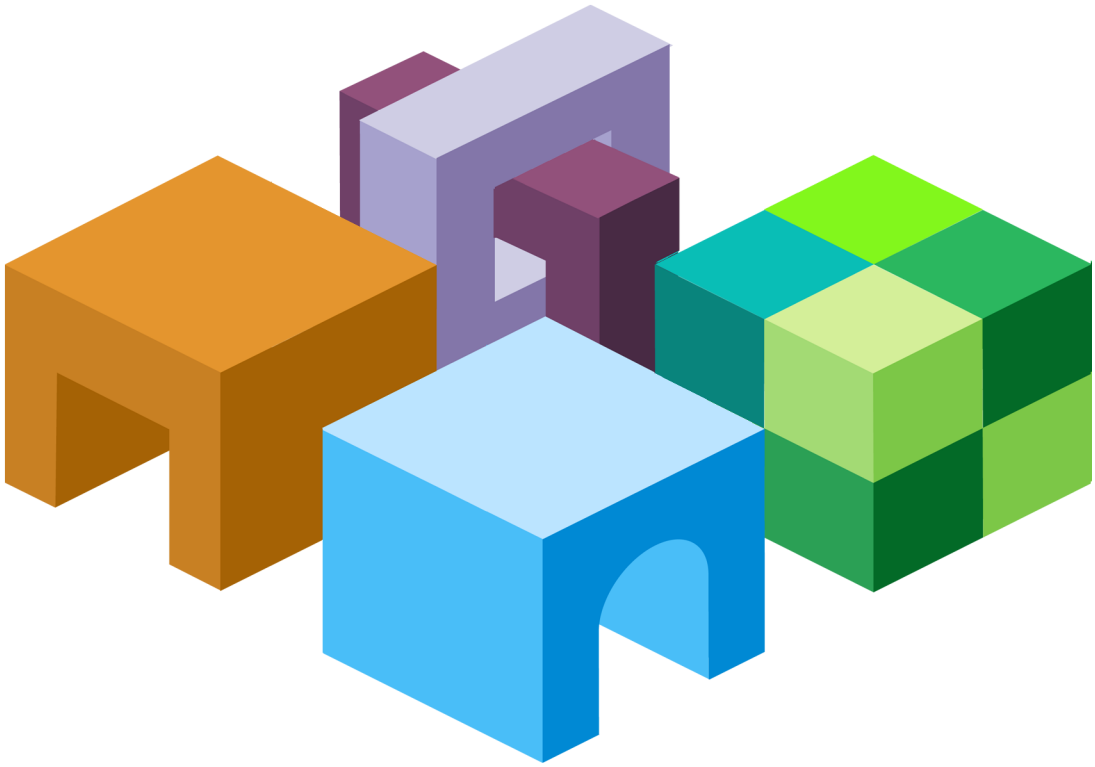


HYPERION® FINANCIAL DATA QUALITY MANAGEMENT

RELEASE 9.3.1

ADMINISTRATOR'S GUIDE

ORACLE | Hyperion

FDM Administrator's Guide, 9.3.1

Authors: John Billinger

Contents

1 Basics of FDM

In This Chapter

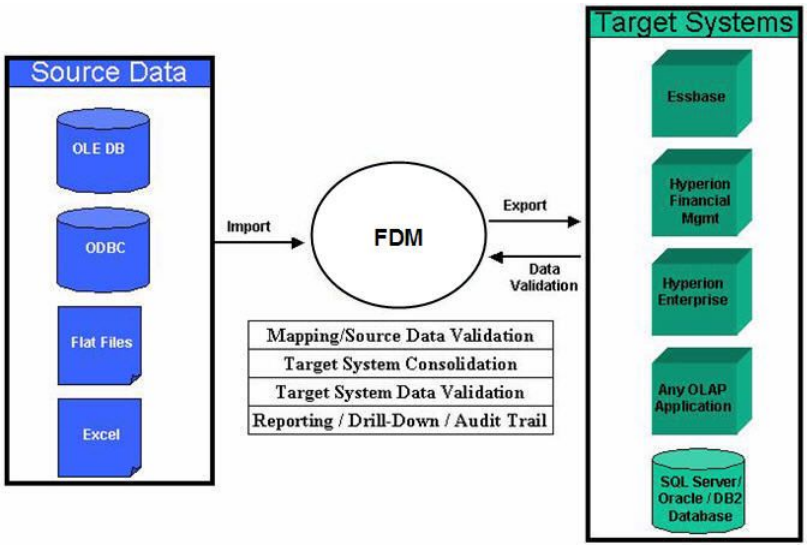
About FDM

Oracle's Hyperion® Financial Data Quality Management (FDM) is an out-of-the-box data transformation tool that feeds source-level financial data to consolidation, reporting, planning, and analytical applications. More than just a data translation tool, it provides an audit trail to the source financial data, ensuring data integrity and mapping consistency that provides easy reconciliation of financial data. Providing an audit trail to the source financial data saves end-users, administrators, and auditors time associated with data error investigation, identification, and correction.

FDM provides a uniform data collection process for all reporting units within the organization. FDM also contains financial controls functionality that assists with internal financial controls processes. FDM features adapters to easily integrate it with Oracle's Hyperion® Enterprise®, Oracle's Hyperion® Financial Management – System 9 (Financial Management), Oracle's Hyperion® Essbase® – System 9 (Essbase), and Oracle's Hyperion® Planning – System 9 (Planning), but it can be used to load data into other financial consolidation and analytical applications.

Six steps comprise the basic FDM process:

1. Import source data
2. Validate source data against mapping tables
3. Export source data to target system
4. Consolidate target system data
5. Validate target system data
6. Review and validate internal financial control



Key Features and Benefits

FDM provides a single, systematic process for loading source data from disparate systems into target analytical applications. The process provides data visibility, integrity and verification.

Features and benefits include the following:

- Acting central repository of all source financial data
- Drill-down audit trail
- Ability to archive source files, error logs, and load files
- Internal controls assessment and certification feature aids in compliance with sections 302 and 404 of the Sarbanes-Oxley Act
- Corporate-wide process monitoring
- Ability to import source data from any formatted text file or data source
- Multiple dimension mapping and validating capability
- Data validation and quality checking
- Error identification and notification
- Consolidation of target system data
- Validation and reporting on target system data
- Load adjustments capability through Excel journals
- Budget data loading for multiple periods
- Advanced reporting and audit functions
- “Lights out” batch loading
- Support for unlimited concurrent users
- Zero footprint Web deployment
- SQL and Oracle database support

Product Set

FDM includes two main products, a Windows client (Workbench) and a Web client.

Workbench

FDM Workbench is a Windows client that can be installed on a PC or terminal server. Workbench provides functionality to application setup, integration, and development features in FDM and should be used only by administrators. FDM Workbench basic functionality includes the following:

- Systems integration
- Script creation and maintenance
- Report creation and maintenance
- Application importing and exporting
- Batch processor setup and monitoring

Web Client

The FDM web client consists of four components:

- Web Server
- Application Server
- Load Balance Manager
- Task Manager

Web Server

The FDM Web Server components can be installed on a Web server running IIS 6.0 or higher. These components enable users to access FDM applications from the Internet or corporate intranet. The Web-based interface provides functionality to all end-user features and most administrator features of FDM.

Application Server

The FDM Application Server enables FDM to execute resource-intensive tasks on one or more application servers instead of the Web server. To minimize network traffic and increase data transfer rates, installing the Application Server component on a different server other than the Web and data (SQL or Oracle) servers is recommended.

Load Balance Manager

The FDMLoad Balance Manager enables FDM to manage the application servers that execute resource-intensive tasks. Installing Load Balance Manager on the first application server is recommended.

Task Manager

The FDM Task Manager runs scheduled tasks (FDM scripts) at specified intervals. You can also use Task Manager to add, modify, and delete tasks. It can be installed on any server that has access to the FDM application. You may schedule tasks to run repeatedly, daily, weekly, or monthly.

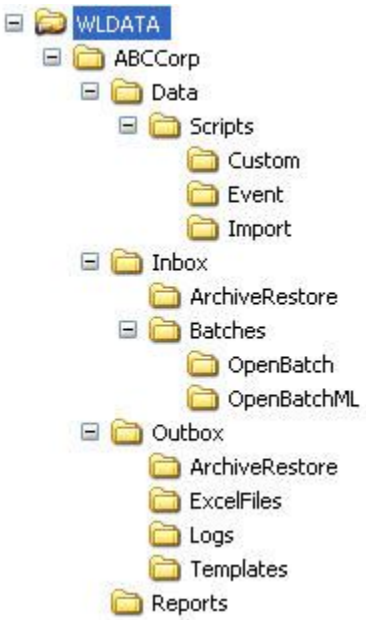
Definition of a FDM Application

A FDM application consists of a Relational Database Management System (RDBMS) database and directories that contain all the transactional data, metadata, reports, and other files that are used to integrate FDM with target applications.

One FDM application can load to multiple target applications of different systems. For example, one FDM application can load to two different Hyperion Enterprise applications, a Financial Management application, and three different Oracle's Hyperion® Planning – System 9 applications.

FDM Application Architecture

The following diagram shows the directories that are created when a new FDM application is created. In this example, WLDATA is a user-created directory used to store all FDM applications. When a new application was created and named “ABC Corp,” the directory named ABCCorp and all subdirectories were created automatically by FDM. Each application contains four main directories: Data, Inbox, Outbox, and Reports.



Data

`Data` is where FDM archives data files imported to and exported from FDM including imported source files, import logs, journal entries, multiload files, target system load files, and any attached memo documents. Each file in the `Data` directory is assigned a unique name, can be retrieved from the Import and Export screens, and provides a complete audit trail. See "Data Archiving" on page 130 for additional information.

Scripts

The `Data` directory contains a `Scripts` subdirectory where FDM scripts are stored. Source files, journals, multiload files, logs, output data files, and attachments are also archived here.`Scripts` contains subdirectories for each type of script.

Custom—Contains scripts that can be executed through a custom menu created in the Menu Maker screen.

Event—Contains scripts that are executed when a particular event runs. All scripts contained in this directory are named for the event with which they correspond.

Import—Contains import scripts that are created when import formats are defined. Import scripts are executed during the source file import process.

Inbox

This is the default directory from which to import source files. It can be used as a central repository for all ledger extract files. Because source files can be retrieved from any accessible directory, you are not required to place import files in this directory. `Inbox` includes two subdirectories: `Archive Restore` and `Batches`.

Archive Restore

This directory is used to store restored import source files and logs that were previously archived. FDM stores the original archived source files and logs in the `Data\Archive` directory. See "Validation Entity Groups" on page 127 and "Data Archiving" on page 130 for additional information.

Batches

Files used for batch loading are stored in `Batches`. Standard batch files loaded using Batch Loader must be placed in the `OpenBatch` directory. Multiload batch files loaded using Batch Loader must be placed in the `OpenBatchML` directory.

Outbox

This directory provides a central location for all FDM export files that are subsequently loaded to target systems. `Outbox` also contains four subdirectories; `Excel Files`, `Logs`, `Templates`, and `ArchiveRestore`.

Excel Files

When FDM exports the contents of a grid, the resulting Excel files are stored here.

Logs

`Logs` primarily contains log files created when source files are imported. These logs contain the data lines that FDM was not able to import as well as a description of why the data line was not imported. `Logs` also contains error logs which are named per the following convention (`<username>.err`), where <username> is the user that is logged into FDM, and `.err` is the common extension used by FDM to identify error logs. This directory can be purged to reclaim disk space.

Templates

The `Templates` directory is used by reporting locations to manage and distribute custom journal or Multiload templates. The contents of this directory are displayed as links within the FDM Web client. Publish a template by placing it in this directory.

Archive Restore

This directory stores restored data load files that were previously archived. FDM stores the original archived data load files in the `Data` directory. See "Data Archiving" on page 130 for additional information. The contents of this directory are deleted when a compact is performed.

Reports Directory

`Reports` stores the Active Report files. Active Report files use a `*.rpx` extension.

Setting Up the Server Load Balance Group

Before accessing Workbench, you must assign a load balance server group. The server group defines the load balance servers that are used by FDM. The Load Balance Server Group Setup screen opens automatically when Workbench is launched for the first time.

➤ To set up server load balance groups:

1 Open the Workbench Logon screen by selecting **Start > Programs > Hyperion > Financial Data Quality Management > Workbench > Workbench Client**.

The Load Balance Server Group dialog box is displayed (if this is the first time you have opened Workbench).

2 Click **Add**.

The Load Balance Server Group form is displayed.

3 Enter a server group name and description.

4 Enter the names of the primary load balance server and backup load balance server (optional).

Browse to find available servers by clicking  .

5 Click **Connect**.

You can define multiple load balance server groups using the Load Balance Server Group form. This enables Workbench to attach to multiple FDM applications on different servers.

Note:

If a load balance server group has already been defined and you want to add or modify a server group, you can access the Load Balance Server Group form by logging into Workbench and selecting File > Load Balance Server Group.

Adapters

Adapters are software codes that communicate with various source and target applications. Each adapter is designed to integrate with a specific target system (for example, Oracle's Hyperion® Essbase® – System 9 or Financial Management) or source system (SAP, SQL, and so on). Workbench enables you to import and maintain the adapters used by the FDM application.

Adapters are comprised of an XML metadata component and a DLL/EXE (target adapters only). The DLL/EXE contains the system-specific instructions for interacting with the target system. The XML contains all data relating to the FDM application.

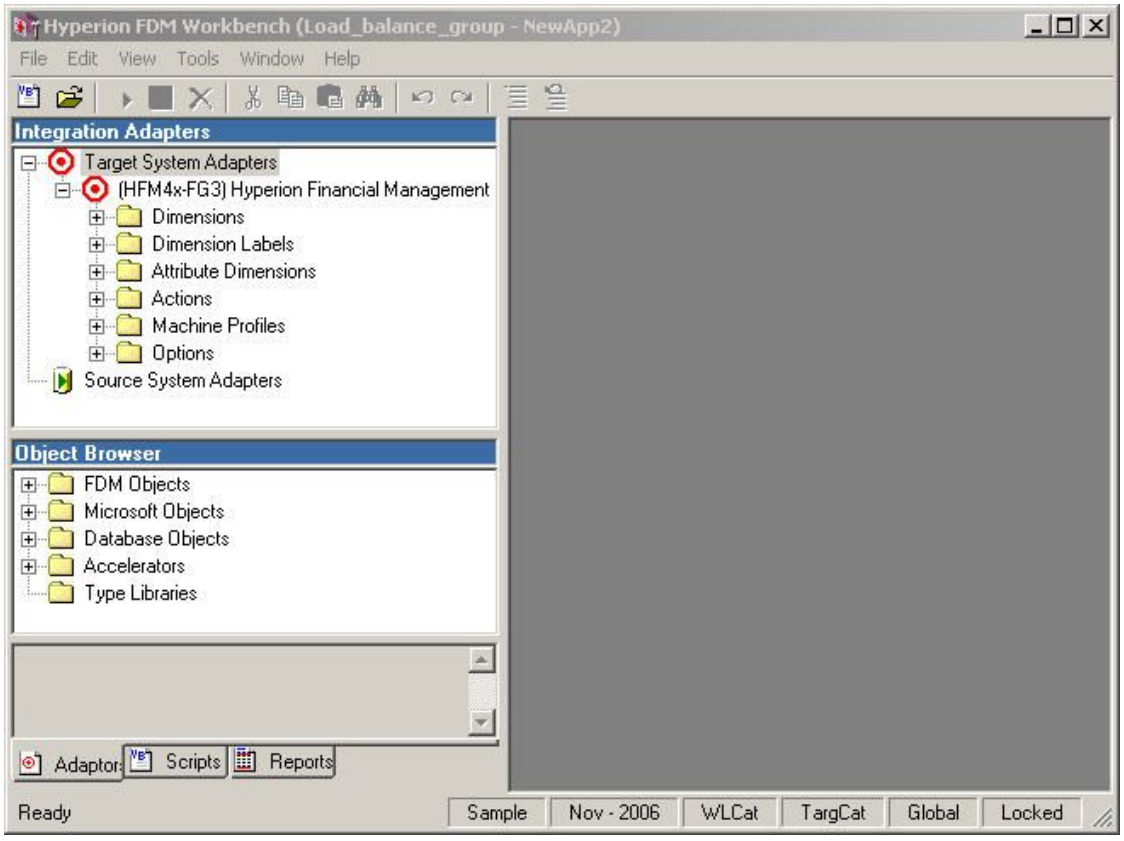
Adapter DLL/EXE

The target system adapter `DLL` or `EXE` acts as a buffer between FDM and the target application. The `DLL` contains all of the instructions for FDM to communicate with the specified target application, such as connecting to the database, loading data, extracting data, and so on. Each target system adapter `DLL` contains the API calls for the specific target application. This enables FDM to integrate with many target systems without having to maintain large amounts of application-specific calls within the program.

Adapter XML

The adapter `XML` acts as a second layer between FDM and the target application. The `XML` stores application setup parameters and options, and scripts that call API functions, and returns the results from the target application. The `XML` file may also contain the information required to configure a FDM application (locations, security, import formats, reports, and so on).

Access the Adapters menu from the Workbench desktop by clicking the Adapters tab at the bottom of the left pane.



Each adapter listed in the Adapters screen contains the following sections:

- Dimensions
- Dimension Labels
- Attribute Dimensions
- Actions
- Machine Profiles
- Options

Dimensions

The Dimensions section contains all of the available dimension lists for the target application. Each item under the main Dimension section represents one available dimension and contains a script that retrieves a list of all members in the selected dimension from the target application. Only target adapters employ the Dimensions section. Source system adapters do not contain the dimension section.

Actions

These are scripts that interact with the target application and use the adapter DLL API calls to perform such functions as `Connect`, `Load`, `ValueGet`, `Export`, Drill Down, and so on.

Machine Profile

This section contains information that enables FDM to determine on what computers or servers the source and target databases are located. The machine profile also stores global logon information for connecting to target applications. When the Global Login option is selected, FDM will always use this username and password to log on to FDM and the target application. Therefore, the user account must have access to the server where the target application resides.

