimension{Parent.[SystemList]}
```

Table 30 Syntax for SelectablePOVList Keyword

Parameter	Description
Dimension	One of these characters to represent the dimension that is selectable: ● S for Scenario ● W for View ● E for Entity ● V for Value ● A for Account ● I for ICP ● C1 for Custom1 ● C2 for Custom2 ● C3 for Custom3 ● C4 for Custom4
Parent	A valid parent for the dimension.
MemberList	Name of a valid member list.

Parameter	Description
SystemList	Name of a valid system list.

Example

```
SelectablePOVList=S{ActualBudget}.Y{Years}.P{Months}.E{[Hierarchy]}.
C1{ProductsAllocate}.C2{Customers}.C3{AllChannels}
```

ShowDescriptions

Use this keyword to specify whether descriptions are displayed for the dimension elements in the form. The default is False.

Note: If no description exists, the label is shown. If there is no description for an entity, only the entity label is shown; the parent label is not included.

Syntax

```
ShowDescriptions=Boolean
```

Replace *Boolean* with True or False.

Example

```
ShowDescriptions=True
```

ShowLabels

Use this keyword to specify whether labels are displayed for the dimension elements in the form. The default is True.

If [ShowLabels](#) and [ShowDescriptions](#) are set to True, then labels and descriptions are separated by hyphens. If both are set to False, the system displays labels.

Syntax

```
ShowLabels=Boolean
```

Replace *Boolean* with True or False.

Example

```
ShowLabels=False
```

String

Use this option to specify a text string in a column, row, or cell. Use this option within a row or column definition or within a cell override definition. The String option is considered a server-side calculation, therefore it can be used anywhere that SCalc is used.

Example

```
C4=String("Show this read-only string")
```

Style

Use this option to specify the style attributes for a row, column, cell, custom header, or dimension header. Use this option within a row, column, cell override, custom header or header option definition.

Caution! Use care when applying style attributes to a form. The Style option is a powerful feature and, if used incorrectly, can significantly change the appearance of a form.

Note: When you export to Excel, the style formatting is maintained.

Syntax

The Style option is interpreted by the browser, not by Financial Management, so it is limited only by what the browser supports. The Style option uses the standards supported by the W3C. See <http://www.w3.org/tr/rec-css2>. The Style keyword consists of one or more style properties, separated by semicolons. Each property consists of the property name, a colon, and the value. The value can be one or more words, separated by spaces. For example:

```
Style:color:red;text-align:right
```

Note: You can use the Style option keyword one time on a line and add multiple Property:Value pairs. For example:

```
C3=S#Actual.Y#@CUR(-1),Style:color:blue;background-color:red;font-weight:bold
```

The Style Example table lists some of the properties and potential values that can be used. For a full list, see the Property Index at <http://www.w3.org/TR/REC-CSS2/propidx.html>.

Table 31 Style Examples

Property	Value
Color (foreground) Background-color	The color name or the standard hexadecimal RGB notation. For example: <pre>Style: color: red</pre> <pre>Style: color: rgb(255,0,0)</pre> <pre>Style: background-color: #ff0000</pre> <pre>Style: background-color: yellow</pre> A quick reference for colors can be found at http://www.cross-browser.com/docs/color_charts.html
Font-family	The font name. For example: <pre>Style: font-family: Arial</pre>
Font	The font style. For example: <pre>Style: font: italic 13px Courier</pre> Note: You can combine up to six font properties separated by a space for one Style value. For example: <pre>Style: font:extra-bold Italic 12pt Arial</pre>
Font-weight	The font weight. Values include demi-bold, demi-light, extra-bold, light, extra-light, demi-light. For example: <pre>Style: font-weight: extra-bold</pre>
Font-size	The font point size. For example: <pre>Style: font-size: 12pt</pre>
Alignment	The text alignment. Values include left, right, center, and justify. For example: <pre>Style: text-align: center</pre>

You can also mix styles where they intersect by including a semicolon after the Style keyword. Note that without the semicolon after the Style keyword, the style defined for the row is used because row values supersede column values when they conflict in the form. To mix the styles from the row and column definitions, you must include the semicolon on the row keyword.

Each cell in the grid displays only its right and bottom borders, so that a single pixel separates adjoining cells. You can use Style to change the color or set the line to dashed for the right or bottom borders. However, if you enable the top or left border, it will be in addition to the bottom border of the above cell and the right border of the cell to the left, respectively. This means you will have two borders. You can avoid double borders by turning off the adjacent border.

You also need to set the CustomHeaderStyle of an axis to keep the headers aligned with the grid.

Example

In the following example, the styles from Row 1 (pink background) and Column 1 (pink background and bold, green text) are mixed by adding a semicolon after the Style keyword in the row definition. See the example for [SuppressColHeaderRepeats](#) to see what the data entry form looks like if this syntax is used.

```
R1=Blank, CustomheaderStyle: font-weight: bold, CustomHeader: Balance sheet accounts,
Style;; Background-color: pink
```

C1=S#Actual.Y#@CUR, **Style: Background-color: pink; color: green; font-weight: bold**

	<i>Actual</i>	<i>Budget</i>
Balance sheet accounts		
Cash		1,746,137.00
Short Term Receivables		9,951.32
Inventories		-10,746.57
Short Term Investments		2,831.47
Total Short Term Assets		1,748,173.21
Computers	33,333.00	13,382.68
Buildings		4,351.48
Office Equipment	50,000.00	-948.88
Transportation	80,000.00	4,719.12
Fixtures	800,000.00	931.97
Tangible Assets	963,333.00	22,436.37
Accumulated Depreciation	22,222.00	-87,002.34

SuppressColHeaderRepeats

Use this keyword to specify whether to suppress repeated column headers. The default is True.

This example shows the data form where repeated column headers are suppressed. If suppression is turned off, the column header Actual would also be displayed above August.

	<i>Actual</i>	
	<i>July</i>	<i>August</i>
Sales	500,000.00	600,000.00
Purchases	250,000.00	300,000.00

Syntax

SuppressColHeaderRepeats=*Boolean*

Replace *Boolean* with True or False.

Example

SuppressColHeaderRepeats=False

SuppressInvalidColumns

Use this keyword to specify whether columns containing invalid cells are suppressed from the form. The default is False.

Note: Regardless of this setting, invalid columns are suppressed for non-administrator users.

Syntax

`SuppressInvalidColumns=Boolean`

Replace *Boolean* with True to suppress invalid columns or False to display invalid columns.

Example

`SuppressInvalidColumns=False`

SuppressInvalidRows

Use this keyword to specify whether rows containing invalid cells are suppressed from the form. The default is False.

Note: Regardless of this setting, invalid rows are suppressed for non-administrator users.

Syntax

`SuppressInvalidRows=Boolean`

Replace *Boolean* with True to suppress invalid rows or False to display invalid rows.

Example

`SuppressInvalidRows=False`

SuppressNoDataColumns

Use this keyword to specify whether columns containing nodata cells are suppressed from the form. The default is False.

Syntax

`SuppressNoDataColumns=Boolean`

Replace *Boolean* with True to suppress columns with no data or False to display columns with no data.

Example

`SuppressNoDataColumns=False`

SuppressNoDataRows

Use this keyword to specify whether rows containing nodata cells are suppressed from the form. The default is False.

Syntax

`SuppressNoDataRows=Boolean`

Replace *Boolean* with True to suppress rows with no data or False to display rows with no data.

Example

`SuppressNoDataRows=False`

SuppressRowHeaderRepeats

Use this keyword to specify whether to suppress repeated row headers. The default is True.

Syntax

`SuppressRowHeaderRepeats=Boolean`

Replace *Boolean* with True or False.

Example

`SuppressRowHeaderRepeats=False`

SuppressZeroColumns

Use this keyword to specify whether columns containing zeros are suppressed from the form. The default is False.

Syntax

`SuppressZeroColumns=Boolean`

Replace *Boolean* with True to suppress columns with zeros or False to display columns with zeros.

Example

`SuppressZeroColumns=False`

SuppressZeroRows

Use this keyword to specify whether rows containing zeros are suppressed from the form. The default is False.

Syntax

`SuppressZeroRows=Boolean`

Replace *Boolean* with True to suppress rows with zeros or False to display rows with zeros.

Example

SuppressZeroRows=False

Using Relative Periods and Years

You can use these year and period functions:

- @Cur
- @First
- @Last
- @Next
- @Prev

Note: Year and period values are not case sensitive.

For the system to apply the relative year when the period extends past the current year, or if you need to refer to a prior year, you must specify Y#@Cur as part of the row or column definition.

For example, you need to specify C1=Y#@Cur.P#@Cur(+1) to return the correct year if the current period is the last period. If the current Point of View is 2008 December:

C1=Y#@Cur.P#@Cur(+1) returns 2009 January.

If the current Point of View is 2008 December: C2=P#@Cur(+1) returns 2009 January (if January is the first period in the year, and December is the last). Therefore, for the system to correctly reflect the year information, you specify the year in the column definition. For example,

C2=Y#@Cur.P#@Cur(+1)

The @CUR function can be used for other dimensions to retrieve the current POV. However, @CUR for the other dimensions does not support using an offset, for example, +2, since the other dimensions are not chronological. For example, if the current POV had the Actual scenario, a valid example for a column would be to use C1=S#@CUR to return “Actual”. If you changed the POV scenario to Budget, this same column definition would return “Budget”.

Order of Precedence for Conflicting Attributes

There are multiple ways to set the number of decimals, scale, and POV for data forms. For example, you can set the number of decimals:

- For the form - FormNumDecimals=3
- For a column - C4=A#Inventory,NumDecimals:1
- For a row - R4=A#Sales,NumDecimals:2
- For a cell override - R2=S#Actual.P#August, Override(2,7,P#July,NumDecimals:4)

In some instances, the setting for one of these attributes may intersect with a conflicting setting for the same attribute. For example, a column may have a scale setting of 1, while an intersecting row may have a scale setting of 2. It is important to understand that the value that is applied for the number of decimals and scale attributes is based on this order of precedence:

- Cell Override
- Row
- Column
- Form
- Default - For number of decimals, the default comes from the account. For scale, the default is taken from the entity currency.

Note: In the POV, the value for each dimension is independently resolved. For example, the Account dimension might be set at the form level and the Scenario dimension at the row level.

The system resolves conflicting attributes according to the order of precedence. For example, if the number of decimals attribute is defined on a cell override, this value is used instead of the form, row, column or default.

Similarly, if a cell has conflicting data and calculations specified, this order of precedence is applied:

- SCalc
- Data

Note: SCalc formulas in a form are evaluated in a left to right, top to bottom order. For example, all of the columns in row 1 are evaluated sequentially, then all of the columns for row 2 are evaluated. An SCalc formula can refer to another SCalc cell that precedes it in the evaluation order.

Editing Data Forms

You can edit a data form using the Form Builder.

Note: After you edit a data form, you must load the updated script to the application to see the changes that you made.

➤ To edit a data form:

- 1 Open the application.
- 2 Select **Administration**, then **Manage Documents**.
- 3 Select the **Data Entry Forms** tab.
- 4 To edit a form, do one of these tasks:

- Select the check box next to a form and click **Edit**.
- Click the Edit icon  next to a form.

Note: You can select only one form at a time to edit.

- 5 Edit the form information on each tab of the Form Builder. You can move between tabs by clicking the tab or by clicking **Back** and **Next** at the bottom of the Form Builder.

Note: To post changes to the server, click Update.

Tip: If you make a mistake, click Reset at the bottom of the Form Builder to reset values.

- 6 To scan the form for proper syntax, click **Scan** at the bottom of the Form Builder.
- 7 Click **Save** at the bottom of the Form Builder.
- 8 Select a file name and location.

Creating Data Form Folders

You can create a folder hierarchy for data forms. A Root folder is available by default and cannot be deleted. New folders are created under the Root folder. Organizing data forms into folders may help if you have many forms.

► To create folders for data forms:

- 1 Open the application.
- 2 Select **Administration**, then **Manage Documents**.
- 3 Select the **Data Entry Forms** tab.
- 4 Select **New Folder**.
- 5 Enter a folder name.

The name can contain a maximum of 20 characters, including spaces.

- 6 **Optional:** Enter a folder description.

The description can contain a maximum of 40 characters, including spaces.

- 7 From **Security Class**, select the security class to assign to the folder.

Note: Users of this folder must have access rights to the specified security class.

- 8 Click **OK**.

Viewing and Hiding Private Data Forms

You can view private documents of other users. This feature is available only to the application administrator. To view only private documents, click Show Private Documents; to view only public documents, click Hide Private Documents.

Loading Data Forms

After you create data form scripts, you load them into an application. For information on creating a data form script, see [“Data Form Script Syntax” on page 144](#).

Note: Oracle recommends that you add Financial Management to the exceptions for your Web pop-up blocker. When you perform some Financial Management tasks on the Web such as loading data, a status window pops up showing the task status. If you have a pop-up blocker enabled on your computer, the status window is not displayed.

► To load data forms:

- 1 Open the application.
- 2 Select **Administration**, then **Manage Documents**.
- 3 Select the **Data Entry Forms** tab.
- 4 Click **Load**.
- 5 Click **Browse** and locate the data form script to load, then click **Open**.

Note: By default, data form scripts use the WDF file extension. The load process accepts other file extensions such as TXT or CSV, however, Oracle recommends that you use the WDF file extension.

6 **Optional.** Perform one of these steps:

- To override the security class specified in the file being loaded, select **Override**, then select the security class from **Security Class**.
- To make this form available only to you, select **Private**. If you select this check box, the **Override** check box and **Security Class** drop-down list are not available.

- 7 Repeat [step 5](#) and [step 6](#) until you have added the forms to load.
- 8 **Optional:** Select **Overwrite Existing Documents** if you are updating data forms.
- 9 Click **OK**.

Note: All data forms that you load must include a valid ReportLabel. If you have selected to load multiple forms and one of the forms has an invalid ReportLabel, none of the selected forms are loaded.

Extracting Data Forms

You can extract data form scripts from an application. Extracting the script does not delete the script from the folder or from the application. It only extracts the contents of the script to a location you select.

After you extract a data form script, you can modify it and load it to the application. Make sure to overwrite the old file when you load the script. You can also use the model for a new script.

Note: If you are loading a script to an application, you must select the Overwrite Existing Documents option to replace the old file in the application.

► To extract data forms:

- 1 Open the application.
- 2 Select **Administration**, then **Manage Documents**.
- 3 Select the **Data Entry Forms** tab.
- 4 Click **Extract**.
- 5 Click the file name link for the file to extract.
- 6 Click **Save**.
- 7 Specify the name and location for the extracted file, or leave the default name and path.

Note: By default, data forms use the WDF file extension.

- 8 Click **Save**.
- 9 Repeat steps 5-8 until you have extracted all data forms that you selected in step 3.
- 10 Click **OK**.

Deleting Data Forms

To be able to delete data forms, you must be an administrator with the security role of Manage Data Entry Forms.

► To delete data forms:

- 1 Open the application.
- 2 Select **Administration**, then **Manage Documents**.
- 3 Select the **Data Entry Forms** tab.
- 4 Select the check box next to the data form to delete.

Note: You must delete data forms from a folder before you can delete a folder.

- 5 Click **Delete**.



Using Extended Analytics

In This Chapter

Creating a Data Link (UDL) File for Extended Analytics	177
Extended Analytics Registry Settings	181
Star Schema	181
Creating and Exporting Data to a Star Schema	185
Updating a Star Schema	187
Deleting a Star Schema	188
Creating a Star Schema Template	188
Deleting a Star Schema Template	189

An Extended Analytics star schema enables you to use Essbase to analyze data and produce reports. To do so, you use the star schema to send data to an Oracle Essbase database.

Note: You create the database after you export to the star schema. See the *Database Administrator's Guide* for your release.

You can also use a star schema with third-party products.

To use the Extended Analytics module, you must set up an OLAP data source name (DSN) for the database that will store star schemas, then configure the DSN with the Financial Management configuration utility. For instructions, see the *Oracle Hyperion Enterprise Performance Management System Installation and Configuration Guide*.

Note: To store star schemas in multiple databases, you can create a DSN for each database.

To use Extended Analytics, you must be assigned the security role of Administrator or Extended Analytics.

Creating a Data Link (UDL) File for Extended Analytics

To use Extended Analytics, you must create a universal data link (UDL) file, which provides a connection between the database server and application server. You can also encrypt the UDL file.

► To create a data link (UDL) file:

1 From Windows Explorer, select the folder in which to store the file.

Note: You should store the file in a directory other than the Financial Management installation directory, so that you can retain the file if you uninstall Financial Management.

2 Select **File**, then **New**, and then **Text Document**.

3 Rename the file, removing all spaces, and change its file extension to `.udl`, for example, `hfm.udl`.

4 Configure the UDL file.

See these topics:

- [“Configuring the Data Link for Oracle” on page 178](#)
- [“Configuring the Data Link for Microsoft SQL Server” on page 179](#)
- [“Configuring the Data Link for IBM DB2” on page 179](#)

Configuring the Data Link for Oracle

The data link specifies the database server name and other database-related information, such as the username and password of a user with full access rights to the database server.

For Oracle, Financial Management requires the Oracle OLE DB Provider, which you can download from the Oracle Web site.

► To configure the data link for Oracle:

1 From Windows Explorer, double-click the UDL file that you created.

2 From **Data Link Properties**, select the **Provider** tab, and from the **OLE DB Providers** list, select **Oracle Provider for OLE DB**.

3 Click **Next**.

4 On the **Connection** tab:

- Enter the server name *<Oracle Global Database Name>*.
- Enter a username and password.

Note: Ensure that the user is granted privileges to create, update, and delete tables.

5 Select **Allow saving password**.

6 Select the database to which you want to connect.

7 Click **Test Connection**.

8 On the **Test Connection Succeeded** box, click **OK**.

9 Click **OK** to save the connection and close the dialog box.

Configuring the Data Link for Microsoft SQL Server

The data link specifies the database server name and other database-related information, such as the username and password of a user with full access rights to the database server.

► To configure the data link for Microsoft SQL Server:

- 1 From Windows Explorer, double-click the UDL file that you created.
- 2 From **Data Link Properties**, select the **Provider** tab, and from the **OLE DB Providers** list, select **Microsoft OLE DB Provider for SQL Server**.
- 3 Click **Next**.
- 4 On the **Connection** tab:
 - a. Enter the server name, which is the computer hosting the relational database.
 - b. Select **Use a Specific User Name and Password**.
 - c. Enter a username and password for a user with full access rights to the database.

Note: You can use the default username `sa`. If you use the default name, leave the password text box blank, and select **Blank Password**. If you do not want to use the default username, you must enter the name of a user other than the default user who has full access rights. Ensure that this user is granted privileges to create, update, and delete tables.
 - d. Select **Allow Saving Password**.
- 5 Select the database to which you want to connect.
- 6 Click **Test Connection**.
- 7 On the **Test Connection Succeeded** box, click **OK**.
- 8 Click **OK** to save the connection, and close the dialog box.
- 9 Right-click the UDL file, and select **Open** to set the selected UDL file as the system data link file.

Configuring the Data Link for IBM DB2

The data link specifies the database server name and other database-related information, such as the username and password of a user with full access rights to the database server.

► To configure the data link for IBM DB2:

- 1 From Windows Explorer, double-click the UDL file that you created.
- 2 From **Data Link Properties**, select the **Provider** tab, and from the **OLE DB Providers** list, select **IBM OLE DB Provider for DB2 Servers**.
- 3 Click **Next**.
- 4 On the **Connection** tab:
 - a. Select **Use data source name option**.

This option selects the data source name that you established during the IBM DB2 installation.

Note: If the data source name is not displayed in the list, go to the Client Configuration Assistant, Database Properties box, and select As a system data source.

- b. Enter a username and password.

Note: Ensure that the user is granted privileges to create, update, and delete tables.

- 5 Select **Allow saving password**.
- 6 Click **Test Connection**.
- 7 On the **Test Connection Succeeded** box, click **OK**.
- 8 Click **OK** to save the connection and close the dialog box.
- 9 Right-click the UDL file and select **Open** to set the UDL file as the system data link file.

Encrypting UDL Files

Financial Management provides a utility to encrypt the UDL file used by the application server to communicate with the database. Based on the UDL file extension, Financial Management detects whether the file is encrypted. If the file extension is `.UDL`, the system assumes that the file is not encrypted. If the file extension is `.hfmudl`, the system assumes that the file is encrypted.

To encrypt the UDL file, you use the `EncryptHFMUDL` utility located in the `FinancialManagement/Server` directory.

► To encrypt the UDL file:

- 1 Do one of these tasks to access the MS-DOS command prompt:
 - Select **Start > Programs > Accessories > Command Prompt**.
 - Select **Start > Run**, and type `cmd`.
- 2 Enter `encrypthfmudl hfm.udl hfm.hfmudl`

where *hfmUDL* is the location and name of your original UDL file, and *hfm.hfmUDL* is the location for the encrypted UDL file.

Note: The encrypted UDL file extension must be `.hfmudl`. After encrypting the file, you can delete the original unencrypted file.

► To decrypt the UDL file:

- 1 Do one of these tasks to access the MS-DOS command prompt:
 - Select **Start > Programs > Accessories > Command Prompt**.
 - Select **Start > Run**, and type `cmd`.
- 2 Enter `encrypthfmudl hfm.hfmudl hfm.udl /u`.

where *hfm.hfmudl* is the location for the encrypted UDL file, and *hfm.udl* is the location for the decrypted UDL file.

Extended Analytics Registry Settings

These registry keys can be set in the HKEY_LOCAL_MACHINE\SOFTWARE\Hyperion Solutions\Hyperion Financial Management\Server registry location.

Table 32 Extended Analytics Registry Settings

Entry Name	Data Type	Default Value	Description
EACommandTimeout	REG_DWORD	0	The number of seconds that must elapse before a SQL query is determined to be deadlocked. Specify a value from 0, which means no time out, to 65535.
NumEAThreads	REG_DWORD	8	The number of threads used during processing of data. Specify a value from 1 to 32.
EAIgnoreInvalidMemberIDs	REG_DWORD	0	Determines whether to fail if an invalid dimension member ID is encountered. Specify 1 to ignore invalid IDs. Specify 0 to have the system fail if an invalid ID is encountered.
EAUseSQLBinding	REG_DWORD	1	Determines whether to use SQL binding. Specify 1 to use SQL binding or 0 to disable SQL binding. Caution! If you disable SQL binding, performance may be adversely affected.
ConnectionPoolingTrace	REG_DWORD	0	Determines whether to trace connection pool activity to log. Specify 1 to trace or 0 to disable trace.
EATraceADOErrors	REG_DWORD	0	Determines whether to trace ActiveX Data Object (ADO) activity to log. Specify 1 to trace or 0 to disable trace. This setting is not applicable to ODBC.
EAUseODBC	REG_DWORD	1	Determines whether to use ODBC to interact with the external data store. Specify 1 to use ODBC or specify 0 to use ADO.
EALogEntirePOV	REG_DWORD	0	Determines whether to log the pov being captured in the Extended Analytics dump to the log file (in the temp directory). Specify 1 to log the POV or 0 to disable the logging.

Star Schema

The Extended Analytics module enables you to create multiple star schemas per application. The table names in each star schema begin with a prefix that you specify. You can select multiple members in all dimensions to create the star schema that best reflects the information to export.

Note: Cell text and line item detail are not exported to the star schema.

The data combinations in the star schema are created based on the dimension members that you select to export. The more dimension members selected, the more possible data combinations that need to be created in the star schema, and the more time needed to complete the export process. You can calculate the number of data combinations by multiplying the number of members selected for each dimension.

Caution! Do not select to export all members from every dimension; select segments of data to export. Depending on the application size, the number of data combinations, and the amount of time to complete the export time could be excessive.

For example, you can export this data:

- Scenario - Actual
- Year - 2009
- Period - July
- View - Periodic
- Entity - Regional, United States, Florida, Connecticut
- Value - USD
- Account - Gross Margin, Sales, Total Costs
- ICP - [None]
- C1 - Balls, Tennis Balls, Golf Balls
- C2 - All Customers, Customer2, Customer3, Customer4, Customer5
- C3 - [None]
- C4 - [None]

The star schema that is created can then be used by Oracle Essbase Integration Services to create one or many data cubes to reflect the audience that needs to see and use the Budget information. The star schema will contain 180 data combinations for these members ($1 * 1 * 1 * 1 * 4 * 1 * 3 * 1 * 3 * 5 * 1 * 1 = 180$).

You can extract only local currency data if you prefer. To extract only local currency data, use the Entity Currency member from the Value dimension in the Point of View. Entity Currency acts as placeholder for the currency and extracts the default currency for each entity that is selected.

If the selected Scenario is YTD, Extended Analytics extracts periodic derived data. For example, suppose the Scenario is ZeroView=YTD. There is a value of 100 in an expense account in the first period. In the second period, no data is reported for that account. The derived periodic value for the second period is -100, forcing the YTD amount to 0. If Extended Analytics is run for this Point of View on a periodic basis, for the first three periods, the values are 100, -100, and 0, respectively.

Star Schema Formats

You select one of these extract format options when you create a star schema:

- Standard
- Metadata Only
- Selected Metadata Only
- Essbase
- Data Warehouse
- Flat File with Header
- Flat File No Header

The extract format option that you select determines the schema format used. Each schema format generates a different set of tables. These schema formats are available:

- Standard Essbase Schema - this schema is used for the Standard, Metadata Only, and Selected Metadata Only extract format types.
- SQL and Essbase Schema - this schema is used for the Essbase extract format type.
- Warehouse Normalized Hierarchy Schema - this schema is used for the Data Warehouse extract format type.

Prefix Tables

For each schema format, the system creates a *PREFIX_FACT* table that contains keys to the 13 dimension tables and one data field. The system also creates *PREFIX_DIMENSION* tables, *HFM_EA_EXTRACT* table to track extract timestamps for metadata, and a *PREFIX_LOCK_ACCESS* table to track writer and reader locks. A writer lock is used when metadata is being changed, for example, when the create or replace process is being used. A reader lock is used when metadata is not being changed and the system is updating data in the *FACT* table, for example when an update process is being used. The rules for locking are:

- Only one writer at a time can execute against the same prefix on the same DBMS instance.
- If a writer is executing or is in the queue, no readers can execute until the writer has completed.
- Multiple readers can execute simultaneously regardless of the point of view.

Note: The system creates two tables for the Entity dimension: *PREFIX_ENTITY* and *PREFIX_PARENT*.

For the SQL and Essbase Aggregation Schema format, the system also creates *PREFIX_DIMENSION_BASE* tables.

For the Warehouse Normalized Hierarchy Schema format, the system creates *PREFIX_DIMENSION_PARENT* tables.

Note: Base and Parent tables are not created for the View, Year, and Parent dimensions.

For example, if the Relational Table Prefix is DEMO, the system creates these tables for the Essbase format:

- HFM_EA_EXTRACT1
- HFM_LOCK_ACCESS1
- DEMO_FACT
- DEMO_YEAR
- DEMO_VIEW
- DEMO_PARENT
- DEMO_SCENARIO and DEMO_SCENARIO_BASE
- DEMO_PERIOD and DEMO_PERIOD_BASE
- DEMO_VALUE and DEMO_VALUE_BASE
- DEMO_ENTITY and DEMO_ENTITY_BASE
- DEMO_ICP and DEMO_ICP_BASE
- DEMO_ACCOUNT and DEMO_ACCOUNT_BASE
- DEMO_CUSTOM1 and DEMO_CUSTOM1_BASE
- DEMO_CUSTOM2 and DEMO_CUSTOM2_BASE
- DEMO_CUSTOM3 and DEMO_CUSTOM3_BASE
- DEMO_CUSTOM4 and DEMO_CUSTOM4_BASE

Flat File Extract Options

When you extract members in a flat file format, you can select whether to include a header for the file.

Example: Flat File with Header

```
Scenario=Actual
Year=2009
Period=[Year]
View=YTD
Entity=CORP_OPS
Parent=GROUP
Value=[Parent Total]
Possible Duplicated Records=No
SalesIC;CHINA;[None];[None];[None];[None];100000
```


SalesIC;CHINA;[None];AllCustom2;[None];[None];100000

Scenario=Actual

Year=2009

Period=[Year]

View=Periodic

Entity=CORP_OPS

Parent=GROUP

Value=[Parent Total]

Possible Duplicated Records=No

SalesIC;CHINA;[None];[None];[None];[None];100000

SalesIC;CHINA;[None];AllCustom2;[None];[None];100000

Example: Flat File No Header

!Data

Actual;2009;[Year];YTD;GROUP.CORP_OPS;[Parent Total];SalesIC;CHINA;[None];[None];[None];
[None];100000

Actual;2009;[Year];YTD;GROUP.CORP_OPS;[Parent Total];SalesIC;CHINA;[None];AllCustom2;
[None];[None];100000

Actual;2009;[Year];Periodic;GROUP.CORP_OPS;[Parent Total];SalesIC;CHINA;[None];[None];
[None];[None];100000

Actual;2009;[Year];Periodic;GROUP.CORP_OPS;[Parent Total];SalesIC;CHINA;
[None];AllCustom2;[None];[None];100000

Creating and Exporting Data to a Star Schema

You can create a star schema by specifying the prefix that identifies the tables for the schema and the dimension members of the cells to be exported. Before you export the data, make sure that the application data is consolidated.

Caution! Do not use the Financial Management database as the destination database for the data export.

Note: For Oracle database users: The extracted text data is stored in NVARCHAR(Unicode) format. Use the Oracle “translate” command in SELECT statements to convert the text from Unicode to ANSI format. For example, SELECT translate(LABEL using CHAR_CS) as LABEL FROM DEMO_ACCOUNT.

You can also extract selected data to a flat file. The data is extracted as a .dat file to the HFM Server Working folder on the application server.

Note: If you are extracting a large amount of data, make sure that you have adequate disk space for the extract.

Oracle recommends that you add Financial Management to the exceptions for your Web pop-up blocker. When you perform some Financial Management tasks on the Web such as loading data, a status window pops up showing the task status. If you have a pop-up blocker enabled on your computer, the status window is not displayed.

► To create a star schema and export Financial Management data to it:

- 1 **Open the application.**
- 2 **Select *Administration*, then *Extended Analytics*.**
- 3 **Do one of these tasks:**
 - From *Template*, select the saved POV template to use.
 - Select the POV dimension members to export.
- 4 **From *Destination Database (DSN)*, select the database to which you are exporting the star schema.**

Note: Do not use the same Financial Management database that you are exporting data from as the destination database.

- 5 **For *Relational Table Prefix*, enter the prefix that will identify the tables for the star schema.**

Note: The prefix can contain up to 10 alpha-numeric characters and must start with a letter. It cannot contain an underscore. The default prefix is the application name.

- 6 **From *Extract Format*, select the type of extract to perform on the data:**
 - Standard
 - Metadata Only - extracts metadata only
 - Selected Metadata Only - extracts only the metadata for the selected dimension members.
 - Essbase
 - Data Warehouse
 - Flat File with Header
 - Flat File No Header
- 7 **Optional: Uncheck the *Exclude Dynamic Accounts* option to include dynamic accounts in the star schema.**
- 8 **Select *Create Star Schema*.**

Updating a Star Schema

You can export data to a previously defined star schema. When you update a star schema, you can specify different dimension members.

Note: When you update a star schema, the system clears data from the Fact table only.

Oracle recommends that you add Financial Management to the exceptions for your Web pop-up blocker. When you perform some Financial Management tasks on the Web such as loading data, a status window pops up showing the task status. If you have a pop-up blocker enabled on your computer, the status window is not displayed.

► To update a star schema:

- 1 Open the application.
- 2 Select **Administration**, then **Extended Analytics**.
- 3 Do one of these tasks:
 - From **Template**, select the saved POV template to use.
 - Select the POV dimension members to export.
- 4 From **Destination Database (DSN)**, select the database that contains the star schema.
- 5 For **Relational Table Prefix**, type the prefix that identifies the tables for the star schema.

Note: The prefix can contain up to 10 alpha-numeric characters and must start with a letter. The default prefix is the application name.

- 6 From **Extract Format**, select the type of extract to perform on the data.
 - Standard
 - Metadata Only - extracts metadata only
 - Selected Metadata Only - extracts only the metadata for the selected dimension members.
 - Essbase
 - Data Warehouse

Note: The Flat File option is not available for this action.

- 7 **Optional:** Uncheck the **Exclude Dynamic Accounts** option to include dynamic accounts in the star schema.
- 8 Select **Update Star Schema**.

Deleting a Star Schema

You can delete a star schema that you no longer need. Deleting a star schema deletes all of the star schema data, metadata, and tables.

► To delete a star schema:

- 1 Open the application.
- 2 Select **Administration**, then **Extended Analytics**.
- 3 From **Destination Database (DSN)**, select the database that contains the star schema.
- 4 For **Relational Table Prefix**, type the prefix that identifies the star schema's tables.

Note: The prefix can contain up to 10 alpha-numeric characters and must start with a letter. The default prefix is the application name.

- 5 Select **Delete Star Schema**.

Creating a Star Schema Template

You can create a star schema template, which enables you to name and save POVs so that you can use them again.

► To create a star schema template:

- 1 Open the application.
- 2 Select **Administration**, then **Extended Analytics**.
- 3 Select the POV dimension members.
- 4 From **Destination Database (DSN)**, select the database to which you are exporting the star schema.
- 5 For **Relational Table Prefix**, type the prefix that identifies the star schema's tables.

Note: The prefix can contain up to 10 alpha-numeric characters and must start with a letter. The default prefix is the application name.

- 6 Click .
- 7 Type a template name.
- 8 **Optional:** To overwrite a template, select **Overwrite**.
- 9 Click **Save**.
- 10 Click **OK**.

Deleting a Star Schema Template

► To delete a star schema template:

- 1 Open the application.
- 2 Select **Administration**, then **Extended Analytics**.
- 3 From **Template**, select the template to delete.
- 4 Click .
- 5 Click **OK** to confirm.
- 6 Click **OK**.

9

Defining Reports

In This Chapter

Defining Reports	191
Explore Data Report Example	192
Creating Journal Reports	193
Creating Intercompany Partner Matching Reports	194
Creating Intercompany Transaction Reports	203

You can generate these report types in Financial Management:

- Explore Data reports, which display information from data grids
- Journal reports, which display information for a specific journal or list of journals based on criteria that you select in the Journals module
- Intercompany Partner (ICP) Matching reports, which display the intercompany transactions that are to be eliminated during consolidation

You can run reports from the System Reports module on the Desktop or from the Reporting tab on the Web.

Tip: To access the Reporting tab on the Web, click the **Administration** link in the navigation bar, and then **Manage Documents**.

Defining Reports

You define a report by assigning values to report tags using the reports editor or a text editor. You can create a new report definition for each report, or you can open and modify a report definition. You can save the report definition with an RPT or HDE format.

You must specify these attributes for every report:

- Report Type
- Description
- Point of View

Caution! When you define reports, you must follow the syntax as shown in the following examples. The ReportType must start with a capital letter. Do not include spaces between words or on either side of the equal sign.

This example shows a Report Type, Report Description, and Point of View for a Journal report.