For FDM to load and retrieve values to and from the target application, the user account must have the appropriate security privileges in the target application regardless of the user who is logged into FDM. You must configure a machine profile for every computer that has the FDM Application Server or Task Manager component installed.

Options

This section of the `XML` file contains the integration setting defaults and application settings for the target application with which FDM is integrating.

2 Managing Applications

In This Chapter

Working with the Application Screen

Applications are managed by using the New Application screen and Open Application Screen. From these screens, you can add, modify, and remove applications.

➤ To create FDM applications:

1 Access the FDM **Logon** page by launching the Web client or Workbench:

- Web Client—Select Start > Hyperion > Financial Data Quality Management > Web Server Components > Web Logon.
- Workbench—Select Start > Hyperion > Financial Data Quality Management > Workbench > Workbench Client.

2 From **Application**, select **<New Application>**.

3 Enter a name and password of the administrator of the new application.

4 Click **Logon** (Web client) or **OK** (Workbench).

The New Application dialog box is displayed and features two tabs–General and Database.

5 Enter the appropriate information on the **General** tab.

The General tab is used to create the application name and description, and defines where the application architecture is stored.

- Name—Name of the application. May contain up to twenty alphanumeric characters. Do not use spaces. Underscores (_) are acceptable.
- Description—Enter a description of the application.
- Path—Define the file path where the new application is created. The application consists of the directory that includes the `Inbox`, `Outbox`, `Data`, and `Reports` directories. If users access the application through multiple Web and application servers, it is recommended to use the UNC naming convention to avoid problems with inconsistent drive letter mapping.

When the new application is created, a new directory of the same name is created. The application name is also added to the `HyperionFDMApplications.xml` file, which is stored in the FDM installation directory.

6 Select the **Database** tab and input the required information.

The Database tab is used to define the properties of the RDBMS database that stores the FDM transaction data.

- OLE DB Provider—Specify the database used by the application. The default database is SQLOLEDB. Oracle is also supported.
- Database Server—The location of the database used by the FDM application. This option is visible only when the OLE DB Provider is set to SQLOLEDB.
- Database Name—The database name (typically the same as the application name). This option is only available when OLE DB Provider is set to SQLOLEDB.
- Service—Enter the Oracle service used for connecting to the Oracle database. This option is only available when the OLE DB Provider is set to ORAOLEBD.
- Username—Enter the database administrator ID here. For example, the SQLOLEDB administrator ID.
- Password—Enter the database administrator password here.
- Options—Used to override of the default table space where the application is created. This option is available when the OLE DB Provider is set to ORAOLEB.

7 Click **OK**.

Note:

If override table spaces are not specified, then all tablespaces default to "Users." This can severely degrade performance. See the *DBA Guide* for detailed tuning instructions prior to creating the FDM application.

➤ To remove applications:

1 Access the FDM **Logon** page by launching the Web client or Workbench:

- Web Client—Select Start > Hyperion > Financial Data Quality Management > Web Server Components > Web Logon.
- Workbench—Select Start > Hyperion > Financial Data Quality Management > Workbench > Workbench Client.

2 From **Applications**, select **<Add Application>**.

3 Enter the user name and password for the application.

4 Click **Logon** (Web client) or **OK** (Workbench).

The Add Application screen is displayed.

5 Select an application.

6 Click **Remove**.

7 Click **OK**.

Note:

Removing an application does not delete or impact the data in the application. This task only deletes the current user's pointer to the application from the application's XML file. The directory that contains all the information remains intact.

➤ To modify applications:

1 Access the FDM **Logon** page by launching the Web client or Workbench:

- Web Client—Select Start > Programs > Hyperion > Financial Data Quality Management > Web Server Components> Web Logon.
- Workbench—Select Start > Programs > Hyperion > Financial Data Quality Management > Workbench > Workbench Client.

2 From **Application**, select **<Add Application>**.

3 Enter the administrator user name and password.

4 Click **OK**.

The Open Application dialog box is displayed.

5 Select an application.

6 Click **Modify**.

The Modify Application dialog box is displayed.

7 In **Modify Application**, change the settings as desired.

The dialog box provides a view of all settings for the selected application and enables you to modify the settings. You can change any application attributes except the application name. To change the name of the application you must remove it and then add it with a new name using the Add Application feature.

8 Click **OK**.

Note:

The procedure for creating a new application is used when the application does not exist. The procedure for adding an application is used when the application exists, but there is no pointer to the application in the application's `XML` file.

➤ To add FDM applications:

1 Access the FDM **Logon** page by launching the Web client or Workbench:

- Web Client—Select Start > Hyperion > Financial Data Quality Management > Web Server Components > Web Logon.
- Workbench—Select Start > Hyperion > Financial Data Quality Management > Workbench > Workbench Client.

2 From **Application**, select **<Add Application>**.

3 Enter the username and password for the application.

Note:

The domain name may be required for certain configurations.

4 Click **Logon**.

The Applications screen is displayed and lists all applications that have been defined on the computer.

5 Click **Add**.

The Add Application screen is displayed and is composed of two tabs—General and Database—that contain information about the name and location of an application.

6 On the **General** tab, enter the appropriate information.

- Name—The name of the application. Do not use spaces. Underscores (_) are acceptable.
- Description—Enter a description of the application.
- Path—Specify the file path where the application resides.

7 Select the **Database** tab and enter the appropriate information.

The Database tab is used to name the RDBMS database that stores all of the FDM transaction data.

- OLE DB Provider—Defines the database used by the application. The default database is SQLOLEDB (SQL). Oracle is also supported.
- Database Server—The location of the database used by the application. (only available when SQLOLEDB is selected in OLE DB Provider).
- Database Name—Typically the same as the application name. The database name you define will be the name of the RDBMS database (only available when SQLOLEDB is selected in OLE DB Provider).
- Service—Service used for connecting to the Oracle database (only available when ORAOLEDB (Oracle) is selected in OLE DB Provider).
- Username—The database administrator ID. For example, the SQLOLEDB administrator ID.
- Password—The database administrator password.
- Options—Overrides of the default table space where the application is created (only available when ORAOLEDB is selected in OLE DB Provider).

8 Click **OK**.

The `.ini` file and registry settings for the application that is referenced are created.

Logging into an Application for the First Time

The FDM login screen enables users to log in to applications, create new applications, or add existing applications. The user name that you used to create the application is the only user name (user ID) that is valid until the application has been configured to accept other users.

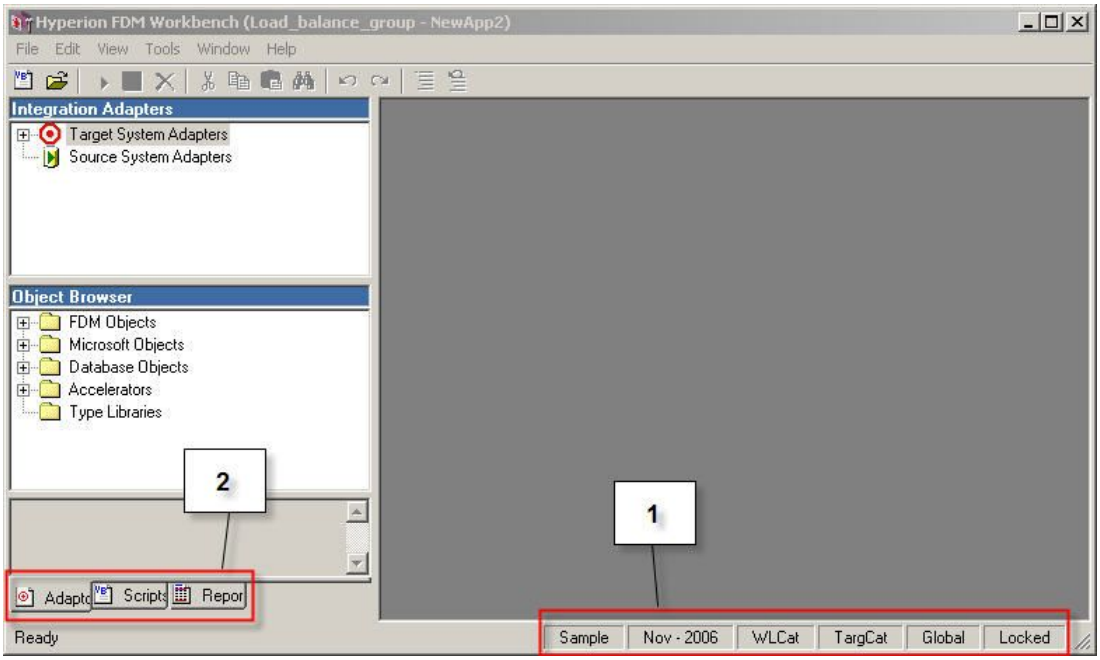
When logging on to a new application for the first time, you are prompted to set up a new FDM integration. See the *FDM Installation Guide* for information regarding integration adapter setup, and Chapter 4, "Configuring System Settings" for information regarding configuration settings.

3 Navigating FDM

In This Chapter

Functions in Workbench

The following diagram depicts the Workbench desktop.



1. Point-of-View (POV) Bar—Use this to select the FDM location, FDM period, FDM category, target system category, global or local mode, and system lock.
2. Screen Selector Tabs—Click to display the Adapters screen, Scripts screen, and Reports screen.

Command Buttons

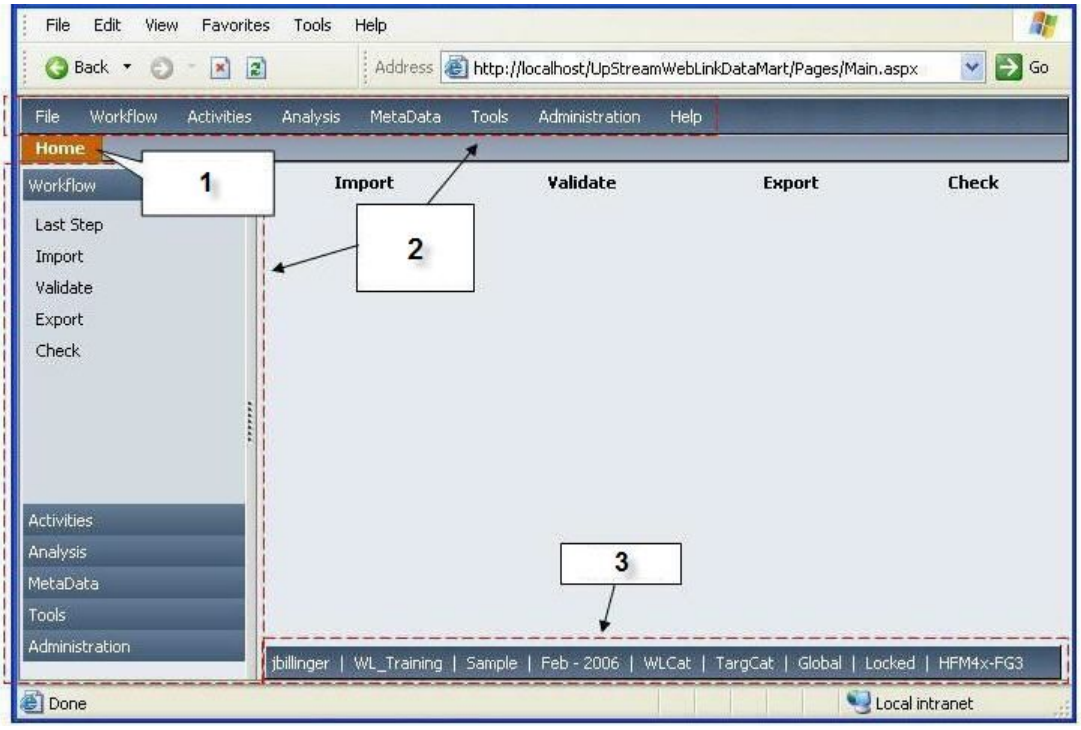
The following table lists the keyboard shortcuts that access various components of the FDM user interface.

Function	Key Combination	Action
Open Forms/Dialogs	Alt+F4	Close Program
	Ctrl + O	Open the Application Login form
	Ctrl + N	Open a new script
Help	F1	FDM Help

Functions in Web Client

Desktop

The FDM desktop is the control center where an application is built and administered. After logging in to an application, the desktop is displayed. Three areas to note on the desktop are Screen Indicator, FDM Menus, and the POV bar.



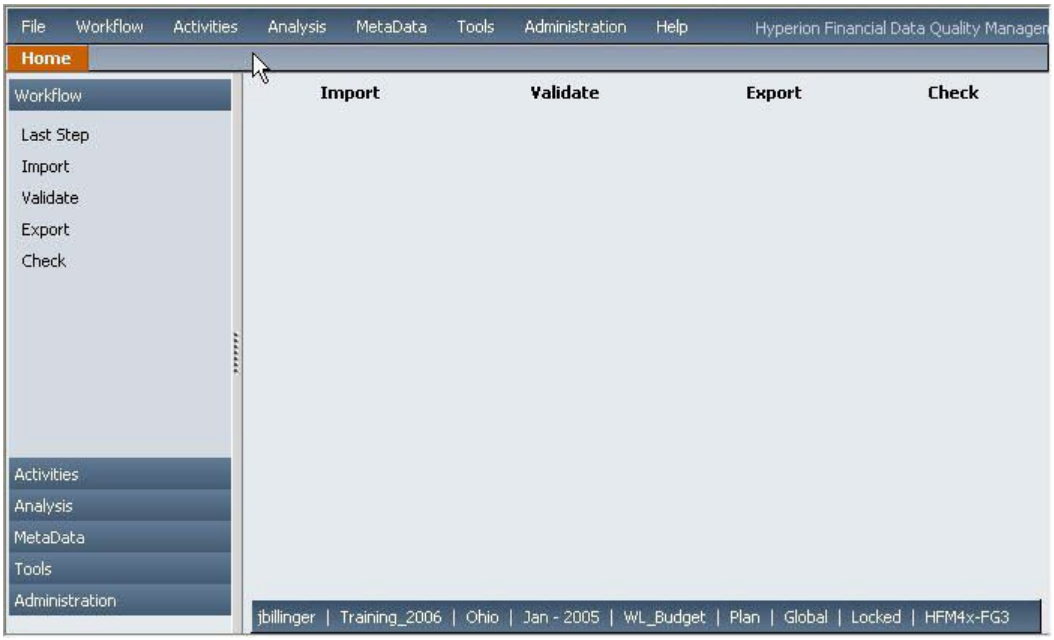
1. Screen Indicator—Displays the current FDM screen.
2. FDM Menu Rollup—Contains six tabs. Select a tab to expose the underlying menu. All FDM functions available in the Web client can be accessed from these menus. The example figure

shows that the Workflow tab is selected and the Workflow menu is displayed. You can also access all functions from the menu bar at the top of the FDM desktop.

3. POV Bar (Point of View)—Displays the current user, FDM application, FDM location, FDM period, FDM category, target category, mode (global or local), system lock status, and the adapter used by the current location.

Workflow Menu

The Workflow menu enables users to quickly open any workflow task without having to reprocess the workflow steps. Users can open the process step and then decide if they need to re-validate or export. There are five steps in the workflow process.



- Last Step—Opens the last Workflow Process screen that was run for the current POV (whether successful or not). If a user has attempted to validate data and failed, clicking on Last Step displays the Validate screen.
- Import—Displays the Import screen.
- Validate—Displays the Validate screen (even if the Import process has not been run for the current POV) but does not validate the data.
- Export—Displays the Export screen (even if the current POV has not validated its data) but does not initiate the Export process.
- Check—Displays the Check report for the current POV. If there is no check report data for the current POV, a blank page is displayed.

Working with Data in Grids

Most forms (screens) in FDM display data in one or more grids. These are the basic grid operations used to manipulate data:

- Add Record—To add a new record, click Add.
- Delete—To delete the selected record, click Delete.
- Delete All—To delete all records in a grid, click Delete All.

Note:

If the form contains more than one grid, you cannot delete the contents of a grid.

- Edit Record—To edit a record, click in the cell and start typing.
- Cancel Edit—To cancel all changes made to a row, click to the left of the cell and select Cancel Changes from the context menu.
- Save Record—To save a record, click Update Grid. This sends all changes on the current grid to the database.
- Update Grid—Click Update Grid to send updated rows (adds, deletes, and changes) to the database.
- Excel—Click Excel to output the contents of a grid to an Excel spreadsheet. The Excel spreadsheet is saved to the `Excel` subdirectory inside the `Outbox` directory specified in the Connections Dialog form.

Note:

Screens exported to Excel are configured for re-import (where available) into FDM. The named range is automatically defined, Import table name set, and column names set.

Sorting Columns

To sort a column in the grid, click a column heading to activate the column, then click the up-down arrow to the left of the column description. Click the up-down arrow again to re-sort the column the opposite way.

Note:

If a form contains more than one grid, you cannot sort a column in a grid.

Searching

➤ To search for a record:

1. Click a column heading in the grid.
2. Click the Search button ( ).
3. In the **Find** text box, enter the search criteria.
4. Click the Find Next button ( ) to cycle through all records that match the specified criteria.

Filtering

➤ To filter the records in a grid:

1 Click a column heading in the grid.

2 Click the filter button ( ).

The Filter dialog box is displayed.

3 From **Filter Method**, select an operator.

4 In **Filter Value**, enter a value by which to filter.

To remove filtering on the grid, select All.

Note:

If the form contains more than one grid, you cannot use the filter function.

Searching Control Trees

Screens that include control trees, such as the Locations screen, include a search function.

➤ To search a control tree:

1 Right-click on a node and select **Search**.

The Search dialog box is displayed.

2 In the **Search** dialog box, enter the search criteria.

The search criteria can be an exact phrase or part of a location name. The input is not case sensitive.

3 Click **OK**.

Customizing the Interface

You can customize the Web interface on a global application level or an individual user level.

Web Settings

The Web Settings screen enables you to control status bar icons, themes, number of grid rows to display per page, and the operation of the information bar. Access Web settings by selecting Administration > Web Settings. The Web settings screen features five tabs:

- General
- Grid
- Theme
- Info Bar

- Cache

General Tab

Allow Status Bar Icon Override—Enables users to override the default status bar icons. The status bar icons are the pass or fail icons displayed in the Workflow area under Import, Validate, Export and Check. When enabled, users can set individual default icons.

Default Status Bar Icons—Status bar icons displayed under the Workflow processes (Import > Check). The selected icon group is used unless a user has selected an override group. The available groups are stored on the Web server under`\Hyperion\ FDM\ WebServerComponent\ WebSite\ StatusBarIcons`.

To create additional status bar icon groups, create a directory under in the`StatusBarIcons` directory and place the additional icons in this directory.

Note:

The icons must have the same name as the icons in the existing directories.

Allow Default Report Publishing Type Override—Enables users to override the default report publishing type. The default report publishing type is the format used for reports that are displayed without allowing user selection of Type. These include the Check report and Process Explorer reports. When enabled, users can set an individual default report publishing type.

Default Report Publishing Type—A report type used when the Check report and Process Explorer reports are run. The Publish-type list on the Reports page is set to this value by default. Selections for this field are PDF, Excel, Word, Rich Text, and HTML.

Generate Debugging Information—Enable the info bar to display stack trace information if an error occurs. Stack trace information is helpful if you need to contact Hyperion Technical support for troubleshooting assistance.

Note:

This option should only be enabled if an error occurs and Hyperion Technical Support requires additional information.

Grid Tab

Allow Grid Override—Enables users to override the default number of grid rows displayed on a page. The default is 100. When enabled, users can set the number of rows displayed on a page on their screens (any number up to the Maximum Grid Page Size; set by the administrator).

Default Grid Page Size—Number of grid rows displayed on a page unless a user has specified an override value.

Maximum Grid Page Size—Maximum number of grid rows that users can specify when overriding the default number of grid rows per page. To avoid performance degradation, the maximum size for this field is 500.

Theme Tab

Allow Theme Override—Enables users to override the FDM default color scheme. The default theme contains all of the colors, styles and general-use icons that are displayed on the Web interface. If this option is enabled, users can select from the list of available themes in the Default Theme list on the User Settings page.

Default Theme—Default color scheme used on the FDM interface for this application. The selected theme is used unless the individual user has selected an override group. The groups available are stored on the Web Server under `\\Hyperion\ FDM\ WebServerComponent\ WebSite\ Themes`.

You can create additional themes by adding a directory in the `Themes` directory and placing the additional icons and style files in this directory.

Note:

The icons must have the same name as the icons in the existing directory.

Info Bar Tab

Only Display Errors and Script Messages—Disables confirmation messages displayed in the FDM Info Bar (Update Grid button clicks, records updated, rows deleted and so on). If this option is enabled, FDM only displays script-generated messages to the Info Bar.

Info Bar Timer—Sets the amount of time the Info Bar is displayed before hiding. After hiding, the Info Bar is accessed by clicking on the Information icon ().

Cache Tab

Clear Web Cache—Clears the Web server memory of all `XML` values stored for the table editor. This does not delete any files from the Web server, but only the cached view of these files.

Note:

The FDM session must be restarted for user interface changes to take effect. This applies to all Web sessions.

User Settings

Access User Settings by selecting Tools > User Settings. These settings allow each user to control themes, status bar icons, number of grid rows to display per page, and their default report publishing type using four options:

- Themes
- Status Bar
- Grid Page Size
- Default Report Publishing Type

Themes—Enables the user to select their own default theme. The default theme contains all of the colors, styles and general-use icons that are displayed on the Web Interface. Users can select a theme from the Themes list.

Status Bar Icons—Enables users to select their own default status bar icons. The status bar icons are the pass/fail icons that are displayed in the workflow area under Import, Validate, Export and Check. Users can select a status bar group from the Status Bar Groups list.

Grid Page Size—Enables users to set the default number of grid rows that are displayed per page. The maximum number of rows that a user can enter is the Maximum Grid Page Size set on the administrator Web Settings page.

Default Report Publishing Type—Enables users to select their own default report type used when the Check report and Process Explorer reports are run. By default, the Publish Type field on the Reports page is set to this value. Selections for this field are PDF, Excel, Word, Rich Text, and HTML.

Note:

The FDM session must be restarted for the changes to take effect. These options can only be set if the administrator enables overrides for the selected option.

Locking and Unlocking the POV

The Point-of-View Lock Settings consist of the following options:

- Lock Current Point-of-View
- Unlock Current Point-of-View
- Lock All Locations (Current Category/Period)
- Unlock All Locations (Current Category/Period)

Locking the POV prevents locations from modifying their FDM data. When a location has been locked for a particular period or category, users cannot import, validate, export, or re-run the validation report. Run the Process Monitor report to view the lock status for all locations. When a location is locked, a lock symbol () is displayed in the POV bar.

The POV can be locked or unlocked for individual locations by users or administrators. There is also an option, available only to administrators, to lock or unlock all locations simultaneously.

➤ To lock or unlock individual locations (all users):

1. From the Web client, set the POV to the category and location you want to lock or unlock.
2. Select the appropriate menu item:
   - To lock—Select Tools > Lock Current Point-of-View
   - To unlock—Select Tools > Unlock Current Point-of-View

➤ To lock or unlock all locations for the current category and period (administrators only):

1 From the Web client, set the POV to the category and period you want to lock or unlock.

2 Select the appropriate menu item:

- To lock all locations—Select Administration > Lock All Locations (Current Category/Period)
- To unlock — Select Administration > Unlock All Locations (Current Category/Period)

Locking and Unlocking the POV Mode

Use this option to control the availability of the Local POV mode When this option is On, end users cannot change to the Local POV mode. When this option is off, Local POV mode is available to all users on the system. See "Switching POV Mode" on page 37

for information on the POV mode.

➤ To change the POV Mode lock:

1 From the Web client, select **Administration > Point-of-View Mode Lock**.

2 Select or clear **On/Off**.

3 Click **OK**.

Functions Common to Web Client and Workbench

Use the POV to set the FDM data focus. When the POV Lock is enabled, the period and category values are globally controlled across the system, and only the system administrator can change them. This ensures that end users can only load data to the proper period and category.

You can disable the POV Lock for more end-user flexibility. This enables end users to change to Local POV mode to process data for any category or period.

Switching POV Mode

The mode indicator in the POV Bar displays the current POV mode—Global or Local. When the system is in Local mode, double-click Local in the POV bar to switch to Global mode. When the system is in Global mode, double-click Global to switch to Local mode.

- Local POV Mode—Allows any category or period to be selected for processing.
- Global POV Mode—Restricts category and period processing to a global value that is active for all users on the system.

Note:

System administrators can change their POV mode to Local, but end users can only access this option if the POV Lock is disabled.

Setting Location POV

Users can only view locations that they have permission to access. Location permission is determined and granted by system administrators. All users must be assigned one default location. When a users log in, the POV is set to their default location.

➤ To set Location POV:

1 In the **POV Bar** , double-click the current location.

The POV dialog box (Locations tab) is displayed.

2 Select a location.

3 Click **OK**.

Setting Category POV

The FDM administrator controls the data category that is active for all users. This prevents users from inadvertently loading data to incorrect categories.

➤ To set the Category POV:

1 In the **POV Bar**, double-click the current category.

The POV dialog box (Category tab) is displayed.

2 Select a category.

3 Click **OK**.

Changing the FDM category also changes the target system category. When a FDM category is defined on the Control Tables form, it is associated with a target system category.

Setting Period POV

The FDM administrator controls the accounting period that is active for all users. This feature prevents users from inadvertently loading data into an incorrect period.

When a user logs on to FDM, the application checks the global period value and automatically sets the POV to the current value.

➤ To set the Period POV:

1 In the **POV** bar, double-click the current period.

The Period tab of the POV dialog box is displayed.

2 Highlight a period.

3 Click **OK**.

Changing the FDM period also changes the target system period.

Searching the POV

➤ To search from within the Web Client POV:

1 Open the **POV** dialog box by double-clicking the current location, period, or category from the **POV Bar**.

2 Click **Search**.

3 In the **Explorer User Prompt** dialog box, enter a full or partial string to search.

4 Click **OK**.

➤ To search from within Workbench POV:

1 Open the **POV** dialog box by selecting the current location, period, or category from the **POV** bar.

2 In **Select Location**, enter a full or partial string to search.

3 Click **>**.

Locking and Unlocking the System

Use this option to terminate current user sessions and to disallow new users from logging in. You can also type in a message you want displayed to users when they are disconnected. You must terminate user sessions before performing database maintenance activities.

When the system is locked, users cannot log on. User sessions are not terminated if they are writing records to the database.

➤ To lock and unlock the system:

1 In the **POV Bar** (Web client or Workbench), double-click the System Lock category.

The System Lock category is the right-most option in the POV Bar and will be labeled either "Locked" or "Open."

The System Lock dialog box is displayed.

2 Toggle the **On/Off** option to change the system lock status.

3 (Optional) In **System Lock Message**, type the message you want displayed to users when they are disconnected.

4 Click **OK**.

The new system lock status will appear in the POV ("Locked or Open").

Note:

System locking and unlocking privileges are restricted to administrators.

How Users Default Points-of-View are Determined

When a user logs in to FDM, the following sequence of events executes to set the POV:

1. Location POV is set by retrieving the user's security profile.
2. POV Mode is set to Global.
3. Category POV is set by retrieving the system global category (set by administrator).
4. Period POV is set by retrieving the system global period (set by administrator).

4 Configuring System Settings

In This Chapter

System Options

System options are global values that control the behavior of a FDM application. System options are used as both control mechanisms and performance tuning mechanisms. There are three types of system options:

- Application Settings
- Integration Settings
- Configuration Options

You can configure system options from both Workbench and the Web Client.

Configuring Application Settings

➤ To access application settings:

1 Open the Application Settings dialog box:

- From Workbench: Select Tools > Application Settings.
- From the Web client: Select Administration > Application Settings.

2 From **Options**, select an application setting.

There are fifteen application setting options. See "Application Setting Options" on page 42 for detailed information about each option.

3 Select the option settings.

The option settings are located on the bottom half of the Application Settings dialog box. The options displayed are dependent on which application setting is selected in Option.

Application Setting Options

System Code

Specifies the target system that you integrate with FDM. There must be a target system code for each target system integration adapter used. One FDM application can be associated to an unlimited number of target applications.

Ignore Map Item

This stores a target dimension member identified as the “dump” or “ignore” value. Any source dimension mapped to this value is not loaded into the target system. Source dimensions mapped to this value are displayed in the Export screen, but are omitted from the export data file. The default value is “Ignore.”

Log Map Changes

When enabled, all changes made to the mapping table, for any location, are written to the system log. This enables users to maintain an audit trail of all mapping changes. The default value is On.

Batch Loader

Use this option to enable Batch Loader.

Validation Report ID

Use this option to set the ID of the Validation report.

Archive Method

Specifies whether archived files are copied to the archive location or if they are moved to the archive location.

Controls Lock

This option is used with the Financial Controls functionality of FDM. See Chapter 12, “Financial Controls”for detailed information.

Controls w/o Data

This option is used with the Financial Controls functionality of FDM. See Chapter 12, “Financial Controls”for detailed information.

Controls Unsubmit

This option is used with the Financial Controls functionality of FDM. See Chapter 12, "Financial Controls"for details.

Fiscal Start Period

This option relates to the Financial Controls functionality of FDM. See Chapter 12, "Financial Controls"for information.

Date Format Mask

Used to set the date format of the FDM Periods description field. All subsequent FDM Periods created will adhere to the format specified here. The default value is MMM-YYYY.

Create Location Directory

This setting determines if FDM creates a location directory in the Inbox when FDM locations are created. Options are: Only Data Load Location, All locations, or None.

Delete Location Directory

This setting determines, when FDM locations are deleted, if FDM deletes the location directories from the `Inbox`.

Configuring Integration Settings

Integration setting determine the levels of integration between FDM and the target system and control which integration points are active, the type of integration processes to run, and default values relating to integration actions.

➤ To access integration settings:

1 Open the Integration Settings dialog box.

- From Workbench—Select Tools > Integration Settings.
- From the Web client—Select Administration > Integration Settings.

2 From **Options**, select an integration setting.

3 Select the option settings.

The option settings are located on the bottom half of the Integration Settings dialog box. The options that are displayed are dependent on which integration setting is selected in Option.

There are three categories of integration options:

- Connection
- Integration Point

- Default Load

Connection Options

- Application Name—Specifies the name of the target application to integrate with FDM.
- Logon Method—Controls the method that FDM uses to log on to the target system when FDM makes a connection. There are two logon methods—Unified and Global.
  - Unified—The FDM username and password are used to log on to the target system.
  - Global—The username and password specified in the Global Logon Information option is passed to the target system. Using this option, all users connect to the target system by using the same user name and password.

    Note:

    A machine-specific user name and password in Workbench overrides the logon method specified here for this specific machine.
- Global Logon Information—Used in conjunction with the Logon Method option. When Logon Method is set to "Global," the user name and password specified are used to log on to the target system. Separate the user name and password with a semicolon (UserID;Password).
- Use SSO—An On or Off switch to enable Hyperion Single-Sign-On when a token is passed to FDM.
- Sticky Server—On/Off switch to force usage of one server for Load, Consolidate and Check (required for Financial Management application server cache delay).

Integration Point Options

- Enable Load—Forces FDM to automatically load the export file (`*.dat` file) into the target system after export. If this option is cleared, FDM creates the export file but does not automatically load it.
- Enable Consolidation—Runs a consolidation in the target system after a data load.
- Enable Validation Report—Controls the periods viewed when integrating with the target system. Select this option to show all periods. Clear this option to show only base periods.
- All Periods—Enable to force FDM to run a validation report (if there is one assigned to the location) after the data load and consolidate process.

Default Load Options

- Consolidation Type—Default setting for the Consolidation Type option on the target system load dialog box.
- Load Method—Default setting for the Load Method option on the target system load dialog box.

- Load Operation—Default setting for the Load Process Type option on the target system load dialog box.
- List2—Not Used.
- List3—Not Used.
- Calc—Default setting for the Calculate option on the target system load dialog box.
- Accumulate in File—Default setting for the Accumulate in File option on the target system load dialog box.
- File as Ownership Data—Default setting for the File as Ownership Data option on the target system load dialog box.
- Status 3—Not Used.
- Data Protection Switch—Enables FDM to protect data in the target system from being overwritten during data imports based on a specified protection value. Use this when data is entered into the target system through a method other than FDM.
- Protection Value 1—Stores the value used with the Data Protection Switch. It is the value that is not to be overwritten.
- Protection Operator—Stores an operator (= or <>) to determine if items equal to the value stored in the Protection Value 1 are to be protected, or if items that equal the value stored in the Protection Value 1 are to be protected.
- Line Item Detail Loading Switch—Enables loading of line item detail to Financial Management.
- Line Item Detail Load Type—Sets the Line Item Detail loading type (Summarized or Detail).
- Force Calculate before Consolidation—Enable this option to run the default calculation call prior to running a consolidation.

Setting Configuration Options

Configuration options set database performance and tuning parameters. These database tuning options should be used when network infrastructure requires tuning of the database engine I/O activity.

➤ To access configuration options:

1 Open the Configuration Options dialog box from the FDM Web client or Workbench.

- From Workbench: Select Tools > Configuration Options.
- From the Web client: Select Administration > Configuration Options.

2 From **Options**, select a configuration option.

There are thirty-one options for configuration. Details of each are listed in the sections that follow.

3 Select the option settings.

The option settings are located on the bottom half of the Configuration Options dialog box. The options that are displayed are dependent on which configuration option is selected in Option.

Configuration Option Items

General Options

- DB Version—Shows the version number of the installed FDM software.
- SQL Query Date Format Mask—Select the standard format used in SQL statements when querying dates.
- Decimal Replacement—This enables FDM to specify the delimiter used when loading BCP files into RDBMS server. Change this option only when using a system that has been set up in multiple regional settings (for example, RDBMS server has been installed on a US–English OS and the application server or client machine is using French regional option operating system).
- Insert Batch Size—This option enables FDM to specify the number of rows to insert into the FDM database during file import. Use this only for FDM locations that use the SQL Insert load type.
- Total No. Data Segments—Use to define the default number of segments created in the RDBMS database.

SQL Tuning Configuration Options

- Data Seg Table File Group—This option allows FDM to specify that RDBMS Server use a different data segment table file group other than the RDBMS primary group (SQL Server databases only).
- Data Map Seg Table File Group—This option allows FDM to specify that RDBMS Server user a different Data Map Segment Table File Group other that the RDBMS primary group. (SQL Server databases only).
- Work Table File Group—This option allows FDM to specify that RDBMS Server user a different Work Table File Group other that the RDBMS primary group (SQL Server databases only).
- Work Table Index File Group—This option allows FDM to specify that RDBMS Server user a different Work Table Index File Group other that the RDBMS primary group (SQL Server databases only).

Oracle Tuning Configuration Options

- Oracle Seg Insert Hint—Defines how SQL loader loads data to the tDataSeg tables. Default option setting is Append which enables FDM to quickly append data to the segment tables without having to find available space on the table as the Insert setting requires.
- Oracle Work TableSpaceName—The tablespace name for FDM work tables; default is “Users.”