```
ReportType=Journal
ReportDescription=TaxJournals
POV=S#Actual.Y#2009.P#July.V#<Entity Curr Adjs>
```

The Point of View definitions are different for each report. For detailed examples of reports, see [“Explore Data Report Example” on page 192](#), [“Journal Report Example” on page 193](#), and [“Creating Intercompany Partner Matching Reports” on page 194](#).

The date, time, and user fields are automatically displayed as header information on all reports.

When you create a Journals, Explore Data or Intercompany Partner Matching Report on the Desktop, the report definition is created automatically and opened in the System Reports module. When you create a report from the Journals module on the Desktop, it opens in the View tab in the Reports module. When you create a report from the Journals module on the Web, you can preview the report or save the definition. When you create a report from the Explore Data module, it opens in the Design tab. You can use the Design view to modify a report script before you generate the report. You can then set a report description, apply a new style sheet, and save the report locally or remotely.

When you use reports on the Web, the system creates temporary files and log files containing information about these operations. You should set up a file transfer directory other than the Financial Management Web directory to store these temporary files. For information about creating this File Transfer directory, see the *Oracle Hyperion Enterprise Performance Management System Installation and Configuration Guide*.

Explore Data Report Example

This example shows a sample Explore Data report definition script:

```
ReportType=ExploreData
ReportDescription=Default Data Grid
Scenario=Budget
Year=2009
Period=July
View=<Scenario View>
Entity=UnitedStates.Connecticut
Value=USD
Account=Sales
ICP=[ICP None]
Custom1=[None]
Custom2=[None]
Custom3=[None]
Custom4=[None]
Scenarios=Budget
Years=2009
Periods=
```



```

Views=<Scenario View>
Entities=Regional,UnitedStates.Connecticut
Values=USD
Accounts=GrossMargin,Sales
ICPs=[ICP Top]
Custom1s=[None]
Custom2s=[None]
Custom3s=[None]
Custom4s=[None]
ScenarioMemberList=[Hierarchy]
YearMemberList=[Hierarchy]
PeriodMemberList=[Hierarchy]
ViewMemberList=[Hierarchy]
ValueMemberList=[Hierarchy]
AccountMemberList=[Hierarchy]
ICPMemberList=[Hierarchy]
Custom1MemberList=[Hierarchy]
Custom2MemberList=[Hierarchy]
Custom3MemberList=[Hierarchy]
Custom4MemberList=[Hierarchy]
RowDimensions=Account
ColDimensions=Entity
DisplayMode=Data
MetadataDisplayMode=Labels
Row=~~ExchangeRates
Row=~~NetProfit
Row=~~NetIncome
Row=~~GrossMargin
OBPSettings=0

```

Creating Journal Reports

You create journal reports to check the status of journals and review journal adjustments. You can create a journal report to display information for a specific journal, or list of journals based on criteria that you select in the Journals module. For example, you can select to display only journals with a specific status, balance type, entity, or account. You can also select the columns that display on the report and change their sort order.

You can set single entity and account filters for journal reports in the report definition. The syntax for Entity filtering is `EntityFilter=ParentName.ChildName` (for example, `UnitedStates.Maryland`). The syntax for Account filtering is `AccountFilter=MemberName` (for example, `Sales`).

Journal Report Example

This example shows a sample Journal report definition:

```

ReportType=Journal
ReportDescription=Tax Journals
POV=S#Actual.Y#2009.P#July.V#<Entity Curr Adjs>
ShowDescriptions=0
DisplayColumn_0=Label,Ascending,Repeat,Yes
DisplayColumn_1=Group,Ascending,Repeat,Yes

```

```
DisplayColumn_2=Description
DisplayColumn_3=Entity
StatusFilter=Working, Submitted, Rejected, Approved, Posted
TypeFilter=Regular
BalanceTypeFilter=Balanced, Unbalanced
EntityFilter=UnitedStates.Maryland
AccountFilter=Sales
```

Creating Intercompany Partner Matching Reports

You create Intercompany Partner (ICP) Matching reports to list the intercompany transactions that are eliminated during consolidation. Intercompany transactions are managed across the Intercompany Partner dimension. The Intercompany dimension contains all intercompany balances that might exist for an account. You can enter intercompany transactions through data grids, Data Load, Journals, or data forms. Financial Management can track and eliminate intercompany transaction details across accounts and custom dimensions. Intercompany transactions are eliminated as of the first common parent of two intercompany entities. They are eliminated through the [Elimination] member of the Value dimension.

Intercompany Matching reports help you track intercompany transactions for analysis and auditing purposes. The Intercompany Matching report shows matches for entities and intercompany partner dimensions that have been set up with the Intercompany Partner (IsICP) attribute enabled.

You can create Intercompany Matching reports by using the user interface or by using a text editor to specify report options in a script. To edit a report, you must modify the report script.

You can select accounts for the report, or use the plug account option, in which the system generates the account and matching account based on the plug account. You can also select to suppress reversed transactions, custom dimensions or intercompany details. When you print a report, you can override report settings to customize the report for your needs.

Selecting Member Lists for Intercompany Matching Reports

You can select member lists for the report Entity and Partner. For example, you could select the Regional member list for the Entity field. If you select a member list for Entity and Partner, the system processes the ICP transactions for all entities in the Entity list against all entities in the Partner list. The system only processes transactions for entities that have the ICP attribute enabled.

Selecting Accounts for Intercompany Matching Reports

You must define the account and matching accounts for the report. If you would like the system to generate the account and matching account based on the plug account, you can specify the plug account option and the system automatically generates the account and matching account.

Specifying Decimal Places in Intercompany Matching Reports

You can add the Decimal keyword to an Intercompany Matching report definition to specify the number of decimal places to display in the report. The possible values for the decimal are default, 0-9. If you do not specify a decimal value, the system uses the default decimal setting as defined in the account.

Selecting Style Sheets for Intercompany Matching Reports

You can select a style sheet from a drop-down list when you are creating a report. You can also add the StyleSheet keyword to the report definition to specify the style sheet to use for the report; for example, `StyleSheet=HFM_IntercompanyDefault.xsl`. If you do not specify a style sheet in the report definition, the system uses the default style sheet.

Specifying Currencies in Intercompany Matching Reports

You can generate an Intercompany Matching report in a currency that you specify. This enables you to run the report and validate intercompany transactions in a common currency before the amounts are consolidated. For example, to check the values in the EUR currency, you could change the Value in the Point of View from USD to EUR and generate the report.

If you define a report using a currency that has not been translated, the system performs the translation process using the translation rules defined in the Sub Translate section of your rule file. The system also stores the translated amounts in the corresponding currency Value dimension member. However, if the reporting currency has previously been translated and the translation status of the entity is OK, the system does not need to re-translate and uses the stored translated amounts for processing the Intercompany Matching report.

For example, if you run an Intercompany Match report for the currency EUR, the system first checks if the translated data has been created for the EUR currency (V#EUR). The system also checks to ensure that translation status is OK. If the translation status of the entity is TR (requires translation), the system re-translates to ensure that the translated data is valid. Otherwise, the system uses the stored translated amounts for processing. However, if the data has not yet been translated to EUR, the system performs the translation process as defined in Sub Translate and stores the translated amounts in the EUR value member. The EUR translated amounts are also used for the Intercompany Matching report.

Suppression Options for Intercompany Matching Reports

When you create an Intercompany Matching report, you can suppress several types of intercompany transactions and detail from the report. You can also select these suppression options when you print the report.

Suppressing Matches

When you create an Intercompany Matching report, you can show or suppress matching Entity/ Partner transactions. If you select to suppress them, the system suppresses the transactions if the Entity and Partner amounts are within the matching tolerance amount or percentage. For example, if an Entity has an amount of 299 and the Partner has 200, the difference is 99. If the matching tolerance amount is 100 and the difference between the Entity and Partner is less than 100 as in this example, the system suppresses the transactions because it is within the matching tolerance.

If you select not to suppress the Entity/Partner transactions, the system does not suppress them even if the Entity and Partner amounts are within the matching tolerance amount or percentage. Using the previous example, even if the difference amount is 99 and if it is within the matching tolerance, the system does not suppress the transactions. The matching tolerance specified is displayed in units.

Suppressing Reversed Transactions

By default, the system displays the reversed transactions from an Intercompany Partner every time a transaction is displayed for the Entity. You can select to suppress these reversed transactions when you create a report.

This option is very useful when you have one matching account for the report. For example, you might have a “Cash” account used to store intercompany transactions for Revenue and Expense. In this case, you must use the Suppress Reversed Transaction option to avoid a double entry because there is only one matching account for the report.

Suppressing Details

For an Intercompany Matching report to display only the difference amount, you can suppress the intercompany details when you create a report. When you select this option, the report does not display the intercompany transactions and prints only the total difference for each Entity/ Partner section. If there is a discrepancy and you need to view each intercompany transaction, you can regenerate the report and show intercompany details.

Suppressing Custom Dimensions

You can select to suppress the columns for the Custom 1, 2, 3 or 4 dimensions.

Member Display Option

You can display the label, description, or both for the dimension member in the report.

Group By Option

You can group your intercompany partner transactions by Custom 1, 2, 3, or 4. The system sorts the details based on this option and provides a subtotal for the group.

Defining Intercompany Partner Matching Report Scripts

You use the keywords in this section to define Intercompany Partner Matching report scripts. After you create a script, save it with the RPT file name extension.

Sample Intercompany Partner Matching report scripts are included when you install Sample Applications for Financial Management. The files are located in the Sample Apps folder in the directory to which you installed Financial Management.

Note: Intercompany Partner Matching report script keywords are not case-sensitive.

ReportType

This keyword specifies the report type. This keyword is required in the script.

Syntax

`ReportType=Intercompany`

ReportDescription

This keyword specifies the description for the report. The report description can contain a maximum of 40 characters. This keyword is required in the script.

Syntax

`ReportDescription=ReportDescription`

Replace *ReportDescription* with the description for the report. For example:

`ReportDescription=Intercompany Elimination Report`

StyleSheet

This keyword specifies the style sheet to use for the report.

Syntax

`StyleSheet=StyleSheetFileName`

Replace *StyleSheetFileName* with the style sheet for the report. For example:

`ReportDescription=HFM_IntercompanyDefault.xsl`

If you do not specify a style sheet in the report definition, the system uses the default style sheet.

POV

This keyword specifies the point of view for the report. This keyword is required in the script.

Syntax

```
POV=S#Scenario.Y#Year.P#Period.V#Value.W#View
```

Replace *Scenario*, *Year*, *Period*, *Value*, and *View* with valid dimension members. For example:

```
POV=S#Actual.Y#2009.P#July.V#USD.W#YTD
```

Note: Adjs value members, for example parent curr adjs, are not supported.

Entity

This keyword specifies the entity or entity member list to be displayed on the report.

Syntax

```
Entity=E#Parent.Entity  
Entity=E{EntityList}
```

Replace *Entity.Parent* with the entity-parent combination. Replace *EntityList* with the name of a valid member list. For example:

```
Entity=E#UnitedStates.Connecticut  
Entity=E{Geographical.[Base]}
```

Partner

This keyword specifies the partner or partner member list to be displayed on the report.

Syntax

```
Partner=E#PartnerParent.PartnerEntity  
Partner=E{PartnerList}
```

Replace *PartnerParent.PartnerEntity* with the partner parent-entity combination. Replace *PartnerList* with the name of a valid partner member list. For example:

```
Partner=E#UnitedStates.Florida  
Partner=E{Geographical.[Base]}
```

AccountEntity and AccountPartner

The *AccountEntity_x* and *AccountPartner_x* keywords specify the accounts for matching. For each account pair to match, you specify *AccountEntity_x* and *AccountPartner_x* starting with zero. To create a One to Many or Many to Many matching report, specify additional accounts using the correct keyword. You cannot use duplicate keywords within one report. For example, you should not have *AccountEntity_0* more than once within one report.

Syntax

```
AccountEntity_0=A#Sales.C1#GolfBalls.C2#Customer2
```

```
AccountPartner_0=A#Purchases.C1#GolfBalls.C2#Customer2
```

This example uses these keywords to show all accounts in one matching report:

```
AccountEntity_0=A#1004780
AccountEntity_1=A#1004790
AccountEntity_2=A#1005850
AccountEntity_3=A#1005850
AccountPartner_0=A#2000100
AccountPartner_1=A#2000140
AccountPartner_2=A#2000210
AccountPartner_3=A#2000250
AccountPartner_4=A#2000320
AccountPartner_5=A#2000430
AccountPartner_6=A#2000560
AccountPartner_7=A#2000630
AccountPartner_8=A#2000680
```

This example uses the Entity and Partner keywords to create different reports with different account pairs. This is an example of one report with one account matching two accounts (one to many):

```
Report 1
AccountEntity_0=A#1004780
AccountPartner_0=A#2000100
AccountPartner_1=A#2000140
```

This example shows one report with two accounts matching one account (many to one):

```
Report 2
AccountEntity_0=A#1004790
AccountEntity_1=A#2000210
AccountPartner_1=A#2000250
```

Report 3 is an example of one report with one account matching with one account (one to one).

```
Report 3
AccountEntity_0=A#1005850
AccountPartner_0=A#2000320
```

Report 4 is an example of one report with two accounts matching with four accounts (many to many).

```
Report 4
AccountEntity_0=A#1005850
AccountEntity_1=A#1005860
AccountPartner_0=A#2000430
AccountPartner_1=A#2000560
AccountPartner_2=A#2000630
AccountPartner_3=A#2000680
```

SuppressIfMatch

This keyword suppresses transactions if the entity and partner amounts are within the matching tolerance amount.

Syntax

`SuppressIfMatch=Boolean`

Replace *Boolean* with Yes or No. For example:

`SuppressIfMatch=Yes`

For example, if an Entity has an amount of 299 and the Partner has 200, the difference is 99. If the matching tolerance amount is 100 and the difference between the Entity and Partner is less than 100, as in this example, the system suppresses the transactions because it is within the matching tolerance.

If you select not to suppress the Entity/Partner transactions, the system does not suppress these even if the Entity and Partner amounts are within the matching tolerance amount. Using the previous example, even if the difference amount is 99 and if it is within the matching tolerance, the system does not suppress the transactions.

Note: Matching tolerance is specified in units.

SuppressReversedTransactions

This keyword suppresses reversed transactions from partners for each corresponding entity transaction.

Syntax

`SuppressReversedTransactions=Boolean`

Replace *Boolean* with Yes or No. For example:

`SuppressReversedTransactions=Yes`

By default, the system displays the reversed transactions from an Intercompany Partner every time a transaction is displayed for the Entity. You can select to suppress these reversed transactions when you create a report.

This option is useful when you have one matching account for the report. For example, if you have one “wash” account to state intercompany transactions for Revenue and Expense. In this case, you must use the suppress reversed transactions option to avoid the double entry because there is only one matching account for the report.

SuppressDetails

This keyword suppresses intercompany detail and prints only the Total difference for each Entity/Partner section.

Syntax

`SuppressDetails=Boolean`

Replace *Boolean* with Yes or No. For example:

`SuppressDetails=Yes`

For an Intercompany Matching report to display only the difference amount, you can suppress the intercompany details when you create a report. When you select this option, the report does not display the intercompany transactions and only prints the Total difference for each Entity/ Partner section. If there is a discrepancy and you need to view each intercompany transaction, you can regenerate the report and show intercompany details.

MatchingTolerance

This keyword lets you specify a value to see only out of balance transactions over a certain amount, or use the default value of 0.

Note: Matching tolerance is specified in units.

Syntax

```
MatchingTolerance=ToleranceValue
```

Replace *ToleranceValue* with a number that is less than 1 billion. The limit for this keyword is 999999999. For example:

```
MatchingTolerance=100
```

For example, if an Entity has an amount of 299 and the Partner has 200, the difference is 99. If the matching tolerance amount is 100 and the difference between the Entity and Partner is less than 100 as in this example, you can use the SuppressIfMatch keyword to have the system suppress the transaction because it is within the matching tolerance.

SuppressCustoms

This keyword suppresses custom dimensions.

Syntax

```
DisplayCustomn=Boolean
```

Replace *n* with the number of the custom dimension and replace *Boolean* with Yes or No. The default is Yes. For example:

```
DisplayCustom1=Yes  
DisplayCustom2=Yes  
DisplayCustom3=No  
DisplayCustom4=No
```

ScaleFactor

This keyword specifies the unit in which amounts are displayed by identifying where the decimal point is placed. For example, if you enter a scale factor of 3, the report amount is displayed in thousands. If the scale factor is 0, the report amount is displayed in units.

Syntax

`ScaleFactor=Scale`

Replace *Scale* with one of these numbers:

- 0 = Units
- 1 = Tens
- 2 = Hundreds
- 3 = Thousands
- 4 = Ten Thousands
- 5 = Hundred Thousands
- 6 = Millions
- 7 = Ten Millions
- 8 = Hundred Millions
- 9 = Billions

For example:

`ScaleFactor=3`

In this example, the number 12345.78 is displayed as 12.345678 on the report.

Decimal

This keyword specifies the number of decimal places to display in the report and can override the number of decimal places defined in the account.

Syntax

`Decimal=NumberDecimalPlaces`

Replace *NumberDecimalPlaces* with a number 0-6. If you do not specify a decimal value, the system uses the default decimal setting as defined in the account. For example:

`Decimal=3`

In this example, the number 123.4567 is displayed as 123.457.

DisplayLabels

This keyword specifies if member labels are displayed on the report.

Syntax

`DisplayLabels=Boolean`

Replace *Boolean* with Yes or No. The default is Yes. For example:

`DisplayLabels=Yes`

DisplayDescriptions

This keyword specifies if member descriptions are displayed on the report.

Syntax

`DisplayDescriptions=Boolean`

Replace *Boolean* with Yes or No. The default is No. For example:

`DisplayDescriptions=Yes`

DisplayPlugElimAccts

This keyword specifies if a summary of plug accounts affected by the intercompany transactions is displayed.

Syntax

`DisplayPlugElimAccts=Boolean`

Replace *Boolean* with Yes or No. For example:

`DisplayPlugElimAccts=Yes`

GroupByCustom

This keyword groups custom dimension transactions by Custom 1, Custom 2, Custom 3, or Custom 4 dimension. This keyword is optional.

Syntax

`GroupByCustom=Custom`

Replace *Custom* with the number of the custom dimension by which to group the Custom dimensions in the report. For example:

`GroupByCustom=1`

Creating Intercompany Transaction Reports

You can create these types of intercompany transaction reports:

- IC Transactions - create a list of transactions.
- IC Matching by Account - create matching reports based on accounts selected.
- IC Matching by Transaction ID - create matching reports based on transaction ID.
- IC Template

See [“IC Transaction Report Syntax” on page 204](#).

IC Transaction Report Syntax

These keywords are used in creating an Intercompany Transaction report script.

Table 33 Keywords for IC Transaction Report Scripts

Keyword	Description
ReportType	Specify the report type. For example, ReportType=ICTransactions
ReportLabel	Specify the report name. For example, ReportLabel=Transaction Report
ReportDescription	Specify a report description. For example, ReportDescription=Intercompany Transaction Detail Report
ReportSecurityClass	Specify the security class for the report. The default is [Default].
POV	Specify a valid point of view for the report. For example, POV=S#ActMon.Y#2009.P#January
ScaleFactor	Specify a scale factor for the report. The scale can be a value from 0 to 9.
Decimal	Specify the number of decimals to display in the report. The number of decimals can be a value from 0 to 9.
IncludeMatched	Specify True to include matched transactions in the report, otherwise False.
IncludeUnMatched	Specify True to include unmatched transactions in the report, otherwise False.
IncludeMisMatched	Specify True to include mismatched transactions in the report, otherwise False.
IncludePosted	Specify True to include posted transactions in the report, otherwise False.
IncludeUnPosted	Specify True to include unposted transactions in the report, otherwise False.
Entity	The entity must be a valid ICP base entity, not a parent entity.
Partner	The partner entity must be a valid ICP entity for the account.
Entity Account	If you are displaying Entity transactions, specify an entity account.
Partner Account	If you are displaying Partner transactions, specify a partner account.
TransactionID	This ID is required. You must enter an ID for the transaction, with a maximum of 40 characters. When combined with the Sub ID, this ID becomes a unique identifier for the Entity/Partner/Account/C1/C2/C3/C4 within the Scenario/Year/Period.
Transaction Sub ID	Specify a transaction Sub ID.
TransactionCurrency	This is the currency used for the Invoice transaction. It must be a valid currency defined in the application.
ReferenceID	This is optional. You can enter a Reference ID to store reference information for the transaction. For example, the entity might have its own set of invoice numbering that is different from the entity that issued the invoice. You can enter additional information in this Reference ID for information purposes only. You can enter the entity reference ID in the Transaction ID and enter the corresponding invoice number from the partner entity.

Keyword	Description
MatchCode	This is optional. The match code must be one of these prefixes to distinguish the different types of matching processes: ● A - Auto-matching performed using Accounts ● I - Auto-matching performed using the Transaction ID ● R - Auto-matching performed using the Reference ID ● M - Manual matching performed
TransactionAmount	Specify a range of transaction amounts.
ReasonCode	This is optional. The reason code must be a valid reason code defined by the administrator. The main purpose of the reason code is to indicate why a transaction has a MisMatched status - for example, because of a missing invoice from the partner entity, or an incorrect amount entered by the partner. If the transaction has a Matched status, you do not need to assign a reason code for the transaction. You cannot assign a reason code to transactions with an UnMatched status.
FromDate	This is optional. This must be a valid date.
ToDate	This is optional. This must be a valid date.
DisplayColumns Section	Specifies the columns that are displayed in the report and how they are displayed. Syntax is: <Displayed Columns>, <Sort>, <Options>, <Repeat>, <Totals> where <Displayed Columns> is a valid column, <Sort> is Ascending or Descending, <Options> is Label, Description or Both, <Repeat> is Repeat or NoRepeat, and <Totals> is Total or NoTotal. For example: Entity, Ascending, Label, Repeat, Total
DisplayEntityTransactions	Specify True to display entity transactions in the report, otherwise False. You can select to display only intercompany transactions for a specific entity and partner, or also display the corresponding transactions from the partner with the entity. For example, if you select only the transactions for Entity A with Partner B in the Entity and Partner selections, the system displays only the transactions that Entity A has with Partner B. However, if you would like to see the corresponding transactions for Entity B with Partner A, you can select to include Entity transactions and Partner transactions.
DisplayPartnerTransactions	Specify True to display partner transactions in the report, otherwise False.
Suppress Details	Specify True to suppress transaction detail and display only the subtotal row.

In This Chapter

Rule Types	208
Rule Considerations	209
Rule Execution During Consolidation.....	217
Default Translation	218
Financial Management Objects.....	219
Commonly Used Rules	220
Creating Rules in the Rules Editor.....	223
Creating Rules Files.....	225
Loading Rules	227
Extracting Rules	228

You use Financial Management rules to automate the calculations of data within an application. You can use rules for these purposes:

- Calculate data entry level amounts for a specific entity, scenario, and period.
- Prevent data entry for a specific cell in a specific entity, scenario, and period.
- Allow input at the Parent entity level.
- Calculate data that cannot be calculated through a hierarchical aggregation, such as ratios or variance analysis.
- Perform allocations from a parent entity to a list of base entities.
- Perform complex currency conversions, calculate exchange rate differences, or perform other calculations necessary for your consolidation.
- Define formulas to dynamically calculate accounts.
- Specify the accounts in the application that support intercompany transactions.

You can write rules in a text editor, or you can use the Financial Management Rules Editor to create rules.

Note: To work with rules in Calculation Manager, see [Chapter 12, “Creating Rules Using Calculation Manager”](#).

Rule Types

You can write rules that set and clear values, calculate data, translate currency, consolidate data, allocate data from one location to another, and prevent data input.

Table 34 Rule Types

Rule Type	Description
Calculation	Calculation rules run when users run calculations. You can use Calculation rules to perform calculations that cannot be calculated through the natural order of the dimension hierarchies. For example, you can create calculations to derive ratios or opening balances. The Calculate() routine is executed when you calculate or consolidate data.
Translation	Translation rules run when users run translations. You can use Translation rules to perform calculations related to non-standard translations. The Translate() routine is executed when you translate or consolidate data. For example, if the application is using the default ClosingRate to translate Assets and LIABILITY accounts and AverageRate to translate REVENUE and EXPENSE accounts, you may want to use a different translation rate to calculate the translation difference related to Net REVENUE. Financial Management executes Translation rules in these cases: ● When a user runs a currency translation by right-clicking in the Explore Data module and selecting Translate or Force Translate from the menu that is displayed. This is useful for performing translations as you enter data. ● When a user performs a consolidation and a parent entity's default currency is different from a child entity's default currency.
Consolidation	Consolidation rules run when users run consolidations. You can use consolidation rules to perform non-standard consolidations, most commonly found in statutory applications. The Consolidate() routine is executed when you consolidate data.
Allocation	Allocation rules allocate data from one entity to a list of entities. For example, you can use Allocation rules to calculate administrative expenses for a list of entities based on the total administrative expenses for some other entity.
Input	Input rules allow input at the Parent entity level. Only the entity currency Value dimension is supported. Note that contribution values from children to the Parent entity's entity currency Value dimension are not rolled up. The contribution value for the Parent.Child combination is stored.
NoInput	NoInput rules prevent input at the Base entity level, thus reserving the cells for calculations. You can use the NoInput function multiple times in a NoInput rule to prohibit data input into several non-adjacent cells. These limitations and guidelines apply to NoInput rules: ● Only these Financial Management functions are supported for NoInput rules: ○ NoInput ○ List Note: All VBScript methods are supported for NoInput rules. ● For the List function, fixed lists, system lists, and dynamic lists are supported. A dynamic list can reference metadata attributes supported by the List function. ● An If...Then structure can test for metadata attributes. However, NoInput rules do not support testing of members in the current Point of View. For example, you cannot test If HS.Entity.Member = "CT". ● Be careful when using the NoInput function in loops. A few simple statements with loops may end up loading thousands of cells in memory, so be sure to test the performance impact of loops that include NoInput.

Rule Type	Description
Dynamic Calculation	Dynamic rules enable you to define formulas to dynamically calculate accounts. You can dynamically calculate Base accounts only. You cannot use Dynamic rules on Parent accounts. Use these guidelines for writing dynamic calculation rules: ● The right side of the equation must reference the same Scenario/Year/Entity combination. This means you cannot reference prior year amounts in your calculations. ● Only dynamic accounts are valid on the left side of the equation. ● Dynamic accounts cannot be used on the right side of the equation. ● Only Account and View are valid on the left side of the equation. ● If View is not specified, the calculation executes for YTD and Periodic. If View is specified, the calculation is executed only for the specified view. ● HS.View.PeriodNumber is the only HS statement that can be used in a HS.Dynamic calculation. ● All statements in the Sub Dynamic section are executed sequentially.
Transactions	Transactions rules specify the accounts in the application that support intercompany transactions. Cells supporting transactions are read-only in Web grids and data forms.
Equity Pickup	Equity pickup rules specify the owned entity, owner entity, and percentage of ownership This is the default point of view when the SUB EquityPickup section is run: ● Current scenario, year, and period ● Entity: owner of the pair processed ● Value: <Entity Currency>

You create rules for Financial Management in a unique script, which is based on the Microsoft VBScript language. Rules are constructed through the combination of functions, objects, and other arguments to generate the desired scripting syntax. Within each routine, you use two types of functions to write rules:

- Financial Management functions that define calculations
- VBScript functions that are used to write conditional statements

Rule Considerations

Following are considerations for writing rules for applications:

- Rules are executed when users perform calculations, translations, consolidations, and allocations. Calculation rules execute one time for each Entity/Value dimension intersection to which the calculation or consolidation applies. See [“Calculation Rules with Calculation Commands” on page 210](#).
- The dimension members to which Financial Management applies calculations depend on the data grid cell where the user’s cursor is placed and the members specified in the Point of View bar. See [“Current Dimension Members” on page 214](#).
- When a function puts data into a currency-related Value member, Financial Management might delete the current value in the Value member. See [“Functions Automatically Clear Data” on page 216](#).

- Rules execute in sequential order within a routine and there is a set order in which routines run before other routines. See [“Rule Execution During Consolidation” on page 217](#).

Calculation Commands

Calculate routines are executed for a given intersection of scenario, year, period, entity, and value. Executing Calculate routines results in writing or clearing data in the current data table, which corresponds to the current intersection of scenario, year, period, entity, and value. When users execute Calculate routines, the system can read data from anywhere in the application. However, data is only written to the current data table.

Note: The calculate routine will fail if you have the [None] entity in a hierarchy when you consolidate.

Users run these processes in data grids by selecting one of these commands:

- Calculate
- Force Calculate
- Calculate Contribution
- Force Calculate Contribution
- Consolidate
- Consolidate All With Data
- Consolidate All
- Translate
- Force Translate

When a user selects a calculation command, Financial Management executes the Sub Calculate() routine in the RLE file. The Calculate() routine calculates accounts and custom dimension members for a given Entity-Value combination within a given Scenario, Year, and Period.

Force Calculate

The Force Calculate option forces rules to run only on the Value member selected, and any Value member on which it depends. For example, Force Calculate on entity currency runs rules on the entity currency member only. Force Calculate on the entity currency total runs rules on entity currency, entity currency adj, and entity currency total. Force Calculate is the only command for which it is possible to affect only a single member of a value triplet.

Calculation Rules with Calculation Commands

The number of times that a Calculation rule is executed depends upon the calculate command selected by the user and by other factors.

Caution! Carefully read these sections before writing rules. You might want an operation to occur only for certain members of the Value dimension, and if this is the case you must test for the current member with VBScript's If structure and the `Member` function before executing the operation. For steps and examples on using If structures and the `Member` function, see [“Conditional Rules” on page 221](#).

When a user selects the Calculate or Force Calculate command, Financial Management runs the application's Calculation rule for the intersection of the current entity member and the Value member for the entity's default currency, Entity Currency. If the entity's AllowAdjs attribute is enabled in the metadata, Financial Management also runs the rule a second time, applying the rule to the intersection of the entity and the member of the Value dimension that stores adjustments to the entity's default currency, Entity Curr Adjs.

Example

For example, if an entity named California has a default currency of USD and its AllowAdjs attribute is enabled in the metadata, the calculation rule runs twice, once for the intersection of California with USD and once for the intersection of California with USD Adjs.

Calculation Rules with Consolidation Commands

When a user selects one of the Consolidate commands, Financial Management runs the Calculation rule for several of the Value dimensions that intersect each previously unconsolidated child entity. After executing for the children, Financial Management runs the rule for the intersection of the parent entity and the member of the Value dimension for the parent's default currency.

1. The rule is run for the intersection of the child entity and the Value member that stores the child entity's default currency (Entity Currency).
2. If the child entity's AllowAdjs attribute is set to Y, the rule is run for the intersection of the child entity and the Value member that stores adjustments in the child entity's default currency (Entity Curr Adjs).
3. If the child's currency differs from the parent's currency, the rule is run for the intersection of the child entity and the Value member that stores amounts translated to the parent's home currency (Parent Currency).
4. If a child's currency differs from the parent's currency and the rules file contains a Translation rule, Financial Management executes the Translation rule before step 3.
5. If the child's currency differs from the parent's currency, and the child entity's AllowAdjs attribute is set to Y, the rule is run for the intersection of the child entity and the Value member that stores adjustments translated to the parent's home currency (Parent Curr Adjs).
6. If the parent entity's AllowAdjFromChildren attribute is set to Y, the Logic rule is run for the intersection of the child entity and the Parent Adjs value.
7. The rule is run for the intersection of the child entity and the Proportion value.
8. The rule is run for the intersection of the child entity and the Elimination value.

9. If the parent entity's AllowAdjFromChildren attribute is set to Y, the rule is run for the intersection of the child entity and the Contribution Adjs value.
10. For each additional child entity that contains previously unconsolidated data, repeat steps 1 through 9.
11. The rule is run for the intersection of the parent entity and the Value member that stores the parent entity's default currency.

Example

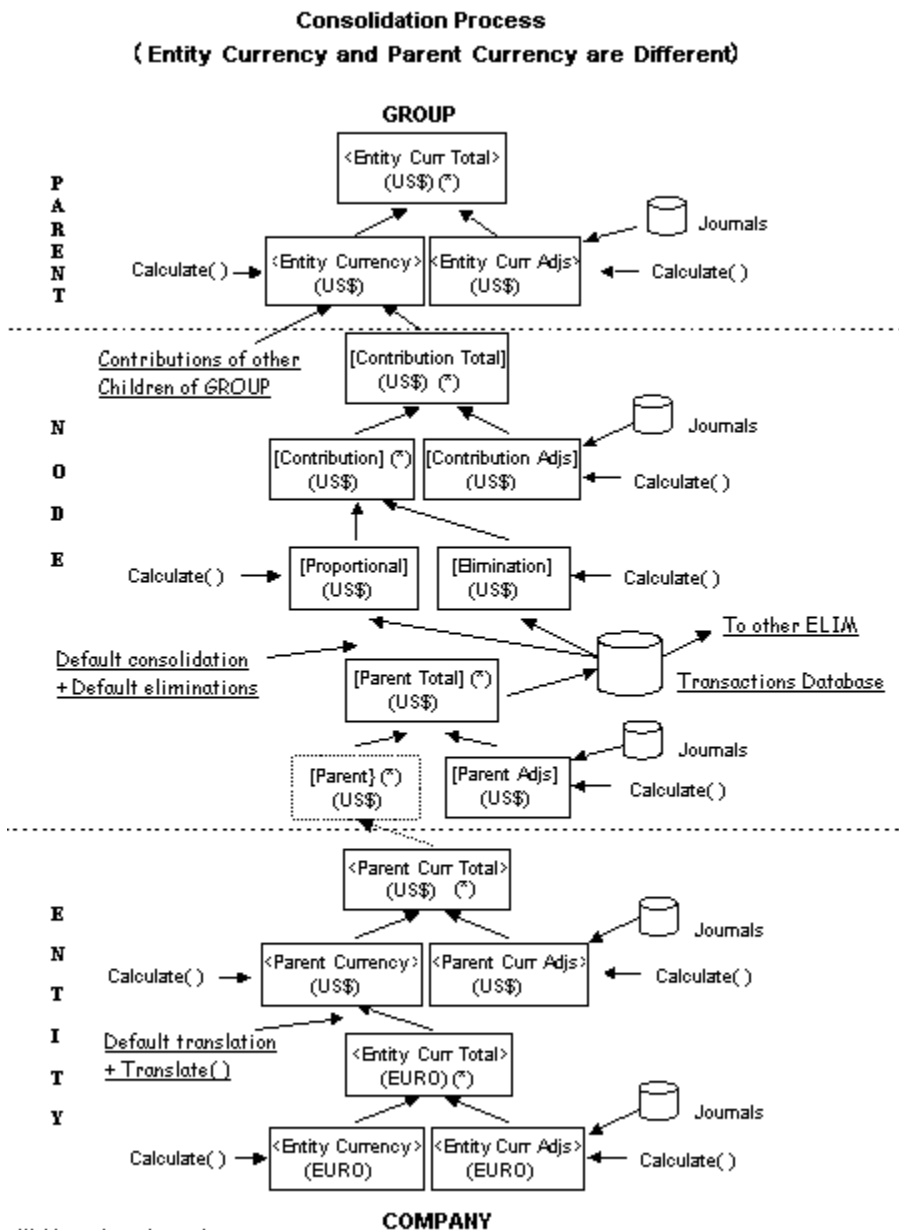
For example, a parent entity named UnitedStates has children named EastUS and WestUS. The children have the AllowAdjs attribute enabled. The UnitedStates entity has the AllowAdjs and the AllowAdjFromChildren attributes enabled. All three entities share a default currency of USD.

If you change data for EastUS and WestUS and consolidate UnitedStates, Financial Management runs the rule for each of these intersections of the Entity and Value dimensions:

1. EastUS and Entity Currency.
2. EastUS and Entity Currency Adjs. (EastUS's AllowAdjs attribute has been set to Y.)
3. EastUS and Parent.
4. EastUS and Parent Adjs. (UnitedStates' AllowAdjFromChildren attribute has been set to Y.)
5. EastUS and Proportion.
6. EastUS and Elimination.
7. EastUS and Contribution Adjs. (UnitedStates' AllowAdjFromChildren attribute has been set to Y.)
8. WestUS and Entity Currency.
9. WestUS and Entity Currency Adjs. (WestUS's AllowAdjs attribute has been set to Y.)
10. WestUS and Parent.
11. WestUS and Parent Adjs. (UnitedStates' AllowAdjFromChildren attribute has been set to Y.)
12. WestUS and Proportion.
13. WestUS and Elimination.
14. WestUS and Contribution Adjs. (UnitedStates' AllowAdjFromChildren attribute has been set to Y.)
15. UnitedStates and Entity Currency.

Following are examples of the consolidation process.

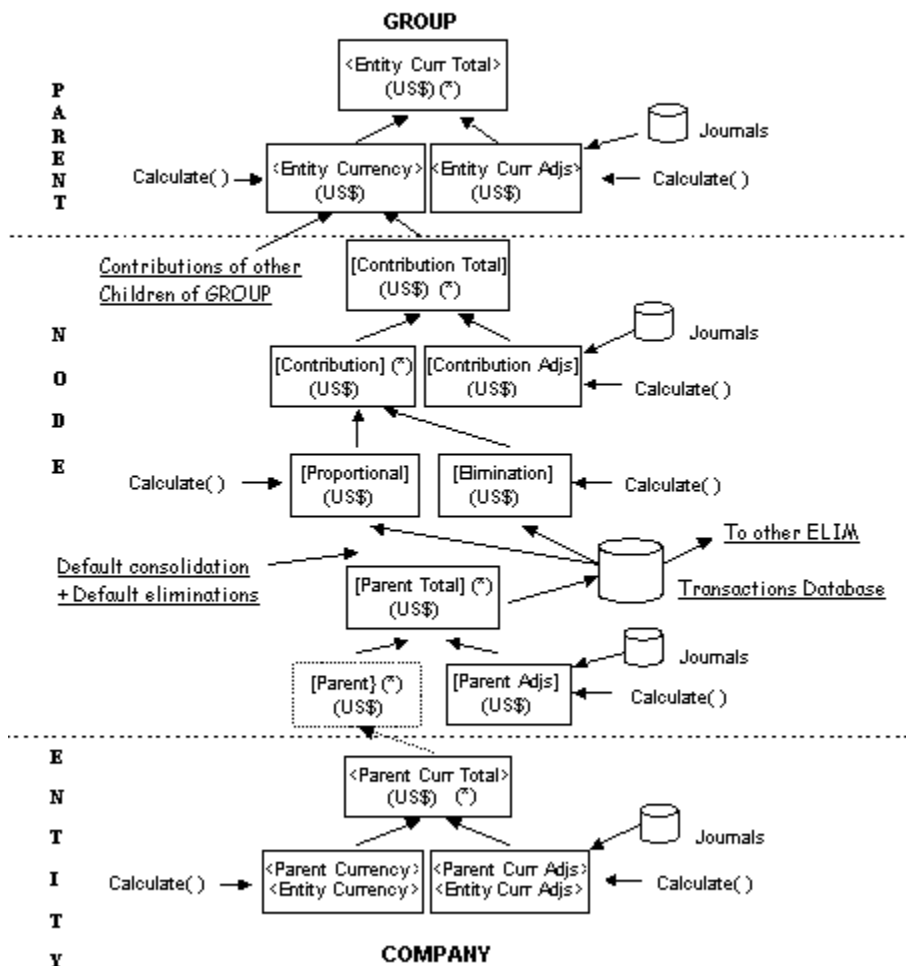
The first example shows the process when the entity currency and the parent currency are different.



(*) Non-stored members

The following example shows the process when the entity currency and the parent currency are the same.

Consolidation Process (Entity Currency and Parent Currency are the Same)



(*) Non-stored members

Current Dimension Members

By default, Financial Management applies Calculation rules to the current dimension members at the time that the user selects a calculation command. See [“Calculation Rules with Calculation Commands” on page 210](#). The current dimension members are determined by these factors:

- The cell in the data grid in which the user’s cursor is placed when the user runs a calculation, translation, or consolidation.
- The dimension members specified in the Point of View bar.
- For consolidations, the current members of the Entity and Value dimensions change each time the rule is executed. See [“Calculation Rules with Consolidation Commands” on page 211](#).
- Rules process for the currently selected dimension members, except for calculation rules in which case the calculation rules process across all accounts in the application.

If a dimension member of the cell in which the cursor is placed differs from the corresponding member of the corresponding dimension in the Point of View bar, the cell's dimension member overrides the Point of View bar's dimension member. For example, if a user has specified an entity named Europe in the Point of View bar and runs a translation with the cursor placed in a cell for an entity named Germany, Germany is the current entity.

By specifying a dimension member as an argument, you can use some functions to work with a specific dimension member regardless of the current dimension. For example, you can use the NumBase function with the Entity object to get the number of base entities for the entity that you specify in the argument.

For more complex functions, you can create Account Expressions to specify the dimension members.

Account Expressions

Some functions require an Account Expression as an argument. In its simplest form, an Account Expression is a String that specifies the account to which Financial Management applies the function.

The Account Expression characters are listed in the following table.

Table 35 Characters Used to Specify Dimensions in Account Expressions

Character	Dimension
A#	Account
I#	Intercompany Partner
C1#	Custom1
C2#	Custom2
C3#	Custom3
C4#	Custom4
S#	Scenario
Y#	Year
P#	Period
W#	View
E#	Entity
V#	Value

To understand Account Expressions, consider the Clear function, which removes values from the dimension members specified in the function's argument. In this example, the argument "A#Sales" is an Account Expression:

```
HS.Clear "A#Sales"
```

The A# characters represent the Account dimension, and the word Sales is the member name of the Account dimension to which Financial Management applies the Clear function. In other words, this Account Expression tells Financial Management to clear the data stored in the Sales account.

When you use an Account Expression, Financial Management applies the function to the intersection of the account that you specify in the Account Expression and the current members of these dimensions:

- Entity
- Period
- Scenario
- Value
- View
- Year
- Custom - Uses the CustomTop member that was set for the account in the metadata. For example, if the Account Expression does not specify a member of the Custom 2 dimension, Financial Management uses all valid Custom 2 members as defined by the CustomTop member specified for the account.
- Intercompany Partner - All valid ICP members.

You can override the Intercompany and Custom dimension defaults by specifying members in the Account Expression. Each dimension is represented by certain characters. When you include more than one dimension in an Account Expression, you must separate the dimensions with periods.

When you create an Account Expression, you do not have to specify all of these dimension members; you can specify the members to which to apply the function. For example, this line clears the intersection of the Sales account and the Hardware ICP:

```
HS.Clear "A#Sales.I#Hardware"
```

Functions Automatically Clear Data

When a function puts data into a Value member that relates to currencies, Financial Management automatically clears data from the member if either of these conditions apply:

- The intersecting Entity member is a parent.
- The intersecting Account member is a calculated account.

Tip: The currency-related Value members are the system-generated Entity Currency member and the user-defined currency members such as USD, EURO, LIRA.

In addition, if a function puts data into a Value member for adjustments, Financial Management automatically clears data from the member if the intersecting Account member is a calculated account.

Tip: The adjustment-related Value members are those that include the code “Adjs” in their name, for example, USD Adjs, EURO Adjs, LIRA Adjs.

Error Messages

When Financial Management detects a syntax error, it displays an error message that contains this information:

- The line number in the RLE file that is causing the error.
- The applicable Financial Management object and function.
- An error description.

Rule Execution During Consolidation

During the consolidation process, rules are executed in a pre-defined sequence. For each base child of a given parent, the calculation sequence for the various elements in the Value dimension takes place in this order:

1. Accounts defined as IsCalculated in the metadata are cleared in EntityCurrency.
2. Accounts defined as IsCalculated in the metadata are cleared in EntityCurrAdjs.
3. The Sub Calculate() routine is executed on EntityCurrency.
4. The Sub Calculate() routine is executed on EntityCurrAdjs.
5. The ParentCurrency data is cleared.
6. Default translation is applied to all accounts defined as Revenue, Expense, Asset, Liability for the total amount of EntityCurrency and EntityCurrAdjs. For accounts with attribute Flow or Balance, the total amount of EntityCurrency and EntityCurrAdjs is rolled up into Parent Currency.
7. The Sub Translate() routine is executed.
8. The Sub Calculate() routine is executed on ParentCurrency.
9. Accounts defined as “IsCalculated” in the metadata are cleared in ParentCurrAdjs.
10. The Sub Calculate() routine is executed on ParentCurrAdjs.
11. Accounts defined as “IsCalculated” in the metadata are cleared in ParentAdjs.
12. The Sub Calculate() routine is executed on ParentAdjs.
13. Proportion and Elimination data are cleared.
14. Default consolidation and eliminations are performed for the total amount of Parent and ParentAdjs.
15. The Sub Calculate() routine is executed on Proportion and Elimination.
16. Accounts defined as “IsCalculated” in the metadata are cleared in ContributionAdjs.
17. The Sub Calculate() routine is executed on ContributionAdjs.

After the previous steps have been repeated for each base child, this sequence takes place for the parent entity:

1. The EntityCurrency data is cleared.
2. The sum of the total of Proportion, Elimination, and ContributionAdjs for every child is written into EntityCurrency of the parent entity.
3. The Sub Calculate() routine is executed on EntityCurrency.
4. Accounts defined as “IsCalculated” in the metadata are cleared in EntityCurrAdjs.
5. The Sub Calculate() routine is executed on EntityCurrAdjs.

Note: If a parent is further consolidated into another parent, this sequence continues with step 5 from the child consolidation sequence.

Default Translation

Following is the sequence in which default translation takes place.

1. The system checks the current entity for the direct translation rate and uses that rate for translation.
2. If the translation rate is not found, the system derives the direct rate from the indirect rate in the current entity.
3. If neither the direct rate nor the indirect rate is found in the current entity, the system looks at the [None] entity and uses the direct rate.
4. If the direct rate is not found in the [None] entity, the system derives the direct rate from the indirect rate in the [None] entity.
5. If the indirect rate for the [None] entity does not exist, the system derives the rate by triangulation using the application currency in the [None] entity.
6. If triangulation fails, the entity is not translated.

Note: The system first looks for a translation rate within the current entity. If not found in the current entity, the system looks for a translation rate within the [None] entity. If the system cannot find a translation rate in the [None] entity, the system translates using triangulation. Triangulation is a way to convert balances from one currency to another using a third, common currency.

For example, if you want to convert EURO to YEN, but the system cannot find a direct or indirect rate to perform the translation, if EURO and YEN can both translate into USD then, using triangulation, the system can convert the EURO balance to USD and then convert the USD balance to YEN.

Financial Management Objects

This section explains the syntax you must use to represent Financial Management objects when using functions.

HS Object

The top-level object in Financial Management is the HS object. This means that when you use a Financial Management function, the first three characters must be the letters HS followed by a period:

HS.

This example demonstrates how to write the Clear function:

HS.Clear

If you do not precede Clear with the HS. characters, an error occurs.

Caution! You cannot use the HS. characters before a standard VBScript function. If you do, an error occurs. Use the HS. characters only before a Financial Management function.

Some Financial Management functions apply only to objects that are children of the HS object. These functions require you to put the applicable object's name between the HS. characters and the function name. These objects are children of the HS object and are used in Financial Management:

- Account
- AppSettings
- Custom1 through Custom4
- DataUnit
- Entity
- ICP
- Node
- Parent
- Period
- Scenario
- Value
- Year

Commonly Used Rules

These sections show you how to write some simple and commonly used rules. The descriptions of these rules contain step-by-step procedures for readers who are not comfortable with VBScript. These procedures are followed by examples. If you have experience with VBScript, you might prefer to skip the procedures and instead focus on the examples.

Tip: These procedures assume that you have a rules file and code you write is placed in the Calculate() subroutine. See [“Creating Rules Files” on page 225](#).

Reusing Data

Use the EXP function to insert data from one account into another account. EXP’s argument contains the account to be set and the account from which the value is retrieved. The argument is in the form of an equation, with the target account on the left side of the equal sign and the source account on the right side.

Note: EXP inserts data into the intersection of an account with the current dimension members (see [“Current Dimension Members” on page 214](#)). In addition, you can use Account Expression characters to override the current Custom and ICP members.

In this example, the Calculation rule sets the PrevCash account to the value in the Cash account:

```
HS.EXP "A#PrevCash = A#Cash"
```

You can use Account Expression characters to specify dimension members on both sides of the equal sign in EXP’s argument. See [“Exp” on page 262](#). This example inserts the data from the previous year’s intersection of the PrevCash account and the Golf member of the Custom 1 dimension into the current year’s intersection of PrevCash and Golf:

```
HS.EXP "A#PrevCash.C1#Golf = A#Cash.Y#Prior.C1#Golf"
```

Tip: The Prior keyword that follows the Y# Account Expression characters causes EXP to retrieve the previous year’s data. There are several similar keywords that apply to Year and Period in Account Expressions. See [“Period and Year Keywords” on page 263](#).

Setting Accounts by Calculating Amounts

Another common task is to calculate the amounts contained in two accounts and then insert the result into another account. The EXP function supports addition, subtraction, multiplication, and division on the right side of the equal sign in its argument.

In this example, the Calculation rule divides the Sales account’s value by the UnitsSold account’s value, and inserts the quotient in the AvgPrice account:

```
HS.EXP "A#AvgPrice = A#Sales / A#UnitsSold"
```

Conditional Rules

You may want a rule to execute an action only when certain dimension members are the current members in the Point of View. For example, you might want an account's value to be calculated in one way when Actual is the current scenario and a different way when Budget is the current scenario.

Tip: For information on how Financial Management determines the current dimension members, see [“Current Dimension Members” on page 214](#).

To do this, use the Member function in a VBScript `If` structure. Member gets the name of the current member of these dimensions:

- Entity (Use the Entity object to get the current entity or the Parent object to get the parent of the current entity.)
- Period
- Scenario
- Value
- Year

If structures enable you to execute statements only if certain conditions are true. The following sections show a few different ways of using Member with If structures to test for dimension members.

Tip: These sections cover only a few of the If structure aspects. For more details on If structures, consult Microsoft's VBScript documentation. (You can download VBScript documentation from Microsoft's Web site.)

Testing for a Dimension Member

To have Financial Management execute an action only if a particular dimension member is the current member, use an If structure that tests the return value of the Member function.

In this example, if the current scenario is Budget, Financial Management multiplies the amounts in the UnitsSold and Price accounts and inserts the product in the Sales account.

```
If HS.Scenario.Member = "Budget" Then
    HS.EXP "A#Sales = A#UnitsSold * A#Price"
End If
```

Tip: All If structures must begin with an `If . . . Then` statement and end with an `End If` statement. The actions to be executed if the condition is met are sandwiched between the `If . . . Then` and `End If` statements as shown above.

Testing for More Than One Member

You can test for more than one member in an `If . . . Then` statement. In other words, you can execute an action for two or more members of a dimension. Consider the example in the Testing for a Dimension Member section. You might want the Sales account's value to be calculated if the current scenario is Budget or Forecast.

To test for more than one member, use two `Member` functions and VBScript's `Or` keyword in the `If . . . Then` statement. Place `Or` after the first `Member` function, then place the second `Member` function between `Or` and `Then`.

Tip: You can use this technique to test for more than two members. For each member to be tested, include an additional combination of the `Member` function and the `Or` keyword.

In this example, if the current scenario is Budget or Forecast, Financial Management multiplies the amounts in the UnitsSold and Price accounts and inserts the product in the Sales account:

```
If HS.Scenario.Member = "Budget" Or HS.Scenario.Member = "Forecast" Then
    HS.EXP "A#Sales = A#UnitsSold * A#Price"
End If
```

Performing Different Actions for Different Members

You can have a rule perform different actions for different members of a dimension. For example, you might want one calculation to occur if the current scenario is Budget and a different calculation to occur if the current scenario is Actual.

To conditionally perform different actions, include one or more `ElseIf` statements in an `If` structure. Have each `ElseIf` statement test for a different member; place the actions to be performed for a member beneath its `ElseIf` statement.

In this example, different accounts will be updated depending upon whether the current scenario is Budget or Actual:

- If the current scenario is Budget, Financial Management multiplies the amounts in the UnitsSold and Price accounts and inserts the product in the Sales account.
- If the current scenario is Actual, Financial Management divides the Sales account's amount by the UnitsSold account's amount and inserts the quotient in the Price account.

```
If HS.Scenario.Member = "Budget" Then
    HS.EXP "A#Sales = A#UnitsSold * A#Price"
    ElseIf HS.Scenario.Member = "Actual" Then
        HS.EXP "A#Price = A#Sales / A#UnitsSold"
End If
```

Tip: To have an action occur if none of the specified conditions in the `If . . . Then` and `ElseIf` statements are met, VBScript enables you to add an `Else` statement to an `If` structure. See Microsoft's VBScript documentation for details.

Setting Opening Balances of All Accounts

To set the opening balances of accounts, use the `Exp` and `IsFirst` functions in an `If` structure. `Exp` and `If` structures are introduced in the previous sections; `IsFirst` determines whether the current period is the first period in the default frequency of the current scenario. For example, if a scenario has a default frequency of `Monthly`, `IsFirst` determines whether the current period is the first month in the year.

To set opening balances, place `IsFirst` in an `If` structure's `If . . . Then` statement, then place `Exp` between this statement and the `End If` statement. While you can include a specific account in `Exp`'s argument, you probably will want to set the opening balances of all the accounts.

This example shows you how to set the opening balances of all accounts. You can just retype or copy this example into a Calculation rule without modifications:

```
HS.EXP "A#ALL = A#ALL.P#Prior"
```

Note: In this example, `A#` is followed by the keyword `ALL`; this means that the rule applies to all accounts. In addition, the `P#` characters are followed by the keyword `Prior`; this means that `EXP` gets the account data from the period prior to the current period.

Creating Rules in the Rules Editor

The Rules Editor provides a graphical interface for creating rules. The Rules Editor can be launched from the Windows Desktop of the client computer. After you launch the program, you can create new rules or you can open existing rules.

- To start the Rules Editor from the Windows Desktop, double-click the Financial Management Rules Editor icon or select it from the Start menu.

Configuring the Rules Editor

You can configure the color and fonts of the different parts of a rules file. This is useful for quickly identifying the parts of the rules file. For example, you can set Financial Management objects and methods to be a different color than Visual Basic objects.

You can change the color and font of these parts of a rules file:

- Text
- Text Selections
- Numbers
- Operators
- Comments
- Strings
- Statements
- Financial Management Methods

- Visual Basic Objects
- Properties
- Events
- Functions
- Constants
- Financial Management Objects

➤ To configure the Rules Editor:

1 Select Tools, then Configure.

2 For Font and Color Settings, select the part of the rules file to configure.

3 Do one or all of these tasks:

- From **Foreground**, select a color for the text for this part of the rules file.

Note: Make sure the **Automatic** check box is not selected.

- From **Background**, select a color for the background for this part of the rules file.

Note: Make sure the **Automatic** check box is not selected.

- Click **Choose Font** select the font and font size for the text in this part of the rules file, and click **OK**.

Note: You can click **Reset All** to reset the colors and fonts for all parts of the rules file to the default settings.

4 Repeat steps 2-3 until you have selected all parts of the rules file to configure.

5 Click OK.

Using the Function Wizard

You can use the Function Wizard in the Rules Editor to select objects and functions to use in rules. When you select the object and function to use, the fields are available for coding. For example, to build an `Exp` function, choose the object `HS`, the function `Exp` and then select the source point of view and the destination point of view by clicking the **POV** button.

By default, rules files use the `RLE` file extension.

Showing the Point of View

You use the Point of View selector to create and edit rules using the metadata stored in your Financial Management application. You can select members to use in the rules.

Scanning Rules

You can scan rules to verify that the Visual Basic Script syntax and the metadata are valid. The scan also validates the destination for the rule. If it is not a valid destination, an error occurs. After you scan the file, you can load it directly into the application from the Rules Editor.

- To scan rules, select **HFM**, then **Scan rule**.

Loading Rules from the Rules Editor

You can load rules in an application directly from the Rules Editor.

For considerations before loading large rule files, see [“Loading Rules” on page 227](#).

- To load rules to an application, select **HFM**, then **Load rule**.

Extracting Rules into the Rules Editor

You can extract rules from an open application into the Rules Editor.

- To extract rules from an application, select **HFM**, then **Extract rule**.

Creating Rules Files

You can create rules in a text editor such as Microsoft Notepad. Rules files can be in an ASCII format that supports multibyte character sets (MBCS), or a file encoded with Unicode format, using Little Endian byte ordering. You use Visual Basic Script functions and Financial Management functions in rules files. By default, rules files use the RLE file extension.

You can include all types of Financial Management rules in any order in the rules file. [Table 36](#) shows the basic syntax to define each routine.

Table 36 Rules Routines

Rule Routine	Syntax
Sub Calculate	Sub Calculate() Type your Calculation rule here. End Sub
Sub Translate	Sub Translate() Type your Translation rule here. End Sub

Rule Routine	Syntax
Sub Allocate	<pre>Sub Allocate() Type your Allocation rule here. End Sub</pre>
Sub Input	<pre>Sub Input () Type your Input rule here. End Sub</pre>
Sub NoInput	<pre>Sub NoInput () Type your NoInput rule here. End Sub</pre>
Sub Consolidate	<pre>Sub Consolidate() Type your Consolidation rule here. End Sub</pre>
Sub Dynamic	<pre>Sub Dynamic() Type your Dynamic rule here. End Sub</pre>
Sub Transactions	<pre>Sub Transactions() Type your Transactions rule here. End Sub</pre>
Sub Equity Pickup	<pre>Sub Equity Pick Up() Type your Equity Pickup rule here. End Sub</pre>

Equity PickUp Rules Example

The following section shows a sample Equity Pickup Rules section. To calculate Equity Pickup, the application administrator must create a new section in the Rules file named SUB EquityPickup, where EPU calculations are defined. The default Point of View when the section is run is as follows:

- Current Scenario, Year, and Period
- Entity: Owner of the pair processed
- Value: Entity currency

Example:

```
Sub EquityPick Up()

Owner = Hs.Entity.Member
```

```

Owned = Hs.Entity.Owned

OwnerDefaultCurrency = HS.Entity.DefCurrency("")

lPown = Hs.GetCell("E#" & Owned & ".I#" & Owner & ".V#[None].A#[Shares
%Owned].C1#[None].C2#[None].C3#[None].C4#[None]")

Hs.Clear "A#IncomeFromSubsidiary.I#" & Owned

Hs.Exp "A#IncomeFromSubsidiary.I#" & Owned & " = E#" & Owned & ".V#" &
OwnerDefaultCurrency & ".A#NetIncome.I#[ICP Top] *" & lPown

End Sub

```

Loading Rules

Rules changes can affect data and are dependent on metadata. As a result, a rules load operation has to put a global lock on the Financial Management server cluster. The rules load cannot proceed until any previously started operations of these types have finished:

- Consolidation
- Data entry
- Data, Journal or Security Load
- Extended Analytics
- Metadata load
- Member list load

Rules scan follows the same restrictions as rules load to enable the system to validate dimension members and other parameters. Rules scan and load processes are queued up and started automatically after the blocking tasks are finished. Oracle recommends that you load rules during periods of light activity across the server cluster instead of, for example, during a long-running consolidation. You can check the Running Tasks page to see which consolidation or data loads, for example, are in progress.

After you load a rules file to an application, users using that application are notified that the system has changed and that they must log off from the application and log back on.

For intercompany transactions, there is an option to check the posted transactions in the application against new transactions in the rules file. The Sub Transactions section of the rules file defines the accounts that support intercompany transaction detail.

Windows Procedure

► To load rules:

- 1 **Open the application.**
- 2 **From the navigation frame, select **Load Rules**.**
- 3 **For **Rules Filename**, enter the file name to load, or click  to find the file.**

Note: By default, rules files use the RLE file extension. The load process accepts other file extensions such as TXT or CSV, however, Oracle recommends that you use the RLE file extension.

- 4 For **Log Filename**, enter a log file name, or click  to find the file.
- 5 **Optional:** Select **Check Integrity for Intercompany Transactions** to verify that posted transactions are valid with the statements in the Sub Transactions section of the rule file that you are loading.
- 6 **Optional:** Click **Scan** to verify that the file format is correct.
- 7 Click **Load**.

Caution! If you make changes to existing rules, you must load the updated RLE file into the application before your changes take effect.

Web Procedure

Note: Oracle recommends that you add Financial Management to the exceptions for your Web pop-up blocker. When you perform some Financial Management tasks on the Web such as loading data, a status window pops up showing the task status. If you have a pop-up blocker enabled on your computer, the status window is not displayed.

► To load rules:

- 1 Open the application.
- 2 In the **Browser View**, expand **Tasks** and select **Load Tasks**.
- 3 Select **Load Rules**.
- 4 For **Rules File**, enter the file name to load, or click **Browse** to find the file.

Note: By default, rules files use the RLE file extension. The load process accepts other file extensions such as TXT or CSV, however, Oracle recommends that you use the RLE file extension.

- 5 **Optional:** Select **Check Integrity for Intercompany Transactions** to verify that posted transactions are valid with the statements in the Sub Transactions section of the rule file that you are loading.
- 6 **Optional:** Click **Scan** to verify that the file format is correct.
- 7 Click **Load**.

Extracting Rules

When you extract rules, they are saved to an ASCII file that supports multibyte character sets (MBCS). By default, rules files use the RLE file extension. After you extract rules, you can view and modify them in a text editor.

Windows Procedure

► To extract rules:

- 1 Open the application.
- 2 From the navigation frame, select **Extract Rules**.
- 3 For **Rules Filename**, enter a file name to extract, or click  to find the file.

Note: By default, rules files use the RLE file extension.

- 4 For **Log Filename**, enter a log file name, or click  to find the file.
- 5 Click **Extract**.

Note: After you extract rules, you can click the View button next to Log Filename to display the log file which notes errors encountered during the extract.

Web Procedure

► To extract rules:

- 1 Open an application.
- 2 In the **Browser View**, expand **Tasks** and select **Extract Tasks**.
- 3 Select **Extract Rules**.
- 4 Follow the download instructions displayed in the browser.

The instructions vary depending on the Web browser that you are using. Make sure to save the file in the Web directory that you set up.

In This Chapter

Functions Overview	234
ABSExp.....	239
AccountType	240
AccountTypeID	241
AddEntityToList	242
AddEntityToListUsingIDs	243
AddMemberToList	244
AddMemberToListUsingIDs	244
Alloc	245
AllowAdjFromChildren	246
AllowAdjs	247
CalculateExchangeRate	247
CalculateRate	248
CalcStatus	249
Clear.....	250
Con	251
Consol1, Consol2, Consol3	252
ConsolidateYTD	253
ContainsCellText.....	253
CreateInputCache.....	254
Currency	254
Custom1Top, Custom2Top, Custom3Top, and Custom4Top.....	255
Decimal	256
DefaultFreq	257
DefaultTranslate.....	257
DefaultView	258
DefCurrency.....	259
Down.....	260
Dynamic	261
Exp	262
GetCell	268
GetCellNoData	269
GetCellRealData.....	270

GetCellText.....	271
GetCellType.....	271
GetItem.....	272
GetItemIDs2.....	273
GetNumItems.....	274
GetNumLID.....	274
GetRate.....	275
GetSubmissionGroup.....	275
GetSubmissionPhase.....	276
Holding.....	276
ICPTopMember.....	277
ICPWeight.....	278
IDFromMember.....	278
ImpactStatus.....	279
Input.....	280
IsAlmostEqual.....	281
IsBase.....	281
IsCalculated.....	283
IsChild.....	283
IsConsolidated.....	285
IsDescendant.....	285
IsFirst.....	287
IsICP.....	288
IsLast.....	289
IsTransCur.....	289
IsTransCurAdj.....	290
IsValidDest.....	290
IsZero.....	291
List.....	292
Member.....	293
MemberFromID.....	293
Method.....	295
NoInput.....	295
NoRound.....	296
NumBase.....	296
Number.....	298
NumChild.....	298
NumDescendant.....	300
NumPerInGen.....	302
NumPeriods.....	302
OpenDataUnit.....	303
OpenDataUnitSorted.....	304
Owned.....	304

Owner	305
PCon	305
PEPU	306
PeriodNumber	306
PlugAcct	307
POwn	308
PVAForBalance	309
PVAForFlow	309
RateForBalance	310
RateForFlow	311
ReviewStatus	311
ReviewStatusUsingPhaseID	312
Round	313
Scale	314
SecurityAsPartner	315
SecurityClass	316
SetData	317
SetDataWithPOV	317
SubmissionGroup	318
SupportsProcessManagement	319
SupportsTran	319
SwitchSign	320
SwitchType	321
Trans	322
TransPeriodic	323
UD1...3	324
ValidationAccount	325
ValidationAccountEx	325
XBRLTags	326

You can write these rule types using these functions:

- Calculation rules
- Translation rules
- Consolidation rules
- Allocation rules
- Input rules
- NoInput rules
- Dynamic calculation rules
- Transactions rules
- Equity Pickup rules

Some functions are unique to specific routines, while others can be used with multiple types of rules within multiple routines.

For an overview of all functions, see [“Functions Overview” on page 234](#).

Functions Overview

Table 37 summarizes the Financial Management functions, the objects with which they can be used, and the type of rules in which they can be used. Functions are listed alphabetically. Detailed sections for each function are provided after the table.

Rules types are abbreviated in this table as follows:

- Alloc - Allocation
- Calc - Calculation
- Con - Consolidation
- Tran - Translation
- Dyn - Dynamic Calculation
- Trans - Transactions
- EPU - EquityPickUp

Table 37 Financial Management Functions

Function	Description	Objects	Types of Rules
ABSExp	Executes a calculation expression and stores the result as an absolute value.	HS	Calc, Tran, Alloc
AccountType	Gets the account type for the member.	Account	Calc, Tran, Con, Alloc
AccountTypeID	Gets the account type ID for the member.	Account	Calc, Tran, Con, Alloc
AddEntityToList	Adds the specified entity and parent to a member list.	HS	Member List
AddEntityToListUsingIDs	Using entity and parent IDs, adds the specified entity and parent to an internal list.	HS	Member List
AddMemberToList	Adds the member to the member list.	HS	Member List
AddMemberToListUsingIDs	Using member IDs, adds the specified member to an internal list.	HS	Member List
Alloc	Allocates data to a cell.	HS	Alloc
AllowAdjFromChildren	Determines if journal postings from children are allowed for the member.	Entity, Parent	Calc, Tran, Con, Alloc
AllowAdjs	Determines if journal postings are allowed for the member.	Entity, Parent	Calc, Tran, Con, Alloc

Function	Description	Objects	Types of Rules
CalculateExchangeRate	Calculates the exchange rate from one currency to another.	HS	Calc, Tran, Con, Alloc, Dyn, Trans
CalculateRate	Gets the current exchange rate for the specified entity.	HS	Calc, Tran, Con, Alloc, Dyn, Trans
CalcStatus	Gets the calculation status for the cell.	HS	Calc, Tran, Con, Alloc
Clear	Removes data from a cell.	HS	Calc, Tran, Con, Alloc
Con	Puts data into the [Proportion] and [Elimination] accounts.	HS	Con
Consol1, Consol2, Consol3	Gets the Consol1...3 system account value for the node.	Node	Calc, Tran, Con, Alloc
ConsolidateYTD	Determines if the scenario is consolidated using the YTD or periodic method.	Scenario	Calc, Tran, Con, Alloc
ContainsCellText	Determines if the cell contains cell text.	HS	Calc, Tran, Con, Alloc
CreateInputCache	Creates a buffer to be used when data is being consolidated.	HS	Calc, Tran, Con, Alloc, Dyn, Trans
Currency	Gets the currency type for the application or the value member.	AppSettings, Value	Calc, Tran, Con, Alloc
Custom1Top, Custom2Top, Custom3Top, and Custom4Top	Gets the Top Custom member.	Account	Calc, Tran, Con, Alloc
Decimal	Gets the number of decimal places for the specified account.	Account	Calc, Tran, Con, Alloc
DefaultFreq	Gets the default scenario frequency.	Scenario	Calc, Tran, Con, Alloc
DefaultTranslate	Calculates translation by bypassing rules.	HS	Tran
DefaultView	Gets the default scenario view.	Scenario	Calc, Tran, Con, Alloc
DefCurrency	Gets the default currency for the entity or parent.	Entity, Parent	Calc, Tran, Con, Alloc
DOWn	Gets the percentage of Direct Ownership (DOWn) for the node.	Node	Calc, Tran, Con, Alloc
Dynamic	Specifies the formula for the dynamic accounts that need calculations.	HS	Dyn
Exp	Executes a calculation expression and puts data into a specified point of view.	HS	Calc, Tran, Alloc
GetCell	Gets the data contained in a cell.	HS	Calc, Tran, Con, Alloc
GetCellNoData	Gets the data contained in a cell and indicates if the cell contains no data.	HS	Calc, Tran, Con, Alloc

Function	Description	Objects	Types of Rules
GetCellRealData	Gets the data contained in a cell and indicates if the cell contains real data.	HS	Calc, Tran, Con, Alloc
GetCellText	Gets the cell text for a specified Point of View.	HS	Calc, Tran, Con, Alloc
GetCellType	Gets the cell type.	HS	Calc, Tran, Con, Alloc
GetItem	Gets an individual record to process for consolidation.	DataUnit	Calc, Tran, Con
GetItemIDs2	Gets an individual record to process for consolidation using dimension ID numbers.	DataUnit	Calc, Tran, Con
GetNumItems	Gets the number of records to process for consolidation.	DataUnit	Calc, Tran, Con
GetNumLID	Gets the number of line-item details for the specified POV.	HS	Calc, Tran, Con, Alloc
GetRate	Gets the currency rate for a cell.	HS	Calc, Tran, Con, Alloc
GetSubmissionGroup	Gets the submission group for a cell.	HS	Calc, Tran, Con, Alloc
GetSubmissionPhase	Gets the submission phase for a cell.	HS	Calc, Tran, Con, Alloc
Holding	Gets the holding company for the member.	Entity, Parent	Calc, Tran, Con, Alloc
ICPTopMember	Gets the ICP Top Member of the current Account dimension member or the specified account member.	Account	Calc, Tran, Con, Alloc
ICPWeight	Gets the percentage of ICP entity balances that aggregate to the [ICP Top] value member.	AppSettings	Calc, Tran, Con, Alloc
IDFromMember	Gets the ID number for the specified member.	Account, Custom1 through Custom4, Entity, ICP, Parent, Year, Period, Scenario, Value, View	Calc, Tran, Con, Alloc
ImpactStatus	Changes the status of the specified data unit to impacted.	HS	Calc
Input	Enables input at parent entity level for specified account.	HS	Input
IsAlmostEqual	Checks to see if two values are equal.	HS	Calc, Tran, Con, Alloc, Dyn, Trans
IsBase	Determines if the member is a base member.	Account, Custom1 through Custom4, Entity, Parent, Node	Calc, Tran, Con, Alloc
IsCalculated	Determines if the account is calculated.	Account	Calc, Tran, Con, Alloc

Function	Description	Objects	Types of Rules
IsChild	Determines if the member is a child of another member.	Account, Custom1 through Custom4, Entity, Parent, Node	Calc, Tran, Con, Alloc
IsConsolidated	Determines if the account is consolidated.	Account	Calc, Tran, Con, Alloc
IsDescendant	Determines if the member is a descendant of another member.	Account, Custom1 through Custom4, Entity, Parent, Node	Calc, Tran, Con, Alloc
IsFirst	Determines if the period or year is the first for the application.	Period, Year	Calc, Tran, Con, Alloc
IsICP	Determines if the member is an ICP.	Account, Entity, Parent	Calc, Tran, Con, Alloc
IsLast	Determines if the year or period is the last for the application.	Period, Year	Calc, Tran, Con, Alloc
IsTransCur	Determines if the value member is a translated currency member.	Value	Calc
IsTransCurAdj	Determines if the value member is a translated currency Adj member.	Value	Calc
IsValidDest	Determines if the specified POV is a valid destination.	HS	Calc, Tran, Con, Alloc
IsZero	Checks to see if the passed in value is zero.	HS	Calc, Tran, Con, Alloc, Dyn, Trans
List	Gets the elements contained in a list.	Account, Custom1 through Custom4, Entity, Parent, ICP, Node, Scenario	Calc, Tran, Con, Alloc
Member	Gets the member name.	Entity, Parent, Period, Scenario, Value, Year, View	Calc, Tran, Con, Alloc
MemberFromID	Gets the member for the specified ID number.	Account, Custom1 through Custom4, Entity, ICP, Parent, Year, Period, Scenario, Value, View	Calc, Tran, Con, Alloc
Method	Gets the consolidation method for the member.	Node	Calc, Tran, Con, Alloc
NoInput	Prevents input into cells.	HS	Noinput
NoRound	Turns off rounding.	HS	Calc, Tran, Con, Alloc
NumBase	Gets the number of base members.	Account, Custom1 through Custom4, Entity, Parent, Node, Period	Calc, Tran, Con, Alloc

Function	Description	Objects	Types of Rules
Number	Gets the current period number.	Period	Calc, Tran, Con, Alloc
NumChild	Gets the number of children for the member.	Account, Custom1 through Custom4, Entity, Parent, Node	Calc, Tran, Con, Alloc
NumDescendant	Gets the number of descendants for the member.	Account, Custom1 through Custom4, Entity, Parent, Node	Calc, Tran, Con, Alloc
NumPerInGen	Gets the number of periods in the generation for the current period being processed.	Period	Dynamic
NumPeriods	Gets the number of periods defined for the frequency of the specified scenario.	Scenario	Dynamic
OpenDataUnit	Gets a data unit for consolidation.	HS	Calc, Tran, Con
OpenDataUnitSorted	Gets the data units for calculation, translation, or consolidation, in sorted order.	HS	Calc, Tran, Con
Owned	Gets the Owned entity of the pair currently processed.	Entity	Equity Pickup
Owner	Gets the Owner entity of the pair currently processed.	Entity	Equity Pickup
PCon	Gets the percentage of consolidation.	Node	Calc, Tran, Con, Alloc
PEPU	Gets the percentage of ownership from the EPU table.	HS	Equity Pickup
PeriodNumber	Gets the period number in the view for the data that is being retrieved.	View	Dynamic
PlugAcct	Gets the plug account.	Account	Calc, Tran, Con, Alloc
POwn	Gets the percentage of ownership.	Node	Calc, Tran, Con, Alloc
PVAFForBalance	Determines default translation method for BALANCE accounts.	AppSettings	Calc, Tran, Con, Alloc
PVAFForFlow	Determines default translation method for FLOW accounts.	AppSettings	Calc, Tran, Con, Alloc
RateForBalance	Gets the default rate for balance.	AppSettings	Calc, Tran, Con, Alloc
RateForFlow	Gets the default rate for flow.	AppSettings	Calc, Tran, Con, Alloc
ReviewStatus	Gets the process management review status for the cell.	HS	Calc, Tran, Con, Alloc
ReviewStatusUsingPhaseID	Gets the process management review status by phase ID for the cell.	HS	Calc, Tran, Con, Alloc
Round	Rounds the data.	HS	Calc, Tran, Con, Alloc

Function	Description	Objects	Types of Rules
Scale	Gets the scale of the specified currency.	Currency	Calc, Tran, Con, Alloc
SecurityAsPartner	Gets the security class for the ICP entity.	Entity, Parent	Calc, Tran, Con, Alloc
SecurityClass	Gets the security class for a dimension member.	Account, Scenario, Entity, Parent, Custom1 through Custom4	Calc, Tran, Con, Alloc
SetData	Sets an individual record.	HS	Calc, Tran
SetDataWithPOV	Inserts data into the node or currency cube.	HS	Calc, Tran
SubmissionGroup	Gets the process management submission group for a dimension member.	HS	Calc, Tran, Con, Alloc
SupportsProcessManagement	Determines if a scenario supports process management.	Scenario	Calc, Tran, Con, Alloc
SupportsTran	Defines the accounts in the application that require Intercompany Transaction detail support.	Scenario, Year, Entity, Account, C1... 4	Tran
SwitchSign	Determines if credits are switched to debits for the member.	Custom1 through Custom4	Calc, Tran, Con, Alloc
SwitchType	Determines if account types are switched for the member.	Custom1 through Custom4	Calc, Tran, Con, Alloc
Trans	Translates using YTD method.	HS	Tran
TransPeriodic	Translates using periodic method.	HS	Tran
UD1...3	Gets the user-defined attribute for the member.	Account, Entity, Parent, Scenario, Custom1 through Custom4	Calc, Tran, Con, Alloc
ValidationAccount	Gets the validation account.	AppSettings	Calc, Tran, Con, Alloc
ValidationAccountEx	Gets the validation account for the process management submission phase.	AppSettings	Calc, Tran, Con, Alloc
XBRLTags	Gets the XBRL tag for the account.	Account	Calc, Tran, Con, Alloc

ABSExp

Executes a calculation expression and stores the result as an absolute value. This function is the same as the Exp function except that it stores the resulting value as an absolute value. This function can be used in these types of rules:

- Calculation
- Translation

- Allocation

Syntax

```
HS.ABSExp "DestPOV = Expression"
```

Table 38 Syntax for ABSExp Function

Parameter	Description
<i>DestPOV</i>	A destination point of view that identifies where to put the data You must specify an Account member, and you can optionally specify ICP and Custom members. Note these usage rules: ● If you do not specify an ICP member, the default is [ICP None]. ● To avoid populating the database with unwanted values, rules should be as explicit as possible in terms of defining where data should reside. A good practice is to include clearly defined Custom dimension intersections for the Account dimension. Such clearly defined intersections utilize Financial Management validation checks to avoid writing data to invalid intersections. See “Dimension Intersection Considerations” on page 263.
<i>Expression</i>	A calculation expression

Return Value

None.

Example

This example sets the amount in the StateTax account. The example calculates the absolute amount by multiplying the amount in the Sales account for 2009 by the rate in the StateRate account for 2009.

```
HS.ABSExp "A#StateTax = A#Sales.Y#2009 * A#StateRate.Y#2009"
```

AccountType

Gets the account type for the current Account member or for a specified account. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Syntax

```
HS.Account.AccountType ("Account")
```

```
HS.Account.AccountType ( " " )
```

where, *Account* is the name of a valid Account member.

You can use a blank string (" ") to apply this function to the current member only if you are using the function in the Sub Consolidate routine. Otherwise, specify an account when using this function.

Return Value

The account type for the specified account.

Note: Account types use all capital letters.

Valid account types are:

- ASSET
- LIABILITY
- REVENUE
- EXPENSE
- FLOW
- BALANCE
- BALANCERECURRING
- CURRENCYRATE
- GROUPLABEL

Example

In this example, if the account type for the Sales account is REVENUE, then statements between the If...Then and End If statements are executed.