- Oracle Work Storage Clause—Specifies storage defaults for the Work tablespace. This overrides the system defaults to become the defaults for objects created in the specified tablespace.
- Oracle Work Pct Free—Specifies the portion of the data block that is not filled by rows as they are inserted into the Work tablespace, but are reserved for later updates made to the rows in that block. The default for PCTFREE is 10 percent. You can use any integer between 0 and 99, inclusive, as long as the sum of PCTFREE and PCTUSED does not exceed 100.
- Oracle Work Init Trans—Specifies the initial number of transaction entries allocated within each data block allocated to the table. The value can range from 1 to 255. Each transaction that updates a block requires a transaction entry. The size of a transaction entry depends on your operating system. This parameter ensures that a minimum number of concurrent transactions can update the block, and also helps to avoid the overhead of dynamically allocating a transaction entry.
- Oracle Work Max Trans—Specifies the maximum number of concurrent transactions that can update a data block allocated to the Work table. This value can range from 1 to 255.
- Oracle Segment Table Logging Value—Used to turn logging on or off for the segment tables.
- Oracle Work Table Index TableSpaceName—Defines the FDM Work Table Index tablespace name. Default is “Users.”
- Oracle Data Map Seg TableSpaceName—Defines the FDM Data Map Seg table tablespace name. Default is “Users.”
- Oracle Data Map Seg Storage Clause—Specifies storage defaults for the Data Map Seg tablespace. This overrides the system defaults to become the defaults for objects created in the specified tablespace.
- Oracle Data Map Seg Pct Free—Specifies the portion of the data block that is not filled by rows when they are inserted into the Data Map Seg table, but are reserved for later updates made to the rows in that block. The default for PCTFREE is 10 percent. You can use any integer between 0 and 99, inclusive, as long as the sum of PCTFREE and PCTUSED does not exceed 100.
- Oracle Data Map Seg Init Trans—Specifies the initial number of transaction entries allocated within each data block allocated to the Data Map tablespaces. This value can range from 1 to 255. Each transaction that updates a block requires a transaction entry in the block. The size of a transaction entry depends on your operating system. This parameter ensures that a minimum number of concurrent transactions can update the bloc, and also helps to avoid the overhead of dynamically allocating a transaction entry.
- Oracle Data Map Seg Max Trans—Specifies the maximum number of concurrent transactions that can update a data block allocated to the Data Map Seg table. This value can range from 1 to 255.
- Oracle Data Seg TableSpaceName—The tablespace name for FDM Data Seg tables. Default is “Users.”
- Oracle Data Seg Storage Clause—Specifies storage defaults for the Data Seg tablespace. This overrides the system defaults to become the defaults for objects created in the specified tablespace.

- Oracle Data Seg Pct Free—Specifies the portion of the data block that is not filled by rows as they are inserted into the Data Seg table, but are reserved for later updates made to the rows in that block. The default for PCTFREE is 10 percent. You can use any integer between 0 and 99, inclusive, as long as the sum of PCTFREE and PCTUSED does not exceed 100.
- Oracle Data Seg Init Trans—Specifies the initial number of transaction entries allocated within each data block allocated to the Data Seg tables. This value can range from 1 to 255. Each transaction that updates a block requires a transaction entry in the block. The size of a transaction entry depends on your operating system. This parameter ensures that a minimum number of concurrent transactions can update the block, and also helps to avoid the overhead of dynamically allocating a transaction entry.
- Oracle Data Seg Max Trans—Specifies the maximum number of concurrent transactions that can update a data block allocated to the Data Seg table. This value can range from 1 to 255.

Other General Configuration Options

- Allows custom description in period—Enables the administrator to create custom period descriptions.
- Lookup Indicator: Target System Data—Specifies the terminating character for use by validation rules that pull data from the target systems. Default value is the pipe character (|).
- Lookup Indicator: FDM Converted Data—Specifies the terminating character for use by validation rules that pull data from FDM converted data. Default value is the grave accent character (`).
- Lookup Indicator: FDM Source Data—Specifies the terminating character for use by validation rules that pull data from FDM source data. Default value is the tilde character (~).

5 Assigning FDM Security and Managing Users

In This Chapter

Security

Within FDM, you can secure almost every menu item, button, screen, and report. Security features are restricted to administrators. FDM supports two levels of security:

- Application object security—Governs access to forms and form controls
- Location security—Governs access to locations

Security levels are applied to users and application objects, and the levels assigned to users and objects are compared at runtime. If a user is assigned a level that equals or exceeds the level assigned to the object that the user is trying to access, the object is available to the user. Object-maintenance tasks are performed in the Web client.

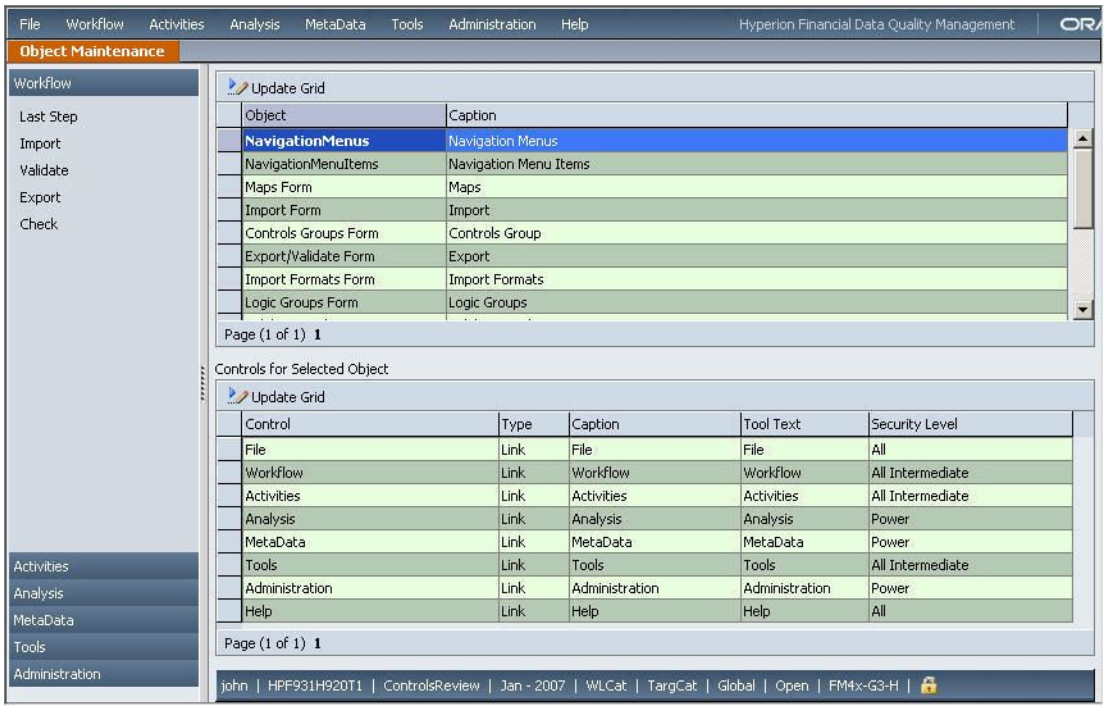
Assigning Application Object Security

Use the Object Maintenance form in the FDM Web client to assign minimum security rights for FDM application objects and to control the caption and tool-text properties of the objects. The Object Maintenance form lists the application objects and their dependant controls.

➤ To assign or change application object security:

1 From the FDM Web client, select **Administration > Object Maintenance.**

The Object Maintenance screen is displayed.



2 From the top grid, select an application object.

3 From the bottom grid, select a control for the object.

4 Double-click in the Security Level column, and select a security level.

Application security is based on a numeric scheme. Initial security levels for users and objects are set to All, which equals the value 99. Administrators are assigned a level of 1, and users are assigned a level between 2 and 98. The lower the user level value, the more access the user has to application objects and reports.

5 **Optional:** To change the caption or the tool text, in the appropriate field, enter a value. .

If a control caption is changed, you should test to ensure that all text is visible. Controls do not re-size to fit the caption.

Location Security

Location security for FDM is configured by using the User Maintenance options.

Managing Users

Access the User Maintenance form from within the Web client. The User Maintenance form enables you to perform the following tasks:

- Add users
- Set user rights
- Delete users

- Change target-system passwords

Adding Users

➤ To add users:

1 From within the Web client, select **Administration > User Maintenance**.

The User Maintenance form is displayed.

2 Click **New User**.

The New User dialog box is displayed.

3 In **Username**, enter a user name.

4 **Optional**: To assign a target system password, select **Use Target System Password**, and enter and confirm the password.

You assign a target-system password if you are using an authentication mode other than LDAP and you want FDM, when using Unified Logon Integration, to pass logon information to the target system.

5 Click **OK**.

The User Rights dialog box is displayed.

Setting User Application Rights

Application rights govern access to forms and form controls. Administrators have access to every location, form, and control. Application security associates each user with a security level that is evaluated against each application object security level. This process determines which components of the user interface each user can access.

➤ To grant users application rights:

1 From within the Web client, select **Administration > User Maintenance**.

The User Rights screen is displayed.

2 Double-click a user.

3 Perform an action:

- To grant administrator rights—from User Level, select Power.
- To grant non-administrator rights—from User Level, select a user group.

4 Click **OK**.

5 **Optional:** To grant the user administrator rights for financial controls, select Auditor.

Setting User Location Rights

Every user must be assigned at least one location. To users assigned a user level of Power, all locations are available. To users assigned user-level application rights, only locations assigned to the users' profiles are available.

➤ To assign locations to users:

1 From the Web client, select **Administration > User Maintenance**.

The User Rights screen is displayed.

2 Double-click a user.

3 Click **Add**.

4 From the new row, double-click the **Location** column, and select a location.

5 **Optional:** To set the selected location as the user's default, select **Default**. .

6 Click **Update Grid**.

Note:

Each user can be assigned multiple locations, but every user must be assigned one default location. . Select Access to All Locations to allow non-administrator users to access all locations.

Location Override Security

The Security Level field enables administrators to configure alternate application rights for assigned locations. For a location, the level specified in the Security Level field, rather than the level specified for a user, is used.

For example, a user with default application rights of Intermediate-7 is assigned three locations. If one of the locations has a security level of Intermediate-5, when the POV is set to the Intermediate-5 location, the user's application rights are set to Intermediate-5.

Email Address

The Email Address field stores the user's E-mail address. You can create a script to E-mail users on the loading status of FDM locations.

Deleting Users

➤ To delete users:

1 From within the Web client, select **Administration > User Maintenance**.

The User Maintenance screen is displayed.

2 Next to a user name, click **Delete.**

3 Click **Update Grid**.

Changing Target-System Passwords

When LDAP security is not being used, users can change their target-system passwords.

➤ To change passwords:

1 From within the Web client, select **Tools > Change Target System Password**.

2 Enter and confirm entry of the old password and the new password.

3 Click **OK**.

6 Dimensions

In This Chapter

Activating Mappable Dimensions

Activate dimensions by using the Web client. Select MetaData > Dimensions to configure the target system mappable dimensions that FDM loads. When Loading Hyperion Enterprise, it is standard to only enable the Account and Entity dimensions. When loading other target systems, every FDM dimension can be loaded. The Dimensions screen contains five columns:

- Dimension
- Alias
- Calc Sequence
- Enabled
- Use as Lookup

Dimension

The Dimension column contains all the predefined FDM dimensions. The dimension name cannot be modified.

Alias

The Alias column allows the dimension to be set with an alternate label. Type in the alias of the dimension that corresponds to the target system dimension name. FDM displays the alias in the list box within the mapping tables and also in the Validate and Export screens.

Calc Sequence

The Calc Sequence column enables FDM administrators to override the default dimension calculation order. Use this when performing conditional mapping. The default order is the order in which the accounts are displayed on the Dimensions screen.

Enabled

Select Enabled to activate the selected dimension. When a dimension is enabled, you can import, map and load data into this dimension.

Use As Lookup

The Use as Lookup column enables administrators to use a dimension for a custom lookup table. When selected, the Enabled field cannot be selected. The custom lookup dimension is used for custom scripting only.

Labeling Dimensions

Select MetaData > Dimension Labels to assign source and target labels to the FDM dimensions. You cannot modify the dimension name, only the dimension label. The dimension labels are displayed in the mapping tables and also in the Import Formats screen.

1. Source Label
2. Target Label

7 Working with Control Tables

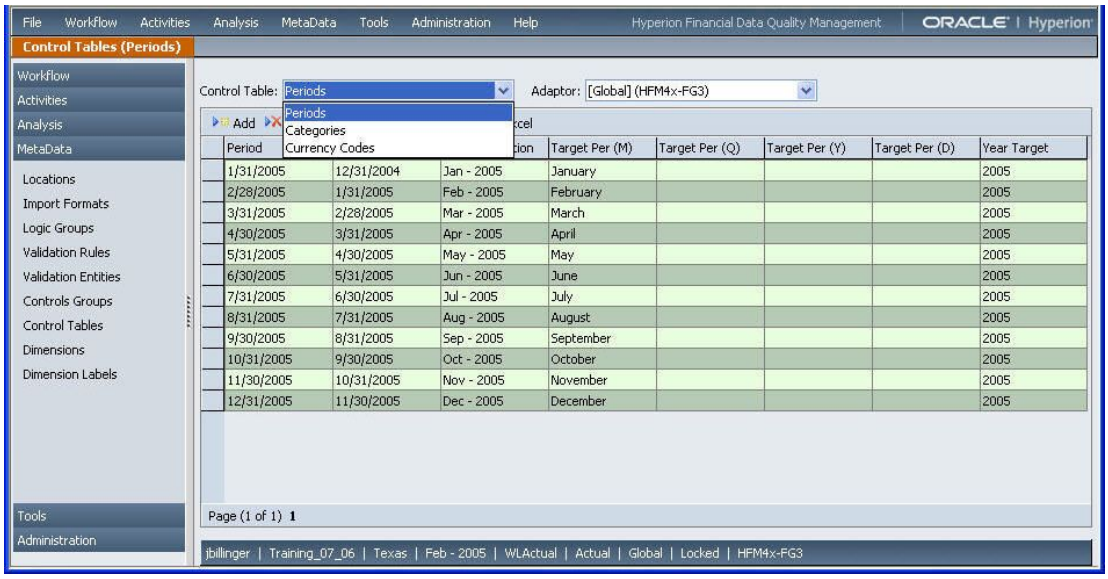
In This Chapter

About Control Tables

Control tables enable system administrators to control the system options and POV values that users can select. For each target system, FDM uses two control tables (Periods and Categories). A third table, Currencies, is global for all target systems. You use the Web client to configure and maintain control tables.

Setting Up Periods

You use the Periods control table to define the fiscal periods that can be used by FDM.



The values in the Periods table identify the periods that can be selected from the period POV. Each FDM period is associated with a target system period. If the FDM period changes, where data is loaded in the target system changes.

- Period—Date value that is stored in the database during the trial-balance load process and that becomes part of a key that identifies a set of trial-balance records
- Prior Date Key—Prior fiscal period key that is used during export to determine whether a $0.00 entry must be made (to prevent ghosting) in Hyperion Enterprise versions 4.3 or earlier (the entry must be made if YTD values were loaded into a periodic category)
- Text Description—Text-based description of the date key (the format of the text description is controlled by the Date Format Mask system configuration. The format defaults to months but can be changed to enable loading of more than twelve periods)

The "Allows custom description in period" option enables administrators to create custom period descriptions that override the default values. Text override option restrictions apply to Text Description Override (value cannot resolve to a date) and Period Key Date Format Mask (option must be set to *MM-DD-YYYY*).

The options identify the target period to which data is loaded and represent months, quarters, years, or days, respectively:

- Target (M)
- Target (Q)
- Target (Y)
- Target (D)

➤ To insert periods into tables:

1 From within the Web client, select **Metadata** > **Control Tables**.

The Control Tables screen is displayed.

2 From **Control Table**, select **Periods**.

3 Select **Add**.

4 Select the cell to which to add or update information.

5 In the **Period** column, select **Browse**, and select a target period.

To indicate that the change is pending,  is displayed left of the row.

6 Click **Update Grid**.

The change-pending icon is removed.

➤ To delete periods from tables:

1 Select a row.

2 Click **Delete**.

 is displayed left of the row, indicating that a deletion is pending.

3 Click **Update Grid**.

➤ To cancel pending changes:

1 Left of the row, click  or  .

2 Select **Cancel Row Changes**.

Setting Up Periods Control Tables for Override Adapters

The procedure "Setting Up Periods" on page 57 illustrates a setup that uses the global integration adapter (default). A Periods control table must be built for each override adapter used in the application.

➤ To set up control tables for override adapters:

1 Select **Metadata > Control Tables.**

The Control Tables screen is displayed.

2 From **Control Table**, select **Periods**.

3 From **Adapters**, select an integration adapter.

A new Periods control table is displayed. Only the Period column is populated.

4 Configure the table to integrate with the override target application.

Note:

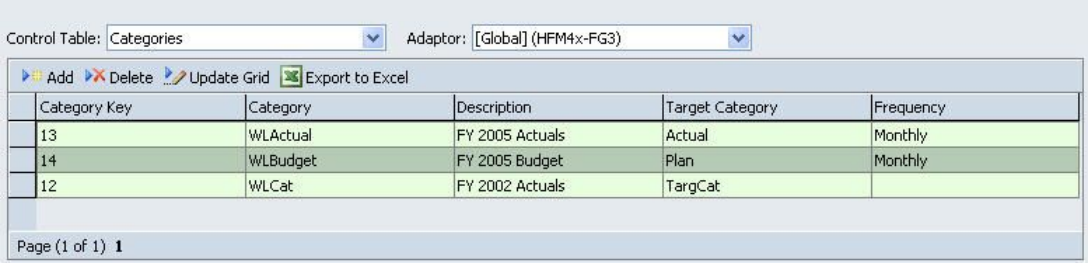
Override adapters that are not in the Adapter list, have not been imported to and configured in the FDM application. See the installation guide for information about adding adapters to applications.

Data Maintenance Considerations

Deleted periods are removed from all categories in all locations. Before deletion, the system must be locked. After deletion, the database should be compacted. Database compaction is performed by the DB administrator.

Setting Up Categories

The Categories table contains definitions of data categories (containers into which data can be loaded).



The values in the table identify the categories that can be selected from the category POV link. Each FDM category is associated with a target system category. Changing the FDM category changes where data is loaded in the target system.

- Category Key—Numeric keys that are stored during the trial-balance load process and generated by FDM (each value identifies a set of trial-balance records)
- Category—The FDM category name
- Description—Description of the FDM category
- Target Category—The target system category to where the data is loaded
- Frequency—The category frequency

➤ To insert categories into tables:

1 From within the Web client, select **Metadata > Control Tables.**

The Control Tables screen is displayed.

2 From **Control Table**, select **Categories**.

3 Select **Add**.

4 Select the cell to which to add or update information.

5 In the **Category** column, select **Browse**.

6 Select the **Target Category** column, and select a target category.

 is displayed left of the row to indicate a pending change.

7 Click **Update Grid**.

The change-pending icon is removed.

➤ To delete categories from tables:

1 Select a row.

2 Click **Delete**.

 is displayed left of the row, indicating that a deletion is pending.

3 Click **Update Grid**.

The deleted rows are removed.

➤ To cancel pending changes:

1 Left of the row, click  or  .

2 Select **Cancel Row Changes**.

Setting Up Category Control Tables for Override Adapters

➤ To set up control tables for override adapters:

1 Select **Metadata > Control Tables.**

The Control Tables screen is displayed.

2 From **Control Table**, select **Categories**.

3 From **Adapters**, select an integration adapter.

A new Categories control table is displayed. Only the Periods column is populated.

4 Configure the table to integrate with the override target application. For information about setting up categories, see "Setting Up Categories" on page 59.

Note:

If the override adapter you wish to select is not in the Adapter list, then the adapter has not been imported and configured in the FDM application. See the installation guide for information about adding adapters to the application.

Data Maintenance Considerations

Deleted periods are removed from all categories in all locations. Before deletion, the system must be locked. After deletion, the database should be compacted. Database compaction is performed by the DB administrator.

Setting up Currency Codes

The Currency Codes table contains a list of currency codes. Currency codes are assigned to locations and displayed on reports. Because currency codes are used only for notation, they do not impact calculations.

- Currency Code—The short description of the currency
- Description—A detailed description of the currency

➤ To insert currencies into the Currency Codes table:

1 From within the Web client, select **Metadata > Control Tables.**

The Control Tables screen is displayed.

2 From **Control Table**, select **Currencies**.

3 Click **Add**.

4 In the new row, provide the currency information.

5 Click **Update Grid**.

8 Working with Data Load Locations

In This Chapter

Note:

This chapter details only FDM data load locations. See Chapter 12, "Financial Controls" for details about controls review locations.

About Data Load Locations

Data load locations are reporting units that are responsible for submitting source data into the target system. Typically, there is one FDM location for each source file loaded to the target system.

End users are assigned access to one or more locations by the system administrator with one location being designated as the default location. When a user logs into FDM, the POV is set to their default location. End users only have access to the locations that were granted by the administrator.

FDM Location Types

FDM contains two types of locations, Data Load and Controls Review:

- Data Load—Location to where source system data is loaded. A data load location can also be assigned financial controls. Data load locations can only be children in the controls structure, they cannot be parents. Data load locations are designated by the  icon.
- Controls Review—Location assigned only to financial controls. No data is loaded to or from a controls review location. Controls review locations are typically parents in the controls structure. A controls review location can also be a child of another controls review location. A controls review location is designated by the  icon.

Creating FDM Locations

When a new FDM application is created, one sample financial controls location named ControlsReview, and one sample data load location named Sample is created. The controls review location is only used for the financial controls functionality of FDM. Create locations using the Web client.

➤ To create data load locations:

1 From within the Web client, select **Metadata** > **Locations**.

The Locations screen is displayed.

2 Right-click on the **Controls Review** location in the tree and select **Add Child**.

3 From **Location Type**, select **Data Load**.

4 Enter the name of the FDM location.

5 Click **OK**.

Location names are restricted to alpha and numeric characters with no spaces. After a location has been created you cannot change the name. Therefore, it is prudent to develop a naming convention before you begin.

Creating Location Directories

When creating locations on the Location screen, FDM creates a directory in the `Inbox` for each data load location added. This function is controlled by the Create Location Folder option in Application Settings. If this option in enabled, FDM creates directory for all locations (data load and controls review).

Location Attributes

The Location screen contains four tabs – General, Workflow Behaviors, Financial Controls, and Integration Options.

General Tab

- Description—Additional information about the location useful for tracking which GL systems are in use or other system-related information about a location. When creating a new location, the default description is the same as the name of the location.
- Currency—The currency that is loaded into the location; used only for identification purposes in some FDM reports and other forms that display data. Because FDM does not perform currency translation, the currency code does not impact calculations (optional).
- Parent Location—A location assigned a parent uses the same mapping table as the parent. Multiple locations can share a parent. This is useful when multiple locations use the same chart of accounts. If a change is made to a child or parent mapping table, the change takes place for all children and parent locations (optional).
- Load Type—Each data load location uses one of two load types—SQL Insert or Bulk Insert. SQL Insert is the default load method and should be used for locations loading smaller files (6000 records or less). The Bulk Insert load type is more efficient for loading larger files.
- Seq Map—By default, FDM wildcard maps (Likes, Ins, Betweens) are sorted in the Maps form alphabetically by the rule name. The Seq. Map option allows sorting and processing maps by a numeric value rather than the default. Use this for locations that have complicated conversion rules predicated upon processing order (for example, a source account is included in multiple wildcard maps, therefore the sort order of the map is important to determine by which map the account must be processed). Use this option only for locations that use complicated translation rules. Click on the Seq Map checkbox to display a new sequencing field in the Maps form. Enter a number in this field to assign a numeric sort order to each map (optional).
- Group Tag—You can group multiple locations by entering a value in this field. In some process monitor reports, grouped locations are displayed in their grouping order rather than alphabetically (optional).
- Data Value—An extra dimension that is only used when integrating with multi-dimension target systems. This dimension is associated with a FDM location. When FDM creates the load file, this dimension value is entered for every data line loaded by this location. For example, the Data Value dimension is associated with the Value dimension in Financial Management. By default, if no value is entered in this field, when integrating with Financial Management, the Data Value <Entity Currency> is the default value.
- Target Adapter (Adapter by Location)—When left at [None], the location uses the global adapter defined during the adapter setup process. Any selection here overrides the global adapter for this location. This setting enables FDM to integrate with multiple target applications. Each location can load data to a unique target application.

Note:

When working in FDM screens that browse for target categories, FDM browses by using the adapter selected for the current location. Unless the target systems contain common member values for category and period, control tables (Categories and Periods) are not partitioned by adapter and require separate entries for each adapter .

Workflow Behaviors Tab

- Import Format—Each data load location must be assigned an import format for a source file to be loaded into the location. The import format defines the structure of the source file. When a location is initially created, you do not need to initially assign an import format. You can set up all reporting locations without knowing all the source file structures (required).
- Logic Group—Logic groups contain one or more logic accounts that are generated after loading the source file. Logic accounts are calculated accounts derived from the source data (optional).
- Validation Rules—A validation rule group contains a set of validation rules that are displayed on the validation report. These validation rules are used by FDM to retrieve target system data from the target application after completing a data load (optional).
- Validation Entities—A validation entity group contains a set of target system Validation entities that can be consolidated and displayed in the validation report (optional).

Financial Controls Tab

See Chapter 12, "Financial Controls" for information on Controls Review locations.

Integration Options Tab

This screen shows available custom adapter options for the respective location.

Organizational Changes

You can add, delete, and move locations. To record the organization structure at the time of submission, FDM logs the organization structure for each period in the tDataSubmitHierarchy table.

9 Logic Accounts

In This Chapter

Logic Accounts

Logic accounts are dynamically generated accounts that are used to calculate supplemental values not provided in the source file. Logic accounts can be mapped and loaded into the target system like any other source account. Logic accounts can be used to provide a variety of functions:

- Statistical Loading—Map a single source account to multiple target accounts
- Conditional Mapping—Map a source account based on its value
- Arithmetic Mapping—Perform arithmetic operations on source values

Logic accounts are created in FDM, therefore detailed audit trails on the logic account values are not available. Access the Logic Accounts screen by selecting MetaData > Logic Groups.

Creating Logic Groups

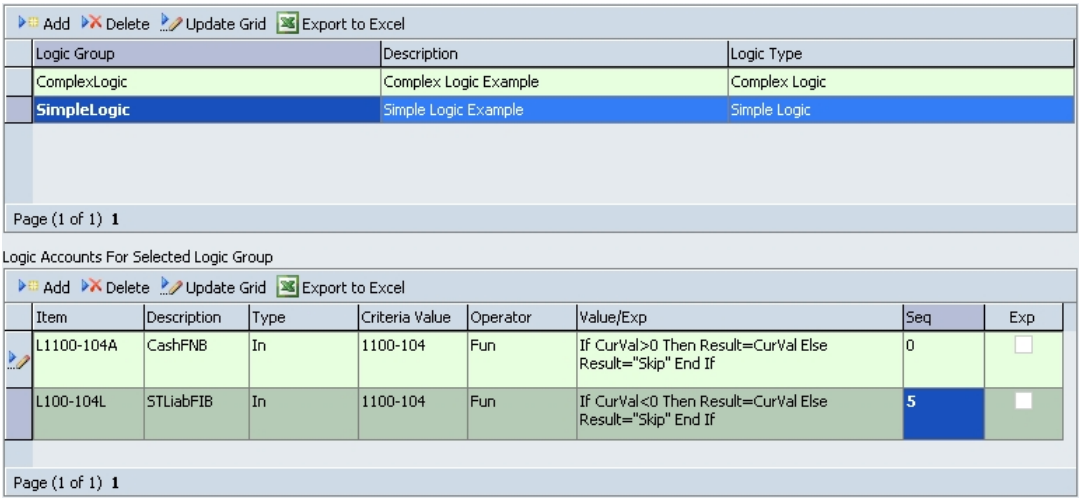
The first step in creating a logic account is to create a logic group. Each logic group can have multiple logic accounts. A logic group is then assigned to one or more locations. If a location has been assigned a logic group, associated logic accounts are generated when a source file is loaded to that location.

Logic Type

Logic groups must be defined as simple or complex. Simple logic groups allow logic items only to be derived from the source account dimension. Complex logic groups allow logic items to be derived from any combination of dimensions.

Simple Logic Accounts

After a simple logic group has been defined, you can define individual logic accounts for that group. A description of each logic item follows:



Item

Enter in the name of the logic item in the Item field. The Item field is displayed in the Account field on the Import screen. It is recommended to precede the name of the logic item with an "L" to distinguish logic accounts from standard source accounts. If the logic account is loaded to the target system, it must be mapped to a target account.

Description

Enter in the description of the logic item here. This description is displayed in the Account Description field on the Import form.

Type / Criteria Value

The operator in the Type field works in tandem with the source account specified in the Criteria field to determine from which source accounts the logic item are derived. A single logic item can be derived from multiple source accounts. The following tables contain examples of possible operators and criteria value.

Between Operator—Use the *Between* operator to specify a range of source accounts in the Criteria field. Separate the two accounts by a comma.

Type Field	Criteria Field
Between	1000,1999

Like Operator—Use the *Like* operator if the source accounts in the Criteria field contain wildcard characters. Use questions marks (?) as placeholders and use an asterisk (*) to signify an indeterminate number of account characters.

Type Field	Criteria Field
Like	1??0
Like	10*

In Operator – Use the *In* operator to include a single source account or a list of nonsequential source accounts.

Type Field	Criteria Field
In	1000
In	1000,1005,2001

Grouping by Center

By default, a separate logic account is created for each source center found in the trial balance.

Criteria Field	Result
12300	A logic account is created for each source center associated with account 12300.

You can create a single logic account that summarizes multiple source centers by placing a semicolon after the account in the Criteria field and entering a number that corresponds to the number of characters in the center you want to group by.

Criteria Field	Result
12300;4	A summarized logic account is created for account 12300 that includes all source centers that have the same first four characters in the source center. The center assigned to this logic account is these four characters.
12300;3,4	A summarized logic account is created for account 12300 that includes all source centers that have the same three characters in the source center, starting at position 4. The center assigned to this logic account is the three characters starting at position 4.

You can create a logic account that summarizes all source centers by placing a semicolon after the account in the Criteria field and entering a text value. This hard-coded text value becomes the new center for the summarized logic account.

Criteria Field	Result
12300;Dept100	A summarized logic account is created for account 12300 that includes all source centers. The center assigned to this logic account is “Dept100.”

Sequence Field

This field specifies the order in which the logic accounts will be processed. This allows one logic account to be used by another, providing the dependant account was processed first.

Export Field

This is a Yes/No switch that determines whether or not the logic account should be considered an export account and subjected to the conversion table validation process. If the switch is set to Yes, then the logic account must be mapped in the Account Conversion table.

Operator and Value/Exp Fields

Defining Operator and Value/Exp Fields

To perform calculations and derive values for the logic account, select an operator in the Operator field to work in tandem with the value in the Value/Expression field.

Numeric Operators

Use a numeric operator to perform simple mathematical calculations.

+ Addition

- Subtraction

* Multiplication

/ Division

NA (No Operator)

If a numeric operator is specified in the Operator field, then a numeric value must be entered in the Value/Exp field unless the specified operator is "NA." In this case, the specified source accounts in the Criteria Value field are summed.

Function Operator (Fun)—Used to execute a custom logic function defined in the Value/Exp field. To write a function, double-click in the Value/Exp field to display the logic function editor. Logic functions are normally used for conditional mapping and other complex operations that involve multiple source accounts. Logic functions allow the use of variable and If statements to enable conditional mapping. The FDM Lookup function can also be used within a logic function.

The following function parameters can be used in a function:

- CurVal—Value of the logic account operation
- StrLocation—Active location name
- strCenter—Logic account center
- strCatKey—Active FDM category key (not name)
- strPerKey—Active FDM period

Assigning Function Return Values

The result of a Logic Function must be assigned to the keyword `RESULT`.

```
RESULT = CURVAL + (|810| * .5)
```

If no return value is assigned to the result keyword then the logic engine automatically sets the value of result to zero. This causes the calculation to be skipped and the logic account is not created.

The following function assigns the result of the logic account calculation (using the CURVAL parameter) to the logic account (RESULT) if the logic account calculation returns a value greater than zero.

```
If CURVAL > 0 Then
    RESULT = CURVAL
Else
    Result=”Skip”
End If
```

This is an example of conditional mapping. If the source accounts specified in the criteria column is less than zero, then this logic account is not created because of the keyword “Skip.” This logic account must be mapped to a target account.

Another logic account is then created to store the result of the logic account calculation if the result is less than zero. This logic account is then be mapped to a different target account.

```
If CURVAL < 0 Then
    RESULT = CURVAL
Else
    Result=”Skip”
End If
```

The following function only assigns the result of the logic account calculation to the logic account if “10” is the active FDM category key.

```
If strCatKey = “10” then
    RESULT = CURVAL
Else
    Result=”Skip”
End If
```

This function assigns the result of the logic account calculation to the logic account only if the Criteria Account center is “000.”

```
If strCenter = “000” then
    RESULT = CURVAL * 100
Else
    Result=”Skip”
End If
```

This function uses the FDM Lookup function to add a source account to the value of the logic account if the current FDM period is “Dec 2003.”

```
If strPerKey = “12/31/2003” then
    RESULT = CURVAL + |810|
Else
    Result=”Skip”
End If
```

This function uses the FDM Lookup function to add another source account from a different source center, FDM category, and FDM period to the value of the logic account if the active location is “Texas.”

```
If strLocation = “Texas” then =
    RESULT = CURVAL + |000,10,09/30/01,810|
Else
    Result=”Skip”
End If
```

Expression Operator (Exp)—Used to execute a custom logic expression defined in the Value/Exp field. Logic expressions are simpler than logic functions but cannot use variables or `If` statements. Expressions do not have built-in parameters except for `|CURVAL|` which must be enclosed in pipes. There is no need to assign the value of an expression to the `RESULT =` constant like in logic functions. Expressions execute faster than logic functions. The FDM Lookup function can be used within an expression in the same manner as within a logic function. To write a custom expression, double-click the Value/Exp field to open the expression editor.

The following are examples of custom logic expressions:

```
|CURVAL| + |810| + |238|
```

This function uses the FDM Lookup function to add two source accounts to the value of the logic account. Notice that the CURVAL parameter can be used within expressions as in logic functions except that it must be enclosed in pipes.

```
(|CURVAL| + |000,10,09/30/01,810|) * 100
```

This function uses the FDM Lookup function to add another source account from a different center, FDM category, and FDM period to the value of the logic account and multiplies the entire result by 100.

FDM Lookup—Use this function to return the value of any source account that resides in FDM. This function can be used within a logic function or within a logic expression.