```
If HS.Account.AccountType("Sales") = "REVENUE" Then
...
End If
```

AccountTypeID

Gets the account type ID for the current account member or for a specified account. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation
- Dynamic Calculation
- Transactions

Syntax

```
HS.Account.AccountTypeID("Account")
```

```
HS.Account.AccountTypeID(" ")
```

where *Account* is the name of a valid Account member

You can use a blank string (" ") to apply this function to the current member only if you are using the function in the Sub Consolidate routine. Otherwise, you must specify an account when using this function.

Return Value

The ID for the specified account. [Table 39](#) lists valid account types with corresponding IDs.

Table 39 Account Type IDs

Account Type	ID
REVENUE (INCOME in previous releases)	0
EXPENSE	1
ASSET	2
LIABILITY	3
BALANCE	4
FLOW	5
CURRENCYRATE	7
GROUPLABEL	10
BALANCERECURRING	11
DYNAMIC	12

Example

```
If HS.Account.AccountTypeID("Investments") = 2 Then  
...  
End If
```

AddEntityToList

Adds the specified entity and parent to a member list. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation

- Allocation
- Dynamic Calculation
- Transactions

Syntax

```
HS.AddEntityToList("Parent", "Child")
```

Table 40 Syntax for AddEntityToList Function

Parameter	Description
<i>Parent</i>	Name of a valid parent entity.
<i>Child</i>	Name of a child of the parent entity.

Return Value

None

Example

```
HS.AddEntityToList "UnitedStates", "Maine"
```

AddEntityToListUsingIDs

Using the ID for the entity and parent, adds the specified entity and parent to the internal Financial Management list to be used later. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation
- Dynamic Calculation
- Transactions

Syntax

```
HS.AddEntityToListUsingIDs(ParentID, ChildID)
```

Table 41 Syntax for AddEntityToListUsingIDs Function

Parameter	Description
<i>ParentID</i>	The ID for the parent entity.
<i>ChildID</i>	The ID for the child of the parent entity.

Return Value

None

AddMemberToList

Adds the specified member to a member list. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation
- Dynamic Calculation
- Transactions

Syntax

```
HS.AddMemberToList ( "Member" )
```

where *Member* is the name of a valid dimension member.

Return Value

None

Example

```
HS.AddMemberToList "July"
```

AddMemberToListUsingIDs

Adds the specified member to the internal Financial Management list to be used later. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation
- Dynamic Calculation
- Transactions

Syntax

```
HS.AddMemberToListUsingIDs (MemberID)
```

where *MemberID* is the ID for a valid dimension member.

Return Value

None

Alloc

Allocates data from one point of view to another. This function can be used in Allocation rules.

Syntax

```
HS.Alloc ("SourcePOV", "DestPOV", "EntityList", "AllocExp", "PlugAccount")
```

Table 42 Syntax for Alloc Function

Parameter	Description
<i>SourcePOV</i>	A source point of view for the data to be allocated. You must specify an Account member, and you can optionally specify ICP and Custom members. If you do not specify ICP and Custom members: ● The default ICP member is ICP Top. ● The default Custom member is Top Member for that account.
<i>DestPOV</i>	A destination point of view that identifies where to allocate the data. You must specify an Account member, and you can optionally specify ICP and Custom members. Note these usage rules: ● If you do not specify an ICP member, the default is [ICP None]. ● If you do not specify Custom members, the default is [None]. ● If you specify an Entity member, it is used as the parent if the <i>EntityList</i> argument is [Base].
<i>EntityList</i>	A member list that identifies the entities to which the data is be allocated. You can use the system-defined entity list [Base] or you can use a user-defined list. If you use the system-defined list [Base], the system uses the entity specified in the destination point of view as the parent member. If you use a user-defined list and that list contains parent members, they are skipped.
<i>AllocExp</i>	An expression that identifies the data to be allocated to each entity. This expression can contain these types of values: ● Numbers ● Account Expressions that identify a numeric value. You can specify an Account member, and you can optionally specify members of the ICP and Custom dimensions. Note these usage rules: - If you do not specify Custom members, the default is Top Member. - If you do not specify an ICP member, the default is ICP Top. - If you do not specify Scenario, Year, Period, View, or Value members, the default is Current - If you do not specify an Entity member, the default is the destination entity.

Parameter	Description
<i>PlugAccount</i>	The name of a plug account. This argument is optional, and is used to reverse the source point of view amount, taking into consideration the attribute of the source point of view account versus the plug account. You must specify an Account member, and you can optionally specify ICP and Custom members. Note these usage rules: ● You cannot use this argument if the source point of view Entity member is a parent, or if the source point of view Value member is not Entity Currency. ● If you do not specify an ICP member, the default is ICP None. ● If you do not specify Custom members, the default is None.

Return Value

None.

Example

In this example, data from the TangibleAssets account is allocated to the Cash account.

```
Call HS.ALLOC ("A#TangibleAssets", "A#Cash", "NewEngland", "A#TangibleAssets", "A#Plug")
```

AllowAdjFromChildren

Specifies if journal postings from children are allowed for the specified entity. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Note: A member is a child if it is one level directly below a member in a tree hierarchy.

Syntax

```
HS.Entity.AllowAdjFromChildren("Entity")
HS.Entity.AllowAdjFromChildren(" ")
```

where *Entity* is the name of a valid Entity member.

Use a blank string (" ") to apply this function to the current entity.

Return Value

A Boolean expression that is True if journal postings from children are permitted for the specified entity, False if journal postings from children are not permitted.

Example

In this example, if journal postings from children of France are allowed, then statements between the If...Then and End If statements are executed.

```
If HS.Entity.AllowAdjFromChildren("France") = TRUE then
    ...
End If
```

AllowAdjs

Specifies if journal postings are allowed for the specified entity. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Syntax

```
HS.Entity.AllowAdjs("Entity")
HS.Entity.AllowAdjs(" ")
```

where *Entity* is the name of a valid Entity member.

Use a blank string (" ") to apply this function to the current entity.

Return Value

A Boolean expression that is True if journal postings are permitted for the specified entity, False if journal postings are not permitted.

Example

In this example, if journal postings for France are allowed, then statements between the If...Then and End If statements are executed.

```
If HS.Entity.AllowAdjs("France") = TRUE then
    ...
End If
```

CalculateExchangeRate

Calculate the exchange rate from one currency to another.

Syntax

```
HS.CalculateExchangeRate (ScenarioID, YearID, PeriodID, EntityID, RateAccountID,
FromCurrencyID, ToCurrencyID)
```

Table 43 Syntax for CalculateExchangeRate Function

Parameter	Description
<i>ScenarioID</i>	The scenario ID.
<i>YearID</i>	The year ID.
<i>PeriodID</i>	The period ID.
<i>EntityID</i>	The entity ID.
<i>RateAccountID</i>	The rate account ID.
<i>FromCurrencyID</i>	The ID for the currency that you are going from.
<i>ToCurrencyID</i>	The ID for the currency that you are going to.

Return Value

The exchange rate.

CalculateRate

Gets the current exchange rate for the specified entity.

Syntax

HS.CalculateRate (*ScenarioID*, *YearID*, *PeriodID*, *EntityID*, *CurrID*, *RateAccountID*)

Table 44 Syntax for CalculateRate Function

Parameter	Description
<i>ScenarioID</i>	The scenario ID.
<i>YearID</i>	The year ID.
<i>PeriodID</i>	The period ID.
<i>EntityID</i>	The entity ID.
<i>CurrID</i>	The currency ID.
<i>RateAccountID</i>	The rate account ID.

Return Value

The exchange rate for the specified entity.

CalcStatus

Gets the calculation status for the specified point of view. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Syntax

```
HS.CalcStatus("Scenario.Year.Period.Entity.Value")
```

Table 45 Syntax for CalcStatus Function

Parameter	Description
<i>Scenario</i>	Name of a valid Scenario member.
<i>Year</i>	A valid year.
<i>Period</i>	A valid period.
<i>Entity</i>	Name of a valid Entity member.
<i>Value</i>	Name of a valid Value member.

Return Value

A String that contains the calculation status for the specified point of view. Valid status codes are listed below.

Table 46 Calculation Status Codes

Status Code	Description
OK	None of the data for the specified Scenario, Year, Period, Entity, and Value dimensions has changed.
OK ND	OK - No data. Some data may have changed.
OK SC	OK - System changed. Some of the data for the specified Scenario, Year, Period, Entity, and Value dimensions has changed. For example, a new rules file or metadata file has been loaded.
CH	Needs Calculation. At least one data cell for the specified Scenario, Year, Period, Entity, and Value dimensions has been changed, or metadata parameters or rules have changed. As a result, other data cells in this Scenario, Year, Period, Entity, and Value dimension may not be up to date because the SubCalculate routine has not been executed. For base level entities, the data cell may have been entered by manual data entry or by a bulk data file load. For entities, the data cell may have been entered by a journal posting.
CH ND	Needs Calculation — No Data. This indicates the first time that calculation will be performed on the cell.
TR	Needs Translation. The selected Value dimension member is not the entity's default currency, and its translated values may not be up to date.

Status Code	Description
TR ND	Needs Translation — No Data. This indicates the first time that translation will be performed on the cell.
CN	Needs Consolidation. The data for the specified Scenario, Year, Period, Entity, and Value dimensions may not be up to date because data for a child entity has changed, data for the same entity's default currency has changed, or metadata parameters or rules have changed.
CN ND	Needs Consolidation - No Data. The parent has no data but a child of the parent has changed. This indicates the first time that consolidation will be performed on the cell.
Locked	The data for the specified Scenario, Year, Period, Entity, and Value dimensions has been locked by an administrator and can no longer be modified by manual data entry or calculations. Note: You can use the Alloc function to modify data in a locked destination POV.
NoData	No data exists for the specified Scenario, Year, Period, Entity, and Value dimensions.

Example

In this example, if the status for the specified point of view is “OK”, then statements between the If...Then and End If statements are executed.

```
If HS.CalcStatus("S#Actual.Y#2009.P#January.E#Connecticut.
V#<EntityCurrency>") = "OK" Then
    ...
End If
```

Clear

Removes data from combinations of Account, ICP, and Custom members. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Syntax

```
HS.Clear "Account.ICP.Custom1.Custom2.Custom3.Custom4"
```

Table 47 Syntax for Clear Function

Parameter	Description
<i>Account</i>	Name of a valid Account member.
<i>ICP</i>	Name of a valid ICP member.
<i>Custom1, Custom2, Custom3, Custom4</i>	Name of valid Custom1, Custom2, Custom3, and Custom4 members.

To remove data from all cells that intersect the current point of view, which consists of the current Entity, Period, Scenario, Value, View, and Year members, place the ALL keyword after the A# characters as in this example:

```
HS.Clear "A#ALL"
```

To clear all intersections of cells and Custom or ICP dimensions, use the ALL keyword or omit the A# characters. This example omits the A# characters to clear data from all account intersections with the GolfBalls member of the Custom1 dimension:

```
HS.Clear "C1#GolfBalls"
```

Return Value

None.

Example

This example clears the data stored in the intersection of the Sales account and the GolfBalls member of the Custom1 dimension.

```
HS.Clear "A#Sales.C1#GolfBalls"
```

Con

Puts data into the [Proportion] and [Elimination] Value dimension members. This function can be used in Consolidation rules.

Syntax

```
HS.Con ("DestPOV", Factor, "Nature")
```

Table 48 Syntax For Con Function

Parameter	Description
<i>DestPOV</i>	Combination of these dimensions: ● Account ● Custom1, Custom2, Custom3, Custom4 ● Intercompany ● Entity ● Value
<i>Factor</i>	A number or an expression using mathematical operators (+ - * /) or functions like HS.GetCell.
<i>Nature</i>	A string used for audit purposes. This string is stored in the database and provides information about the accounting purpose of the transaction. To allow users to view consolidation source and destination transactions after running a consolidation, you must include text in this parameter. If you do not include text, the transaction information is not stored. You can view audit transaction information from the Source or Destination Transaction options in data grids, the Entity Detail report, or automated consolidation journals.

Return Value

None.

Examples

```
Call HS.Con ("V#[Elimination]", -1*dPCon, "")  
Call HS.Con ("V#[Elimination]", -1*dPCon, "Elimination")  
Call HS.Con ("V#[Elimination]"PCON, "DefaultConsolidation")
```

Consol1, Consol2, Consol3

Gets the value in Consol1, Consol2, or Consol3 account for the specified parent.entity node. This function can be used in these types of rules:

- Calculation
- Allocation

Syntax

Combination of scenario, year, period, and parent.entity members.

```
HS.Node.Consoln ("S#Scenario.Y#Year.P#Period.E#Parent.Entity")  
HS.Node.Consoln ("")
```

Note: Use a blank string (" ") to apply this function to the current scenario, year, period, and entity.

Table 49 Syntax for Consol1, Consol2, Consol3 Functions

Parameter	Description
<i>Scenario</i>	Name of a valid Scenario member.
<i>Year</i>	A valid year.
<i>Period</i>	A valid period.
<i>Parent.Entity</i>	Name of a valid Parent.Entity node.

Return Value

The value in the Consol1, Consol2, or Consol3 system account.

Example

This example gets the Consol1 value for the Group1.Ent1 node in the actual scenario.

```
dVar1 = HS.Node.Consol1 ("S#Actual.E#Group1.Ent1")
```

ConsolidateYTD

Determines if the current Scenario member or a specified scenario member is consolidated year to date. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Syntax

```
HS.Scenario.ConsolidateYTD("Scenario")  
HS.Scenario.ConsolidateYTD(" ")
```

where Scenario is the name of a valid Scenario member.

Use a blank string (" ") to apply this function to the current member.

Return Value

A Boolean expression that is True if the scenario is consolidated using the year-to-date method, False if the scenario is consolidated using the periodic method.

Example

In this example, if the Actual scenario is set up to be consolidated using the year-to-date method, then statements between the If...Then and End If statements are executed.

```
If HS.Scenario.ConsolidateYTD("Actual") = TRUE Then  
    ...  
End If
```

ContainsCellText

Determines if the specified cell contains cell text. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Syntax

```
HS.ContainsCellText("POVExpression")
```

where *POVExpression* is a combination of members. If you do not specify a dimension, these default values are used.

- Account - [none]
- ICP - [ICP None]
- Custom1...4 - [None]
- Scenario - Current Scenario member
- Entity - Current Entity member
- Value - Current Value member
- Year and Period - Current member

Return Value

A Boolean expression that is True if the specified data cell contains cell text; False otherwise.

Example

In this example, if the specified cell does not contain cell text, then statements between the If...Then and End If lines are executed.

```
If HS.ContainsCellText("A#Sales.C1#Prod1.C2#Region1") = "False" then
    ...
End If
```

CreateInputCache

Creates a buffer to be used when data is being consolidated. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation
- Dynamic Calculation
- Transactions

Syntax

```
HS.CreateInputCache()
```

Return Value

The cached object.

Currency

Gets the currency for the application or for the current Value dimension member. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Syntax

```
HS.AppSettings.Currency
HS.Value.Currency
```

Return Value

A String that contains the currency name for the application or for the Value member.

Example

In this example, if Euro is the currency of the application then statements between the If...Then and End If statements are executed.

```
If HS.AppSettings.Currency = "Euro" Then
    ...
End If
```

Custom1Top, Custom2Top, Custom3Top, and Custom4Top

Gets the C1Top through C4Top member for the current Account member or for a specified account member. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Syntax

```
HS.Account.<Object>("Account")
HS.Account.<Object>(" ")
```

Note: Use a blank string (" ") to apply this function to the current member.

Table 50 Syntax for C1Top...C4Top Function

Parameter	Description
<i><Object></i>	One of these object keywords: ● C1Top ● C2Top ● C3Top ● C4Top
<i>Account</i>	Name of a valid Account member.

Return Value

The top member for the specified Custom account member.

Example

In this example, the C1Top member is the TotalCosts account.

```
strVariable = HS.Account.C1Top("TotalCosts")
```

Decimal

Gets the number of decimal places for the specified account. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Syntax

```
HS.Account.Decimal("AccountName")
HS.Account.Decimal(" ")
```

Note: Use a blank string (" ") to apply this function to the current account when using a Sub Consolidate routine.

```
HS.Account.Decimal(Var1)
```

Table 51 Syntax for Decimal Function

Parameter	Description
<i>AccountName</i>	Name of a valid Account member.
<i>Var1</i>	VBScript variable representing an Account member.

Return Value

An integer representing the decimal value assigned to the account. Valid values are 0 to 9.

Example

In this example, if the number of decimal places assigned to the Sales account is 2, then statements between the If...Then and End If statements are executed.

```
If HS.Account.Decimal("Sales") = "2" Then
    ...
End If
```

DefaultFreq

Gets the default frequency for the current Scenario member or for a specified scenario member. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Syntax

```
HS.Scenario.DefaultFreq("Scenario")
HS.Scenario.DefaultFreq("")
```

Note: Use a blank string ("") to apply this function to the current member.

Where *Scenario* is the name of a valid Scenario member.

Return Value

A String that contains the default frequency for the scenario.

Example

In this example, if the default frequency for the Actual scenario is YTD, then statements between the If...Then and End If statements are executed.

```
If HS.Scenario.DefaultFreq("Actual") = "YTD" Then
    ...
End If
```

DefaultTranslate

Calculates translation by bypassing rules. This function overrides the Application settings and can be used in SubTranslate rules only.

Syntax

```
HS.DefaultTranslate(dRateForBalanceAccounts, dRateForFlowAccounts,  
bUsePVAForFlowAccounts, bUsePVAForBalanceAccounts)
```

Table 52 Syntax for DefaultTranslate Function

Parameter	Description
<i>dRateForBalanceAccounts</i>	Number for rate
<i>dRateForFlowAccounts</i>	Number for rate
<i>bUserPVAForFlowAccounts</i>	True or False
<i>bUsePVAForBalanceAccounts</i>	True or False

Return Value

None.

Example

In this example, if the parent member is United States, then statements between the If...Then and End If statements are executed.

```
If HS.Parent.Member="UnitedStates" Then  
  HS.DefaultTranslate .25, .27, True, False  
End If
```

DefaultView

Gets the default view for the current Scenario member or for a specified scenario member. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Syntax

```
HS.Scenario.DefaultView("Scenario")  
HS.Scenario.DefaultView(" ")
```

Note: Use a blank string (" ") to apply this function to the current member.

Where *Scenario* is the name of a valid Scenario member.

Return Value

A String that contains the default view for the specified scenario. Valid values are YTD and Periodic.

Example

In this example, if the default view for the Actual scenario is YTD, then statements between the If...Then and End If statements are executed.

```
If HS.Scenario.DefaultView("Actual") = "YTD" Then
    ...
End If
```

DefCurrency

Gets the default currency for the current entity or parent member, or for the specified entity or parent member. If you specify an entity, the system returns the entity currency. To obtain the parent currency, you must specify the parent entity. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Syntax

```
HS.<Object>.DefCurrency("Entity")
HS.<Object>.DefCurrency(" ")
```

Note: Use a blank string (" ") to apply this function to the current member.

Table 53 Syntax for DefCurrency Function

Parameter	Description
<Object>	One of these object keywords: ● Entity ● Parent
Entity	Name of a valid Entity member.

Return Value

A string that contains the default currency for the specified entity or parent.

Example

In this example, if the default currency for Connecticut is different from the default currency for EastRegion, then statements between the If...Then and End If statements are executed.

```
If HS.Entity.DefCurrency("Connecticut") <> HS.Parent.DefCurrency("EastRegion") Then
    ...
End If
```

DOWn

Gets the percentage of direct ownership for the specified parent.entity node. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Syntax

Combination of scenario, year, period, and parent.entity members.

```
HS.Node.DOWn("S#Scenario.Y#Year.P#Period.E#Parent.Entity")
HS.Node.DOWn(" ")
```

Note: Use a blank string (" ") to apply this function to the current node.

Table 54 Syntax for DOWn Function

Parameter	Description
<i>Scenario</i>	Name of a valid Scenario member.
<i>Year</i>	A valid year.
<i>Period</i>	A valid period.
<i>Parent.Entity</i>	Name of a valid Parent.Entity node.

Return Value

A number that is a percentage of direct ownership.

Example

```
dVar1 = HS.Node.DOWn("S#Actual.Y#2009.P#Q1.E#Group1.Ent1")
```

Dynamic

Specifies the formula for the dynamic accounts that need calculations. This function can be used in Dynamic rules.

This function can only reference data in the same subcube. If you need to reference data from a different subcube, you may need to create a “parking” account to store information from other cubes. For example, to reference a prior year’s data in the formula, you need to use a “parking” account to store the last year’s data information so that it can be referenced in the dynamic calculation within the same cube.

Note: You can embed the `HS.View.PeriodNumber` function in the Dynamic function. For example:

```
"HS.Dynamic "A#AvgUnits.I#ICP None.C1#None.C2#None = A#AccumUnits.I#ICP
None.C1#None.C2#None / HS.View.PeriodNumber"
```

Syntax

```
HS.Dynamic "DestPOV = Expression"
```

Table 55 Syntax for Dynamic Function

Parameter	Description
<i>DestPOV</i>	A valid Account member with type set to Dynamic. You may also specify a view for which to execute the calculation. Note: If you do not specify a view, the formula is executed for Periodic and YTD. To have different formulas for different views, you must specify Periodic or YTD in the formula.
<i>Expression</i>	A calculation expression.

Note: The Dynamic function does not support IF...THEN statements.

Return Value

None.

Example

```
Sub Dynamic
  HS.Dynamic "A#GM% = A#GM/A#Sales * 100)/HS.View.PeriodNumber"
End Sub
```

Expected results for the GM% account:

Custom1	Sales	GM	GM% (Dynamic Calculation)	Calculation based on formula
Product	600	140	23.33%	140 / 600 * 100

Custom1	Sales	GM	GM% (Dynamic Calculation)	Calculation based on formula
P1	100	10	10%	10 / 100 * 100
P2	200	40	20%	40 / 200 * 100
P3	300	90	30%	90 / 300 * 100

Exp

Puts data into a combination of Account, ICP, and Custom1...4 members. This function can be used in these types of rules:

- Calculation
- Translation
- Allocation

Syntax

`HS.Exp "DestPOV = Expression"`

Table 56 Syntax for Exp Function

Parameter	Description
<i>DestPOV</i>	A destination point of view that identifies where to put the data. You must specify an Account member, and you can optionally specify ICP and Custom members. Note these usage rules: • If you do not specify an ICP member, the default is [ICP None]. • To avoid populating the database with unwanted values, rules should be as explicit as possible in terms of defining where data should reside. A good practice is to include clearly defined Custom dimension intersections for the Account dimension. Such clearly defined intersections utilize Financial Management validation checks to avoid writing data to invalid intersections.
<i>Expression</i>	A calculation expression.

The destination for the data is on the left side of the equal sign, and the data to be assigned is on the right side. This example sets the cell that intersects the UnitsSold account and the [None] members of the Custom dimensions to 40000:

```
HS.Exp "A#UnitsSold.C1#[None].C2#[None].C3#[None].C4#[None]" _
& " = 40000"
```

On the right side of the equal sign, you can use Account Expression characters to represent dimension members, not just the Intercompany Partner and Custom dimensions. Thus, you can assign data for a group of cells to another group of cells. This example sets the cell that intersects the Taxes account and the [None] members of the Custom dimensions to 6 percent of the data in the cell that intersects the Sales account and the specified Custom dimensions:

```
HS.Exp "A#Taxes.C1#[None].C2#[None].C3#[None].C4#[None]" _
& " = A#Sales.C1#AllProducts.C2#AllCustomers.C3#[None]" _
& ".C4#[None] * .06"
```

Tip: You can set multiple accounts with one Exp statement. See [“Simultaneously Setting Several Accounts” on page 265](#).

Dimension Intersection Considerations

If you do not specify the dimension intersections from which Exp gets data and into which Exp places data, the source and destination intersections are determined by these factors:

- **Destination.** If no member of the Intercompany partner dimension or of a Custom dimension is specified on the left side of the equal sign, Exp places data into each valid intersection of the account and the dimension. If you do not specify a destination account, Financial Management will insert data into all accounts that are valid for the current point of view. See [“Simultaneously Setting Several Accounts” on page 265](#).
- **Source.** If a member of a dimension is not specified on the right side of the equal sign, there are several possibilities:
 - If a dimension has only one member, then Exp gets data from the intersection of this member and the source account.
 - If a dimension has only one valid intersection with the source account, then Exp gets data from this intersection.
 - If a dimension has several intersecting members with the source account, then the source intersection of the data is determined by the left side of the equation:
 - If a member is specified on the left side, then Exp attempts to get the data from the intersection of this member and the source account.
 - If a member is not specified on the left side, then Exp attempts to put data into each valid intersection of the destination account and the dimension’s members. Exp gets the data for the destination intersections from the corresponding intersections of the members and the source account.

Note: If a source intersection is invalid, then Exp does not change the data in the corresponding destination intersection.

For detailed examples that illustrate these considerations, see [“Exp and Dimension Intersection Considerations” on page 265](#).

Period and Year Keywords

To create dynamic rules, you can use the keywords in [Table 57](#) instead of member names to represent members of the destination Year or Period dimensions:

Table 57 Period and Year Keywords and Descriptions

Keyword	Description
CUR	The current period or year.

Keyword	Description
FIRST	The first period or year that is valid for the application.
LAST	The last period or year that is valid for the application.
NEXT	The period or year that follows the current period or year.
PRIOR	The period or year that precedes the current period or year.

Note: Period and Year keywords are case-sensitive and must be in all uppercase letters.

You can use the plus (+) and minus (-) signs with the Period and Year keywords. This example sets the MiscPast account to the amount in the Misc account two years before the current year.

```
HS.Exp "A#MiscPast = A#Misc.Y#CUR-2"
```

If you use the keywords Prior, First, Last, Current, or Next, immediately followed by the plus (+) and minus (-) signs and a digit, you must be careful to use the correct syntax for the order of the equation. In these cases, you can use one of these methods to write the rule:

- Always use parentheses to correctly separate the variable. For example:

```
HS.Exp "A#9001_Group.C4#[None] = A#9001_Group.P#Prior" & "+" & VAR  
& "A#9001_Group.V#[ParentTotal]"
```

Or

```
HS.Exp "A#9001_Group.C4#[None] = (A#9001_Group.P#Prior" & "+" & VAR  
& ") *A#9001_Group.V#[Parent Total]"
```

- Insert a space after keywords Prior, First, Last, Current and Next if you don't want to use numbers as part of the POV. The above rule can be written as:

```
HS.Exp "A#9001_Group.C4#[None] = A#9001_Group.P#Prior" & "+" & VAR &  
"A#9001_Group.V#[Parent Total]"
```

Mathematical Calculations

You can add, subtract, multiply, and divide on the right side of the equal sign. You must use these standard VBScript characters:

+ - * /

This example adds the values in the prior year's Taxes and Purchases accounts and then inserts the sum in the Cash account.

```
HS.Exp "A#Cash = A#Taxes.Y#PRIOR + A#Purchases.Y#PRIOR"
```

Note: If you multiply or divide with an account that has a NoData status, the data in the account on the left side of the equal sign will not be changed. Zero (0) is considered data. In other words, an account that contains 0.00 as data will *not* have a NoData status.

Placing Other Functions Within Exp

If a function returns a single numeric value, you can nest the function within the Exp function. However, if you nest a function that contains a String argument, you cannot surround the String with quotation marks. In this example, the NumBase function is nested in the Exp function, and so its String argument is *not* surrounded by quotation marks.

```
HS.Exp "A#SalesAlloc = A#Sales/HS.Entity.NumBase(Regional) "
```

Simultaneously Setting Several Accounts

To insert data into all accounts that intersect the current point of view, use All with the account expression. You can use this to set the opening balances of all accounts. In this example, the IsFirst function tests whether the current period is the first period. If it is the first period, then Exp sets each account's value for the current period to the account's value from the last period of the prior year.

```
If HS.Period.IsFirst = TRUE Then  
    HS.Exp "A#ALL = A#ALL.P#PRIOR"  
End If
```

To insert data into all intersections of accounts and Custom or Intercompany Partner dimensions, use the All keyword or omit the A# characters. This example, which omits the A# characters, inserts data into each account that intersects the GolfBalls member of the Custom1 dimension. For each valid intersection of GolfBalls and an account, the amount in the prior period's intersection is placed in the current period's intersection.

```
HS.Exp "C1#GolfBalls = C1#GolfBalls.P#PRIOR"
```

Return Value

None.

Example

This example sets the amount in the StateTax account. The example calculates this amount by multiplying the amount in the Sales account for 2009 by the rate in the StateRate account for 2009.

```
HS.Exp "A#StateTax = A#Sales.Y#2009 * A#StateRate.Y#2009"
```

Exp and Dimension Intersection Considerations

The following examples illustrate the considerations introduced in [“Dimension Intersection Considerations” on page 263](#). These types of intersections are covered:

- All intersections are valid for the source and destination accounts. See [“All Intersections Valid” on page 266](#).
- Some intersections are valid and others invalid for the source and destination accounts. See [“Invalid Intersections” on page 267](#).

- One member is valid for the source account. See [“One Valid Member on the Right Side” on page 268](#).

All of these examples use accounts named TargAcct and SourceAcct in conjunction with members of the Custom1 dimension named Member001, Member002, and Member003. The source intersection data for all the examples is listed in [Table 58](#):

Table 58 Data for the Dimension Intersection Examples

Member	Data in SourceAcct Intersection
Member001	10
Member002	NoData status
Member003	15

All Intersections Valid

For this example, all intersections of the TargAcct and SourceAcct accounts and the Custom1 members are valid:

```
HS.Exp "A#TargAcct = A#SourceAcct"
```

The function puts this data into the intersections of the TargAcct account and the Custom1 members:

Custom1 Member	Data	Intersection
Member001	10	SourceAcct and Member001
Member002	---	<i>Not applicable.</i> The intersection of TargAcct and Member002 is unchanged because the intersection of SourceAcct and Member002 has a NoData status.
Member003	15	SourceAcct and Member003

This example uses Exp with the Member001 member on the left side of the equal sign:

```
HS.Exp "A#TargAcct.C1#Member001 = A#SourceAcct"
```

The intersection of TargAcct and Member001 is set to 10. Exp gets the data from the intersection of SourceAcct and Member001 because Member001 is specified on the left side.

This example uses EXP with Member003 on the right side of the equal sign:

```
HS.Exp "A#TargAcct = A#SourceAcct.C1#Member003"
```

The function puts this data into the intersections of the Custom1 members and the TargAcct account:

Custom1 Member	Data	Intersection
Member001	15	SourceAcct and Member003

Custom1 Member	Data	Intersection
Member002	15	SourceAcct and Member003
Member003	15	SourceAcct and Member003

Invalid Intersections

In these examples, the source and destination accounts each have an invalid intersection.

- SourceAcct. Member002 and Member003 are valid, and Member001 is invalid.
- TargAcct. Member001 and Member002 are valid, and Member003 is invalid.

In this example, Exp is used without specifying a Custom1 member on either side of the equal sign:

```
HS.Exp "A#TargAcct = A#SourceAcct"
```

Exp does not change data in the TargAcct account because Exp attempts to perform these operations:

- TargAcct.Member001 = SourceAcct.Member001. SourceAcct and Member001 is an invalid intersection.
- TargAcct.Member002 = SourceAcct.Member002. Since the intersection of SourceAcct and Member002 has a NoData status, the intersection of TargAcct and Member002 remains unchanged.
- TargAcct.Member003 = SourceAcct.Member003. TargAcct and Member003 is an invalid intersection.

In this example, Exp is used with Member001 specified on the left side of the equal sign:

```
HS.Exp "A#TargAcct.C1#Member001 = A#SourceAcct"
```

TargAcct.Member001 remains unchanged because Exp attempts to retrieve data from an invalid intersection (SourceAcct and Member001).

In this example, Exp is used with Member003 specified on the right side of the equal sign:

```
HS.Exp "A#TargAcct = A#SourceAcct.C1#Member003"
```

The function puts this data into the intersections of the Custom1 members and the TargAcct account:

Custom1 Member	Data	Intersection
Member001	15	SourceAcct and Member003
Member002	15	SourceAcct and Member003
Member003	N/A	Not applicable. Member003 is an invalid intersection for the TargAcct account.

One Valid Member on the Right Side

In these examples, the source account has only one valid member, and the destination account has two valid members.

- SourceAcct. Member003 is the only valid intersection.
- TargAcct. Member001 and Member002 are valid, and Member003 is invalid.

In this example, Exp is used without specifying a Custom1 member on either side of the equal sign:

```
HS.Exp "A#TargAcct = A#SourceAcct"
```

The function puts this data into the intersections of the Custom1 members and the TargAcct account:

Custom1 Member	Data	Intersection
Member001	15	SourceAcct and Member003 (the only valid intersection for the SourceAcct account)
Member002	15	SourceAcct and Member003 (the only valid intersection for the SourceAcct account)
Member003	N/A	<i>Not applicable.</i> Member003 is an invalid intersection for the TargAcct account.

In this example, Exp is used with Member001 specified on the left side of the equal sign:

```
HS.Exp "A#TargAcct.C1#Member001 = A#SourceAcct"
```

The intersection of TargAcct and Member001 is set to 15, which is the data in the intersection of SourceAcct and Member003.

Tip: If there were more than one valid intersection for the SourceAcct account and the Custom1 dimension, Exp would attempt to get data from the intersection of SourceAcct and Member001. If this were an invalid intersection, then Exp would leave the destination account unchanged.

GetCell

Gets the data contained in a cell. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Syntax

```
HS.GetCell ("POVExpression")
```

where *POVExpression* is a valid point of view.

Return Value

The data stored in the specified cell.

Note: If the function returns more than one value, an error occurs.

Example

This example assigns to the `dData` variable the amount stored in the intersection of the Sales account and the Golfballs member of the Custom1 dimension:

```
Dim dData
dData = HS.GetCell("A#Sales.I#[ICP
None].C1#Golfballs.C2#Customer2.C3#Increases.C4#[None] ")
```

GetCellNoData

Gets the data contained in a cell and also indicates whether the cell contains data. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Syntax

```
HS.GetCellNoData("POV", Var1)
```

Table 59 Syntax for GetCellNoData Function

Parameter	Description
<i>POV</i>	A valid point of view.
<i>Var1</i>	A variable that specifies if there is data in the cell.

Return Value

The possible return values depend on what is found in the cell:

- If the cell contains real data, the data value is returned and the boolean value returned for *Var1* is False.
- If the cell contains no data, 0 is returned for the data value and the boolean value returned for *Var1* is True.
- If the cell contains derived data, the derived value is returned and the boolean value returned for *Var1* is False.

Caution! If the argument causes `GetCellNoData` to return more than one value, an error occurs.

Example

In this example, the amount in the Sales account is assigned to the `dSalesAmt` variable. If the Sales account has no data, the statements between the `IfThen` and `End If` statements are executed.

```
dSalesAmt = HS.GetCellNoData("A#Sales.I#[ICP  
None].C1#[None].C2#[None].C3#[None].C4#[None]", bIsNoData)  
If bIsNoData = TRUE then  
    ...  
End If
```

GetCellRealData

Gets the data contained in a cell and also indicates whether the cell contains real data. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Syntax

```
HS.GetCellRealData("POV", Var1)
```

Table 60 Syntax for `GetCellRealData` Function

Parameter	Description
<i>POV</i>	A valid point of view.
<i>Var1</i>	A variable that specifies if there is real data in the cell.

Return Value

The possible return values depend on what is found in the cell:

- If the cell contains real data, the data value is returned and the boolean value returned for *Var1* is `True`.
- If the cell contains no data, 0 is returned for the data value and the boolean value returned for *Var1* is `False`.
- If the cell contains derived data, the derived value is returned and the boolean value returned for *Var1* is `False`.

Example

```
dData = HS.GetCellRealData("A#Sales.C1#Prod1", bIsRealData)
```

```
If bIsRealData = TRUE then
...
End If
```

GetCellText

Gets the cell text from the point of view. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Syntax

```
HS.GetCellText("POVExpression")
```

where *POVExpression* is a valid point of view.

Return Value

The cell text for the POV.

Example

```
HS.GetCellText("S#Actual.Y#2009.P#January.E#Connecticut.V#<Entity  
Currency>.A#Sales.I#[ICP None].C1#[None].C2#[None].C3#[None].C4#[None]")
```

GetCellType

Gets the cell type. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Note: Account types use all capital letters.

Syntax

```
HS.GetCellType("POVExpression")
```

where *POVExpression* is a valid point of view.

Return Value

The type for the specified cell.

Valid types are:

- ASSET
- LIABILITY
- REVENUE
- EXPENSE
- FLOW
- BALANCE
- BALANCERECURRING
- CURRENCYRATE
- GROUPLABEL

Example

This example checks to see if the cell type is EXPENSE. If it is, then statements between the If...Then and End If statements are executed.

```
If HS.GetCellType("S#Actual.C4#[None]") = "EXPENSE" Then
    ...
End If
```

GetItem

Gets an individual record to process for consolidation. This function can be used in Consolidation rules.

Syntax

```
HS.DataUnit.GetItem (lItem, strAccount, strICP, strCustom1, strCustom2, strCustom3,  
strCustom4, dData)
```

Table 61 Syntax for GetItem Function

Parameter	Description
<i>lItem</i>	A record number.
<i>strAccount</i>	Name of a valid Account dimension member.
<i>strICP</i>	Name of a valid ICP dimension member.
<i>strCustom1</i>	Name of a valid Custom1 dimension member.
<i>strCustom2</i>	Name of a valid Custom2 dimension member.
<i>strCustom3</i>	Name of a valid Custom3 dimension member.
<i>strCustom4</i>	Name of a valid Custom4 dimension member.
<i>dData</i>	The data in the specified cell.

Return Value

An array containing the account, ICP, Custom1...4, data.

Example

```
Call DataUnit.GetItem(lItem, strAccount, strICP, strCustom1, strCustom2, strCustom3,
strCustom4, dData)
If dData = 0 Then
    ...
End If
```

GetItemIDs2

Using dimension IDs, gets an individual record to process for consolidation. This function can be used in Consolidation rules.

Syntax

```
HS.DataUnit.GetItemIDs2(lItem, lAccountID, lICPID, lCustom1ID, lCustom2ID, lCustom3ID,
lCustom4ID, dData)
```

Table 62 Syntax for GetItemIDs2Function

Parameter	Description
<i>lItem</i>	A record number.
<i>lAccountID</i>	ID number of a valid Account dimension member.
<i>lICPID</i>	ID number of a valid ICP dimension member.
<i>lCustom1ID</i>	ID number of a valid Custom1 dimension member.
<i>lCustom2ID</i>	ID number of a valid Custom2 dimension member.
<i>lCustom3ID</i>	ID number of a valid Custom3 dimension member.
<i>lCustom4ID</i>	ID number of a valid Custom4 dimension member.
<i>dData</i>	The data in the specified cell.

Return Value

An array containing the account, ICP, Custom1...4, data.

Example

```
Call DataUnit.GetItemIDs2(lItem, lAccount, lICP, lCustom1, lCustom2, lCustom3, lCustom4,
dData)
If dData = 0 Then
    ...
End If
```

GetNumItems

Gets the number of records to process for consolidation. This function can be used in Consolidation rules.

Syntax

```
NumItems = HS.DataUnit.GetNumItems
```

Return Value

The number of records in the data unit.

Example

```
Set dataUnit = HS.OpenDataUnit("")
lNumItems = dataUnit.GetNumItems
for lItem = 0 to lNumItems - 1
' Get the next item from the Data Unit
Call dataUnit.GetItem(lItem, strAccount, strICP, strCustom1, strCustom2, strCustom3,
strCustom4, dData)
```

GetNumLID

Gets the number of line item details for the specified point of view. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Syntax

```
HS.GetNumLID("POVExpression")
```

where *POVExpression* is a valid POV combination. If you do not specify a dimension, these values are used:

- Account - [none]
- ICP - [ICP None]
- Custom1...4 - [None]
- Scenario - Current Scenario member
- Entity - Current Entity member
- Value - <entity currency>
- Year and Period - Current member

Note: If an invalid intersection is specified, the return value is 0.

Return Value

The number of line item details entered for the specified cell.

Example

In this example, if no line item details have been entered for the specified cell, then statements between the If...Then and End If lines are executed.