To return a value of a source account for the active logic account source Center, FDM category, and FDM period, use the following format:

```
Syntax: |GLAcct|
Example: |810|
```

To return the value of a GL account for a different GL Center, FDM category key, and FDM period use the following format:

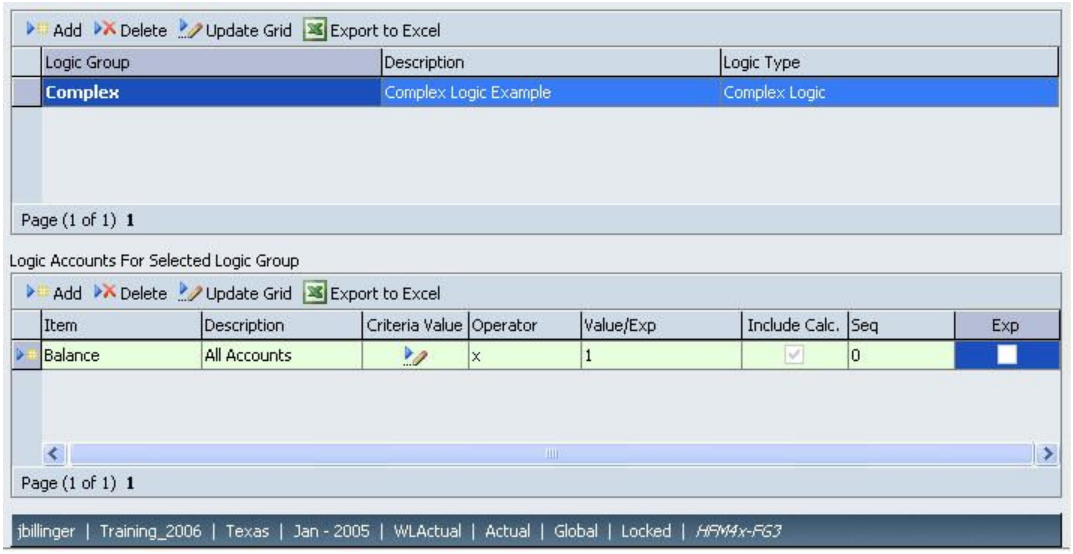
```
Syntax: |GLDept , CategoryKey , PeriodKey , GLAcct|
Example: |000, 10, 12/31/01, 810|
```

You can also use a combination of formats. The following example uses the default source center and FDM period but specifies a particular FDM category key.

```
|, 10 , , 810|
```

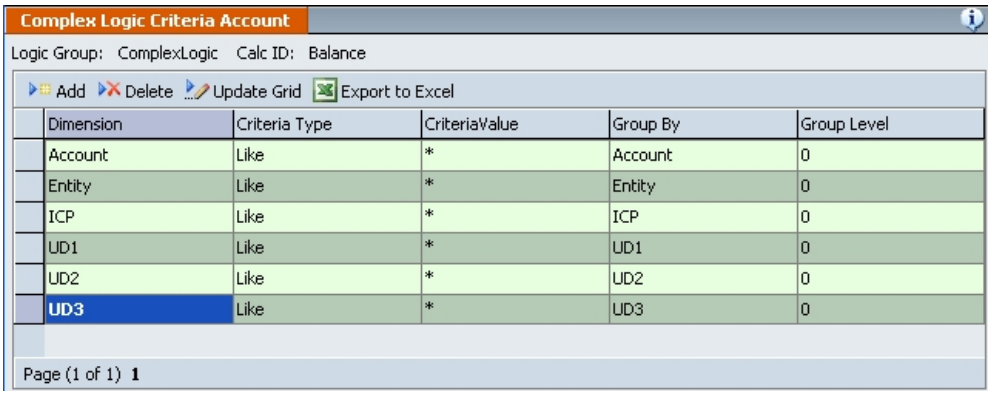
Creating Complex Logic Accounts

Individual logic items are defined within a complex logic group. Each of the fields for complex logic operates the same as simple logic except for the Criteria Value and Include Calc fields.



Criteria Value

To enter criteria for each dimension, click the Criteria Value field to open a new form. The logic item is created only from the source line items that meet the specified criteria for each dimension. Descriptions of each complex logic criteria field follows:



Dimension

This field allows the selection of any enabled source dimension. Each dimension can only be selected once.

Criteria Type

This field works in tandem with the Source Dimension and Criteria Value fields to determine from which source values the logic items are derived. Criteria types available are In, Between, and Like. The Criteria Type determines how the criteria value is interpreted.

Criteria Value

This fields the value that the criteria type uses to determine what members should be included in the logic calculation for any given logic dimension.

Group By

When viewing the derived logic item in the Import screen, the Group By field allows the logic item to override the displayed member in the appropriate dimensions field. Overriding the displayed member allows the dimension to be grouped based on the value entered in the Group By field. This field can be used to hard code the returned member or it can append hard-coded values to the original members by entering a hard-coded member and an asterisk (*) in the Group By field.

For example, placing the word “Cash” in the row with account selected for dimension, the Import form displays “Cash” in the Account field for the logic item. If “L-*” is placed in the Group By field, the import form displays “L-1100” where 1100 is the original account that passed the logic criteria.

If no value is entered in the Group By field, no grouping occurs for this dimension and a separate logic item is created for each unique dimension member.

Group Level

When viewing the logic item in the Import screen, the Group Level field works with the Group By field to override the displayed member in the appropriate dimensions field. This field only accepts numeric values.

Entering a value of “3” in the Group Level field returns the left three characters of the Group By field. If no value is entered in the Group By field, then specifying a value of “3” in the Group Level field returns the first three characters of the original source dimension member.

Using of the Group By and the Group Level fields, the logic items displayed on the Import form can be grouped to the desired level. For example, if “L-*” is entered in the Group By field, the logic item displays in the Import form as “L-1100,” where 1100 is the original account that passed the logic criteria. If a Group Level of “2” is entered for the same row, then the Import form displays “L-11.” If a Group Level of “1” is entered for this row, then the Import form displays “L-1.”

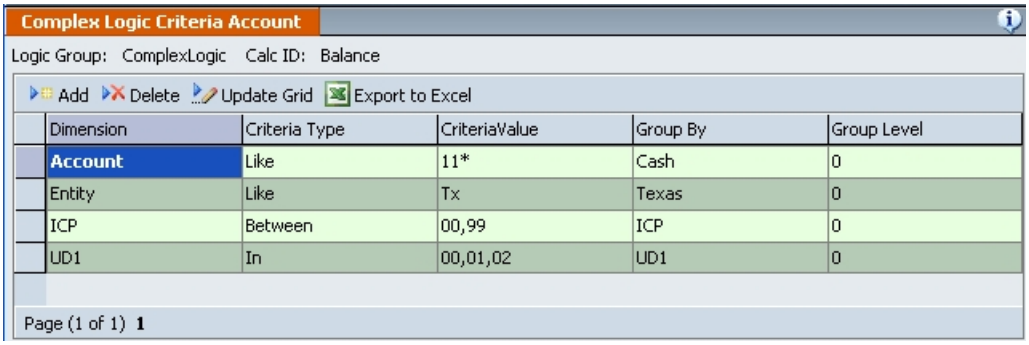
Include Calc Field

If it meets the logic item criteria, Include Calc field allows the logic item to include previously calculated FDM values in its calculations.

Note:

Each logic item has a sequence attached, and the logic items are calculated in this sequence. If the second, or later, logic item has this field enabled, then any previously calculated logic items are included (provided they meet the logic criteria).

Complex Logic Example 1



The first row specifies that any accounts that begin with “11” will be included in the calculated result for “Calc Item: CashTx.” The second row further qualifies the results by specifying that the source record must also have the entity equal to “TX.” The third line reduces the results to only those source records that have an ICP value between 00 and 09. The last line reduces the results to only those source records that have a Custom 1 (UD1) of either: 00, 01 or 02. Any imported source line that does not meet all of the listed criteria is excluded from the calculated results.

As shown in the following table, only one new logic item is derived from multiple source records. Using the preceding graphic example as the logic criteria, and the first grid that follows as the source line items, you can see how FDM derives the value of a single logic item. Note the Group By field. Each Group By field includes a hard-coded value. Therefore, for every line that passes the specified criteria, the original imported member is replaced with the member listed in the Group By field.

Sample Imported Values

Account	Entity	ICP	UD1	Amount	Include / Exclude
1150	TX	07	01	50,401.07	Include
1176	TX	04	02	10,996.00	Include
1201	TX	01	00	500.00	Exclude

Logic Members Created

Account	Entity	ICP	UD1	Amount
Cash	Texas	ICP	UD1	50,401.07

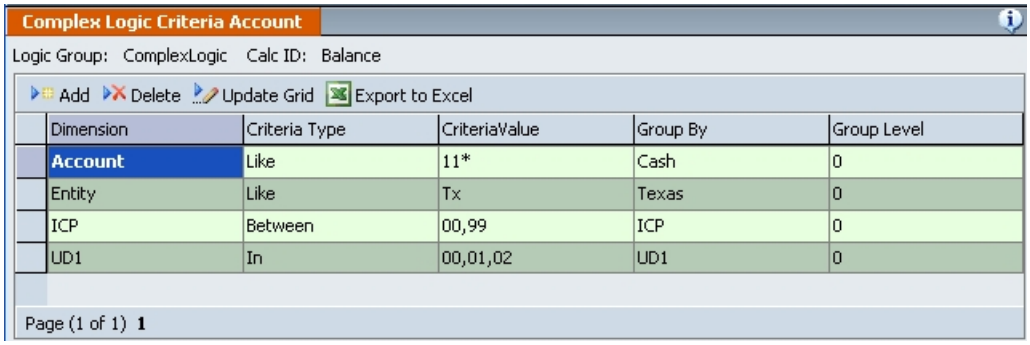
Account	Entity	ICP	UD1	Amount
Cash	Texas	ICP	UD1	10,996.00

FDM groups and summarizes each of the rows where the members are all the same, creating the following final results:

Final Logic Members Results

Account	Entity	ICP	UD1	Amount
Cash	Texas	ICP	UD1	61,397.07

Complex Logic Example 2



The first row in the preceding graphic specifies that accounts that begin with “11” are to be included in the calculated result for “Calc Item: CashTx.” The second row further qualifies the results by specifying that the source record must also have the entity equal to “TX.” The third line reduces the results to only those source records that have an ICP value between 000 and 100. The last line then reduces the results to only those source records that have a Custom 1 (UD1) of either: “00,” “01.” or “02.” Any imported line that does not meet all of the listed criteria are excluded from the calculated results.

As shown in the following tables, two logic items are derived from the source records because of the values entered in the Group By and Group Level fields. Two of the Group By fields have hard-coded values listed and two have an asterisk. Therefore, for every line that passes the specified criteria, the original imported members for the Account and Entity dimensions are replaced with the member listed in the Group By field. The other dimensions return all or part of the original members based on the Group Level entered.

Sample Imported Values

Account	Entity	ICP	UD1	Amount	Include / Exclude
1150	TX	070	01	50,401.07	Include
1176	TX	040	02	10,996.00	Include

Account	Entity	ICP	UD1	Amount	Include / Exclude
1121	TX	045	02	9,050.41	Include
1201	TX	100	00	500.00	Exclude

Logic Members Created

Account	Entity	ICP	UD1	Amount
Cash	Texas	07	UD1-01	50,401.07
Cash	Texas	04	UD1-02	10,996.00
Cash	Texas	04	UD1-02	9,050.41

FDM groups and summarizes each of the rows where the members are all the same, creating the following results:

Final Logic Members Results

Account	Entity	ICP	UD1	Amount
Cash	Texas	07	UD1-01	50,401.07
Cash	Texas	04	UD1-02	20,046.41

10 Import Formats

In This Chapter

Overview

Each data load location must be assigned an import group that it uses when importing data files. Import groups instruct FDM how to interpret data files and transform them into data that can be loaded into the FDM data structure.

The Import Formats screen is arranged in a master-and-detail format. The top grid shows import groups, and the bottom grid is where the fields for a group are defined. After the import format is created, it is important that the source file layout be unchanged. If the format of the source file changes, you must update the import format.

Creating New Import Groups

➤ To create import groups:

1 Select **Metadata** > **Import Formats**.

The Import Formats screen is displayed,

2 Click **Add**.

This adds an additional line in the upper table of the Import Formats screen.

3 Specify the name and description of the import group.

4 From the **File Type** field, select **Fixed** or **Delimited**.

5 If the file type is delimited, in the delimiter column, select the type of delimiter that is used for the file.

6 Click **Update Grid**.

Note:

By selecting Script or Adapter in the File Type column, you can also build an Integration Script or use a target system adapter to import the ledger data directly from an ODBC-compliant data source rather than importing a text file.

Defining Fields for Import Groups

Each row in the lower grid of the Import Formats screen represents one import field. You can assign multiple source fields to the same FDM field name, and FDM automatically concatenates the string values for you during the file load process. If you want to skip records that contain certain values, select "Skip" and enter a text value in the Expression field.

Defining Fields for Fixed Import Groups

- Field Name—The field to which to import.
- Start—The start location of the field to import.
- Length—The length of the field.
- Expression—Specify an expression. This value overwrites whatever is in that field location unless the field is defined as a "Skip" field.

Defining Fields for a Delimited Import Group

- Field Name—The field name in which to import.
- Field No—The field number that you want to import.
- # of Fields—The total number of fields for a row in the text file.
- Expression—An expression in this field overwrites whatever is in that field location unless the field is defined as "Skip."

Drag and Drop Field Definition

Instead of counting the start and length locations of your fields, you can create your field definition by using the Drag and Drop feature from the Import Format Builder.

➤ To use Drag and Drop to build an import format:

1 In the **Import Format** grid, select the field to edit.

2 Click the **Build** button located above the lower table.

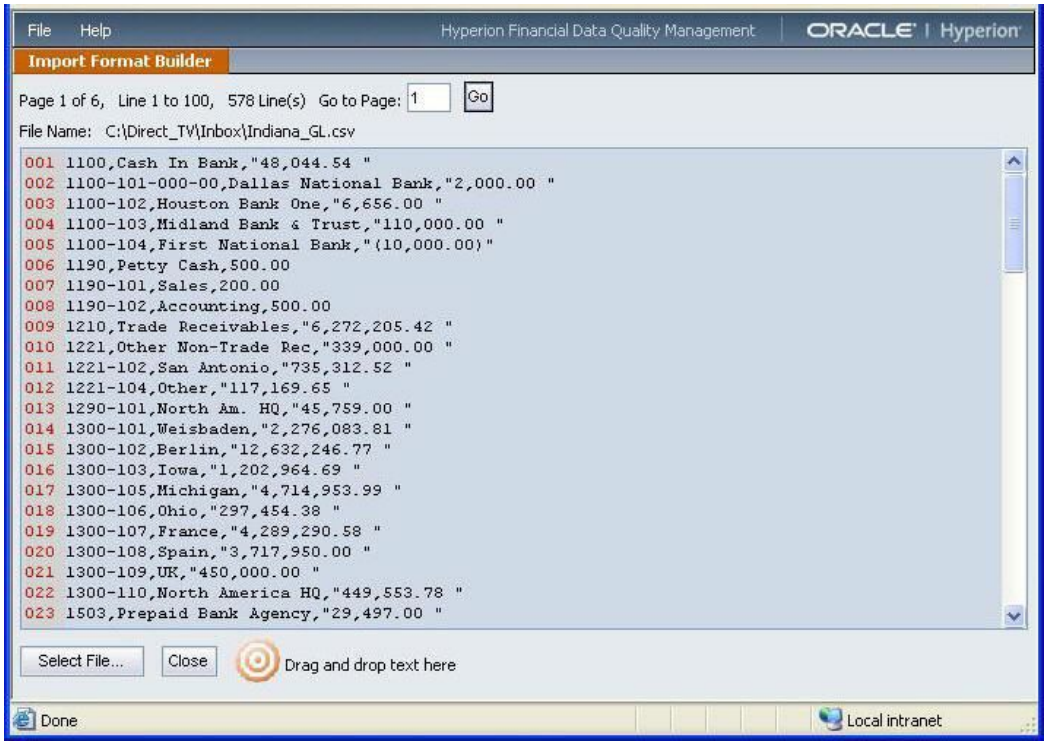
The Import Format Builder screen is displayed.

3 From the **Import Format Builder** screen, click **Select File**.

4 Locate and select the file to import.

5 Click **OK**.

After importing, the file contents are shown in the Import Format Builder window.



6 In the **Import Format Builder** screen, highlight the text to import for the field.

7 Click and drag the highlighted field to the **Drag and Drop Text Here** box.

The Field Number and Number of Fields are automatically be populated.

8 Repeat steps 5 and 6 for each field name.

A field name can be used more than once. FDM automatically concatenates the multiple field names during the load process.

Assigning Import Formats

Import formats are not used until they have been assigned to a location.

➤ To assign import formats to locations:

1 Select **Metadata > Locations**.

The Locations screen is displayed.

2 Click the **Workflow Behaviors** tab.

3 From **Import Format**, select the import format to use for the selected location.

This format is used every time a trial balance is imported for that location. A single import format can be used for many locations. If your company has a standard trial balance file format, then all locations can share the same import format.

Advanced Import Features

Adding Import Expressions and Import Scripts

FDM has a set of powerful import expressions that enable it to read and parse virtually any trial balance file into the FDM database. Advanced expressions are entered into the Fields grid under the Expression column. Import expressions and scripts operate on the value that is read from the import file.

In the previous examples of import formats, expressions were used to hard-code the Center value. An advanced import expression (`sign=,cr`) is used for the NA Amount field to distinguish the credit balances.

➤ To add import expressions or import scripts:

1. On the **Import** form, double-click the **Expression** column.
2. Select **Add Expression**.

   The Add Expression dialog box is displayed.
3. From **Expression Type**, select the type of expression to use.

   The number and type of options shown in this field depend on the field that is being modified in the Import form (Account, Account Description, Amount, and so on.)
4. Enter the value to accompany the expression.
5. Click **OK**.

   The new expression is entered into the Expression field on the Import form.

Sign

Use this expression to mange nonstandard numeric sign conventions. FDM interprets leading and trailing minus signs, as well as parentheses, as negative numbers. However, some files trail numbers with other characters to indicate a negative amount. To define a custom sign the expression should follow this form: `Sign = [Positive String],[Negative String]`.

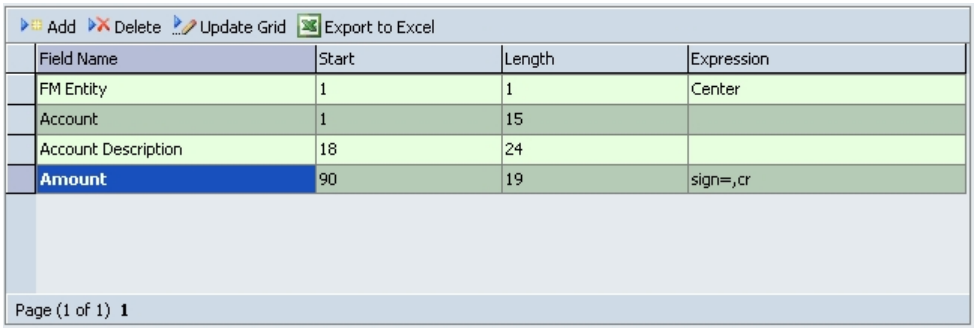
Example

Positive numbers are followed by a “DR” (`1,000.00DR`), and negative numbers are followed by a “CR” (`1,000.00CR`).

Expression should be: `Sign=DR,CR`.

Positive numbers are unsigned (`1,000.00`), and negative numbers are followed by a “CR” (`1,000.00CR`).

Expression should be: `Sign=,CR`.



DRCRSplit = (Split Debit and Credit Columns)

DRCR Split is used to parse split numeric columns. By default, FDM assumes that any numeric value found in the position defined for the Amount field is a debit. However, many files position debit values in the left side of the field and credit values in the right side.

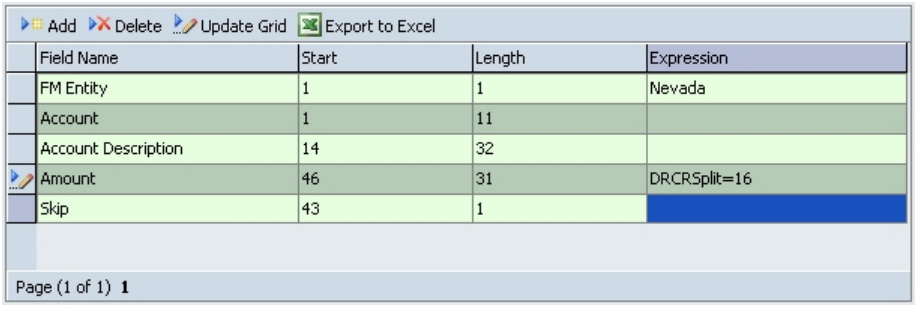
```
Run Date: 03/10/2004  FDM Training                    Page: 1

G/L Date: 03/10/2004TRIAL BALANCE FOR PERIOD ENDING 01/31/04    Time: 03:38
PM
```

ACCOUNT NUMBER	DESCRIPTION	DEBIT	CREDIT
1000-000-00	Cash-Operating Account	68,603.91	
1010-000-00	Cash-FANB-AP		177,216.16
1100-000-00	Accounts Receivable	3,165,909.94	
1200-000-00	Allowance for Doubtful Account		487,652.89
1210-000-00	Allowance for Contractual Adj		266,683.46
1211-000-00	Allowance for Admin Discounts		33,811.30
1212-000-00	Allowance for Medicare/caid		214,895.97
1250-000-00	Inventories		65,056.55
1265-000-00	Prepaid Other		9,184.72
1301-000-00	Other Current Assets -Other AR	152,073.14	
1400-000-00	Land	2,180,000.32	
1420-000-00	Buildings-40yr	6,142,095.26	
1440-000-00	Equipment-5yr	128,789.35	
1450-000-00	Equipment-5yr	47,487.37	
1540-000-00	Accum Depr-Equip-10 yr		200,775.20
1600-000-00	Goodwill	1,382,384.29	

ACCOUNT NUMBER	DESCRIPTION	DEBIT	CREDIT
2000-000-00	Accounts Payable-System		315,540.69
2002-000-00	Credit Balances		263,607.00
2301-000-00	WH-Medical Premium		29,832.35
2400-000-00	Accrued Payroll		328,324.68

The DRCRSplit expression enables FDM to determine how to interpret this positioning and to assign the correct sign. DRCRSplit expression follows the format DRCRSplit = [Mid Point of the DR and CR columns].

The Amount column in the source file shown begins at column 46 and is thirty-one characters long. The midpoint of the amount field is 16. Everything left of the midpoint is a debit and everything right of the midpoint is a credit. Therefore, the expression in the import format for Amount is `DRCRSplit=16`.



After importing the file, all credit amounts are designated by a negative sign, and debit amounts remain unchanged.

Fill=EuroToUS

The key word `Fill=EuroToUS` is used with the Amount field to trigger a number format conversion from (.,) to (,.) format.

FillL = (Leading Fill)

Use this expression to fill a field with a particular character (leading fill). This expression evaluates the text value read from the file, and if it is shorter than the fill expression the value is padded according to the fill expression. The following example shows a data file with account numbers of different lengths:

```
gltbrp.p                                       25.15.4 Trial Balance Summary                                 Date: 02/2/05
Page: 1                                        Texas Automotive Systems, Inc.                                 Time: 15:42:

Texas xxxxxxxx                                     Reporting Currency:   USD
                                                    Exchange Rate:

                                              Beginning Balance      Period Activity       Ending Balance
Account          Description                       12/31/04                                     1/31/05         Adjustments  Balance
---------------  ------------------------     -------------------  -------------------  -------------------  -----------  -------
1100             Cash In Bank                         283,767.29         135,722.75cr            48,044.54
1100-101-000-00  Dallas National Bank                   2,000.00                  .00             2,000.00
1100-102          Houston Bank One                      9,986.39           3,330.39cr             6,656.00
1100-103          Midland Bank & Trust                                    110,000.00           110,000.00
1100-104          First National Bank                  50,000.00            60,000.00            10,000.00cr
1190             Petty Cash                               500.00                  .00               500.00
1190-101          Sales                                   200.00                  .00               200.00
1190-102          Accounting                              500.00                  .00               500.00
1210             Trade Receivables                    158,857.30         6,113,348.12         6,272,205.42
1220             Tooling & Prototype Rec               71,087.28          71,087.28cr                  .00
1221             Other Non-Trade Rec                         .00           339,000.00           339,000.00
1221-101           N/T Rvbl Dallas                     93,145.54          93,145.54cr                  .00
1221-102           San Antonio                        712,693.72            22,618.80           735,312.52
1221-103           United Parts                        50,817.31          50,817.31cr                  .00
1221-104           Other                              117,169.65                  .00           117,169.65
1290             Interest Receivable                   91,250.00          91,250.00cr                  .00
1290-101           North Am. HQ                        30,815.65            14,943.35            45,759.00
1299             Bad Debt Reserve                     239,602.03cr         239,602.03                  .00
```

The FillL expression is used as shown in the following example:

Field Name	Start	Length	Expression
Source Account	1	15	FillL=0000000
Account Description	18	24	
Amount	90	19	sign=,cr
Source Entity	1	1	Center
Source ICP	1	15	
Source Department	10	3	
Source Prod/Salary	14	2	

The following example shows the file after import using the FillL expression.

G	Source Ent	Source Account	Account Description	Source ICP	Source Departmen	Source Prod/Salary	Amount
--	Center	0001100	Cash In Bank	1100			48,044.54
--	Center	0001190	Petty Cash	1190			500.00
--	Center	0001210	Trade Receivables	1210			6,272,205.42
--	Center	0001221	Other Non-Trade Rec	1221			339,000.00
--	Center	0001503	Prepaid Bank Agency	1503			29,497.00
--	Center	0001527	Prepaid Managment Fee	1527			27,861.05
--	Center	0001542	Prepaid Other	1542			148,359.67
--	Center	0001543	Prepaid Maintenance	1543			173,468.07
--	Center	0001548	Prepaid Rent	1548			5,339.94

Fill = (Trailing Fill)

Use this expression to fill a field with a particular character. This expression evaluates the text value read from the file, and if it is shorter than the fill expression the value is padded according to the fill expression. For example, if the account number on the first line is 103950-, the account number after import is 103950-000.

Factor = (Value)

Use this expression to factor the amount read from the source file by a user-defined value. To define a custom factor, the expression should follow the format `Factor = [Value]`. The following is a sample of a data file to be imported using Factor.

```
CA,10000.005,56701
CA,10000.015,110000
```

```
CA,10000.020.1200
CA,12000.005,6272205
CA, 12000.999,1237241
CA,125000.MI,4714954
CA,125000.GER,12632247
```

The following example shows Factor being used in the Expression field. All amounts imported are multiplied by 1000.

Field Name	Field Number	Number of Fields	Expression
Source Entity	1	3	
Source Account	2	3	
Amount	3	3	Factor=1000

The Factor feature can also be used to multiply by decimals to divide the source values.

NZP (No Zero Suppress)

Use this expression to disable zero suppression during the data load process. By default FDM bypasses accounts in the trial balance that have a zero balance. In certain circumstances you may want to load in all accounts to ensure that values that should be zero are replaced in the target consolidation system. Enter `NZP` in the Expression column of the Amount field to disable nonzero suppression.

Excel

Use this expression to pass a field value to an Excel file for additional analysis or parsing. FDM uses this expression to automate Excel, to pass the field value into cell A1, and to return the value found in cell A2. This allows an Excel worksheet to function as a custom evaluation tool. The value in cell A2 can contain any standard Excel functions as well as custom VBA functions. To define an Excel parsing expression, follow this form: `Excel=[YourFileName].xls`. Set the Excel file to append "Excel" to description that is imported using the value of `=A1&"Excel` for the value of cell A2. The use of the Excel expression appends "Excel" to imported description.

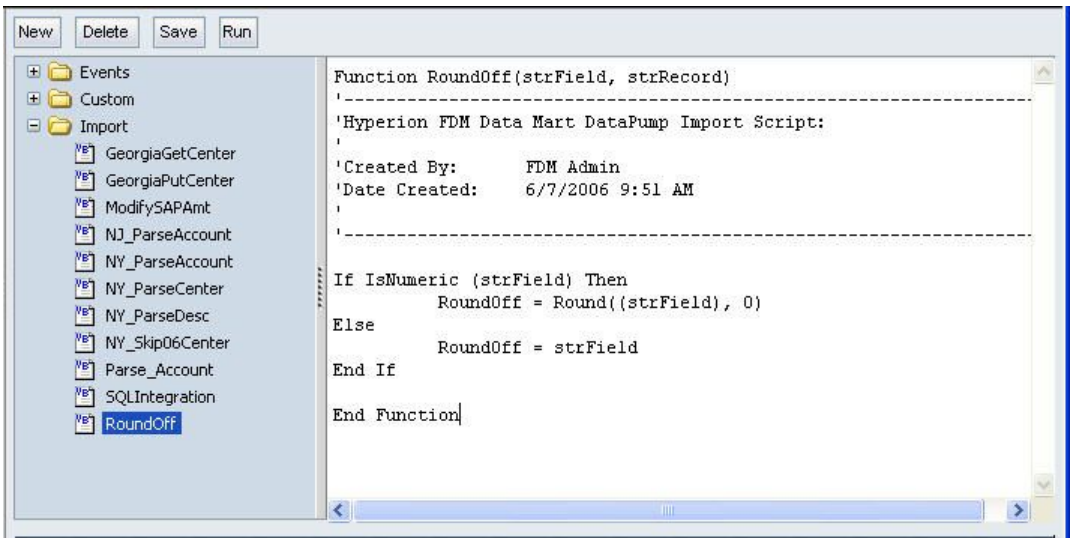
Notes

- The Excel file must be located in the `<application name>\Data\Scripts\Import\` directory.
- This expression may consume a large amount of system resources because Excel must be started and continue to run during the load process. In most cases the `Script=` expression should be used in place of `Excel=`.
- Excel must be installed on the application server.

Script

Use this expression to pass a field value to a FDM Script file for additional analysis or parsing. FDM uses this expression to load and interpret a custom import script that will be run against each line of the import file. Script files are based on the Microsoft® Visual Basic (VB) scripting language and enables end users to extend the power and flexibility of the FDM import process.

The following example show the use of Script to round the amount field and remove decimals.



Note:

This script file must be located in the `<application name>\Data\Scripts\Import\` subdirectory. The file name used for the script file must be the same as the name of the function the file contains. This expression points to a script file that contains a custom function that is run against each line of the import file. This function accepts one argument, the value read for the field from the trial balance file, and returns the value of that argument after the function has operated on it.

Building Import Scripts

If a predefined import expression cannot perform the desired action, then you can create an import script. Import scripts are entered in the Expression column of the row to which the expression will apply.

➤ To build import scripts:

1 Double-click the **Expression** column of the item for which to build the script.

2 Select **Build Script**.

The New Script dialog box is displayed.

3 Click **New**.

4 From Script Type, select **Import (Data Pump)**.

5 Enter a name for the script.

6 Click **OK**.

The Script Editor is displayed.

7 In the script editor, create the import script.

8 Click **Save**.

9 In the **Import Groups** screen, follow the instructions in the "Adding Import Expressions and Import Scripts" on page 82 to add the script to the Expression field.

Building Integration Scripts

Rather than importing a text file, you can build an integration script to import the ledger data directly from an ODBC-compliant data source. Integration scripts can only be run by selecting Script from File Type column in the Import Groups grid (top grid).

➤ To build new integration scripts:

1 Double-click the **Expression** column of the item for which to build the script.

2 Select **Build Script**.

The Script Editor is displayed.

3 Click **New**.

The New Script dialog is displayed.

4 From **Script Type**, select **Import (Integration)**.

5 Enter a name for the script.

6 Click **OK**.

The Script Editor is displayed.

7 In the Script Editor, create the Integration script.

8 Click **Save**.

9 On the **Import Groups** screen, follow the instructions in the section"Adding Import Expressions and Import Scripts" on page 82 to add the integration script to the Expression field.

The following is an example of an integration script that uses an ADO connection to log on to a RDBMS Server database and appends the ledger data to the FDM location's trial balance table.

```
Function SQLIntegration(strLoc, lngCatKey, dblPerKey, strWorkTableName)
'-----------------------------------------------------------------
'FDM Integration Import Script:
'
'Created By:      wladmin
'Date Created:     04/19/2004 2:18:39 PM
'
'Purpose:     Pull data directly from SQL DB
'-----------------------------------------------------------------
Dim objSS        'ADODB.Connection
```

```
Dim strSQL         'SQL string
Dim rs           'Recordset
Dim rsAppend     'tTB table append rs object
'Initialize objects
Set cnSS = CreateObject("ADODB.Connection")
Set rs = CreateObject("ADODB.Recordset")
Set rsAppend = DW.DataAccess.farsTable(strWorkTableName)
'Connect to SQL Server database
cnss.open "Provider=SQLOLEDB.1;Integrated Security=SSPI;Persist Security
Info=False;Initial Catalog=WLDemo;Data Source=DBServerName;"
    'Create query string
    strSQL = "Select * "
    strSQL = strSQL & "FROM tGL "
    'Get data
    rs.Open strSQL, cnSS

       'Check for data
    If rs.bof And rs.eof Then
        RES.PlngActionType = 2
            RES.PstrActionValue = "No Records to load!"
        Exit Function
    End If
  'Loop through records and append to tTB table in location's DB
    If Not rs.bof And Not rs.eof Then
        Do While Not rs.eof
        rsAppend.AddNew
            rsAppend.Fields("PartitionKey") = RES.PlngLocKey
            rsAppend.Fields("CatKey") = RES.PlngCatKey
            rsAppend.Fields("PeriodKey") = RES.PdtePerKey
            rsAppend.Fields("DataView") = "YTD"
            rsAppend.Fields("CalcAcctType") = 9
            rsAppend.Fields("Amount") = rs.fields("dblAmt").Value
            rsAppend.Fields("Desc1") = rs.fields("txtAcctDes").Value
            rsAppend.Fields("Account") = rs.fields("txtAcct").Value
            rsAppend.Fields("Entity") = rs.fields("txtCenter").Value
        rsAppend.Update
        rs.movenext
        Loop
    End If
    'Records loaded
    RES.PlngActionType = 6
      RES.PstrActionValue = "SQL Import successful!"
'Assign Return value
SQLIntegration = True
End Function
```

Using Adapter Import Groups

Rather than importing in a text file, you can use a integration adapter to import the ledger data directly from an ODBC-compliant data source. Configure an integration adapter by selecting Adapter from the dropdown box in the File Type column of the Import Groups grid (top grid).

You can import new source integration adapters into the FDM application by using the Import XML function (File > Import).

Expression Stacking and Processing Order

A single field can use multiple import expressions. To stack expressions, separate the expressions with a semicolon. Consider the order in which the layered expressions are processed.

Expression Stacking Syntax

Field: GL Account

```
Expression List: Script=ParseAcct.txt;Fill=0000000
```

When expressions are stacked, FDM processes the layered expressions in the order listed here. Note that processing order is different for the Amount field

Processing Order

All Fields Except Amount Field

1. Script
2. Excel
3. Fill or FillL

Amount Field

1. DRCRSplit
2. Fill (With EuroToUS Key word only)
3. Script
4. Excel
5. Sign
6. Factor
7. Scale
8. NZP

11 Mapping Tables

In This Chapter

Overview

Mapping tables map the source dimensions members to their corresponding target system dimension members.

Open the Maps screen by selecting Activities > Maps.

The Maps form contains a list used to select a dimension to map. Because FDM only displays valid input values in the target dimension list, you can map source members only to valid input target values.

The Account dimension's mapping table contains an extra field that is used to control sign conversion. This field is titled "-" and contains an option that reverses the sign of the balance of the incoming account.

Note:

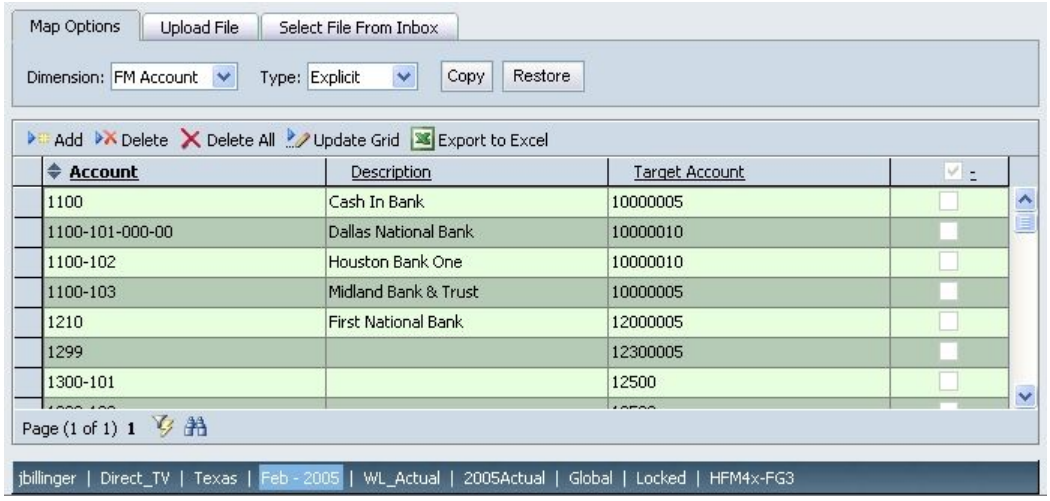
To avoid problems with mappings, do not use the following characters in source or target accounts: Ampersand (&), Asterisk (*), Question Mark (?), Greater Than (>), Less Than (<), or Apostrophe (').