```
If HS.GetNumLID("A#Sales.C1#Prod1.C2#Region1") = "0" then
    ...
End If
```

GetRate

Gets the currency rate for the current point of view or for a specified point of view. This function can be used in the these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Syntax

```
HS.GetRate("POVExpression")
```

where POVExpression is a valid point of view.

Note: You must specify a rate account.

Return Value

The currency rate for the specified point of view.

Example

```
dVar1 = HS.GetRate("S#Actual.Y#2009.P#March.V#Euro.E#Connecticut.A#AvgRate")
```

GetSubmissionGroup

Gets the process management submission group for the cell.

Syntax

```
HS.GetSubmissionGroup("A#Account.C1#Custom1.C2#Custom2.C3#Custom3.C4#Custom4.I#ICP")
```

Return Value

An integer representing the process management submission group. Valid values are 1–99.

Example

```
HS.GetSubmissionGroup("A#Sales.C1#Golfballs.C2#Tennisballs.C3#Soccerballs.C4#Basketballs.I#EastSales")
```

GetSubmissionPhase

Gets the process management submission phase for the cell.

Syntax

```
HS.GetSubmissionPhase("S#Scenario.P#Period.A#Account.C1#Custom1.C2#Custom2.C3#Custom3.C4#Custom4.I#ICP")
```

Return Value

An integer representing the process management submission phase. Valid values are 1–9.

Example

```
HS.GetSubmissionPhase("S#Actual.P#January.A#Sales.C1#Golfballs.C2#Tennisballs.C3#Soccerballs.C4#Basketballs.I#EastSales")
```

Holding

Gets the holding company for the current Entity dimension member or for a specified entity member. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Syntax

```
HS.Entity.Holding("Entity")  
HS.Entity.Holding("")
```

where *Entity* is the name of a valid Entity dimension member. You can use a blank string (“”) to apply this function to the current member.

Return Value

A String that contains the name of the holding company for the specified entity member.

Example

In this example, if Europe is the holding company for the entity France, then statements between the If...Then and End If lines are executed.

```
If HS.Entity.Holding("France") = "Europe" Then
    ...
End If
```

ICPTopMember

Gets the ICP Top Member of the current Account dimension member or the specified account member. This function can be used in the these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Syntax

```
HS.Account.ICPTopMember("AccountName")
HS.Account.ICPTopMember(" ")
HS.Account.ICPTopMember(Var1)
```

Note: You can use a blank string (" ") to apply this function to the current account only if you are using it in a sub consolidate routine.

Table 63 Syntax for ICPTopMember Function

Parameter	Description
<i>AccountName</i>	Name of a valid Account dimension member.
<i>Var1</i>	A VisualBasic variable.

Return Value

A string with the name of the ICP Top Member for the account.

Example

In this example, if the ICP Top Member of the Sales account is TotalProd, then statements between the If...Then and End If statements are executed.

```
If HS.Account.ICPTopMember("Sales") = "TotalProd" Then
    ...
End If
```

ICPWeight

Gets the ICP weight for the application. The percentage of intercompany partner entity [ICP Entities] amounts that aggregate to the [ICP Top] member of the Value dimension. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Syntax

```
HS.AppSettings.ICPWeight
```

Return Value

The percentage of ICP entities that are aggregated into the ICP top member. The value is a percentage scaled to hundreds, with 1.0 equalling 100 percent.

Example

In this example, if the ICPWeight of the current application is 1, then statements between the If...Then and End If lines are executed.

```
If HS.AppSettings.ICPWeight = 1 Then  
    ...  
End If
```

IDFromMember

Gets the ID number for the specified member. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Syntax

```
HS.<Object>.IDFromMember ( "Element" )
```

Table 64 Syntax for IDFromMember Function

Parameter	Description
<i><Object></i>	One of these object keywords: ● Account ● Custom1...4 ● Entity ● ICP ● Parent ● Period ● Scenario ● Value ● Year ● View
<i>Element</i>	Depending on the object selected, the name of a valid member of one of these dimensions: ● Account ● Custom1...4 ● Entity ● ICP ● Parent ● Period ● Scenario ● Value ● Year

Return Value

The ID number of the specified member.

Example

This example gets the ID number for Connecticut:

```
lEntityID = HS.Entity.IDFromMember("Connecticut")
```

ImpactStatus

Changes the status of the specified data unit to impacted. This function can be used in Calculation rules.

Syntax

Combination of Scenario, Year, Period, Entity, and Value members. If the scenario is the same, the year and period combination must be a future period. If no value member is specified, it is assumed to be current.

HS.ImpactStatus "S#Scenario.Y#Year.P#Period.E#Entity.V#Value"

Table 65 Syntax for ImpactStatus Function

Parameter	Description
<i>Scenario</i>	Name of a valid Scenario dimension member.
<i>Year</i>	A valid year.
<i>Period</i>	A valid period.
<i>Entity</i>	Name of a valid Entity dimension member.
<i>Value</i>	Name of a valid Value dimension member.

Note: If the specified POV is the same scenario, year, period, and entity as the data unit being processed, an error occurs and there is

Return Value

None.

Example

HS.ImpactStatus "S#Actual.Y#2009.P#April"

Input

Enables data input into parent entities at base, calculated accounts. Only the entity currency Value dimension is supported. When you use this function, the value at the Parent Entity level is not equal to the sum of its children's contributions.

Because this function enables input at the Parent entity level, the children contributions do not roll up to the Parent entity's entity currency Value dimension. However, the parent-child contribution value is stored, and you can still post journals to the Contribution Adjustments Value dimension. This function can be used in Input rules.

Caution! For accounts where you allow input at the parent entity level, it is important to remember that the value at the Parent entity's entity currency Value member will not be equal to the sum of all the children's contributions.

Syntax

HS.Input "POVExpression"

where *POVExpression* is a point of view.

Return Value

None.

Example

This example enables input into the cells that intersect the Sales account and the Budget scenario:

```
Sub Input
    HS.Input "A#Sales.S#Budget "
End Sub
```

IsAlmostEqual

Checks to see if the passed in values are equal based on a predefined Financial Management epsilon. This function can be used in all types of rules.

A difference of -0.0000000000001 to 0.0000000000001 is considered zero difference

Syntax

```
BooleanValue = HS.IsAlmostEqual(Value1, Value2)
```

Return Value

A Boolean expression that is True if the passed in values are equal ; False if they are not equal.

Example

```
Dim BoolVal
Dim Value1
Dim Value2
Value1 = 10.1299999999
Value2 = 10.13
BoolVal = HS.IsAlmostEqual(Value1, Value2)
If BoolVal = true Then
    // do processing
Else
    // do Processing
End If
```

IsBase

Determines if the current member or a specified member is a base member of the application or of the specified parent. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Note: A member is a base member if it has no children (that is, it is at the end of branch in a tree hierarchy).

Syntax

```
HS.<Object>.IsBase("Parent","Element")
HS.Node.IsBase("Parent"."Entity"."S#Scenario.Y#Year.P#Period")
HS.<Object>.IsBase("","")
```

Note: Use a blank string (“”) to apply this function to the current member.

Table 66 Syntax for IsBase Function

Parameter	Description
<i><Object></i>	One of these object keywords: ● Account ● Custom1...4 ● Entity ● Parent
<i>Parent</i>	A valid Parent member. Note: Parent is mandatory only when used with Node.
<i>Element</i>	Depending on the object selected, the name of a valid member of one of these dimensions: ● Account ● Custom1...4 ● Entity ● Parent
<i>Entity</i>	Name of a valid Entity dimension member.
<i>Scenario</i>	Name of a valid Scenario dimension member.
<i>Year</i>	A valid year.
<i>Period</i>	A valid period.

Note: For Node, determines if the element is an active base member of the specified parent.

Return Value

A Boolean expression that is True if the element is a base member below the specified parent or, when no parent is specified, is a base member in the application. False if the element is not a base member.

For Node, True if the element is an active base entity below the parent in the specified point of view. False if the element is not an active base entity.

Example

In this example, if Connecticut is a base entity under EastRegion, then statements between the If...Then and End If lines are executed.

```
If HS.Entity.IsBase("EastRegion","Connecticut") = TRUE Then
    ...
End If
```

IsCalculated

Determines if the current Account dimension member or a specified account member is a calculated account. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Syntax

```
HS.Account.IsCalculated("Account")
```

where *Account* is the name of a valid Account member.

```
HS.Account.IsConsolidated("")
```

Note: You can use a blank string ("") to apply this function to the current member only if you are using it in a sub consolidate routine.

Return Value

A Boolean expression that is True if the account is a calculated account; False if the account is not a calculated account.

Example

In this example, if the Sales account is calculated, then statements between the If...Then and End If statements are executed.

```
If HS.Account.IsCalculated("Sales") = TRUE Then
    ...
End If
```

IsChild

Determines if the current member or a specified member is a child of the specified parent. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation

- Allocation

Note: A member is a child if it is one level directly below a member in a tree hierarchy.

Syntax

```
HS.<Object>.IsChild("Parent", "Element")
HS.Node.IsChild("Parent"."Entity"."S#Scenario.Y#Year.P#Period")
HS.<Object>.IsChild("Parent", "")
```

Note: Use a blank string (" ") to apply this function to the current member.

Table 67 Syntax for IsChild Function

Parameter	Description
<i><Object></i>	One of these object keywords: ● Account ● Custom1...4 ● Entity ● Parent
<i>Parent</i>	Name of a Parent member. (Parent is mandatory.)
<i>Element</i>	Depending on the object selected, the name of a valid member of one of these dimensions: ● Account ● Custom1...4 ● Entity ● Parent
<i>Entity</i>	Name of a valid Entity dimension member.
<i>Scenario</i>	Name of a valid Scenario dimension member.
<i>Year</i>	A valid year
<i>Period</i>	A valid period

Note: For Node, determines if the member is an active child of the specified parent.

Return Value

A Boolean expression that is True if the element is a child of the specified parent; False if the element is not a child of the specified parent.

For Node, True if the element is an active child of the specified parent; False if the element is not an active child of the specified parent.

Example

In this example, if Connecticut is a child of EastRegion, then statements between the If...Then and End If lines are executed.

```
If HS.Entity.IsChild("EastRegion","Connecticut") = TRUE Then
    ...
End If
```

IsConsolidated

Determines if the current Account dimension member or a specified account member is a consolidated account. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Syntax

```
HS.Account.IsConsolidated("Account")
```

where *Account* is the name of a valid Account member.

```
HS.Account.IsConsolidated(" ")
```

Note: You can use a blank string (" ") to apply this function to the current member only if you are using it in a sub consolidate routine.

Return Value

A Boolean expression that is True if the account is consolidated into a parent account; False if the account is not consolidated into a parent account.

Example

In this example, if the Sales account is consolidated, then statements between the If...Then and End If statements are executed.

```
If HS.Account.IsConsolidated("Sales") = TRUE Then
    ...
End If
```

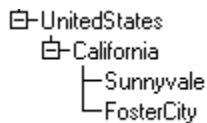
IsDescendant

Determines if the current member or a specified member is a descendant of the specified parent. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Note: A member is a descendant if it is at a level below a parent in a tree hierarchy. Descendants are within the same branch of the tree.

For example, in this hierarchy, FosterCity and Sunnyvale are descendants of California and UnitedStates.



Syntax

```

HS.<Object>.IsDescendant ("Parent", "Element")
HS.Node.IsDescendant ("Parent"."Entity"."S#Scenario.Y#Year.P#Period")
HS.<Object>.IsDescendant ("Parent", "")
  
```

Note: Use a blank string (" ") to apply this function to the current member.

Table 68 Syntax for IsDescendant Function

Parameter	Description
<i><Object></i>	One of these object keywords: ● Account ● Custom1...4 ● Entity ● Parent
<i>Parent</i>	Name of a valid Parent member. Parent is required.
<i>Element</i>	Depending on the object selected, name of a valid member of one of these dimensions: ● Account ● Custom1...4 ● Entity ● Parent
<i>Entity</i>	Name of a valid Entity dimension member.
<i>Scenario</i>	Name of a valid Scenario dimension member.
<i>Year</i>	A valid year.

Parameter	Description
<i>Period</i>	A valid period.

Note: When you use node as the object, the function determines if the member is an active descendant of the specified parent.

Return Value

A Boolean expression that is True if the element is a descendant of the specified parent; False if the element is not a descendant of the specified parent.

For Node, True if the element is an active descendant of the specified parent; False if the element is not an active descendant of the specified parent.

Example

In this example, if Connecticut is a descendant of Regional, then statements between the If...Then and End If lines are executed.

```
If HS.Entity.IsDescendant("Regional","Connecticut") = TRUE Then
    ...
End If
```

IsFirst

Determines if the current period or year is the first period or year of the application. The default frequency of the current scenario is used to determine if the current period or year is the first period or year of the application. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Syntax

HS.<Object>.IsFirst

where <Object> is one of these keywords:

- Period
- Year

Return Value

A Boolean expression that is True if the current period or year is the first period or year; False if the current period or year is not the first period or year.

Example

In this example, if the current period is the first period then statements between the If...Then and End If statements are executed:

```
If HS.Period.IsFirst = TRUE Then
    ...
End If
```

IsICP

Determines if the current Account or Entity dimension member or a specified account or entity member is an intercompany partner (ICP). This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Syntax

```
HS.<Object>.IsICP("Element")
HS.<Object>.IsICP(" ")
```

Note: Use a blank string (" ") to apply this function to the current member.

Table 69 Syntax for IsICP Function

Parameter	Description
<i><Object></i>	One of these object keywords: ● Account ● Entity
<i>Element</i>	Depending on the object selected, name of a valid member of the Account or Entity dimension. ● Account ● Entity

Return Value

A Boolean expression that is True if the account or entity member is an intercompany partner; False if the account or entity member is not an intercompany partner.

Example

In this example, if the Sales account is an intercompany partner, then statements between the If...Then and End If lines are executed.

```
If HS.Account.IsICP("Sales") = TRUE Then
```



```
...  
End If
```

IsLast

Determines if the current period or year is the last period or year of the application. The default frequency of the current scenario is used to determine if the current period or year is the last period or year of the application. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Syntax

```
HS.<Object>.IsLast
```

where <Object> is one of these keywords:

- Period
- Year

Return Value

A Boolean expression that is True if the current period or year is the last period or year; False if the current period or year is not the last period or year.

Example

In this example, if the current period is the last period, then statements between the If...Then and End If statements are executed:

```
If HS.Period.IsLast = TRUE Then  
...  
End If
```

IsTransCur

Determines if the current Value dimension member is a translated currency member. This function can be used in Translation rules.

Syntax

```
HS.Value.IsTransCur
```

Return Value

A Boolean expression that is True if the current Value member is a translated currency member; False if the current Value member is not a translated currency member.

Example

In this example, if the Value member is a translated currency member, then all statements between the If...Then and End If statements are executed.

```
If HS.Value.IsTransCur = TRUE Then
    ...
End If
```

IsTransCurAdj

Determines if the current Value dimension member is a translated currency Adj member. This function can be used in Translation rules.

Syntax

```
HS.Value.IsTransCurAdj
```

Return Value

A Boolean that is True if the current Value member is a translated currency Adj member; False if the current Value member is not a translated currency Adj member.

Example

In this example, if the Value member is a translated currency Adj member, then all statements between the If...Then and End If statements are executed.

```
If HS.Value.IsTransCurAdj = TRUE Then
    ...
End If
```

IsValidDest

Determines if the specified point of view is a valid destination. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Note: This function does not check to see whether the cell is a calculated cell.

Syntax

```
HS.IsValidDest ("POVExpression")
```

where *POVExpression* is a point of view. If you do not specify a dimension, these default values are used:

- Account - Current Account member if used in Sub Consolidate routine. Otherwise, account is required.
- Custom and ICP - Current member is used in Sub Consolidate routine. Otherwise, top member for the account is used.
- Scenario - Current Scenario member
- Entity - Current Entity member
- Value - Current Value member
- Year and Period - Current member

Return Value

A Boolean that is True if the specified point of view is a valid destination; False otherwise.

Example

In this example, if the specified destination is valid, then all statements between the If...Then and End If statements are executed.

```
If HS.IsValidDest("A#Sales.I#CT.C1#P1.C2#R1.C3#[None].C4#[None]") = TRUE Then
    ...
End If
```

IsZero

Checks to see if the passed in value is close to zero based on a predefined Financial Management epsilon. This function can be used in all types of rules.

Syntax

```
BooleanValue = HS.IsZero(Value)
```

Return Value

A Boolean that is True if the passed in value is close to zero. False otherwise.

Example

```
Dim BoolVal
Dim Value
Value = 0.000000001
BoolVal = HS.IsZero(Value)
If BoolVal = true Then
    // do processing
Else
    // do Processing
End If
```

List

Gets the elements contained in the specified list. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Syntax

```
HS.<Object>.List("Parent"."Listname")  
HS.Node.List("Parent"."Listname"."S#Scenario.Y#Year.P#Period")
```

Table 70 Syntax for List Function

Parameter	Description
<i><Object></i>	One of these object keywords: ● Account ● Custom1...4 ● Entity ● Parent ● ICP ● Scenario
<i>Parent</i>	Name of a valid Parent member.
<i>Listname</i>	Name of a valid system list or user-defined list. Note: For node, must be a valid entity system list.
<i>Scenario</i>	Name of a valid Scenario dimension member.
<i>Year</i>	A valid year.
<i>Period</i>	A valid period.

Return Value

An array that contains all elements of the specified list. For Node, only the active elements in the list.

Example

This example gets the elements of the user-defined list MyBaseList for the current account:

```
HS.Account.List("", "MyBaseList")
```

This example gets the elements of the system list [Base] for the TotalAssets account:

```
HS.Account.List("TotalAssets", "[Base]")
```

Member

Gets the name of the current member. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Syntax

HS.<Object>.Member

where <Object> is one of these keywords:

- Entity
- Parent
- Period
- Scenario
- Value
- Year
- View

Return Value

A String that contains the current member name.

For the Value object, Member returns the name of the current Value member, not the currency associated with the value. For example, if the current Value member is Entity Currency and the value is associated with the USD currency, HS.Parent.Member returns Entity Currency, not USD.

Tip: To get the currency of the current Value member, use the DefCurrency function.

Example

In this example, if the current entity is California then statements between the If...Then and End If statements are executed:

```
If HS.Entity.Member = "California" Then
    ...
End If
```

MemberFromID

Gets the dimension member for the specified ID number. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Syntax

`HS.<Object>.MemberFromID(ElementID)`

Table 71 Syntax for MemberFromID Function

Parameter	Description
<i><Object></i>	One of these object keywords: ● Account ● Custom1...4 ● Entity ● ICP ● Parent ● Period ● Scenario ● Value ● Year ● View
<i>Parent</i>	A valid Parent member.
<i>ElementID</i>	Depending on the object selected, the ID number of a valid member of one of these dimensions: ● Account ● Custom1...4 ● Entity ● ICP ● Parent ● Period ● Scenario ● Value ● Year

Return Value

The dimension member name.

Example

This example gets the member for the ID number 001:

```
strEntity = HS.Entity.MemberFromID(001)
```

Method

Gets the consolidation method for the specified member. If there is more than one non-zero value, the system returns the first one found. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Syntax

```
HS.Node.Method("POVExpression")
```

where *POVExpression* is a combination of Scenario, Year, Period, and Entity members.

Return Value

A String that specifies the consolidation method for the specified point of view.

Example

In this example, if the method for the point of view is GLOBAL, then statements between the If...Then and End If statements are executed.

```
If HS.Node.Method("S#Actual"."Y#2009"."P#January"."E#Regional.Connecticut") = "GLOBAL"
Then
    ...
End If
```

NoInput

Prevents users from entering data into specific cells or slices of cells. This is useful when there are accounts that are not input or calculated.

When you set up a calculated account, you in effect, are preventing users from being able to input to that account. However, if you have accounts in which data input is enabled for some dimension intersections but not for others, you can use NoInput. This function can be used in NoInput rules.

Syntax

```
HS.NoInput "POVExpression"
```

where *POVExpression* is a point of view.

Return Value

None.

Example

This example prohibits input into the cells that intersect the Sales account and the Budget scenario for 2009:

```
Sub NoInput
  HS.NoInput "S#Budget.Y#2009.A#Sales"
End Sub
```

NoRound

Turns off rounding for all following `Exp` statements. This function can be used in these types of rules:

- Calculation
- Translation
- Allocation

Tip: You can also turn off rounding by entering 0 as the argument for the `Round` function. For example, `HS.Round(0)` turns off rounding.

Syntax

```
HS.NoRound
```

Return Value

None.

Example

This example rounds the amount inserted into the SalesRound account's cells to the nearest tenth, then uses `NoRound` to turn off rounding for the amount inserted into the SalesNoRound account's cells:

```
HS.Round 0.1
HS.Exp "A#SalesRound" = "A#Sales"
HS.NoRound
HS.Exp "A#SalesNoRound" = "A#Sales"
```

NumBase

Gets the number of base members for the current member or for a specified member. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation

- Allocation

Note: A member is a base member if it has no children (that is, it is at the end of branch in a tree hierarchy).

Syntax

```
HS.<Object>.NumBase("Element")
HS.Node.NumBase("S#Scenario.Y#Year.P#Period.E#Entity")
HS.<Object>.NumBase(" ")
```

Note: Use a blank string (" ") to apply this function to the current member.

Table 72 Syntax for NumBase Function

Parameter	Description
<i><Object></i>	One of these object keywords: ● Account ● Custom1...4 ● Entity ● Parent ● Node
<i>Element</i>	Depending on the object selected, name of a valid member for one of these dimensions: ● Account ● Custom1...4 ● Entity ● Parent ● Node For Account and Custom objects, you must specify the member - you cannot use a blank string. To get the number of base members in the entire dimension, specify ALL within quotation marks, as in this example: <pre>iAcctBase = HS.Account.NumBase("ALL")</pre>
<i>Scenario</i>	Name of a valid Scenario dimension member.
<i>Year</i>	A valid year.
<i>Period</i>	A valid period.
<i>Entity</i>	Name of a valid Entity dimension member.

You can also embed the NumBase function in the Exp function. If you embed the NumBase function, do not surround NumBase's argument with quotation marks, as in this example:

```
HS.Exp "A#AverageSales = A#Sales/HS.Entity.NumBase(Regional) "
```

Return Value

A Long that identifies the number of base members. For Node, gets the number of active base elements of the specified member.

Note: If a base entity appears twice in a branch, the entity is counted twice.

Example

In this example, the application contains an account named SalesAlloc that stores the average sales amount for the base entities under the Regional entity. To calculate the SalesAlloc amount, the example divides the Sales account's amount by the number of base entities under Regional.

```
If HS.Exp"A#SalesAlloc = A#Sales/HS.Entity.NumBase(Regional)" then
...
End If
```

Number

Gets the current period number. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Syntax

```
HS.View.PeriodNumber
```

Return Value

The current period number.

Example

In this example, if the current period is the first period then statements between the If...Then and End If statements are executed.

```
If HS.View.PeriodNumber = 1 Then
...
End If
```

NumChild

Gets the number of child members for the current dimension member or for a specified member. This function can be used in these types of rules:

- Calculation

- Translation
- Consolidation
- Allocation

Note: A member is a child if it is one level directly below a member in a tree hierarchy. Only members one level below the specified object are counted.

Syntax

```
HS.<Object>.NumChild("Element")
HS.Node.NumChild("S#Scenario.Y#Year.P#Period.E#Entity")
HS.<Object>.NumChild(" ")
```

Note: Use a blank string (" ") to apply this function to the current member.

Table 73 Syntax for NumChild Function

Parameter	Description
<i><Object></i>	One of these object keywords: • Account • Custom1-4 • Entity • Node • Parent
<i>Element</i>	Depending on the object selected, name of a valid member of one of these dimensions: • Account • Custom1...4 • Entity • Node • Parent For Account and Custom objects, you must specify the member - you cannot use a blank string. To get the number of child members in the entire dimension, specify ALL within quotation marks, as in this example: <pre>iAcctBase = HS.Account.NumChild("ALL")</pre>
<i>Scenario</i>	Name of a valid Scenario dimension member.
<i>Year</i>	A valid year.
<i>Period</i>	A valid period.
<i>Entity</i>	Name of a valid Entity dimension member.

You can embed the NumChild function in the Exp function. If you embed the NumChild function, do not surround NumChild's argument with quotation marks.

Return Value

A Long that identifies the number of child members. For Node, gets the number of active children of the specified member.

Example

In this example, the application contains an account named SalesChild that stores the average sales amount for the entities immediately under the Regional entity. To calculate the SalesChild amount, the example divides the Sales account's amount by the number of children directly under Regional.

```
HS.Exp "A#SalesChild = A#Sales/HS.Entity.NumChild(Regional) "
```

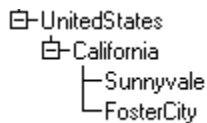
NumDescendant

Gets the number of descendants of the current dimension member or a specified member. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Note: A member is a descendant if it is at a level below a parent in a tree hierarchy. Descendants are within the same branch of the tree.

For example, in this hierarchy, FosterCity and Sunnyvale are descendants of California and UnitedStates.



Syntax

```
HS.<Object>.NumDescendant ("Element")
HS.Node.NumDescendant ("S#Scenario.Y#Year.P#Period.E#Entity")
HS.<Object>.NumDescendant ("")
```

Note: Use a blank string ("") to apply this function to the current member.

Table 74 Syntax for NumDescendant Function

Parameter	Description
<i><Object></i>	One of these object keywords: ● Account ● Custom1-4 ● Entity ● Node ● Parent
<i>Element</i>	Depending on the object selected, name of a valid member of one of these dimensions: ● Account ● Custom1...4 ● Entity ● Node ● Parent For Account and Custom objects, you must specify the member - you cannot use a blank string. To get the number of descendant in the entire dimension, specify ALL within quotation marks, as the argument as in this example: <pre>iAcctBase = HS.Account.NumDescendant ("ALL")</pre>
<i>Scenario</i>	Name of a valid Scenario dimension member.
<i>Year</i>	A valid year.
<i>Period</i>	A valid period.
<i>Entity</i>	Name of a valid Entity dimension member.

Return Value

The number of descendants of the specified member. For node, the number of active descendant entities below the specified member.

Note: If a descendant entity appears twice in a branch, the entity is counted twice.

Example

In this example, if the entity France has no descendants, then statements between the If...Then and End If statements are executed.

```
If HS.Entity.NumDescendant("France") = 0 then
...
End If
```

NumPerInGen

Gets the number of periods in the generation for the current period being processed. This function can be used in Dynamic SUB functions.

Syntax

```
HS.Period.NumPerInGen
```

Return Value

One value for the number of periods of the view.

Example

```
Var1=HS.Period.NumPerInGen
```

For example, if the current period is April, and April is in the fourth generation in the calendar file (monthly generation), the number of periods for the monthly generation is 12. If the current period is Q2, which is in the third generation of the calendar file (quarterly generation), the number of periods is 4.

Monthly generation (4th generation):

January, February, March, April, May, June, July, August, September, October, November, December

System returns 12 for the number of periods in this generation.

Quarterly generation (3rd generation):

Q1, Q2, Q3, Q4

System returns 4 for the number of periods in this generation.

Half-yearly generation (2nd generation):

HY1, HY2

System returns 2 for the number of periods in this generation.

Yearly generation (first generation):

Year

System returns 1 for the number of periods in this generation.

NumPeriods

Gets the number of periods defined for the frequency of the specified scenario. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation

- Allocation

Syntax

```
HS.Scenario.NumPeriod("ScenarioName")
```

or

```
HS.Scenario.NumPeriod("")
```

or

```
HS.Scenario.NumPeriod(Var1)
```

Return Value

Numeric value for the number of periods for the frequency. For example, if the scenario is monthly, the system returns 12 for the number of periods. If the scenario is quarterly, the system returns 4 for the number of periods.

Example

This example returns the number of periods defined for the frequency of the Actual scenario.

```
HS.Scenario.NumPeriod("Actual")
```

OpenDataUnit

Gets the data unit to process during consolidation, calculation, or translation. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation

Syntax

```
HS.OpenDataUnit(POVExpression)
```

where *POVExpression* is a POV. As part of the POV Expression, the function supports user-defined lists for Account, ICP, C1, C2, C3, and C4. System lists are not supported with this function.

Return Value

When used in a sub consolidate routine, returns all records with data but only returns accounts that are flagged as consolidated.

When used in a sub calculate or sub translate routine, returns all records containing data, including accounts that are flagged as consolidated.

Note: An account is consolidated if its IsConsolidated attribute = True.

Example

```
Set DataUnit=HS.OpenDataUnit("A{TotalRev.[Base]}.C1{C1Top.[Base]}.C2{MyC2List}.C3#[None]")
```

OpenDataUnitSorted

Gets the data units to process during calculation, translation, or consolidation, with data sorted in the order specified. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation

Syntax

```
HS.OpenDataUnitSorted(POVExpression, <dimension to be sorted>, Ascending or Descending)
```

where *POVExpression* is a POV string, <dimension to be sorted> is a dimension name string, and can only be one of the following six dimensions: "Account" or "A", "ICP" or "I", "Custom1" or "C1", "Custom2" or "C2", "Custom3" or "C3", "Custom4" or "C4". You must specify one of the following: Account, ICP, Custom1, Custom2, Custom3, Custom4. The third parameter is a string value ("Ascending" or "A", or "Descending" or "D").

Examples

```
Set DataUnit= HS.OpenDataUnitSorted("S#Actual.E#Group1", "C1", "Ascending")
```

```
HS.OpenDataUnitSorted("S#Actual.E#Group1", "Account", "Descending")
```

Owned

Gets the owned entity of the entity pair currently processed. This function is used in Equity Pickup rules.

Syntax

```
HS.Entity.Owned
```

Return Value

The owned entity.

Example

```
Owned=HS.Entity.Owned
```


Owner

Gets the owner of the entity pair currently processed. This function is used in Equity Pickup rules.

Syntax

```
HS.Entity.Owner
```

Return Value

The owner entity.

Example

```
Owner=HS.Entity.Owner
```

PCon

Gets the percentage of consolidation for the current member or a specified member. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Syntax

```
HS.Node.PCon("S#Scenario.Y#Year.P#Period.E#Entity")  
HS.Node.PCon(" ")
```

Note: Use a blank string (" ") to apply this function to the current member.

Table 75 Syntax for PCon Function

Parameter	Description
Scenario	Name of a valid Scenario dimension member.
Year	A valid year.
Period	A valid period.
Entity	Name of a valid Entity dimension member.

Return Value

The percentage of consolidation for the member.

Example

This example gets the percent consolidation for the specified point of view:

```
Pcon = HS.Node.PCon("S#Actual.Y#2009.P#January.E#Regional.Connecticut")
```

PEPU

Gets the percentage of ownership from the EPU table. This function is used in Equity Pickup rules.

Syntax

```
HS.PEPU("S#.Y#.P#", Owner, Owned)
```

Return Value

The ownership percentage from the EPU table.

Example

```
HS.PEPU(S#Actual.Y#2009.P#Jan, Group, CT)
```

or

```
HS.PEPU(, , ,)
```

Default parameters: if the values are blank, the function returns the percentage of ownership for the entity pair in the current Scenario, Year, and Period.

Example

```
Sub EquityPickup ( )  
Owned=Hs.Entity.Owned  
OwnerCurrencyTotl=Hs.Entity.DefCurrency & "Total"  
Hs.Clear "A#Inv.C4#EPU.I#" & Owned  
Hs.Exp "A#Inv.C4#EPU.I#" & Owned & "=A#EQ.C4#C3Tot.I#[ICPTot].E#" &  
Owned & ".V#" & OwnerCurrencyTotl & "*" & Hs.PEPU (, , ,)  
End Sub
```

PeriodNumber

Gets the period number in the view for the data that is being retrieved. This function can be used in Dynamic SUB functions.

Syntax

```
HS.View.PeriodNumber
```

Return Value

One value for the number of periods of the view.

Example

Var1=HS.View.PeriodNumber

	Periodic	YTD	QTD	HYTD
Jan.	1	1	1	1
Feb.	1	2	2	2
Mar.	1	3	3	3
Q1	1	1	1	1
Apr.	1	4	1	4
May	1	5	2	5
June	1	6	3	6
Q2	1	2	1	2
HY1	1	1	1	1
July	1	7	1	1
Aug.	1	8	2	2
Sept.	1	9	3	3
Q3	1	3	1	1
Oct.	1	10	1	4
Nov.	1	11	2	5
Dec.	1	12	3	6
Q4	1	4	1	2
HY2	1	2	2	1
Year	1	1	1	1

PlugAcct

Gets the plug account for the current Account member or for a specified account. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Syntax

```
HS.Account.PlugAcct ("Account")  
HS.Account.PlugAcct (" ")
```

where *Account* is the name of a valid Account dimension member.

Note: You can use a blank string (" ") to apply this function to the current member only if you are using it in a sub consolidate routine.

Return Value

A string that specifies the name of the plug account for the member.

Example

In this example, if the plug account for the Sales account is Plug1 then statements between the If...Then and End If statements are executed:

```
If HS.Account.PlugAcct ("Sales") = "Plug1" Then  
    ...  
End If
```

POwn

Gets the ultimate percentage of ownership for the current member or for a specified member. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Syntax

```
HS.Node.POwn ("S#Scenario.Y#Year.P#Period.E#Entity")  
HS.Node.POwn (" ")
```

Note: Use a blank string (" ") to apply this function to the current member.

Table 76 Syntax for POwn Function

Parameter	Description
<i>Scenario</i>	Name of a valid Scenario dimension member.
<i>Year</i>	A valid year.

Parameter	Description
<i>Period</i>	A valid period.
<i>Entity</i>	Name of a valid Entity dimension member.

Return Value

The percentage of ownership for the member.

Example

This example gets the percent ownership for the specified point of view:

```
POwn = HS.Node.POwn("S#Actual.Y#2009.P#January. E#Regional.Connecticut")
```

PVAForBalance

Determines the default translation method for BALANCE accounts (ASSET and LIABILITY accounts). This function overrides the application defaults for currency conversions during translation. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Syntax

```
HS.AppSettings.PVAForBalance
```

Return Value

A Boolean expression that is True if BALANCE accounts use the periodic value (PVA) translation method; False if BALANCE accounts use the value at exchange rate (VAL) translation method.

Example

In this example, if BALANCE accounts in the application use the periodic value translation method, then statements between the If...Then and End If statements are executed:

```
If HS.AppSettings.PVAForBalance = TRUE Then
    ...
End If
```

PVAForFlow

Determines the default translation method for FLOW accounts (REVENUE and EXPENSE accounts). This function overrides the application defaults for currency conversions during translation. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Syntax

```
HS.AppSettings.PVAForFlow
```

Return Value

A Boolean expression that is True if FLOW accounts use the periodic value (PVA) translation method; False if FLOW accounts use the value at exchange rate (VAL) translation method.

Example

In this example, if FLOW accounts in the application use the value at exchange rate translation method, then statements between the If...Then and End If statements are executed:

```
If HS.AppSettings.PVAForFlow = FALSE Then
    ...
End If
```

RateForBalance

Gets the default translation rate account to use for BALANCE accounts (ASSET and LIABILITY accounts). This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Syntax

```
HS.AppSettings.RateForBalance
```

Return Value

A String that specifies the rate account containing the default translation rate to use for BALANCE accounts.

Example

In this example, if the default translation rate account for BALANCE accounts is Rate1, then statements between the If...Then and End If statements are executed:

```
If HS.AppSettings.RateForBalance = "Rate1" Then
    ...
End If
```

RateForFlow

Gets the default translation rate account to use for FLOW accounts (REVENUE and EXPENSE accounts). This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Syntax

```
HS.AppSettings.RateForFlow
```

Return Value

A string that specifies the rate account containing the default translation rate to use for FLOW accounts.

Example

In this example, if the default translation rate account for FLOW accounts is Rate2, then statements between the If and End If statements are executed:

```
If HS.AppSettings.RateForFlow = "Rate2" Then
    ...
End If
```

ReviewStatus

Gets the review status for the specified point of view. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Syntax

```
HS.ReviewStatus("S#Scenario.Y#Year.P#Period.E#Entity.V#Value")
HS.ReviewStatus(" ")
```

Note: Use a blank string (" ") to apply this function to the current member.

Table 77 Syntax for ReviewStatus Function

Parameter	Description
Scenario	Name of a valid Scenario dimension member.
Year	A valid year
Period	A valid period.
Entity	Name of a valid Entity dimension member.
Value	Name of a Value dimension member.

Return Value

A string that specifies the review status for the member. Valid review statuses are as follows:

- Not Started
- First Pass
- Review Level 1-10
- Submitted
- Approved
- Published
- Not Supported

Example

In this example, if the review status of the specified point of view is Submitted then statements between the If...Then and End If statements are executed:

```
If HS.ReviewStatus("") = "Submitted" Then
    ...
End If
```

ReviewStatusUsingPhaseID

Gets the review status for the specified point of view using the process management submission phase ID.

Syntax

```
HS.ReviewStatusUsingPhaseID("S#Scenario.Y#Year.P#Period.E#Entity",n)
```

where *n* is an integer representing the process management submission phase. Valid values are 1–9.

Return Value

A string that specifies the review status for the member using the Submission Phase ID. Valid review statuses are as follows:

- Not Started
- First Pass
- Review Level 1-10
- Submitted
- Approved
- Published
- Not Supported

Example

```
HS.ReviewStatusUsingPhaseID("S#Actual.Y#2009.P#March.E#Connecticut",3)
```

Round

Rounds data from the Exp function. You specify the degree of rounding in the argument. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation

Tip: If you need to apply various degrees of rounding in a Calculation rule, you can include multiple statements that contain Round.

Syntax

```
HS.Round(Unit)
```

where *Unit* is a factor for rounding. Value of 1 rounds to the nearest whole number. Value of 0.1 rounds to the nearest tenth. Value of 0 turns off rounding.

If you specify 0 for this argument, rounding is turned off for all subsequent Exp functions in a Calculation rule. This syntax has the same effect as HS.NoRound: HS.Round(0)

Caution! The NumDecimalPlaces attribute of an account determines the maximum number of digits that can appear to the right of the decimal point. The Round function does not override this attribute.

Return Value

None.

Example

This example rounds the amount inserted into the SalesRound account to the nearest tenth, then uses NoRound to turn off rounding for the amount inserted into the SalesNoRound account's cells:

```
HS.Round(0.1)
HS.Exp"A#SalesRound = A#Sales"
HS.NoRound
HS.Exp"A#SalesNoRound = A#Sales"
```

Scale

Gets the scale of the specified currency. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Syntax

```
HS.Currency.Scale("Currency")
HS.Currency.Scale(Var1)
```

Table 78 Syntax for Scale Function

Parameter	Description
<i>Currency</i>	Name of a valid currency.
<i>Var1</i>	A VisualBasic variable.

Return Value

A number indicating the scale of the specified currency (0 to 9). Specifies the unit in which amounts are displayed and stored for the currency by identifying where the decimal point is placed. The return values are as follows:

- 0 = Units
- 1 = Tens
- 2 = Hundreds
- 3 = Thousands
- 4 = Ten Thousands
- 5 = Hundred Thousands
- 6 = Millions
- 7 = Ten Millions

- 8 = Hundred Millions
- 9 = Billions

Example

In this example, if the scale for French francs (FF) is 3, then statements between the If...Then and End If statements are executed:

```
If HS.Currency.Scale("FF") = 3 Then
    ...
End If
```

SecurityAsPartner

Gets the security class assigned to the specified entity when the entity is used as an intercompany partner. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Syntax

```
HS.Entity.SecurityAsPartner("Entity")
HS.Entity.SecurityAsPartner("")
HS.Entity.SecurityAsPartner(Var1)
```

Note: Use a blank string (" ") to apply this function to the current entity member.

Table 79 Syntax for SecurityAsPartner Function

Parameter	Description
<i>Entity</i>	Name of a valid Entity dimension member.
<i>Var1</i>	A VisualBasic variable.

Return Value

A string with the security class assigned to the entity when it is used as an ICP.

Example

In this example, if Class1 is the security class for France as it is used as an intercompany partner, then statements between the If...Then and If...End statements are executed:

```
If HS.Entity.SecurityAsPartner("France") = "Class1" Then
    ...
```

End If

SecurityClass

Gets the security class for the specified member. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Syntax

```
HS.<Object>.SecurityClass ("Element")
HS.<Object>.SecurityClass ("")
HS.<Object>.SecurityClass (Var1)
```

Note: Use a blank string ("") to apply this function to the current member.

Table 80 Syntax for SecurityClass Function

Parameter	Description
<Object>	One of these object keywords: ● Account ● Scenario ● Entity ● Custom1...4
Element	Depending on the object selected, name of a valid member of one of these dimensions: ● Account ● Scenario ● Entity ● Custom1...4
Var1	A VisualBasic variable.

Return Value

The name of the security class assigned to the specified member.

Example

In this example, if Class1 is the security class assigned to the Cash account, then statements between the If...Then and End If statements are executed:

```
If HS.Account.SecurityClass ("Cash") = "Class1" Then
...
End If
```

End If

SetData

Sets an individual record. This function can be used in these types of rules:

- Calculation
- Translation

Syntax

```
HS.SetData lView, lAccount, lICP, lCustom1, lCustom2, lCustom3, lCustom4, dData, bAddToExistingData
```

Table 81 Syntax for SetData Function

Parameter	Description
<i>lView</i>	0 = Scenario View 1 = Periodic 2 = YTD
<i>lAccount</i>	ID number of the account to which you are setting data.
<i>lICP</i>	ID number of the ICP to which you are setting data.
<i>lCustom1...4</i>	ID number of the Custom dimension to which you are setting data.
<i>dData</i>	The data value to set.
<i>bAddToExistingData</i>	True = To accumulate the data False = To replace the data

Return Value

None.

Example

```
HS.SetData 2, 002, , , , , 25000, TRUE
```

SetDataWithPOV

Inserts data into the node or currency cube. This function can be used in these following types of rules:

- Calculation
- Translation

Syntax

HS.SetDataWithPOV *POV*, *dData*, *bAddToExistingDataInCache*

Table 82 Syntax for SetData Function

Parameter	Description
<i>POV</i>	Valid POV
<i>dData</i>	The data value to set.
<i>bAddToExistingData</i>	True = To accumulate the data False = To replace the data

Return Value

None.

Example

```
HS.SetDataWithPOV "V#YTD.A#Asset.I#[ICP None].C1#None.C2#None.C3#None.C4#None",  
25000,TRUE
```

SubmissionGroup

Gets the process management submission group for a dimension member.

Syntax

HS.<*Dimension*>.SubmissionGroup(*Dimension member*)

where *Dimension* is one of these dimensions: Account, Custom1...4.

```
HS.Account.SubmissionGroup(Account)
```

```
HS.Custom1.SubmissionGroup(Custom1)
```

```
HS.Custom2.SubmissionGroup(Custom2)
```

```
HS.Custom3.SubmissionGroup(Custom3)
```

```
HS.Custom4.SubmissionGroup(Custom4)
```

Return Value

An integer representing the submission group for the dimension member. Valid values are 1–99.

Example

```
HS.Account.SubmissionGroup("Sales")
```

SupportsProcessManagement

Determines if a scenario supports process management. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Syntax

```
HS.Scenario.SupportsProcessManagement ("Scenario")  
HS.Scenario.EnableProcessManagement ("")
```

Note: Use a blank string ("") to apply this function to the current scenario.

```
HS.Scenario.SupportsProcessManagement (Var1)
```

Table 83 Syntax for SupportsProcessManagement Function

Parameter	Description
<i>Scenario</i>	A valid scenario.
<i>Var1</i>	VBScript variable representing a Scenario member.

Return Value

A Boolean that is True if the scenario has process management enabled; False otherwise.

Example

In this example, if process management is enabled for the actual scenario, statements between the If and End If statements are executed:

```
If HS.Scenario.SupportsProcessManagement ("Actual") = "TRUE" then  
...  
End IF
```

SupportsTran

Specifies the accounts in the application that support intercompany transactions. This function can be used in Transactions rules.

Syntax

```
HS.SupportsTran "POVExpression"
```

where *POVExpression* is a combination of Account, Custom1-4, Scenario, Entity, and year members.

Return Value

None.

Example

```
HS.SupportsTran "S#ActMon.A#RecltIC.C1#Closing"
```

SwitchSign

Determines if credits are switched to debits for the current Custom member or for a specified custom member. This function reverses the debit/credit sign using these rules:

- ASSET to LIABILITY
- LIABILITY to ASSET
- EXPENSE to REVENUE
- REVENUE to EXPENSE
- BALANCE to FLOW
- FLOW to BALANCE

This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Syntax

```
HS.<Object>.SwitchSign("Member")  
HS.<Object>.SwitchSign("")
```

Note: Use a blank string ("") to apply this function to the current member.

Table 84 Syntax for SwitchSign Function

Parameter	Description
<Object>	One of these object keywords: ● Custom1 ● Custom2 ● Custom3 ● Custom4
Member	Name of a valid Custom dimension member.

Return Value

A Boolean expression that is True if credits are switched to debits for the Custom member or False if credits and debits are not switched.

Example

In this example, if the credits and debits are switched, then statements between the If...Then and End If statements are executed:

```
If HS.Custom1.SwitchSign("") = TRUE Then
    ...
End If
```

SwitchType

Determines if the account types are switched for the current Custom member or for a specified custom member. This function changes the account type for the Custom dimension member using these rules:

- ASSET to EXPENSE
- EXPENSE to ASSET
- LIABILITY to REVENUE
- REVENUE to LIABILITY
- BALANCE to FLOW
- FLOW to BALANCE

This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Syntax

```
HS.<Object>.SwitchType("Member")
HS.<Object>.SwitchType(" ")
```

Note: Use a blank string (" ") to apply this function to the current member.

Table 85 Syntax for SwitchType Function

Parameter	Description
<Object>	One of these object keywords: ● Custom1 ● Custom2 ● Custom3 ● Custom4
Member	Name of a valid Custom dimension member.

Return Value

A Boolean expression that is True if account types are switched for the Custom member or False if account types are not switched.

Example

In this example, if the account types are switched for the current Custom1 member, then statements between the If...Then and End If statements are executed:

```
If HS.Custom1.SwitchType(" ") = "TRUE" Then
    ...
End If
```

Trans

Translates a currency using the year-to-date method. This function can be used in Translation rules.

Syntax

```
HS.Trans("DestPOV", "SourcePOV", "Rate1", "Rate2")
```

Table 86 Syntax for Trans Function

Parameter	Description
DestPOV	The destination point of view. The destination can be a combination of Account, Custom1...4, and ICP members. For each unspecified dimension, the system writes to all valid members of the dimension. For each specified dimension, the system writes into the specified member only.

Parameter	Description
<i>SourcePOV</i>	The source point of view. The source can be a combination of dimensions. If the Account, Custom1...4, and ICP dimensions are unspecified, the system uses the same member as the Destination member. If the Scenario, Year, Period, and Entity dimensions are not specified, the system uses the current members. If the Value dimension is not specified, the system uses the <EntityCurrTotal> member. If the source point of view is blank, the system uses the destination point of view as the source point of view.
<i>Rate1-2</i>	The exchange rate. The rate can be a constant, an exchange rate account, or a specific cell. Rate accounts are input for entity and for [None] entity. For information about the default translation process, see “Default Translation” on page 218 .

Return Value

None.

Example

This example uses the rate in the Rate1 account to translate the Sales account using the year to date method:

```
HS.Trans("A#Sales", "A#LastYearSales", "A#Rate1", "")
```

TransPeriodic

Translates a currency using the periodic method. This function can be used in Translation rules.

Syntax

```
HS.TransPeriodic("DestPOV", "SourcePOV", "Rate1", "Rate2")
```

Table 87 Syntax for TransPeriodic Function

Parameter	Description
<i>DestPOV</i>	The destination point of view. The destination can be a combination of Account, Custom1-4, and ICP members. For each not specified dimension, the system writes to all valid members of the dimension. For each specified dimension, the system writes into the specified member only.
<i>SourcePOV</i>	The source point of view. The source can be a combination of dimensions. If the Account, Custom1, Custom 2, Custom 3, Custom 4, and ICP dimensions are not specified, the system uses the same member as the Destination member. If the Scenario, Year, Period, and Entity dimensions are not specified, the system uses the current members. If the Value is not specified, the system uses the EntityCurrTotal member. If the source is blank, the system uses the destination as the source.
<i>Rate1-2</i>	The exchange rate. The exchange rate can be a constant, an exchange rate account, or a specific cell. Rate accounts are input for entity and for None entity. For information about the default translation process, see “Default Translation” on page 218 .

Return Value

None

Example

This example uses the exchange rate in the Rate1 account to translate the Sales account using the periodic method:

```
HS.TransPeriodic("A#Sales", "A#LastYearSales", "A#Rate1", "")
```

UD1...3

Gets the text stored in the UserDefined1...3 attribute for the current member or for a specified member. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Syntax

```
HS.<Object>.UD1...3(strElement)  
HS.<Object>.UD1...3("")
```

Note: Use a blank string ("") to apply this function to the current member.

Table 88 Syntax for UD1...3 Functions

Parameter	Description
<Object>	One of these object keywords: ● Account ● Entity ● Parent ● Scenario ● Custom1...4
Element	Depending on the object selected, name of a valid member of one of these dimensions: ● Account ● Entity ● Parent ● Scenario ● Custom1...4

Return Value

A String that contains the user-defined text stored for the member.

Example

In this example, if the user-defined text for the UD1 account is History, then statements between the If...Then and End If statements are executed.

```
If HS.Account.UD1(strAccount) = "History" Then
    ...
End If
```

ValidationAccount

Gets the validation account for an application. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Syntax

```
HS.AppSettings.ValidationAccount
```

Return Value

A String that specifies the name of the validation account for the application.

Example

If the validation account for the application is MyAccount, then statements between the If...Then and the End If statements are executed.

```
If HS.AppSettings.ValidationAccount = "MyAccount" Then
    ...
End If
```

ValidationAccountEx

Gets the validation account for a process management submission phase.

Syntax

```
HS.AppSettings.ValidationAccountEx(n)
```

where *n* is an integer representing the process management submission phase. Valid values are 1 to 9.

Return Value

A String that specifies the name of the validation account for the process management submission phase.

Example

This example returns the validation account defined for Submission Phase 5:

```
HS.AppSettings.ValidationAccountEx(5)
```

XBRLTags

Gets the XBRL tag assigned to the specified Account member. This function can be used in these types of rules:

- Calculation
- Translation
- Consolidation
- Allocation

Syntax

```
HS.Account.XBRLTags("Account")  
HS.Account.XBRLTags(" ")
```

Note: Use a blank string (" ") to apply this function to the current member.

Table 89 Syntax for XBRLTags Functions

Parameter	Description
Account	A valid account.

Return Value

A string that specifies the XBRL tag for the specified account.

In This Chapter

Calculation Manager Security Roles.....	327
Working with Applications in Calculation Manager	328
Migrating Rules to Calculation Manager	328
VB Function Support in Function Selector	329
Special VB Script Functions for Financial Management.....	330

The Calculation Manager module provides a common user interface to create calculation rules for Financial Management. The graphical flow provides a better understanding of the calculation process and enables you to switch between the graphical view and the VB Script view. Calculation Manager provides a central repository to maintain all calculation rules, and share rules among applications. You can import, export, and print calculation rules, and create custom folders for easy navigation.

Calculation Manager Security Roles

These roles are available for Calculation Manager access for Financial Management:

- Rules Administrator - can perform any tasks in Calculation Manager for the specified application, such as create, modify and delete rule objects, templates and variables, and validate and deploy any rule sets
- Rules Designer - can create rules objects and modify or delete those objects
- Rules Viewer - can view and validate rules objects

To access Calculation Manager from Performance Management Architect, you must have the Calculation Manager Administrator or Calculation Designer security role.

To access Calculation Manager from EPM Workspace, you must have the Rules Administrator, Rules Designer, or Rules Viewer security role.

To deploy rule sets, you must have the Calculation Manager Administrator, Calculation Designer, or Rules Administrator security role.

For more information on security roles, see the *Oracle Hyperion Enterprise Performance Management System User and Role Security Guide*.

Working with Applications in Calculation Manager

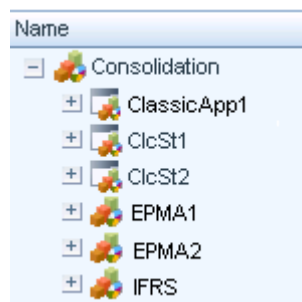
You can work with Calculation Manager in either Performance Management Architect applications or Classic Administration applications. You can install Calculation Manager with Performance Management Architect, or you can install it separately and access it from EPM Workspace.

When you create an application, you can load VB script rules, or use Calculation Manager to design and deploy rules to the Financial Management application.

Note: After you have deployed rules to Calculation Manager, when you attempt to load VB script rules, you will receive a prompt that the Calculation Manager rules will be overwritten. You can either click OK to continue or Cancel.

When you open the Consolidation folder in Calculation Manager, the system displays a list of your applications in alphabetical order. Classic Administration applications are identified with the following icon , and Oracle Hyperion EPM Architect, Fusion Edition applications use this icon: .

Figure 1 Sample Calculation Manager Application List



You can expand the application folder to view Rule Sets, Rules, Formulas, Scripts, and Templates. For information on using Calculation Manager, see the *Oracle Hyperion Calculation Manager Designer's Guide* or online help.

Migrating Rules to Calculation Manager

If you have existing VB Script rule (.xle) files from a previous release, you can migrate the files into Calculation Manager using the Rules Migration utility instead of converting them manually. The utility converts the VB Script file to graphical rules objects in XML format, which you can then load into Calculation Manager. The Rules Migration utility, `FMRulesMigrator.exe`, is installed by default in the `Financial Management\Consultant Utilities` folder.

VB Function Support in Function Selector

The Calculation Manager Function Selector supports these VB functions.

Note: You can also use other VB functions in the script component, even though they are not available in the UI for selection.

Array Functions

Function	Description
Array	Returns a variant containing an array
Filter	Returns a zero-based array that contains a subset of a string array based on a filter criteria
Join	Returns a string that consists of a number of substrings in an array
LBound	Returns the smallest subscript for the indicated dimension of an array
Split	Returns a zero-based, one-dimensional array that contains a specified number of substrings
UBound	Returns the largest subscript for the indicated dimension of an array

Date Time Functions

Function	Description
Date	Returns the current system date
DateAdd	Returns a date to which a specified time interval has been added
DateDiff	Returns the number of intervals between two dates
DatePart	Returns the specified part of a given date
DateSerial	Returns the date for a specified year, month, and day
Day	Returns a number that represents the day of the month (between 1 and 31, inclusive)
Month	Returns a number that represents the month of the year (between 1 and 12, inclusive)
MonthName	Returns the name of a specified month

Mathematical

Function	Description
Abs	Returns the absolute value of a specified number
Fix	Returns the integer part of a specified number
Int	Returns the integer part of a specified number

String

Function	Description
InStr	Returns the position of the first occurrence of one string within another. The search begins at the first character of the string.
InStrRev	Returns the position of the first occurrence of one string within another. The search begins at the last character of the string
LCase	Converts a specified string to lowercase
Left	Returns a specified number of characters from the left side of a string
Len	Returns the number of characters in a string
Mid	Returns a specified number of characters from a string
Right	Returns a specified number of characters from the right side of a string
StrComp	Compares two strings and returns a value that represents the result of the comparison
Trim	Removes spaces on both the left and right side of a string
UCase	Converts a specified string to uppercase

Special VB Script Functions for Financial Management

These special functions were implemented for Financial Management to address array and loop in the Hyperion Calculation Manager UI.

Range

Value	Loop Variable	VBScript Generation
@Range(1-50)	i	Dim i(50) I(1)=1 I(2)=2 I(3)=3 .. I(50)=50
@Range(5-10)	i	Dim i(6) I(1)=5 I(2)=6 I(3)=7 I(4)=8 I(5)=9 I(6)=10

Value	Loop Variable	VBScript Generation
@Range(1,3-5,7-9)	i	Dim i(8) I(1)=1 I(2)=3 I(3)=4 I(4)=5 I(5)=7 I(6)=8 I(7)=9

For / ForStep

Value	Loop Variable	VBScript Generation	Comments
@For(2,10)	Item	For Item=2 to 10	New @ForLoop @For(from, to)
@ForStep(2,10,2)	Item	For Item =2 to 10 step 2	New @ForStep loop function @ForStep(from, to, step) Note: If you need a reverse step, add a negative sign (-) in front of the step, for example: @ForStep(2,10,-2)

ExitFor

@ExitFor - exiting the loop

The system generates these VB script statements:

For each *element* in group

[*statements*]

Exit For

[*statements*]

Next [*element*]

Or

For *counter*=start To end [Step *step*]

[*statements*]

Exit For

[*statements*]

Next [*counter*]

ExitSub

@ExitSub - exiting the rule

The system generates these VB script statements:

```
Sub name [(x,y)]
```

```
[statements]
```

```
ExitSub
```

```
[statements]
```

```
EndSub
```

ReDim

Redimensions one or more dynamic array variables and reallocates their storage space. The optional Preserve keyword can be used to keep the contents of the array intact when it is being redimensioned.

```
{VarArrayX(5)} = @Redim
```

```
{VarArrayXY(5,9)} = @Redim
```

The system generates these VB statements:

```
Redim VarArrayX(5)
```

```
Redim VarArrayXY(5,9))
```

RedimPreserve

```
{VarArrayXY(5)} = @RedimPreserve
```

Or

```
{VarArrayXY(5,9)} = @RedimPreserve
```

Or

```
{VarArrayXY(5,{i})} = @RedimPreserve
```

The system generates this VB statement:

```
RedimPreserve VarArrayX(5)
```

Or

```
RedimPreserve VarArrayXY(5,9)
```

Or

```
RedimPreserve VarArrayXY(5,i)
```

In This Chapter

Managing Intercompany Transaction Periods	333
Locking and Unlocking Entities	336
Managing Reason Codes.....	337
Monitoring Intercompany Transactions	338

Managing Intercompany Transaction Periods

Intercompany transactions are transactions carried out between two units of the same corporation. Before you can enter, load, or process intercompany transactions, you must first open the period for the transactions. After you have finished processing transactions, you can close the period to prevent further input.

See these procedures:

- [“Opening Periods” on page 333](#)
- [“Setting Matching Tolerances” on page 334](#)
- [“Setting Match/Validate Before Post Option” on page 335](#)
- [“Closing Periods” on page 335](#)

Opening Periods

You open a period by specifying the scenario, year, and period. A period can have a status of Unopened, Opened, or Closed. The default status for periods is Unopened. After a period is opened and a transaction has been entered, it can only be changed to Closed. It cannot go back to Unopened.

For each period, you can set the Match/Validate Before Post option and enter matching tolerances that apply to the Auto-Match and Manual Match processes. See [“Setting Match/Validate Before Post Option” on page 335](#) and [“Setting Matching Tolerances” on page 334](#).

➤ To open periods:

- 1 In the **Browser View**, expand **Tasks** and select **IC Transaction Tasks**.
- 2 Select **Manage IC Periods**.
- 3 From **Scenario**, select a scenario for the period.

- 4 From **Year**, select a year for the period.
- 5 Select the check box next to each period to open.
- 6 **Optional:** In the **Matching Tolerance** column, enter a TID Tolerance amount or percentage, Account Tolerance amount, or Manual Matching Tolerance amount for the period.
- 7 In the **Match/Validate Before Post** column, select an option:
 - If you require the system to check the match status before posting transactions, select **Yes or Restricted**.
 - If you do not require the system to check the match status, select **No**.
- 8 Click **Open Period**.
- 9 **Optional:** To save the settings for the period, click **Save Period Settings**.

Setting Matching Tolerances

You can set matching tolerances by period for the Auto-Match and Manual Match processes. You can set amounts for the Account and Manual Matching tolerance. For Transaction ID (TID) tolerance, you can set an amount, a percentage, or both.

If you enter a percentage for Transaction ID, the system uses the smaller amount between the total of the entity's transaction and the total of the partner's transaction and applies the percentage to the amount, resulting in the tolerance amount.

For example, suppose you have three transactions from Entity A with TID 123, and the total of these transactions is 1000. Partner B with TID 123 has five transactions with a total of 1020. The difference between the entity total and the partner total is 20. However, if you enter a tolerance of 3%, the system calculates 3% of the smaller total, which is 1000 times 3%, resulting in 30. If you compare that to the difference, it would be within the tolerance and the transactions would be considered matched.

If in addition to the percentage, you enter an amount, for example, 15, the system compares the percentage amount with the amount entered and uses the smallest amount as comparison. In this example, the difference between the entity total and the partner total is 20 and the percentage tolerance is 30, but the amount tolerance is 15. This would not be considered within the tolerance and the transactions are not matched.

You can also leave the TID amount and percentage blank. If either has a zero value, or if both are blank, the system matches only transactions that have zero transaction difference.

For Account tolerance and Manual Match tolerance, the matching tolerance is represented in the application currency and the amount entered is represented in the scale factor of the application currency. During the matching process, the system converts each transaction into the application currency and compares the total difference amount to the matching tolerance set for the period. The comparison is done in units.

During the TID/RID matching process, the system does not translate the transaction to the application currency when comparing it to the TID tolerance value.

Setting Match/Validate Before Post Option

For each period, you can set the Match/Validate Before Post option. The Match/Validate option defines whether the system needs to check the match status of the transactions before the transactions can be posted and defines the types of validation that need to be done before an entity can be locked or a period can be closed.

The Match/Validate Before Post option can be set to Yes, No, or Restricted.

Table 90 Match/Validate Before Post Option Value Descriptions

Match/ Validate Option Value	Description
No	All transactions can be posted
Yes	The system allows transactions to be posted if either of these conditions are met: • Transactions have the Matched status. • Transactions with a MisMatched status contain a valid Reason Code. Note: Unmatched transactions or mismatched transactions without a reason code cannot be posted. The system also checks that all matched transactions or mismatched transactions with reason codes are posted before the period can be closed or the entity can be locked
Restricted	If the Match/Validate Before Posted option is set to Restricted, the system allows transactions to be posted if either of these conditions are met: • Transactions have the Matched status. • Transactions with a MisMatched status contain a valid Reason Code. Note: Unmatched transactions or mismatched transactions without a reason code cannot be posted. The system does not require that all matched transactions or mismatched transactions with reason code be posted before the period can be closed or the entity can be locked.

Closing Periods

At the end of the closing cycle when you have finished processing intercompany transactions, you can close the period to prevent additional modifications to the transactions. If the Match/Validate Before Post option is set to Yes for the period, all matched transactions or mismatched transactions with a reason code must be posted before a period can be closed. If Match/Validate Before Post is set to No or Restricted, the system does not check before closing the period.

When you close a period, it is locked for future transactions, however you can continue to view transactions or run reports on the period's transactions.

➤ To close a period:

- 1 In the **Browser View**, expand **Tasks** and select **IC Transaction Tasks**.
- 2 Select **Manage IC Periods**.

- 3 From the **Scenario** list, select a scenario for which to close periods.
- 4 From the **Year** list, select a year in which to close periods.
- 5 Select the check box next to each period to close.
- 6 Click **Close Period**.

Locking and Unlocking Entities

You can apply a lock to an entity for a scenario, year, and period to prevent future changes to intercompany transactions for the entity. If the Match/Validate Before Post option is set to Yes for the period, all matched transactions or mismatched transactions with a reason code must be posted before the entity can be locked. If the Match/Validate Before Post option is set to No or Restricted, the system does not check before locking the period.

Note: The transaction lock status is different from the data lock status. For information on data lock status, see the *Oracle Hyperion Financial Management User's Guide*.

When the entity is locked, you are not allowed to enter any new intercompany transactions. You are also not allowed to delete or make any changes to existing transactions for the entity. You cannot post or unpost any transactions to a locked entity, or update the match status of the entity's transaction. Therefore, even if the partner entity is not locked, the partner cannot match their transactions with the entity's transactions because the match status cannot be updated for the entity.

For example, suppose Entity A is locked. You cannot enter any more transactions for Entity A, and no posting or matching can be done to the entity. You can still have intercompany transactions for Entity B with its partner Entity A if Entity B is not locked. However, if you try to match Entity B with Entity A, the process fails because the system cannot update the match status for Entity A.

If you are using submission phases, an entity should not be locked until all phases have Published status.

► To lock an entity:

- 1 In the **Browser View**, expand **Tasks** and select **IC Transaction Tasks**.
- 2 Select **Lock/Unlock IC Entities**.
- 3 From **Scenario**, select a scenario for the entity.
- 4 From **Year**, select a year for the entity.
- 5 From **Period**, select a period for the entity.
- 6 Select the check box next to each entity to lock.
- 7 Click **Lock**.

Tip: To unlock entities, select the check box next to each entity to unlock, then click **Unlock**.

Managing Reason Codes

When intercompany transactions are created in the application, they have a default match status of UnMatched. During the Auto-Match process, the match status is updated to Matched or MisMatched.

You can define reason codes to indicate why a transaction has a MisMatched status. For example, this might be due to a missing invoice from the partner entity, or an incorrect amount entered by the partner. After you define the list of valid reason codes for the application, users can select from the list and assign one of the codes when they enter intercompany transactions.

If the Match/Validate Before Post option is selected for the period, you can post transactions with a Matched status, or transactions with a MisMatched status that contain a valid reason code.

You can add, edit, or delete reason codes for an application.

See these procedures:

- [“Adding Reason Codes” on page 337](#)
- [“Editing Reason Codes” on page 338](#)
- [“Removing Reason Codes” on page 338](#)

Adding Reason Codes

You can create a list of reason codes for mismatched transactions for an application. Users can then assign one of the reason codes for transactions.

You can manually add reason codes or you can load reason codes during the transaction load process. For information on loading transactions, see the *Oracle Hyperion Financial Management User's Guide*.

► To add reason codes:

- 1 In the **Browser View**, expand **Tasks** and select **IC Transaction Tasks**.
- 2 Select **Manage IC Reason Codes**.
- 3 For **New Reason Code**, enter a label for the reason code.

Note: The label can contain a maximum of 20 characters. Note that a space is counted as a character.

- 4 For **Description**, enter a description for the reason code.

Note: The description can contain a maximum of 40 characters.

- 5 Click **Add**.

Editing Reason Codes

After you create a reason code, you can edit the code description.

► To edit reason codes:

- 1 In the **Browser View**, expand **Tasks** and select **IC Transaction Tasks**.
- 2 Select **Manage IC Reason Codes**.
- 3 From the list of reason codes, select the reason code to edit.
- 4 Click **Edit**.
- 5 From the **Description** column, edit the description, and click **OK**.

Removing Reason Codes

You can remove reason codes that you no longer need in the list of reason codes for an application.

► To remove reason codes:

- 1 In the **Browser View**, expand **Tasks** and select **IC Transaction Tasks**.
- 2 Select **Manage IC Reason Codes**.
- 3 From the list of reason codes, select the reason code to remove.
- 4 Click **Remove**.
- 5 Click **Yes** to remove the reason code.

Monitoring Intercompany Transactions

You can use the Monitor Intercompany Transactions feature to monitor the intercompany transaction matching process. When a large number of intercompany transactions are entered to the system in a period, the matching process can be time-consuming to ensure that all transactions are entered and matched successfully. Since not all transactions are entered at the same time, administrators need to monitor the matching process. The Monitor Intercompany Transactions feature enables you to easily find out which intercompany partners have started their intercompany transactions process.

The Intercompany Transactions Monitor window displays a list of intercompany entities with their Process status and Lock status. The entities are links to Intercompany Transactions Monitor Detail information. When you click an entity, the system opens a pop-up window that displays the number of posted and unposted transactions by status, such as Matched, Mismatched, or Unmatched. See [“Viewing the Intercompany Transactions Summary” on page 340](#).

The values in the Intercompany Transactions Monitor Detail window link to the Process IC Transactions window with the filtering to display the detail transactions from the values.

For example, if you click Entity A from the list of entities, the Intercompany Transactions Monitor Detail pop-up window displays the number of posted and unposted transactions by status for Entity A. If you click the value for unposted transactions in the Unmatched column, the system links to the Process IC Transactions window, with the filtering selected to display the unposted transactions with an Unmatched status for Entity A.

You can send email alerts for any of the entities in the Intercompany Transactions Monitor window. See the *Oracle Hyperion Financial Management User's Guide*.

Note: To monitor intercompany transactions, you must be assigned the Intercompany Transaction Administrator security role.

You can monitor the status of intercompany transactions between entities and intercompany partners. The system displays the process status and lock status of the intercompany transactions for the selected entities, and you can filter and sort the list by status.

► To monitor intercompany transactions:

- 1 In the **Browser View**, expand **Tasks** and select **IC Transaction Tasks**.
- 2 Select **Monitor IC Transactions**.
- 3 In the Point of View bar, select a scenario, year, and period.
- 4 For **Entity**, enter or browse for the entity for which to monitor intercompany transaction status.

Note: If you leave Entity blank, the system returns all entities in the list.

- 5 From the **Display** drop-down list, select an option:
 - To view the entity information using the label, select **Label**.
 - To view the entity information using the description, select **Description**.
 - To view the entity information using the label and description, select **Both**.
- 6 **Optional:** To filter the transactions list by process or lock status, select one or more of these transaction types, then click **Update** to refresh the list:
 - Not Started
 - Started
 - Not Lockable
 - Lockable
 - Locked
- 7 **Optional:** To sort the transactions list by **Process** status, click **Sort** to sort by **Ascending** or **Descending**.
- 8 **Optional:** To sort the transactions list by **Lock** status, click **Sort** to sort by **Ascending** or **Descending**.
- 9 Click an entity to link to the Intercompany Transactions Monitor Detail information.
- 10 From the **Intercompany Transactions Monitor Detail** window, click a value from one of the status columns.

The Process IC Transactions window is displayed with the filtering selected.

- 11 When you finish viewing monitor detail, from the **Intercompany Transactions Monitor Detail** window, click **Close Window**.

Viewing the Intercompany Transactions Summary

You can view a summary of the status of the intercompany transactions for a specific entity based on the selections displayed in the Monitor Intercompany Transactions window.

- To view the intercompany transactions summary:
 - 1 From the **Monitor Intercompany Transactions** window, select the entity for which to view a summary of transactions.
 - 2 When you finish viewing the summary, click **Close Window**.

In This Chapter

Defining Submission Phases by Scenario and Period	341
Submission Group Assignments by Scenario and Period	343
Assigning Submission Groups to Phases	345
Viewing Unassigned Submission Groups	346

You can use submission phases for Process Management, which enable you to subdivide a process unit into different phases and apply process management to those phases. During the review process, you can promote each phase of the process unit rather than the entire process unit. This eliminates the need for additional scenarios to enforce the review process.

You can use a maximum of nine submission phases in the review process. Submission phases can vary by period and scenario. For example, in the Actual scenario, you might submit Balance Sheet and Profit/Loss accounts for review in the first submission phase, and supplemental data in the second submission phase. In the Budget scenario, you might submit Intercompany data in the first submission phase, Balance Sheet and Profit/Loss accounts in the second phase, and supplemental data in the third phase.

You define and set up submission phases in metadata files. See the *Oracle Hyperion Enterprise Performance Management Architect Administrator's Guide*.

Defining Submission Phases by Scenario and Period

Your review process requirements may vary from period to period. For example, the monthly close cycle might require a single-phase review process for Balance Sheet and Profit/Loss data in January and February. For a quarterly month such as March, the quarterly closing review process may require multiple phased submission cycles for supplemental data and Balance Sheet and Profit/Loss data.

Phase definition can also differ by scenario. For example, the Actual scenario might require only Balance Sheet and Profit/Loss accounts to be submitted for review. For the Budget scenario, all accounts might be required, and for the Forecast scenario, only Profit/Loss accounts and supplemental data might be required.

To set up submission phases, administrators must complete these procedures:

- In the metadata file, set the application and dimension metadata attributes to use submission phases. Set the UseSubmissionPhase application attribute to Y, and set SupportSubmissionPhaseForAccounts, SupportSubmissionPhaseForCustom, or SupportSubmissionPhaseForICP attributes as required. For example, if the application only needs submission by accounts and not for Custom 1 to 4 or Intercompany (ICP) dimensions, you can select the SupportSubmissionPhaseForAccounts attribute. At least one dimension must be enabled.
- Define submission groups and assign submission groups to dimensional members (accounts, Custom 1 to 4 dimensions, ICP members)
- Assign submission groups to submission phases
- Assign validation accounts to submission groups. Validation accounts are used to ensure that the value equals zero before a process unit can be promoted to the next review level. Assign Submission Group 0 to validation accounts that are not subject to process management.

When multiple dimensions are used for phased submissions, the system determines the cell submission group assignment by the maximum of the group assignments of its dimension members. You should consider all of the group assignments that you need before assigning submission groups.

Example 1:

Account=2

C1=1

C2=2

C3=1

C4=1

ICP=1

The submission group value for the cell is 2 because the maximum submission group number for these dimensions is 2.

Example 2:

Account=1

C1=3

C2=2

C3=5

C4=1

ICP=3

The submission group value for the cell is 5 because the maximum submission group number for these dimensions is 5.

Submission Group Assignments by Scenario and Period

After you set application and dimension member attributes for phases, and define submission groups for dimension members, you can assign submission groups for accounts to each of the submission phases. The assignment applies only to the scenario that supports Process Management. However, the assignment must be done by Scenario and Period.

If a group is not specified, process management is not applied.

These examples show sample submission groups and their assignments to submission phases.

Accounts	Submission Group
HistData	0
Cash	1
Invest	1
ICRec	2
ICPay	2
Liability	3
Equity	3
Revenue	4
Expense	4
SuppData1	5
SuppData2	5
Headcount	6
MiscData	6

C1 (Product)	Submission Group
[None]	1
Golf Balls	7
Tennis Balls	8
Soccer Balls	9

Base accounts do not inherit submission groups from parent accounts, and a parent account does not assume any submission group from its children. You must assign a submission group to each account. If you leave the submission group blank, it defaults to Submission Group 1. In this example, the HistData account has a submission group assignment of 0, which means that the account does not require review process.

This example shows submission group assignments by period for the Actual scenario.

Table 91 Example: Submission Group Assignment to Submission Phases

Period	Submission Phase 1	Submission Phase 2	Submission Phase 3
January (Single Phase)	1, 2, 3, 4	N/A (No supplemental or product data required)	N/A (No supplemental or product data required)
February	1, 2, 3, 4	N/A	N/A
March (Multiple Phases)	2	1, 3, 4, 7, 8, 9	5, 6
April	1, 2, 3, 4	N/A	N/A
May	1, 2, 3, 4	N/A	N/A
June (Multiple Phases)	2	1, 3, 4, 7, 8, 9	5, 6
July	1, 2, 3, 4	N/A	N/A
August	1, 2, 3, 4	N/A	N/A
September (Multiple Phases)	2	1, 3, 4, 7, 8, 9	5, 6
October	1, 2, 3, 4	N/A	N/A
November	1, 2, 3, 4	N/A	N/A
December (Multiple Phases)	2	1, 3, 4, 7, 8, 9	5, 6

January — Single Phase Assignment

For the January monthly close, in this example, Process Management is required for Submission Phase 1 but not required for Submission Phases 2 and 3. Since this is a short monthly close cycle, a Intercompany, Balance Sheet, and Profit/Loss data (Groups 1, 2, 3, and 4) is submitted in the same submission phase. Supplemental data is not required.

March — Multiple Phase Assignment

During the March quarterly close, Process Management utilizes data submission across multiple phases.

Submission Phase 1 in this example requires intercompany data, so includes data for ICRec and ICPay accounts; in this example, all accounts with a submission group assignment of 2.

Submission Phase 2 for March contains Balance Sheet and Profit/Loss accounts (Cash, Invest, Liability, Equity, Revenue, and Expense accounts); in this example, all accounts with a submission group assignment of 1, 3, or 4.

Submission Phase 3 for March includes supplemental data, and contains supplemental data accounts such as SuppData1, SuppData2, Headcount, and MiscData; in this example, all accounts with a submission group assignment of 5 and 6.

Assigning Submission Groups to Phases

You display and manage submission phases using the Manage Submission Phases task. This task only displays if you enable the UseSubmissionPhase application setting in the metadata file. You must also be assigned to the administrator or Review Supervisor security role.

In the Manage Submission Phases window, you can specify the submission phases to display, from 1 to 9. You can change the setting as needed to display different submission phases.

You assign submission groups to phases by Scenario and Period. You can enter one or more groups for a submission phase and use a comma as a separator for multiple groups assignment (for example, 1, 5, 6, 8, 9). You can specify a range of groups. For example, to assign groups 1, 2, 3, 4, 5, 7 and 8 to a submission phase, you can specify 1–5, 7–8. If you enter one or more groups in a range using commas, when you submit and refresh the data, the system displays the groups that are in a range (for example, 1, 2, 3, 4 displays as 1–4).

Valid groups are 1 to 99. The default for Submission Phase 1 is the keyword ALL to indicate all groups. All groups belong to Submission Phase 1 until you change their assignment.

You cannot assign the same group to multiple phases in the same period. For example, you cannot specify Groups 2 through 5 for Phase 1, and Groups 3 and 8 for Phase 2, because Group 3 cannot be assigned to both Phase 1 and Phase 2. A submission group can only be assigned to one phase in the same period. An error message displays if you try to assign a submission group to a phase with one already assigned. If you move all groups out of a submission phase into a different submission phase, the original phase is cleared entirely from the system for the specified scenario and period.

You can skip a submission phase assignment. For example, you can assign groups in Phase 1 and 3 but not in Phase 2. Any groups that are not assigned to a submission phase are not considered part of the review process. Those dimensional members are available to all users with the appropriate security class access without the need to check for review level security. Unassigned cells do not need to be started for process management before you can enter data.

► To assign submission groups for submission phases:

- 1 In the Browser view, select **Tasks**, then **Data Tasks**, and then **Manage Submission Phases**.
- 2 Select the phases to display, and click **OK**.
- 3 To change the scenario, click on it, select a scenario, and click **OK**.
- 4 In each phase column, enter the groups for that submission phase and click **Enter**
 - To enter multiple groups, use a comma as a separator.
 - To specify a range of groups, use a dash as a separator.
 - To indicate all groups, specify ALL.
- 5 Click **Submit Data** to save the data, or click **Reset Data** to refresh the database.

Viewing Unassigned Submission Groups

In Process Control, you can view submission groups that are not assigned to any phase. Submission groups that are not assigned a submission phase are not part of the review process. Viewing unassigned groups enables you to check if you have accidentally omitted groups from the review process.

The system displays groups assigned to dimension members that are not assigned to any submission phase, and groups assigned to a submission phase that are not assigned to any dimension member. If a group has not been assigned to either a dimension member or phase, it is not displayed.

For example, an application has submission groups 1–10 assigned:

Period	Submission Phase 1	Submission Phase 2	Submission Phase 3
January	1,2,3,4	5,7	8,9
February	1,2,3,4	5-8	N/A
March	2	1,3,4	5,6
April	1,2,3,4	5,6	8
May	1,2,3,4	5-8	N/A
June	2	1,3,4	5,6
July	1,2,3,4	N/A	N/A
August	1,2,3,4	N/A	N/A
September	2	1,3,4	5,6
October	1,2,3,4	N/A	N/A
November	1,2,3,4	NA	N/A
December	2	1,3,4	5,6

If you select the option to display unassigned groups, for January in the previous example, these groups are shown as unassigned:

Period	Unassigned Group
January	6,10

► To view unassigned submission groups:

- 1 In the Browser View, select **Tasks**, then **Data Tasks**, and then **Manage Submission Phases**.
- 2 Select the cell for the phase, right-click and select **Show Unassigned Groups**.
- 3 From the popup list, review the list of unassigned groups if applicable, and then click **OK**.

In This Chapter

Setting Up Process Management Alerting	347
Setting Up Intercompany Transaction Alerting	348

You can use e-mail alerting for intercompany transactions and during the process management review process. E-mail alerts help highlight a key event or data change in the system. For example, you can send an e-mail alert that an intercompany transaction is mismatched and needs to be matched, or that a process unit is ready for the next promotion level.

E-mail alerts are sent using standard Simple Mail Transfer Protocol (SMTP), so you can use alerts with any e-mail system that works with Internet e-mail. To use alerts, you must specify the SMTP server name when you run the Financial Management configuration utility. See the *Oracle Hyperion Enterprise Performance Management System Installation and Configuration Guide*.

The alert process uses the e-mail addresses that are stored in your authentication files, such as LDAP, MSAD, or Native Directory.

Before you can send or receive e-mail alerts, you must have set up user and data security rights in the application. The security class assigned to the scenario and entity for the alert must support e-mail alerts, and users must be assigned a security role to receive e-mail alerts.

Setting Up Process Management Alerting

You can set up the Process Control module to trigger e-mail alerts based on a change of status in process control. You can set up alerts for these actions: First Pass, Review levels 1 through 10, Submitted, Approved, or Published.

E-mail alerts are not generated when the process unit is at the Not Started level or for the Sign Off action.

For information on setting security roles, see the *Oracle Hyperion Enterprise Performance Management System User and Role Security Guide*.

► To set up process management e-mail alerts:

- 1 For the scenario in the process unit, set the `SupportsProcessManagement` metadata attribute to “A” to allow alerts.

Note: When you enable this attribute, the scenario generates e-mail alerts during the review process for the users that have the security rights to receive them.

- 2 Assign the user to the Receive E-mail Alerts for Process Management role.
- 3 Assign the user ALL or PROMOTE access to the security classes assigned to the scenario and entity in the process unit and add an alert for each security class.

Users who meet all criteria receive e-mail alerts.

Table 92 Process Management User Roles and Alert Notification

Process Unit Level Before or After Action	Process Management User Roles Notified
First Pass	Users with ALL or PROMOTE access to the entity are notified.
Review Levels 1 through 10	Reviewer at that Review Level and Submitter roles are notified. For example, for Review Level 1, Reviewer 1 and Submitter roles are notified.
Submitted	Review Supervisor role is notified. Only users with this role can approve the submitted process unit.
Approved	Reviewer 1 to Reviewer 10 and Submitter roles are notified.
Published	Users with ALL, READ, or PROMOTE access to the entity are notified.

When a process review action is performed, the system automatically generates e-mail alerts to the appropriate users, according to the security rights that have been set up. The user that performed the action is also notified with a confirmation e-mail.

Note: Users with the Application Administrator role do not receive e-mail alerts. For a user with the Application Administrator role to receive e-mail alerts, set up the administrator as a separate user and assign the role to receive alerts

Setting Up Intercompany Transaction Alerting

You can generate intercompany transaction e-mail alerts for users with the security rights to receive them.

For information on setting security roles, see the *Oracle Hyperion Enterprise Performance Management System User and Role Security Guide*.

► To set up intercompany transaction e-mail alerts:

- 1 Set the security class attribute of SupportAlert=Y for the scenario and entity for the alert.
- 2 Assign the user to the Receive E-mail Alerts for IC Transactions role.
- 3 Assign the user to the Inter-Company Transaction Admin or Inter-Company Transaction User role.
- 4 Assign the user ALL, READ, or PROMOTE access to the security classes that are assigned to the scenario and entity in the transaction and add an alert for each security class.

Users who meet all criteria receive e-mail alerts from the Intercompany Transactions or Intercompany Partner Matching Report modules.

For information on generating e-mail alerts in intercompany transactions, see the *Oracle Hyperion Financial Management User's Guide*.



Accessibility

In This Appendix

Enabling Accessibility for Financial Management.....	351
Using Keyboard Equivalents	351
Shortcut Keys by Module.....	354

Enabling Accessibility for Financial Management

Financial Management accessibility is enabled when you enable the screen reader support for EPM Workspace preferences and then open a Financial Management application.

Note: If the application is already open before enabling the screen reader support for EPM Workspace, you must close it and open it in a new browser window.

If you are using JAWS® Screen Reading Software, Oracle recommends using the Internet Explorer browser. For the JAWS screen reader to announce links and image shortcut keys, Virtual PC Cursor (VPC) mode must be enabled.

➤ To enable accessibility:

- 1 From Oracle Enterprise Performance Management Workspace, Fusion Edition, select **File**, and then **Preferences**.
- 2 From Accessibility Mode, select **Enable Screen Reader Support**.
- 3 Click **OK**.

Using Keyboard Equivalents

These sections describe keyboard shortcuts in Financial Management applications.

File Menu

Use these keyboard shortcuts from the File menu. Select File (Alt + f), then New (n), then an option.

Table 93 File New Menu Keyboard Shortcuts

Action	Key
Data grid	g
Data form	f
Link	l
Related Content	r
Tasklist	k

For new System Reports, use these keyboard shortcuts from the File menu. Select File (Alt + f), then New (n), then System Report (s), then an option.

Table 94 System Report Menu Keyboard Shortcuts

Action	Key
Journal	j
Intercompany	i
Explore Data	d
Intercompany Matching template	t
Intercompany transactions	n
Intercompany Matching by Account	c
Intercompany Matching by Transaction ID	m

View Menu

Use these keyboard shortcuts from the View menu. Select View (Alt + v), then an option.

Table 95 View Menu Keyboard Shortcuts

Action	Key
Advanced Mode	m

Administration Menu

These keyboard shortcuts are used in the Administration menu. Select Administration (Alt + a), then an option.

Table 96 Consolidation Application Administration

Action	Key
Manage Documents	d
Manage Favorites	f
System Messages	s
Users on System	u
Manage Servers and Applications	r
Task Audit	k
Data Audit	t
Running Tasks	i
Extended Analytics	x
Copy Document Link	c
Manage Taskflows	l
View Taskflow Status	v
Manage Smart View Providers	w

Navigating to the Application

Table 97 Application Navigation Shortcuts

Action	Key
iFrame- Main content area	Alt+z or Ctrl+F6, Tab
iFrame - Popups (when iFrame receives focus)	Alt+q
Close popups	Alt+L
Access contents of the selected tab page	Alt+y

Financial Management Preferences

Tab Indexes are provided for Groupboxes, TextBoxes, and Buttons.

Note: The JAWS screen reader can announce the number of items in a group box if you click Alt and the Up arrow key.

Abbreviations such as “POV” for Point of View are used across Financial Management screens. Screen readers will read them as “POV.” To add abbreviation equivalents in the Screen Reader dictionary, such as replacing “POV” with “Point of View,” refer to your screen reader documentation.

Shortcut Keys by Module

Use these shortcut keys for Financial Management modules.

To use shortcut keys, select Alt, then the keyboard shortcut, and then press Enter. For example, to view logged-in users, select Alt+i, then Enter.

For multiple selections, you can use Shift+F8. When this mode is enabled, you can move up or down the list of choices and press the spacebar to select multiple items, or press the spacebar again to deselect an item.

You can use Alt+Z as a shortcut key to go to the beginning of any page. You can also use Alt+Z to navigate to a floating frame when loading a page. In certain cases, when a page is refreshed as a result of an action, the focus goes to the top of the Financial Management content area. You can navigate using Tab or shortcut keys.

All Oracle Hyperion Financial Management, Fusion Edition reports that use the HFM_format style can be accessed using JAWS table reading commands.

Note: Accessibility is supported on Microsoft Internet Explorer only.

Table 98 Administrative System Tasks

Module	Action	Keyboard Shortcut
Users on System	View logged-in users	i
	Log out users	g
	Log out on simple grid. Single key for all rows	o
	Previous page	\
	Next page	/
Manage Servers and Applications	Log out users	g
	Enable connections	e
	Disable connections	s
	View disabled components	i
	Enable (common for all rows)	n

Module	Action	Keyboard Shortcut
System Messages	View system messages	i
	Calendar start	[
	Calendar end	]
	Show Details	s
	Delete system messages	e
	Delete All system messages	l
	Select All system messages	+
	Anchor tag for error details. Same for each row	u
	Previous page	\
	Next page	/
	Print Error Details page	p
	Close	l
Running Tasks and popup menu	View Running Tasks	i
	Stop running task from popup menu	x
	View log from main page and popup menu	q
	From Running Task Log page, navigate to top	q
	Refresh	r
	Close	l
Extended Analytics	Save template as	s
	Delete template	p
	Refresh template	o
	Create Star Schema	c
	Delete Star Schema	e
	Update Star Schema	u
Save Template for Extended Analytics	Save template	s
	Cancel	l

Module	Action	Keyboard Shortcut
Data Audit	Start date	[
	End date	]
	Previous	\
	Next	/
	POV	Same as POV bar
	View	i
	Clear log	l
	Export	x
Task Audit	Start date	[
	End date	]
	Previous	\
	Next	/
	View	i
	Clear log	l
	Export	x
Task Audit Export page	OK	o

Table 99 Load and Extract Tasks

Module	Action	Keyboard Shortcut
Load Security	Security file	m
	Load	l
	Back to Load Security	k
Load Metadata	Metadata file	m
	Load	l
	Scan	s
	Back to Load Metadata	k
	Select All	g
	Deselect All	q

Module	Action	Keyboard Shortcut
Load Member Lists	Member list file	m
	Load	l
	Scan	s
	Back to Load Member Lists	k
Load Rules	Rules file	m
	Load	l
	Scan	s
	Back to Load Rules	k
Load Data	Load	l
	Scan	s
	Back to Load Data	k
Load Journals	Journals file	m
	Load	l
	Back to Load Journals	k
Load Intercompany Transactions	Load	l
	Scan	s
	Back to Load Intercompany Transactions	k
Extract Security	Extract	m
	Back to Extract Security	k
Extract Metadata	Extract	m
	Back to Extract Metadata	k
	Select All	g
	Deselect All	q
Extract Member Lists	Extract	k
Extract Rules	Extract	k
Extract Data	Extract	m
	Back to Extract Data	k

Module	Action	Keyboard Shortcut
Extract Journals	Extract	m
	Back to Extract Journals	k
Extract Intercompany Transactions	Extract	m
	Back to Extract Intercompany Transactions	k

Table 100 Application Tasks

Module	Action	Keyboard Shortcut
Create application	Create	c
Register Application	Register	r
	Close	l
Create Classic application	Navigate	n
	Administer	a
	Classic Application Administration	c
	Consolidation Administration	c
	Create application	c
	Delete application	d
	Register application	r
	Consolidation System Messages	t
	Consolidation Users on system	o
	Manage Consolidation Servers and Applications	v

Table 101 Navigation Tasks

Module	Action	Keyboard Shortcut
Object palette (Left-side navigation)	Browser View: Tree	b
	Browser View: List	0 (zero)
	Project View	j
	Navigate	Use TAB key
	Navigate through the Tree and List	Use the Up and Down arrow keys
	Expand and collapse the tree nodes	Use the Right and Left arrow keys
	Select nodes	Use the Spacebar

Module	Action	Keyboard Shortcut
User preferences	Save	s
	Reset	r
Member Selector	OK	o
	Cancel	l
	Previous	\
	Next	/
	Textbox selected value	– (dash)
	Textbox top name	k
	Dropdown selector	g
	Description	'
	Filter	=
	Validate selected value	i
	Clear selected value	y
	Find previous value	u
	Find next value	v
	Click to select a list	Shift + :
	Drill up	. (period)
	Clear the checked item	[
	Expand selected list	Shift + (pipe key)
	Check all	]
	Move selected item up	Shift + {
	Move selected item down	Shift + }
	Tree	x + Tab
	List	; + Tab
	fn	' (apostrophe)
	Cur	, (comma)
	Sort	Shift + ?

Module	Action	Keyboard Shortcut
POV Bar	Scenario	!
	Year	@
	Period	#
	View	\$
	Entity	%
	Value	^
	Account	&
	ICP	*
	Custom1	(
	Custom2	)
	Custom3	-
	Custom4	+
Calendar	Previous year	r
	Previous month	p
	Next month	n
	Next year	x
	Close calendar	l

Table 102 Data Grid and Data Form Navigation

Module	Action	Keyboard Shortcut
Data Grids and Data Forms	Activate	For data grid: Space Data Forms are activated by default.
	Move to next cell in a row	Right arrow For Data Forms: right arrow and Tab
	Move to previous cell in a row	Left arrow For Data Forms: left arrow, Shift and Tab
	Move to next row	Down arrow
	Move to previous row	Up arrow
	Move to top left cell	Ctrl + Home Data grid works only on the displayed cells. Data forms works on all the data.
	Move to right bottom cell	Ctrl + End Data grid works only on the displayed cells. Data forms works on all the data.
	First cell in a row	HOME
	Last cell in a row	END
	First cell in a column	PAGEUP
	Last cell in a column	PAGEDOWN
	Enter edit mode in a cell	F2
	Finish editing and move one cell down	Enter
	Finish editing and move one cell right	Tab
	Access context menu	Data Grids: F9 Data Forms: F9
	Row header context menu available in Linked form and preview form	Data Grids: unavailable Data Forms: w
	Column header context menu available in Linked form and preview form	Data Grids: unavailable Data Forms: n
	Lines context menu available in Linked form and preview form	Data Grids: unavailable Data Forms: o

Table 103 Document Manager Tasks

Module	Action	Keyboard Shortcut
Document Manager (Main window)	All	1
	Tasklists	2
	Data Forms	3
	Data Grids	4
	Report	5
	Link	6
	Related Content	7
	Custom documents	8
	New folder	l
	Print	p
	New task list	w
	New data grid	w
	New data form	w
	New link	w
	New Web form	w
	New report	w
	Edit	i
	Add to Favorites	p
	Delete	e
	Load	o
	Extract	x
	Show private documents	q
	Hide private documents	q
	Browse	s
Load local reports	Select report	m
	Open	o
	Print	p

Module	Action	Keyboard Shortcut
New Folder	OK	o
	Cancel	l
Load Documents	OK	o
	Cancel	l
	Finished	y
	Load more files	x
Save Documents	OK	o
	Cancel	l
Creating Intercompany Partner Reports	Print/Preview	w
	Save remote	m
	Save local	s
	Close	l
	Entity Selector	q
	Add Entity Account	+
	Remove Entity Account	-
	Add Partner Account	y
	Remove Partner Account	x
Creating Explore Data Reports	Print/preview	w
	Save remote	m
	Save local	s
	View as XML	x
	View as RPT	p
	Close	l
Print report options	OK	o
	Cancel	l

Module	Action	Keyboard Shortcut
Overrides for generation of Intercompany reports	OK	o
	Cancel	l
	Entity selector	g
	Partner selector	q
Overrides for generation of Journal reports	OK	o
Favorites	Remove	m
	Clean	l
Related Content (Tree View)	Resource document	i
	Select resource document	v
	Clear document	u
	Open document	g
	Select resource server	k
	Tree	x
	Client area	y
	Save	s
	Close	l
Links	View	w
	Save	s
	Cancel	l

Table 104 Tasklists

Module	Action	Keyboard Shortcut
Add Tasks	Add to Tasklist	k
	Return to Tasklist	l
	Select All	Shift + '+' (plus sign)

Module	Action	Keyboard Shortcut
Manage Tasklist	Save	s
	Close	l
	Move up	u
	Move down	w
	Select list of tasks	k
	This is my default tasklist	m
	Actions Menu: Add	c, a
	Actions Menu: Remove	c, r

Table 105 Data Grid Tasks

Module	Action	Keyboard Shortcut
Cell Line Item Detail	Add	a
	OK	o
	Cancel	l
Cell Information	OK	o
Cell Text	Attach	a
	Detach	t
	OK	o
	Cancel	l
Attach Documents	Attach	x
	Cancel	y
Grid Settings	Layout	1
	Display Options	2
	OK	o
	Cancel	l
Manage Process	Attach	a
	Detach	t
	OK	o
	Cancel	l

Module	Action	Keyboard Shortcut
Cell History	Close	l
Source Transaction	Close window	l
Destination Transaction	Close window	l
Entity Details	Refresh	r
Close Entity Details Window	Close Entity Details Window	l

Table 106 Intercompany Transaction Tasks

Module	Action	Keyboard Shortcut
Process Intercompany Transactions	Previous	\
	Next	/
	Entity	g
	Partner	q
	Update	u
	Select All	+
	Actions menu: Alert	c, a
	Actions menu: New	c, n
	Actions menu: Edit	c, e
	Actions submenu: Edit transaction	c, e, t
	Actions submenu: Set reason code	c, e, s
	Actions menu: Delete	c, d
	Actions submenu: Delete selected	c, d, s
	Actions submenu: Delete all	c, d, a
	Actions menu: Match	c, m
	Actions submenu: Auto match	c, m, a
	Actions submenu: Manual match	c, m, m
	Actions submenu: Unmatch selected	c, m, u
	Actions submenu: Unmatch all	c, m, i
	Actions submenu: Post	c, p
	Actions submenu: Post selected	c, p, s
	Actions submenu: Post all	c, p, l
	Actions menu: Unpost	c, u
	Actions submenu: Unpost selected	c, u, s
	Actions submenu: Unpost all	c, u, l
	Actions menu: Filter	c, f
	Actions submenu: Column	c, f, c
	Actions submenu: Row	c, f, r
	Actions menu: Report	c, r
	Actions submenu: Report by ID	Shortcut Keys by Module
	Actions submenu: Report by Account	367
	Actions submenu: Report by Transaction	

Module	Action	Keyboard Shortcut
New/Edit	Calendar	[
	New	w
	Save	s
	Reset	k
	Close	l
	Pick members	i
Set reason code	Set	s
	Cancel	l
Manual Match	Entity	g
	Partner	q
	Add Accounts	k
	Remove Accounts	x
	Add Matching Accounts	e
	Remove Matching Accounts	u
	Select All Entity Accounts	+
	Select All Partner Accounts	-
	Match	m
	Save as Template	s
	Close	l

Module	Action	Keyboard Shortcut
Auto Match	Entity	g
	Partner	q
	Add Accounts	k
	Remove Accounts	x
	Add Matching Accounts	e
	Remove Matching Accounts	u
	Select All Entity Accounts	+
	Select All Partner Accounts	-
	Match	m
	Save as Template	s
	Close	l
Unmatch Selected	Unmatch	u
	Close	l
	Select all	k
	Clear all	e

Module	Action	Keyboard Shortcut
Filter-Columns	Restore defaults	u
	Close	l
	Display column list multiselect	k
	Selected column list multiselect	e
	Add selected	\
	Remove selected	/
	Move up	p
	Move down	o
	Actions menu: Sort options	c, s
	Actions submenu: No sort	c, s, n
	Actions submenu: Ascending	c, s, a
	Actions submenu: Descending	c, s, d
	Actions menu: Display options	c, d
	Actions submenu: Label	c, d, l
	Actions submenu: Description	c, d, d
	Actions submenu: Both	c, d, b
Filter-Rows	Entity selector	4
	Partner selector	5
	Entity account selector	6
	Partner account selector	7
	Start Calendar	8
	End Calendar	9
	Restore defaults	u
	Close	l

Module	Action	Keyboard Shortcut
Transaction ID - General	Print/preview	w
	Save remote	m
	Save local	s
	Close	l
	Entity selector	g
	Partner selector	q
Transaction ID - Display Options	No access keys	No access keys
Account - General	Print/preview	w
	Save remote	m
	Save local	s
	Close	l
	Entity selector	g
	Partner selector	q
Account - Display options	No access keys	No access keys
Account - Accounts	Add Accounts	+
	Add Accounts	k
	Matching Accounts	-
	Remove Accounts	x
	Add Matching Accounts	e
	Remove Matching Accounts	u

Module	Action	Keyboard Shortcut
Transactions - Columns	Restore defaults	u
	Close	l
	Display column list multiselect	k
	Selected column list multiselect	e
	Add selected	\
	Remove selected	/
	Move up	p
	Move down	o
	Actions menu: Sort options	c, s
	Actions submenu: No sort	c, s, n
	Actions submenu: Ascending	c, s, o
	Actions submenu: Descending	c, s, d
	Actions menu: Display options	c, d
	Actions submenu: Label	c, d, l
	Actions submenu: Description	c, d, d
Transactions - Rows	Entity selector	4
	Partner selector	5
	Entity account selector	6
	Partner account selector	7
	Calendar 1	8
	Calendar 2	9
	Restore defaults	u
	Close	q
Intercompany Transaction Alerts Options	Send	s
	Cancel	l

Module	Action	Keyboard Shortcut
Monitor Intercompany Transactions	Entity Search	g
	Update	u
	Process Status column	1 (numeric)
	Lock Status column	2
	Previous	\
	Next	/
	Select All	Shift + '+' (plus sign)
	Actions menu: Alert	c, t
	Actions menu: Summary	c, u
Monitor Intercompany Transactions Summary	Close window	l
Intercompany Monitor Detail	Close window	l
Manage Intercompany Transaction Periods	Open Period	p
	Close Period	l
	Save Period Settings	s
Intercompany Open Close Period Status	Close	l
Lock / Unlock Intercompany Entities	Lock	k
	Unlock	u
	Select All	Shift + '+' (plus sign)
Lock / Unlock Intercompany Status	Close window	l
Manage Intercompany Reason Codes	Add	k
	Edit	e
	Remove	m
	Select All	Shift + '+' (plus sign)
	Edit mode: OK	o
	Edit mode: Cancel	l

Table 107 Manage Ownership Tasks

Module	Action	Keyboard Shortcut
Manage Ownership Window	Select All	k
	Fill	i
	Display Options	p
	Calculate Ownership	o
	Save	s
	Reset	e
Display Options	OK	o
	Cancel	l
Calculate Ownership	Calculate	c
	Cancel	l

Table 108 Database Management Tasks

Module	Action	Keyboard Shortcut
Copy Data	Copy Data	c
	Back to Copy Data	k
	Clear Data	2
	Delete Invalid Records	3
	Copy Data	1
	POV: Source Scenario	!
	POV: Source Year	@
	POV: Source Period	#
	POV: Destination Scenario	!
	POV: Destination Year	@
	POV: Destination Period	#
	POV: Copy Members Entity	%
	POV: Copy Members Account	&

Module	Action	Keyboard Shortcut
Clear Data	Copy Data	1
	Clear Data	2
	Delete Invalid Records	3
	Clear Data	c
	Back to Clear Data	k
	POV: Scenario	!
	POV: Year	@
	POV: Period	#
	POV: Entity	%
	POV: Account	&
Delete Invalid Records	Copy Data	1
	Clear Data	2
	Delete Invalid Records	3
	Delete Invalid Records	d
	Scan Only	s
	Back to Delete Invalid Records	k

Table 109 Process Control Tasks

Module	Action	Keyboard Shortcut
Process Control Window from Data Grid	Access keys are the same as data grids	Access keys are the same as data grids
Process Control Window from Process Management Module	Attach	a
	Detach	t
	Start, Promote, Submit, Approve, Publish, Sign Off, Reject	x
Error condition popup	Close	l
	Show Details	s
Options	OK	o
	Cancel	l
Process Status Summary	Close window	l

Module	Action	Keyboard Shortcut
Validation Detail	Update	u
	Close window	l
Process Flow History	Close window	l
Calculation Status Summary	Close window	l
Non-Phased Submissions	OK	o
	Cancel	l
Phased Submissions - Rows	OK	o
	Cancel	l
	Rows	1
	Columns	2
Phased Submissions - Columns	OK	o
	Cancel	l
	Rows	1
	Columns	2
Manage Submission Phases Page from Data Grids	Accessibility is not supported.	Accessibility is not supported.
Show Unassigned Groups	OK	o
Process Control Advanced Options	OK	o
	Cancel	l

Table 110 Equity Pickup Tasks

Module	Action	Keyboard Shortcut
Main Equity Pickup page	Previous	\
	Next	/
	Sort on Level column	1
	Sort on Owner column	2
	Sort on Owned column	3
	Sort on %EPU column	4
	Sort on Status column	5
	Actions menu: Filter	c, f
	Actions menu: Refresh	c, r
	Actions menu: Calculate	c, l
	Actions submenu: Run EPU	c, l, r
	Actions submenu: Force EPU	c, l, f
	Actions menu: Print Report	c, p
	Actions submenu: Full Report	c, p, u
	Actions submenu: Filtered Report	c, p, i
Filter	Owner Search	g
	Owned Search	q
	OK	o
	Cancel	l
Reports	Accessibility is not supported.	Accessibility is not supported.

Table 111 Data Forms

Module	Action	Keyboard Shortcut
Web Data Form (main form)	Read current POV	~
	Actions menu: Submit	c, s
	Actions menu: Refresh	c, r
	Actions menu: Calculate	c, c
	Actions menu: Force Calculate	c, t
	Actions menu: Cell Text	c, x
	Actions menu: Instructions	c, i
	Actions menu: Print	c, p
	Actions menu: Import from Excel	c, m
	Actions menu: Export to Excel 2003	c, 3
	Actions menu: Export to Excel 2007	c, 7
Web Data Form (Linked form, Preview form)	Submit	s
	Refresh	r
	Calculate	c
	Force calculate	f
	Cell text	t
Suppress Rows	Actions menu: Suppress rows	c, w
	Actions submenu: None	c, w, n
	Actions submenu: Nodata	c, w, d
	Actions submenu: Zero	c, w, z
	Actions submenu: Nodata and Zero	c, w, o
	Actions submenu: Invalid	c, w, i
	Actions submenu: Invalid and Nodata	c, w, a
	Actions submenu: Invalid and Zero	c, w, e
	Actions submenu: Invalid and Nodata and Zero	c, w, r

Module	Action	Keyboard Shortcut
Suppress Columns	Actions menu: Suppress columns	c, n
	Actions submenu: None	c, n, n
	Actions submenu: Nodata	c, n, d
	Actions submenu: Zero	c, n, z
	Actions submenu: Nodata and Zero	c, n, o
	Actions submenu: Invalid	c, n, i
	Actions submenu: Invalid and Nodata	c, n, a
	Actions submenu: Invalid and Zero	c, n, e
	Actions submenu: Invalid and Nodata and Zero	c, n, r
Suppress Lines	Actions menu: Suppress lines	c, o
	Actions submenu: None	c, o, n
	Actions submenu: Nodata	c, o, d
	Actions submenu: Zero	c, o, z
	Actions submenu: Nodata and Zero	c, o, o
	Actions submenu: Invalid	c, o, i
	Actions submenu: Invalid and Nodata	c, o, a
	Actions submenu: Invalid and Zero	c, o, e
	Actions submenu: Invalid and Nodata and Zero	c, o, r
Line Item Details	Save	s
	Refresh	r
	Close	L
	New entry	w
	Delete row	e
	Clear row	c
	Select All	Shift + '+'
	Move focus to a cell on which the context menu is to be displayed and click F9.	Use Up and Down arrows to select items.

Module	Action	Keyboard Shortcut
Cell Text	Attach	a
	Detach	t
	OK	o
	Cancel	l
Cell Adjustments	Close window	l
	OK	o
Properties	OK	o
Import from Excel	OK	o
	Cancel	l
	File Browse	u
Instructions	Close	l
View Definition	Close	l
New Web Form	Properties	1
	POV	2
	Columns	3
	Rows	4
	Details	5
	Header	6
	Script	7
Properties	Save	s
	Reset	r
	Scan	c
	Preview	l
	Close	l
	Update	u
	Next	x
	Back	k
POV	Search	q

Module	Action	Keyboard Shortcut
Columns same	Search Common keys: same as Properties module.	q
Rows	Search Common keys: same as Properties module.	q
Details	Common keys: same as Properties module.	Common keys: same as Properties module.
Headers	Common keys: same as Properties module.	Common keys: same as Properties module.
Script	Print	p
	More	o
	Less	e
	Print page: Print	p
	Close print page	l

Table 112 Journals Tasks

Module	Action	Keyboard Shortcut
Process Journals Main Window	Check all	+
	Anchor (common for all rows) Journal Label	L
	New	w
	Open	o
	Delete	e
	Filter	i
	Reports	p
	Submit	c, s
	Unsubmit	c, u
	Approve	c, a
	Reject	c, r
	Post	c, p
	Unpost	c, n
New Journal window	OK	o
	Cancel	l

Module	Action	Keyboard Shortcut
Open/Edit/New	Add row	+
	Copy row	y
	Paste row	i
	Select member	m
	Display context menu	Select a cell and click F9 (only if journal is editable).
	New	w
	Delete	e
	Save	s
	Reset	x
	Scan	k
	Print	p
	Close	l
Filter - Columns	Column	1
	Filter	2
	OK	o
	Cancel	l
	Restore defaults	e
Filter - Rows	Column	1
	Filter	2
	Restore defaults	e
	OK	o
	Cancel	l
	Entity search	q
	Select All	c
	Deselect All	p

Module	Action	Keyboard Shortcut
Reports - Properties	Print/preview	p
	Save remote	m
	Save local	s
	Close	l
	Properties	1
	Columns	2
	Filter	3
Reports - Columns	Print/preview	p
	Save remote	m
	Save local	s
	Close	l
	Properties	1
	Columns	2
	Filter	3
	Repeat	+
	Totals	t
Reports - Filter Rows	Restore defaults	e
	Properties	1
	Columns	2
	Filter	3
	Print/preview	p
	Save remote	m
	Save local	s
	Close	l

Module	Action	Keyboard Shortcut
Journals Template - Main window	Standard	1
	Recurring	2
	New	w
	Open	o
	Delete	e
	Filter	f
	Reports	r
	Check All	+
	Anchor (common for all rows) - Link to journal labels	l
New/Open	New	w
	Delete	e
	Save	s
	Reset	x
	Generate	g
	Print	p
	Close	l
	Add rows	+
	Copy rows	y
	Paste rows	i
	Select members	m
	Enable context menu on line items.	Move the focus to any cell and click F9.
Filter - Columns	Restore defaults	e
	Columns	1
	Filter	2
	OK	o
	Cancel	l

Module	Action	Keyboard Shortcut
Filter	Restore defaults	e
	Columns	1
	Filter	2
	OK	o
	Cancel	l
Manage Periods	Open period	p
	Close period	l
Manage Groups	Add groups	g
	Remove groups	m
	Remove all	v
	Select all	+
New Journals	Move focus to a cell on the context menu to be displayed and click F9.	Use Up and Down arrows to select items.
New Journals Template	Move focus to a cell on the context menu to be displayed and click F9.	Use Up and Down arrows to select items.

Table 113 Security Tasks

Module	Action	Keyboard Shortcut
Select Users and Groups	Search	s
	Show all	w
	Select	1
	Select all	3
	Remove	2
	Remove all	4
	Help	g
	Back	k
	Next	x

Module	Action	Keyboard Shortcut
Select Classes	Select	1
	Select all	3
	Remove	2
	Remove all	4
	Add	b
	Delete	o
Assign Access	Save	s
	Cancel	l
	Set	t
	Pivot	p
	Add alert	e
	Remove alert	r
Security Reports	Launch	n
	Export	r

Glossary

account type A property that determines how an account's value flows over time and its sign behavior. Account type options can include expense, income, asset, liability, and equity.

activity-level authorization Defines user access to applications and the types of activities they can perform on applications, independent of the data that will be operated on.

ancestor A branch member that has members below it. For example, the members Qtr2 and 2006 are ancestors of the member April.

application 1) A software program designed to run a specific task or group of tasks such as a spreadsheet program or database management system. 2) A related set of dimensions and dimension members that are used to meet a specific set of analytical requirements, reporting requirements, or both.

application currency The default reporting currency for the application.

asset account An account type that stores values that represent a company's assets.

attribute A characteristic of a dimension member. For example, Employee dimension members may have attributes of Name, Age, or Address. Product dimension members can have several attributes, such as a size and flavor.

authentication Verification of identity as a security measure. Authentication is typically based on a user name and password. Passwords and digital signatures are forms of authentication.

auto-reversing journal A journal for entering adjustments that you want to reverse in the next period.

balance account An account type that stores unsigned values that relate to a particular time.

balanced journal A journal in which the total debits equal the total credits.

base currency The currency in which daily business transactions are performed.

base entity An entity at the bottom of the organization structure that does not own other entities.

business rules Logical expressions or formulas that are created within an application to produce a desired set of resulting values.

calendar User-defined time periods and their relationship to each other. Q1, Q2, Q3, and Q4 comprise a calendar or fiscal year.

cell (1) The data value at the intersection of dimensions in a multidimensional database; the intersection of a row and a column in a worksheet. (2) A logical group of nodes belonging to one administrative domain.

CHANGED status Consolidation status that indicates data for an entity has changed.

child A member with a parent above it in the database outline.

consolidation The process of aggregating data from dependent entities to parent entities. For example, if the dimension Year consists of the members Qtr1, Qtr2, Qtr3, and Qtr4, its consolidation is Year.

consolidation rule The rule that is executed during the consolidation of the node of the hierarchy. This rule can contain customer-specific formulas appropriate for the correct consolidation of parent balances. Elimination processing can be controlled within these rules.

contribution The value added to a parent from a child entity. Each child has a contribution to its parent.

conversion rate See exchange rate.

currency conversion A process that converts currency values in a database from one currency into another. For example, to convert one U. S. dollar into the European euro, the exchange rate (for example, 0.923702) is multiplied by the dollar (1×0.923702). After conversion, the European euro amount is .92.

custom dimension A dimension created and defined by users. Channel, product, department, project, or region could be custom dimensions.

data form A grid display that enables users to enter data into the database from an interface such as a Web browser, and to view and analyze data or related text. Certain dimension member values are fixed, giving users a specific view into the data.

data lock A feature that prevents changes to data according to specified criteria, such as a period or scenario.

dependent entity An entity that is owned by another entity in the organization.

descendant Any member below a parent in the database outline. In a dimension that includes years, quarters, and months, the members Qtr2 and April are descendants of the member Year.

destination currency The currency to which balances are converted. You enter exchange rates and convert from the source currency to the destination currency. For example, when you convert from EUR to USD, the destination currency is USD.

dimension A data category used to organize business data for the retrieval and preservation of values. Dimensions usually contain hierarchies of related members grouped within them. For example, a Year dimension often includes members for each time period, such as quarters and months.

direct rate A currency rate that you enter in the exchange-rate table. The direct rate is used for currency conversion. For example, to convert balances from JPY to USD, in the exchange-rate table, enter a rate for the period/scenario where the source currency is JPY and the destination currency is USD.

drill-down Navigation through the query result set using the dimensional hierarchy. Drilling down moves the user perspective from aggregated data to detail. For example, drilling down can reveal hierarchical relationships between years and quarters or quarters and months.

dynamic member list A system-created named member set that is based on user-defined criteria. The list is refreshed automatically whenever it is referenced in the application. As dimension members are added and deleted, the list automatically reapplies the criteria to reflect the changes.

dynamic view account An account type indicating that account values are calculated dynamically from the data that is displayed.

elimination The process of zeroing out (eliminating) transactions between entities within an organization.

entity A dimension representing organizational units. Examples: divisions, subsidiaries, plants, regions, products, or other financial reporting units.

exchange rate type An identifier for an exchange rate. Different rate types are used because there may be multiple rates for a period and year. Users traditionally define rates at period end for the average rate of the period and for the end of the period. Additional rate types are historical rates, budget rates, forecast rates, and so on. A rate type applies to a specific time.

expense account An account that stores periodic and year-to-date values that decrease net worth if they are positive.

Extensible Markup Language (XML) A language comprising a set of tags used to assign attributes to data that can be interpreted between applications according to a schema.

external authentication Logging on to Oracle EPM System products with user information stored outside the application. The user account is maintained by the EPM System, but password administration and user authentication are performed by an external service, using a corporate directory such as Oracle Internet Directory (OID) or Microsoft Active Directory (MSAD).

file delimiter A character, such as a comma or tab, that separates fields in a data source.

flow account An unsigned account that stores periodic and year-to-date values.

generation A layer in a hierarchical tree structure that defines member relationships in a database. Generations are ordered incrementally from the top member of the dimension (generation 1) down to the child members. Use the unique generation name to identify a layer in the hierarchical tree structure.

holding company An entity that is part of a legal entity group, with direct or indirect investments in all entities in the group.

IMPACTED status A status that indicates changes in child entities consolidating into parent entities.

INACTIVE status A status that indicates entities deactivated from consolidation for the current period.

intercompany elimination See elimination.

intercompany matching The process of comparing balances for pairs of intercompany accounts within an application. Intercompany receivables are compared to intercompany payables for matches. Matching accounts are used to eliminate intercompany transactions from an organization's consolidated totals.

intercompany matching report A report that compares intercompany account balances and indicates whether the accounts are in balance.

journal entry (JE) A set of debit-credit adjustments to account balances for a scenario and period.

liability account An account type that stores "point in time" balances of a company's liabilities. Examples: accrued expenses, accounts payable, and long-term debt.

locked A user-invoked process that prevents users and processes from modifying data.

LOCKED status A consolidation status indicating that an entity contains data that cannot be modified.

member list A named system- or user-defined group that references members, functions, or member lists within a dimension.

merge A data load option that clears values only from the accounts specified in the data load file and replaces them with values in the data load file.

metadata A set of data that defines and describes the properties and attributes of the data stored in a database or used by an application. Examples of metadata are dimension names, member names, properties, time periods, and security.

NO DATA status A consolidation status indicating that this entity contains no data for the specified period and account.

ODBC Open Database Connectivity. A database access method used from any application regardless of how the database management system (DBMS) processes the information.

OK status A consolidation status indicating that an entity has already been consolidated, and that data has not changed below it in the organization structure.

Open Database Connectivity (ODBC) Standardized application programming interface (API) technology that allows applications to access multiple third-party databases.

organization An entity hierarchy that defines each entity and their relationship to others in the hierarchy.

P&L accounts Profit and loss accounts. P&L refers to a typical grouping of expense and income accounts that comprise a company's income statement.

parent adjustments The journal entries that are posted to a child in relation to its parent.

parents The entities that contain one or more dependent entities that report directly to them. Because parents are entities associated with at least one node, they have entity, node, and parent information associated with them.

percent consolidation The portion of a child's values that is consolidated to its parent.

percent control The extent to which an entity is controlled within the context of its group.

percent ownership The extent to which an entity is owned by its parent.

periodic value method (PVA) A process of currency conversion that applies the periodic exchange rate values over time to derive converted results.

plug account An account in which the system stores any out-of-balance differences between intercompany account pairs during the elimination process.

PVA See periodic value method.

recurring template A journal template for making identical adjustments in every period.

regular journal A feature for entering one-time adjustments for a period. A regular journal can be balanced, balanced by entity, or unbalanced.

relational database A type of database that stores data in related two-dimensional tables. Contrast with multidimensional database.

replace A data load option that clears existing values from all accounts for periods specified in the data load file and loads values from the data load file. If an account is not specified in the load file, its values for the specified periods are cleared.

review level A Process Management review status indicator representing the process unit level, such as Not Started, First Pass, Submitted, Approved, and Published.

scenario A dimension for classifying data; for example, Actuals, Budget, Forecast1, or Forecast2.

Shared Services Registry The part of the Shared Services repository that manages EPM System deployment information for most EPM System products, including installation directories, database settings, computer names, ports, servers, URLs, and dependent service data.

source currency The currency from which values originate and are converted through exchange rates to the destination currency.

standard journal template A journal function used to post adjustments that have common adjustment information for each period. For example, you can create a standard template that contains the common account IDs, entity IDs, or amounts, and then use the template as the basis for many regular journals.

system extract A feature that transfers data from application metadata into an ASCII file.

translation See currency conversion.

triangulation A means of converting balances from one currency to another through a third common currency. In Europe, this currency is the euro for member countries. For example, to convert from the French franc to the Italian lira, the common currency is defined as the European euro. Therefore, to convert balances from the French franc to the Italian lira, balances are converted from the French franc to the European euro and from the European euro to Italian lira.

user-defined member list A named, static set of members within a dimension defined by the user.

value dimension A dimension that is used to define input value, translated value, and consolidation detail.

view A year-to-date or periodic display of data.

XML See Extensible Markup Language.

Index

A

- ABSExp function, [239](#)
- account attributes, [76](#)
- Account dimension, [20](#)
- account expressions, rules, [215](#)
- accounts example, [99](#)
- AccountType attribute, [76](#)
 - behavior, [78](#)
- AccountType function, [240](#)
- AccountTypeID function, [241](#)
- active account, [92](#)
- AddEntityToList function, [242](#)
- AddEntityToListUsingIDs function, [243](#)
- AddMember keyword, Web scripts, [146](#)
- AddMemberToList function, [244](#)
- alerting
 - intercompany transactions, [67](#)
 - process management, [66](#)
- Alloc function, [245](#)
- AllowAdjFromChildren attribute, [81](#)
- AllowAdjFromChildren function, [246](#)
- AllowAdjs attribute, [81](#)
- AllowAdjs function, [247](#)
- APP or XML files, creating, [96](#)
- APP to XML, [106](#)
- application profiles
 - creating, [27](#)
 - defining calendars for, [28](#)
 - saving, [32](#)
 - specifying frequencies for, [29](#)
 - specifying languages for, [27](#)
 - specifying periods for, [29](#)
- application profiles (PER), creating, [27](#)
- application security files (SEC)
 - extracting in Web, [74](#)
 - extracting in Windows, [73](#)
 - loading in Web, [70](#)
 - loading in Windows, [68](#)
- application settings attributes, [84](#)
- application settings example, [98](#)
- applications
 - closing, [38](#)
 - creating, [32](#)
 - creating in Web, [34](#)
 - creating in Windows, [33](#)
 - deleting, [39](#)
 - opening, [37](#)
 - sample files for, [46](#)
- asset account behavior, [78](#)
- assigning access to security classes, [65](#)
- assigning users and groups to roles, [65](#)
- attributes
 - AccountType, [76](#)
 - AllowAdjFromChildren, [81](#)
 - AllowAdjs, [81](#)
 - ConsolidateYTD, [82](#)
 - ConsolidationRules, [85](#)
 - ConsolMethod, [87](#)
 - Control, [88](#)
 - Currency, [91](#)
 - CustomTopMember, [76](#)
 - DefaultCurrency, [85](#)
 - DefaultFreq, [83](#)
 - DefaultParent
 - accounts, [76](#)
 - custom, [80](#)
 - entities, [81](#)
 - scenarios, [83](#)
 - DefaultRateForBalanceAccounts, [85](#)
 - DefaultRateForFlowAccounts, [85](#)
 - DefaultValueForActive, [85](#)
 - DefaultView, [83](#)
 - DefCurrency, [82](#)
 - DefFreqForICTrans, [83](#)
 - Description
 - accounts, [76](#)

- consolidation methods, 88
- currencies, 91
- custom, 80
- entities, 82
- scenarios, 83
- EnableCustomAggr, 76
- EnableDataAudit
 - accounts, 76
 - scenarios, 83
- EnableMetadataSecurityFiltering, 85
- HoldingCompany, 82
- ICPEntitiesAggregationWeight, 85
- ICPTopMember
 - accounts, 76
- IsCalculated
 - accounts, 77
 - custom, 80
- IsConsolidated, 77
- IsHoldingMethod, 88
- IsICP
 - accounts, 77
 - entities, 82
- MaximumReviewLevel, 83
- Member
 - accounts, 77
 - custom, 80
 - entities, 82
 - scenarios, 83
- NodeSecurity, 85
- NumDecimalPlaces, 77
- OrgByPeriodApplication, 85
- PercentConsol, 88
- PlugAcct, 77
- Scale, 91
- SecurityAsPartner, 82
- SecurityClass
 - accounts, 77
 - custom, 81
 - entities, 82
 - scenarios, 84
- Submission Group
 - accounts, 77, 81
- SupportsProcessManagement, 84
- SwitchSignForFlow, 81
- SwitchTypeForFlow, 81
- ToPercentControl, 88
- ToPercentControlComp, 88

- TranslationOperator, 92
- UsedByCalcRoutine, 88
- UsePVAForBalanceAccounts, 86
- UsePVAForFlowAccounts, 86
- UserDefined1...3
 - accounts, 77
 - custom, 81
 - entities, 82
 - scenarios, 84
- UseSecurityForAccounts, 86
- UseSecurityForCustom, 86
- UseSecurityForEntities, 86
- UseSecurityForICP, 86
- UseSecurityForScenarios, 86
- UsesLineItems
 - accounts, 78
 - scenarios, 84
- UseSubmissionPhase, 86
- ValidationAccount, 86
- XBRLTags, 78
- ZeroViewForAdj, 84
- ZeroViewForNonadj, 84
- auditing data, 55
- auditing tasks, 53

B

- BackgroundPOV keyword, Web scripts, 147
- balance account behavior, 78
- balance recurring account behavior, 78
- Blank keyword, Web scripts, 148

C

- Calc1 keyword, Web scripts, 150
- CalcByRow keyword, Web scripts, 150
- CalcStatus function, 249
- CalculateExchangeRate function, 247
- CalculateRate function, 248
- Cell_Link keyword, Web scripts, 150
- CellText keyword, Web scripts, 151
- Clear function, 250
- closing applications, 38
- Cn keyword
 - Web scripts, 149
- Con function, 251
- Consol1, Consol2, Consol3 function, 252
- Consol1...3 account, 93

- ConsolidateYTD attribute, [82](#)
- ConsolidateYTD function, [253](#)
- consolidation accounts
 - active account, [92](#)
 - Consol1...3 account, [93](#)
 - DOWN account, [92](#)
 - method account, [93](#)
 - PCON account, [92](#)
 - PCTRL account, [92](#)
 - POWN account, [92](#)
- consolidation methods, [87](#)
- consolidation rules, execution order, [217](#)
- ConsolidationRules attribute, [85](#)
- ConsolMethod attribute, [87](#)
- ContainsCellText function, [253](#)
- Control attribute, [88](#)
- copying periods, [31](#)
- CreateInputCache function, [254](#)
- creating
 - application profiles, [27](#)
 - application profiles (PER), [27](#)
 - applications, [32](#)
 - data form scripts (WDF), [144](#)
 - data forms in Form Builder, [139](#)
 - folders for data forms, [173](#)
 - in Web applications, [34](#)
 - in Windows applications, [33](#)
 - journal files (JLF), [129](#)
 - member list files (LST), [121](#)
 - metadata files (APP or XML), [96](#)
 - rules files (RLE), [225](#)
 - rules, Rules Editor, [223](#)
 - star schema template, [188](#)
- CSV file
 - data audit log, [57](#)
 - task audit log, [55](#)
- CUR keyword
 - Web scripts, [171](#)
- Currencies attributes, [91](#)
- currencies example, [98](#)
- Currency attribute, [91](#)
- Currency function, [254](#)
- custom attributes, [80](#)
- Custom dimension, [21](#)
- custom dimension example, [100](#)
- Custom1Top, Custom2Top, Custom3Top, and Custom4Top functions, [255](#)

- CustomHeader keyword
 - Web scripts, [151](#)
- CustomHeaderStyle keyword
 - Web scripts, [152](#)
- CustomTopMember attribute, [76](#)

D

- data audit, [55](#)
- data form script syntax, [144](#)
- data form scripts (WDF)
 - creating, [144](#)
 - extracting, [175](#)
 - loading, [174](#)
- data forms
 - creating folders, [173](#)
 - deleting, [175](#)
- data link
 - configuring for IBM DB2, [179](#)
 - configuring for Oracle, [178](#)
 - configuring for SQL Server, [179](#)
 - encrypting, [180](#)
- Decimal function, [256](#)
- default translation, [217](#), [218](#)
- DefaultCurrency attribute, [85](#)
- DefaultFreq attribute, [83](#)
- DefaultFreq function, [257](#)
- DefaultParent attribute
 - accounts, [76](#)
 - custom, [80](#)
 - entities, [81](#)
 - scenarios, [83](#)
- DefaultRateForBalanceAccounts attribute, [85](#)
- DefaultRateForFlowAccounts attribute, [85](#)
- DefaultTranslate function, [257](#)
- DefaultValueForActive attribute, [85](#)
- DefaultView attribute, [83](#)
- DefaultView function, [258](#)
- DefCurrency attribute, [82](#)
- DefCurrency function, [259](#)
- DefFreqForICTrans attribute, [83](#)
- defining
 - account attributes, [76](#)
 - calendars in applications, [28](#)
 - consolidation methods, [87](#)
 - Currencies attributes, [91](#)
 - custom attributes, [80](#)
 - entity attributes, [81](#)

- scenario attributes, [82](#)
- deleting
 - applications, [39](#)
 - data forms, [175](#)
 - periods, [32](#)
 - star schema, [188](#)
 - star schema template, [189](#)
- delimiter characters, [21](#)
- Description attribute
 - accounts, [76](#)
 - consolidation methods, [88](#)
 - currencies, [91](#)
 - custom, [80](#)
 - entities, [82](#)
 - scenarios, [83](#)
- description requirements, [21](#)
- dimensions overview, [18](#)
- disabling connections for users on system, [51](#)
- DOWN account, [92](#)
- DOWn function, [260](#)
- Dynamic function, [261](#)
- dynamic member lists, [124](#)
- DynamicPOV keyword, Web scripts, [152](#)

E

- E-mail alerting
 - intercompany transactions, [67](#)
 - process management, [66](#)
- editing data forms in Form Builder, [172](#)
- editing forms, [172](#)
- email alerts
 - for intercompany transactions, [348](#)
 - for process management, [347](#)
- EnableCustomAggr attribute, [76](#)
- EnableDataAudit attribute
 - accounts, [76](#)
 - scenarios, [83](#)
- EnableMetadataSecurityFiltering attribute, [85](#)
- EnableProcessManagement function, [319](#)
- enabling connections for users on system, [51](#)
- entities example, [100](#)
- entity attributes, [81](#)
- Entity dimension, [19](#)
- Exp function, [262](#)
- expense account behavior, [78](#)
- Explore Data reports, [192](#)
- exporting data to star schema, [185](#)

- Expression Builder
 - using the Function Wizard, [224](#)
- extracting
 - data form scripts (WDF), [175](#)
 - in Web
 - application security files (SEC), [74](#)
 - journal files (JLF), [136](#)
 - member list files (LST), [127](#)
 - metadata files (XML and APP), [120](#)
 - rules files (RLE), [229](#)
 - in Windows
 - application security files (SEC), [73](#)
 - journal files (JLF), [135](#)
 - member list files (LST), [127](#)
 - metadata files (XML and APP), [119](#)
 - rules files (RLE), [229](#)

F

- File Properties
 - creating metadata reports, [114](#)
- Financial Management
 - architecture, [18](#)
 - dimensions, [18](#)
- FIRST keyword
 - Web scripts, [171](#)
- flow account behavior, [78](#)
- Form Builder, [139](#)
 - editing forms, [172](#)
- FormInputBoxLength keyword, Web scripts, [152](#)
- FormNumDecimals keyword
 - Web scripts, [153](#)
- FormRowHeight keyword, Web scripts, [153](#)
- FormScale keyword, Web scripts, [154](#)
- frequencies, [29](#)
- Function Wizard, using in Expression Builder, [224](#)
- functions
 - ABSExp, [239](#)
 - AccountType, [240](#)
 - AccountTypeID, [241](#)
 - AddEntityToList, [242](#)
 - AddEntityToListUsingIDs, [243](#)
 - AddMemberToList, [244](#)
 - Alloc, [245](#)
 - AllowAdjFromChildren, [246](#)
 - AllowAdjs, [247](#)
 - CalcStatus, [249](#)
 - CalculateExchangeRate, [247](#)

CalculateRate, [248](#)
 Clear, [250](#)
 Con, [251](#)
 Consol1, Consol2, Consol3, [252](#)
 ConsolidateYTD, [253](#)
 ContainsCellText, [253](#)
 CreateInputCache, [254](#)
 Currency, [254](#)
 Custom1Top, Custom2Top, Custom3Top, and Custom4Top, [255](#)
 Decimal, [256](#)
 DefaultFreq, [257](#)
 DefaultTranslate, [257](#)
 DefaultView, [258](#)
 DefCurrency, [259](#)
 DOWn, [260](#)
 Dynamic, [261](#)
 Exp, [262](#)
 GetCell, [268](#)
 GetCellNoData, [269](#)
 GetCellRealData, [270](#)
 GetCellText, [271](#)
 GetCellType, [271](#)
 GetItem, [272](#)
 GetItemItemIDs2, [273](#)
 GetNumItems, [274](#)
 GetNumLID, [274](#)
 GetRate, [275](#)
 GetSubmissionGroup, [275](#)
 GetSubmissionPhase, [276](#)
 Holding, [276](#)
 ICPTopMember, [277](#)
 ICPWeight, [278](#)
 IDFromMember, [278](#)
 ImpactStatus, [279](#)
 Input, [280](#)
 IsAlmostEqual, [281](#)
 IsBase, [281](#)
 IsCalculated, [283](#)
 IsChild, [283](#)
 IsConsolidated, [285](#)
 IsDescendant, [285](#)
 IsFirst, [287](#)
 IsICP, [288](#)
 IsLast, [289](#)
 IsTransCur, [289](#)
 IsTransCurAdj, [290](#)
 IsValidDest, [290](#)
 IsZero, [291](#)
 List, [292](#)
 Member, [293](#)
 MemberFromID, [293](#)
 Method, [295](#)
 NoInput, [295](#)
 NoRound, [296](#)
 NumBase, [296](#)
 Number, [298](#)
 NumChild, [298](#)
 NumDescendant, [300](#)
 NumPerInGen, [302](#)
 NumPeriods, [302](#)
 OpenDataUnit, [303](#)
 OpenDataUnitSorted, [304](#)
 overview, [234](#)
 Owned, [304](#)
 Owner, [305](#)
 PCon, [305](#)
 PEPU, [306](#)
 PeriodNumber, [306](#)
 PlugAcct, [307](#)
 POwn, [308](#)
 PVAForBalance, [309](#)
 PVAForFlow, [309](#)
 RateForBalance, [310](#)
 RateForFlow, [311](#)
 ReviewStatus, [311](#)
 ReviewStatusUsingPhaseID, [312](#)
 Round, [313](#)
 Scale, [314](#)
 SecurityAsPartner, [315](#)
 SecurityClass, [316](#)
 SetData, [317](#)
 SetDataWithPOV, [317](#)
 SubmissionGroup, [318](#)
 SupportsProcessManagement, [319](#)
 SupportsTran, [319](#)
 SwitchSign, [320](#)
 SwitchType, [321](#)
 Trans, [322](#)
 TransPeriodic, [323](#)
 UD1...3, [324](#)
 ValidationAccount, [325](#)
 ValidationAccountEx, [325](#)
 XBRLTags, [326](#)

G

GetCell function, 268
 GetCellNoData function, 269
 GetCellRealData function, 270
 GetCellType function, 271
 GetCelText function, 271
 GetItem function, 272
 GetItemIDs2 function, 273
 GetNumItems function, 274
 GetNumLID function, 274
 GetRate function, 275
 GetSubmissionGroup function, 275
 GetSubmissionPhase function, 276

H

HDE files, defining, 191
 Holding function, 276
 HoldingCompany attribute, 82
 HS object, 219

I

ICP, 94
 ICP members
 editing, 95
 system generated, 94
 ICPEntitiesAggregationWeight attribute, 85
 ICPTopMember attribute, 76
 ICPTopMember function, 277
 ICPWeight function, 278
 IDFromMember function, 278
 ImpactStatus function, 279
 Input function, 280
 Instructions keyword, Web scripts, 155
 Intercompany dimension, 20
 intercompany matching reports
 selecting member lists, 194
 selecting style sheets, 195
 specifying currencies, 195
 specifying decimal places, 195
 suppression options, 195
 Intercompany Partner Matching reports, 194
 intercompany partners, setting up, 94
 intercompany transactions
 adding reason codes, 337
 alerting, 67
 closing periods, 335

 editing reason codes, 338
 locking entities, 336
 monitoring, 338
 opening periods for, 333
 removing reason codes, 338
 setting matching tolerances, 334
 unlocking entities, 336
 invalid characters in labels and descriptions, 21
 IsAlmostEqual function, 281
 IsBase function, 281
 IsCalculated attribute
 accounts, 77
 custom, 80
 IsCalculated function, 283
 IsChild function, 283
 IsConsolidated attribute, 77
 IsConsolidated function, 285
 IsDescendant function, 285
 IsFirst function, 287
 IsHoldingMethod attribute, 88
 IsICP attribute
 accounts, 77
 entities, 82
 IsICP function, 288
 IsLast function, 289
 IsTransCur function, 289
 IsTransCurAdj function, 290
 IsValidDest function, 290
 IsZero function, 291

J

JLF files
 creating, 129
 extracting in Web, 136
 extracting in Windows, 135
 loading in Web, 134
 loading in Windows, 134
 journal files (JLF)
 creating, 129
 extracting in Web, 136
 extracting in Windows, 135
 label requirements for, 21
 loading in Web, 134
 loading in Windows, 134
 journal reports
 creating, 193
 example, 193

L

label requirements, [21](#)

LAST keyword, Web scripts, [171](#)

liability account behavior, [78](#)

limitations for characters, [21](#)

LineItemDetailSinglePeriod keyword, Web scripts, [155](#)

Link keyword, Web scripts, [155](#)

List function, [292](#)

List View

- adding and modifying members, [111](#)
- copying, cutting, and pasting members, [112](#)
- deleting members, [112](#)
- showing and hiding columns, [113](#)
- sorting metadata, [113](#)
- validating metadata, [112](#)

loading

- data form scripts (WDF), [174](#)
- in Web
 - application security files (SEC), [70](#)
 - journal files (JLF), [134](#)
 - member list files (LST), [126](#)
 - metadata files (XML and APP), [118](#)
 - rules files (RLE), [228](#)
- in Windows
 - application security files (SEC), [68](#)
 - journal files (JLF), [134](#)
 - member list files (LST), [126](#)
 - metadata files (XML and APP), [116](#)
 - rules files (RLE), [227](#)
- rules in Rules Editor, [225](#)

logging off users on system, [50](#)

LST files

- creating, [121](#)
- extracting in Web, [127](#)
- extracting in Windows, [127](#)
- loading in Web, [126](#)
- loading in Windows, [126](#)

M

Match/Validate Before Post option, [335](#)

MaxCells keyword, Web scripts, [156](#)

MaxColsForSparseRetrievalMethod keyword, Web scripts, [156](#)

MaximumReviewLevel attribute, [83](#)

Member attribute

- accounts, [77](#)

- custom, [80](#)
- entities, [82](#)
- scenarios, [83](#)

Member function, [293](#)

member list files (LST)

- creating, [121](#)
- extracting in Web, [127](#)
- extracting in Windows, [127](#)
- loading in Web, [126](#)
- loading in Windows, [126](#)

member list files (LST), dynamic lists, [124](#)

MemberFromID function, [293](#)

metadata

- accounts example, [99](#)
- application settings example, [98](#)
- currencies example, [98](#)
- custom dimension example, [100](#)
- dimensions in metadata file, [99](#)
- dimensions not in the metadata file, [104](#)
- entities example, [100](#)
- file format, [106](#)
- members in metadata file, [99](#)
- scenarios example, [100](#)

metadata file

- account dimension hierarchies section, [103](#)
- consolidation methods section, [102](#)
- custom dimension hierarchies section, [104](#)
- entity dimension hierarchies section, [104](#)
- hierarchies section, [102](#)
- scenario dimension hierarchies section, [103](#)

metadata files

- extracting in Web, [120](#)
- extracting in Windows, [119](#)
- loading in Web, [118](#)
- loading in Windows, [116](#)

metadata files (APP or XML), creating, [96](#)

metadata integrity, [114](#)

Metadata Item Properties

- creating aliases for Custom dimensions, [113](#)

Metadata Manager

- File Properties, creating metadata reports, [114](#)
- Metadata Item Properties, creating aliases for Custom dimensions, [113](#)
- views defined, [105](#)

Metadata Manager List View

- adding and modifying members, [111](#)
- copying, cutting, and pasting members, [112](#)

- deleting members, [112](#)
- showing and hiding columns, [113](#)
- sorting metadata, [113](#)
- validating metadata, [112](#)
- Metadata Manager Tree View
 - adding and modifying members, [107](#)
 - adding members, [108](#)
 - adding orphaned members, [110](#)
 - deleting and removing members, [109](#)
 - expanding and collapsing the hierarchy, [111](#)
 - modifying node attributes, [108](#)
 - moving members, [108](#)
 - promoting members, [109](#)
 - removing orphaned members, [110](#)
 - repositioning members, [109](#)
- method account, [93](#)
- Method function, [295](#)
- moving periods, [31](#)

N

- NEXT keyword
 - Web scripts, [171](#)
- NodeSecurity attribute, [85](#)
- NoInput function, [295](#)
- NoRound function, [296](#)
- NoSuppress keyword, Web scripts, [156](#)
- NumBase function, [296](#)
- Number function, [298](#)
- NumChild function, [298](#)
- NumDecimalPlaces attribute, [77](#)
- NumDecimals keyword
 - Web scripts, [157](#)
- NumDescendant function, [300](#)
- NumPerInGen function, [302](#)
- NumPeriods function, [302](#)

O

- objects, HS, [219](#)
- OpenDataUnit function, [303](#)
- OpenDataUnitSorted function, [304](#)
- opening applications, [37](#)
- organization by period, [86](#)
- OrgByPeriodApplication attribute, [85](#)
- Override keyword
 - Web scripts, [157](#)
- Owned function, [304](#)

- Owner function, [305](#)
- ownership accounts
 - SharesOutstanding account, [93](#)
 - SharesOwned account, [93](#)
 - VotingOutstanding account, [93](#)
 - VotingOwned account, [93](#)

P

- PCON account, [92](#)
- PCon function, [305](#)
- PCTRL account, [92](#)
- PEPU function, [306](#)
- PER files, creating, [27](#)
- percent consolidation, [92](#)
- percent control, [92](#)
- percent direct ownership, [92](#)
- percent ownership, [92](#)
- PercentConsol attribute, [88](#)
- Period dimension, [19](#)
- Periodic frequency, [29](#)
- PeriodNumber function, [306](#)
- periods
 - copying, [31](#)
 - deleting, [32](#)
 - moving, [31](#)
- phased submissions for process management, [341](#)
- PlugAcct attribute, [77](#)
- PlugAcct function, [307](#)
- POWN account, [92](#)
- POwn function, [308](#)
- PrintNumColsPerPage keyword, Web scripts, [158](#)
- PrintNumRowsPerPage keyword, Web scripts, [159](#)
- PrintRepeatHeadersonAllPages keyword, Web scripts, [159](#)
- process management
 - alerting, [66](#)
 - assigning submission groups, [343](#)
 - defining submission phases, [341](#)
 - submission phase description, [341](#)
 - unassigned submission groups, [346](#)
- PVAForBalance function, [309](#)
- PVAForFlow function, [309](#)

R

- RateForBalance function, [310](#)
- RateForFlow function, [311](#)

ReadOnly keyword, Web scripts, [160](#)
 reason codes for intercompany transactions, [337](#)
 records, deleting invalid, [58](#)
 referential integrity, [114](#)
 registering server clusters, [35](#)
 report definitions
 Explore Data report, [192](#)
 Intercompany Partner Matching Reports, [194](#)
 Journal report, [193](#)
 ReportDescription keyword, Web scripts, [161](#)
 ReportLabel keyword, Web scripts, [161](#)
 reports
 Explore Data reports, [192](#)
 Intercompany Partner Matching reports, [194](#)
 Journal reports, [193](#)
 ReportSecurityClass keyword
 Web scripts, [162](#)
 ReportType keyword, Web scripts, [162](#)
 requirements for user-defined elements, [21](#)
 revenue account behavior, [78](#)
 ReviewStatus function, [311](#)
 ReviewStatusUsingPhaseID function, [312](#)
 RLE files
 creating, [225](#)
 extracting in Web, [229](#)
 extracting in Windows, [229](#)
 loading in Web, [228](#)
 loading in Windows, [227](#)
 Rn keyword, Web scripts, [159](#)
 Round function, [313](#)
 RPT files, defining, [191](#)
 rules
 account expressions, [215](#)
 loading from rules Editor, [225](#)
 types, [208](#)
 Rules Editor
 configuring, [223](#)
 creating rules, [223](#)
 rules files (RLE)
 creating, [225](#)
 extracting in Web, [229](#)
 extracting in Windows, [229](#)
 loading in Web, [228](#)
 loading in Windows, [227](#)
 running security reports, [68](#)

S
 saving application profiles, [32](#)
 SCalc keyword
 Web scripts, [162](#)
 Scale attribute, [91](#)
 Scale function, [314](#)
 Scale keyword
 Web scripts, [164](#)
 scenario attributes, [82](#)
 Scenario dimension, [19](#)
 Scenario View frequency, [29](#)
 scenarios example, [100](#)
 SEC files
 extracting in Web, [74](#)
 extracting in Windows, [73](#)
 loading in Web, [70](#)
 loading in Windows, [68](#)
 security classes
 assigning access, [65](#)
 security reports, [68](#)
 SecurityAsPartner attribute, [82](#)
 SecurityAsPartner function, [315](#)
 SecurityClass attribute
 accounts, [77](#)
 custom, [81](#)
 entities, [82](#)
 scenarios, [84](#)
 SecurityClass function, [316](#)
 SelectablePOVList keyword, Web scripts, [164](#)
 server clusters
 registering, [35](#)
 unregistering, [35](#)
 SetData function, [317](#)
 SetDataWithPOV function, [317](#)
 setting up, [94](#)
 setting up alerting
 intercompany transactions, [67](#)
 process management, [66](#)
 setting up ICP, [94](#)
 Shared Services console, [62](#)
 SharesOutstanding account, [93](#)
 SharesOwned account, [93](#)
 ShowDescription keyword, Web scripts, [165](#)
 ShowLabels keyword, Web scripts, [165](#)
 specifying
 frequencies in applications, [29](#)
 languages in applications, [27](#)

- periods in applications, 29
- star schema, 181
 - creating and exporting data to, 185
 - deleting, 188
 - updating, 187
- star schema template
 - creating, 188
 - deleting, 189
- String keyword, Web scripts, 166
- Style keyword, Web scripts, 166
- Submission Group function, 318
- submission phases for process management, 341
- SupportsProcessManagement attribute, 83, 84
- SupportsTran function, 319
- SuppressColHeaderRepeats keyword, Web scripts, 168
- SuppressInvalidColumns keyword
 - Web scripts, 168
- SuppressInvalidRows keyword
 - Web scripts, 169
- SuppressNoDataColumns keyword, Web scripts, 169
- SuppressNoDataRows keyword, Web scripts, 169
- SuppressRowHeaderRepeats keyword, Web scripts, 170
- SuppressZeroColumns keyword, Web scripts, 170
- SuppressZeroRows keyword, Web scripts, 170
- SwitchSign function, 320
- SwitchSignForFlow attribute, 81
- SwitchType function, 321
- SwitchTypeForFlow attribute, 81
- syntax
 - data form scripts, 144
- system accounts
 - active account, 92
 - Consol1...3 account, 93
 - DOWN account, 92
 - editing, 93
 - method account, 93
 - PCON account, 92
 - PCTRL account, 92
 - POWN account, 92
 - SharesOutstanding account, 93
 - SharesOwned account, 93
 - VotingOutstanding account, 93
 - VotingOwned account, 93

T

- task audit, 53
- ToPercentControl attribute, 88
- ToPercentControlComp attribute, 88
- Trans function, 322
- translation default, 218
- translation, default, 217
- TranslationOperator attribute, 92
- TransPeriodic function, 323
- Tree View
 - adding and modifying members, 107
 - adding members, 108
 - adding orphaned members, 110
 - deleting and removing members, 109
 - expanding and collapsing the hierarchy, 111
 - modifying node attributes, 108
 - moving members, 108
 - promoting members, 109
 - removing orphaned members, 110
 - repositioning members, 109

U

- UD1...3 function, 324
- UDL file
 - configuring for IBM DB2, 179
 - configuring for Oracle, 178
 - configuring for SQL Server, 179
 - encrypting, 180
- updating a star schema, 187
- UsedByCalcRoutine attribute, 88
- UsePVAForBalanceAccounts attribute, 86
- UsePVAForFlowAccounts attribute, 86
- user and group assignments, 63
- user-defined element requirements, 21
- UserDefined1...3 attribute
 - custom, 81
 - entities, 82
 - scenarios, 84
- users on system
 - disabling connections, 51
 - enabling connections, 51
 - logging off users, 50
 - viewing users, 49
- UseSecurityForAccounts attribute, 86
- UseSecurityForCustom attribute, 86
- UseSecurityForEntities attribute, 86
- UseSecurityForICP attribute, 86

UseSecurityForScenarios attribute, [86](#)
 UsesLineItems attribute
 scenarios, [84](#)

V

ValidationAccount attribute, [86](#)
 ValidationAccount function, [325](#)
 ValidationAccountEx function, [325](#)
 Value dimension, [20](#)
 value members
 editing, [95](#)
 system generated, [94](#)
 View dimension, [20](#)
 viewing users on system, [49](#)
 views, corresponding frequencies and, [29](#)
 VotingOutstanding account, [93](#)
 VotingOwned account, [93](#)

W

WDF files
 creating, [144](#)
 extracting, [175](#)
 loading, [174](#)
 Web form builder
 Columns tab, [142](#)
 Details tab, [143](#)
 POV tab, [141](#)
 Properties tab, [140](#)
 Rows tab, [142](#)
 Web scripts
 AddMember keyword, [146](#)
 BackgroundPOV keyword, [147](#)
 Blank keyword, [148](#)
 Calc1 keyword, [150](#)
 CalcByRow keyword, [150](#)
 Cell_Link keyword, [150](#)
 CellText keyword, [151](#)
 Cn keyword, [149](#)
 CUR, FIRST, LAST, NEXT keywords, [171](#)
 CustomHeader keyword, [151](#)
 CustomHeaderStyle keyword, [152](#)
 DynamicPOV keyword, [152](#)
 FormInputBoxLength keyword, [152](#)
 FormNumDecimals keyword, [153](#)
 FormRowHeight keyword, [153](#)
 FormScale keyword, [154](#)

Instructions keyword, [155](#)
 LineItemDetailSinglePeriod keyword, [155](#)
 Link keyword, [155](#)
 MaxCells keyword, [156](#)
 MaxColsForSparseRetrievalMethod keyword, [156](#)
 NoSuppress keyword, [156](#)
 NumDecimals keyword, [157](#)
 Override keyword, [157](#)
 PrintNumColsPerPage keyword, [158](#)
 PrintNumRowsPerPage keyword, [159](#)
 PrintRepeatHeadersonAllPages keyword, [159](#)
 ReadOnly keyword, [160](#)
 ReportDescription keyword, [161](#)
 ReportLabel keyword, [161](#)
 ReportSecurityClass keyword, [162](#)
 ReportType keyword, [162](#)
 Rn keyword, [159](#)
 SCalc keyword, [162](#)
 Scale keyword, [164](#)
 SelectablePOVList keyword, [164](#)
 ShowDescription keyword, [165](#)
 ShowLabels keyword, [165](#)
 String keyword, [166](#)
 Style keyword, [166](#)
 SuppressColHeaderRepeats, [168](#)
 SuppressInvalidColumns keyword, [168](#)
 SuppressInvalidRows keyword, [169](#)
 SuppressNoDataColumns keyword, [169](#)
 SuppressNoDataRows keyword, [169](#)
 SuppressRowHeaderRepeats keyword, [170](#)
 SuppressZeroColumns keyword, [170](#)
 SuppressZeroRows keyword, [170](#)
 Workspace, [18](#)

X

XBRLTags attribute, [78](#)
 XBRLTags function, [326](#)
 XML to APP, [106](#)

Y

Year dimension, [19](#)

Z

ZeroViewForAdj attribute, [84](#)
 ZeroViewForNonadj attribute, [84](#)

A B C D E F G H I J L M N O P R S T U V W X Y Z