Map Types

GL accounts can be converted with either one-to-one mapping, or wildcard mapping. One-to-one mapping is referred to as explicit mapping. In, Between, and Like maps are wildcard conversions. There are four mapping types available for each dimension—Explicit, Between, In, and Like. Use the Type list to select between the different types of maps.

Explicit Mapping (one-to-one)

Explicit mapping is used in circumstances where the translation is one-to-one (each incoming account is assigned a target account).



Mapping a source account to a target account begins by identifying the account structure of the incoming general ledger account. Therefore, each account segment should be identified and documented. Decide which level the account will be translated into the target consolidation system account.

Example Account Structure

TT = Account Type = Division

PP = Primary Account Code = Location

SS = Sub Account = Department

Sample Accounts

001100 Cash - Disbursement

001101 Cash - Lock Box

223500 Accounts Payable Trade

223501 Accounts Payable Intercompany

This sample account structure could be mapped at two different levels. If the target system account structure is aggregated so that no distinction is made between Cash Disbursement and Cash Lock Box, then the incoming source account could be truncated after the primary account segment. On the other hand, if the target consolidation system account structure has a level of detail equal to the incoming general ledger, the account must be mapped at the more detailed sub account level. The sample maps that follow illustrate this point.

Primary Level Map (TTPP)

```
GL Primary Account, Consolidation Account
0011, Cash
2235, AP
Sub Account Level Map (TTPPSS)
GL Primary Account / Sub-Account, Consolidation Account
001101, Cash.01
001102, Cash.02
223501, AP.01
223502, AP.02
```

Wildcard Mapping

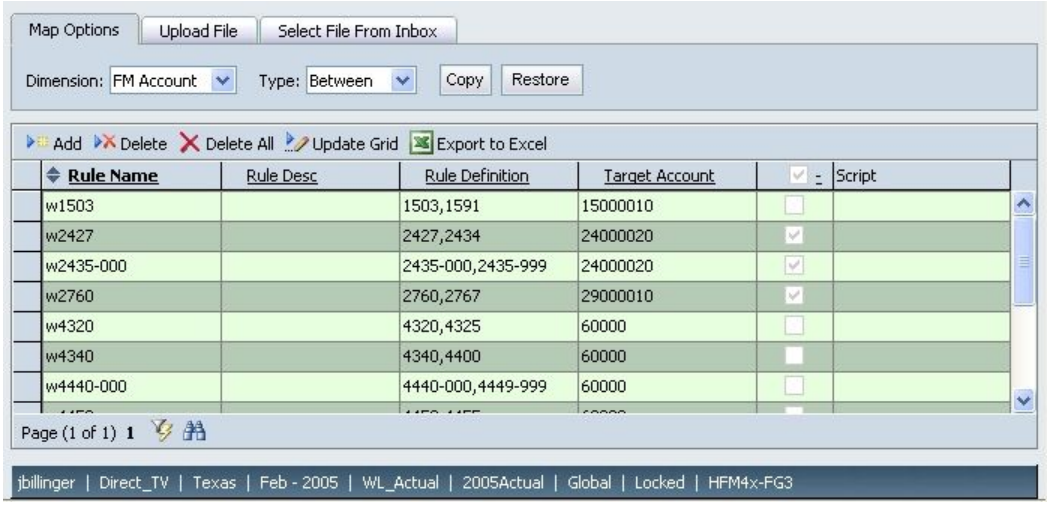
FDM provides the greatest possible flexibility with respect to mapping source records to target records through the use of wildcard translation rules. The use of translation rules allows reduced conversion table maintenance.

All incoming translated records are stamped with the translation rule that was used to provide the record with its target values. This process creates a static map for each location, category, and period combination, ensuring a static audit trail.

FDM offers additional flexibility and power when it comes to account mapping. General ledger accounts can be grouped into a calculated account, and that calculated account could then be mapped to the appropriate target account. There are three basic criteria types that can be used to create a wildcard/calculated account.

The process for creating wildcard accounts is the same as the process as creating standard account mapping. However, a wildcard rule name and rule definition must also be created and a target account must be assigned to the rule.

Between Mapping

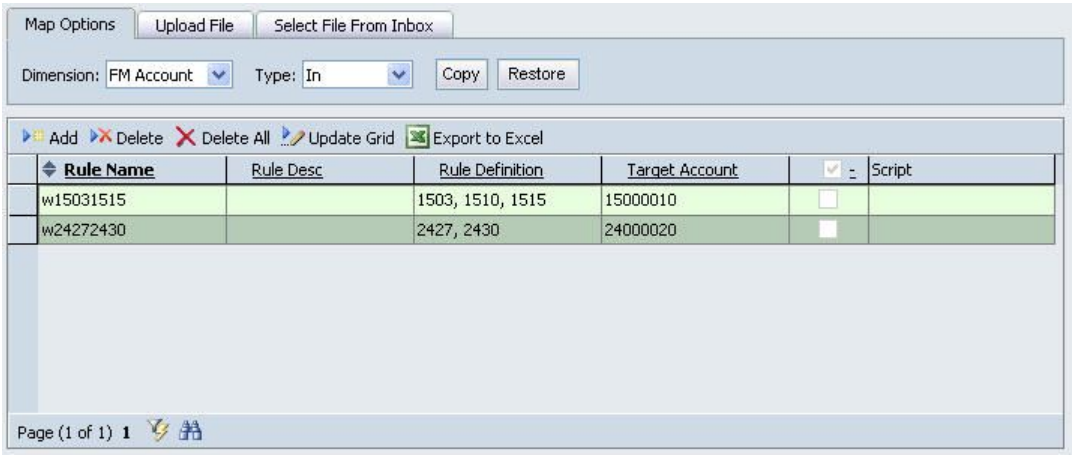


When creating a Between mapping, ensure that each side of the mapping range is the same length. For example: `2000.000, 2991.999` is a valid mapping while `2000.000, 2991` is not.

This is especially important if another mapping would catch any unmapped items. `2000.000, 2991` would fail to include 2991.000 through 2991.999.

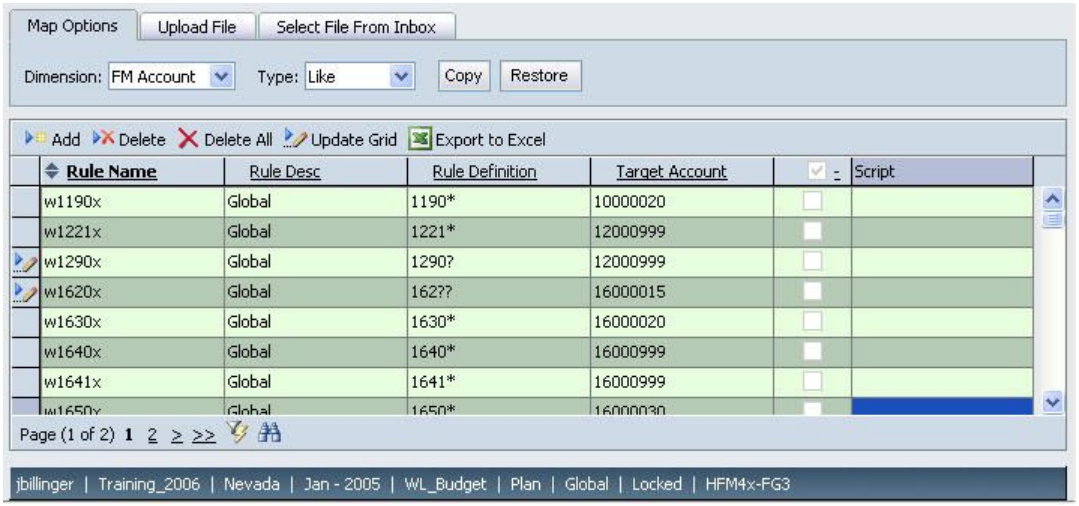
Additionally, Between mappings must be the same length to ensure accounts use the correct map. For example: If mappings `1530, 1540` and `1530000, 1540000` both exist and an account 1540 is imported then it is processed by the second mapping (`1530000, 1540000`).

In Mapping



In mappings allow a list of nonsequential source accounts to be mapped to a single target account. Multiple accounts can be mapped to a single account within a single rule, eliminating the need to create multiple rules in the Explicit Map. The previous example shows that source accounts 1503, 1510, and 1515 are mapped to the target account 15000010.

Like Mapping



Use the wildcard characters "*" and "?" for Like mappings. The asterisk designates a placeholder for any number of characters. In this example, `1190*` will map account 1190, 1190100, and 1190-200 accounts all to the Cash target account.

The question-mark wildcard acts as a placeholder for one character. For example, a source account of `119?` would only map source accounts that contain four characters, and only those that begin with 119.

Automap Wildcarding

FDM allows for the derivation of the target account or entity by allowing you to place wildcard characters (* and ?) in both the source dimension and target dimension of the mapping table. When a mapping table contains an entry that has wildcard characters in the source and target columns the record is considered to be an Automap.

Example Automap

Rule Name	Rule Desc.	Rule Value	Target Account
w0011--	Cash Accts	`0011??`	Cash.??

Sample General Ledger Trial Balance Records

GL Account	Center	Description	Amount
001100	0160000	Cash In Bank	1000.00
001101	0000000	Cash Corp LB	2000.00
001116	0001000	Petty Cash	1000.00
223500	0160000	AP	5000.00

Resulting Record Conversion

GL Account Description	Hyperion Account
001100 0160000	Cash.00
001101 0000000	Cash.01
001116 0160000	Cash.16

Explanation

The criteria for the Automap entry (Like `0011??`) retrieves the first three records from the sample general ledger trial balance. Because the Automap entry contains wildcard characters in the target account field, FDM must search the source account to replace the wildcard characters in the target account with actual characters from the source account.

The characters represented by the two "--" in the source account were used to replace the two question marks (?) in the target account. Next, FDM checks to ensure that the new target

accounts (Cash.00, Cash.01, Cash.16) are valid target accounts. If they are valid, the derived values are assigned to the trial balance as the target account values.

Conditional Mapping

Conditional mapping allows a source dimension value to be mapped to a script expression rather than a hard-coded target system dimension member.

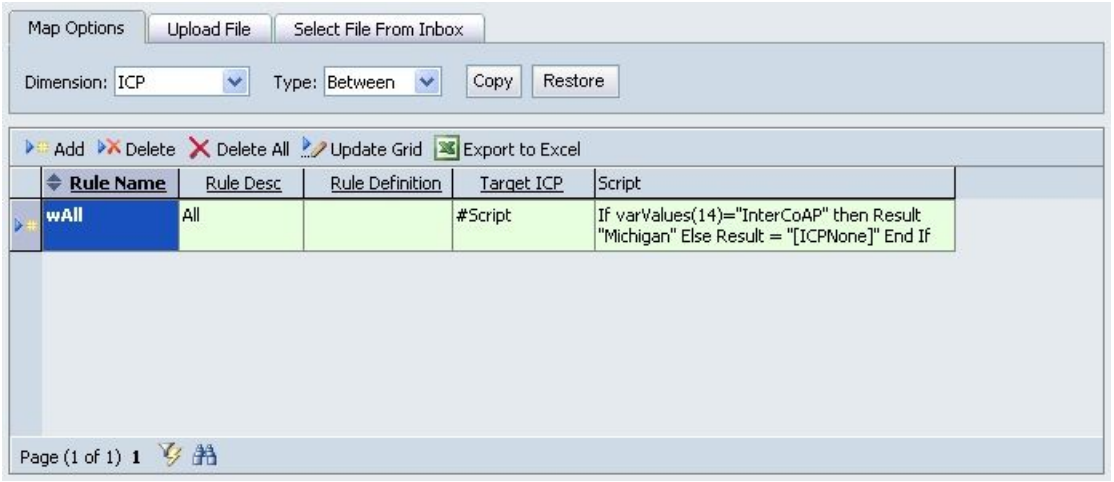
Conditional mapping is only valid for rule-based mapping (Between, In, and Like). Activate conditional mapping by placing value `#Script` in the Target Member column. This triggers the script listed in the Script column to execute.

The script is written in the Script column. Conditional map value logic is written in VBScript and provides all the functionality contained in the VBScript language and access to the running instance of the API. Using conditional mapping in conjunction with dimension processing order allows for mapping to be based on the results of dimension mapping performed on dimensions that have been mapped prior to the current dimensions being processed.

Example

If the Account and Entity dimension process prior to the ICP dimension and the ICP dimension uses a conditional mapping rule, the varValues array contains the mapped values for Account and Entity when the ICP dimension is processed, but the mapped values for all other dimensions are null.

Sample Condition map entry for ICP is shown.



Explanation of Script

ICP dimension conditional Script that checks the mapped value of the ACCOUNT dimension to determine how to map ICP.

```
‘Check element 14 of the varValue array to get converted/mapped account
If varValues(14) = “InterCoAP” Then
    ‘Mapped value = “InterCoAP” map to Michigan
```

```
    Result = “[Michigan]”
Else
    ‘All others, map to none
    Result = “[ICP None]”
End If
```

The following table defines the preset variables are available for use in a conditional script expression:

Values

API—Object reference to running API.

strLoc—String value containing the name of the current location being processed.

VarValues—A variant array containing the record values being mapped and values for proceeding dimensions already mapped.

Variable	Record Value Mapped	Variable Type
varValues(1) =	Current Location ID number	Long Integer
varValues(2) =	Current Category ID number	Long Integer
varValues(3) =	Current Period Key value	Date
varValues(4) =	Current DataView	String, Default = YTD
varValues(5) =	Currency key assigned to the location	String
varValues(6) =	Calculation type for the current record	Long Integer, 9 = Base, 5 = Logic Export, -1 =
Logic	(No Export)	
varValues(7) =	Sign Change flag for current record	Boolean, 0 = False, -1 = True
varValues(8) =	Journal ID for current record	String, Null string if not a journal
varValues(9) =	Amount for current record	Double
varValues(10) =	Supplemental Amount for current record	Double, 0 unless populated by a custom script
varValues(11) =	Account description for current record	String
varValues(12) =	Supplemental Account description for current record	String, Null unless populated by a custom script
varValues(13) =	Source Account	String
varValues(14) =	Target Account	String, Null if this dimension has not processed yet
varValues(15) =	Source Entity	String

Variable	Record Value Mapped	Variable Type
varValues(16) =	Target Entity	String, Null if this dimension has not processed yet
varValues(17) =	Source ICP	String
varValues(18) =	Target ICP	String, Null if this dimension has not processed yet
varValues(19) =	Source UD1	String
varValues(20) =	Target UD1	String, Null if this dimension has not processed yet
varValues(21) =	Source UD2	String
varValues(22) =	Target UD2	String, Null if this dimension has not processed yet
varValues(23) =	Source UD3	String
varValues(24) =	Target UD3	String, Null if this dimension has not processed yet
varValues(25) =	Source UD4	String
varValues(26) =	Target UD4	String, Null if this dimension has not processed yet
varValues(27) =	Source UD5	String
varValues(28) =	Target UD5	String, Null if this dimension has not processed yet
varValues(29) =	Source UD6	String
varValues(30) =	Target UD6	String, Null if this dimension has not processed yet
varValues(31) =	Source UD7	String)
varValues(32) =	Target UD7	String, Null if this dimension has not processed yet
varValues(33) =	Source UD8	String
varValues(34) =	Target UD8	String, Null if this dimension has not processed yet
varValues(35) =	Source UD9	String
varValues(36) =	Target UD9	String, Null if this dimension has not processed yet
varValues(37) =	Source UD10	String

Variable	Record Value Mapped	Variable Type
varValues(38) =	Target UD10	String, Null if this dimension has not processed yet
varValues(39) =	Source UD11	String
varValues(40) =	Target UD11	String, Null if this dimension has not processed yet
varValues(41) =	Source UD12	String
varValues(42) =	Target UD12	String, Null if this dimension has not processed yet
varValues(43) =	Source UD13	String
varValues(44) =	Target UD13	String, Null if this dimension has not processed yet
varValues(45) =	Source UD14	String
varValues(46) =	Target UD14	String, Null if this dimension has not processed yet
varValues(47) =	Source UD15	String
varValues(48) =	Target UD15	String, Null if this dimension has not processed yet
varValues(49) =	Source UD16	String
varValues(50) =	Target UD16	String, Null if this dimension has not processed yet
varValues(51) =	Source UD17	String
varValues(52) =	Target UD17	String, Null if this dimension has not processed yet
varValues(53) =	Source UD18	String
varValues(54) =	Target UD18	String, Null if this dimension has not processed yet
varValues(55) =	Source UD19	String
varValues(56) =	Target UD19	String, Null if this dimension has not processed yet
varValues(57) =	Source UD20	String
varValues(58) =	Target UD20	String, Null if this dimension has not processed yet
varValues(59) =	Archive Document ID number for this record	Long Integer

Map Processing Order

FDM processes maps within a specific order. This mapping order is important if the same source value falls within multiple maps types (for example, Explicit and Between) and also if the same source value falls within the same mapping type multiple times.

Processing Order Across Multiple Mapping Types

FDM maps the same source value when it falls across multiple mapping types in the following order:

1. Explicit
2. Between
3. In
4. Like

Explicit maps override every other mapping type. Between maps override In and Like maps. In maps override Like maps.

Processing Order within the Same Mapping Type

The following summarizes how FDM maps the same source value when it falls within the same mapping type multiple times.

Between Maps

If a source account is valid within multiple Between maps then the source account is assigned to the last Between map in the mapping table.

In Maps

If a source account is valid within multiple In maps then the source account is assigned to the first In map in the mapping table.

Like Maps

If a source account is valid within multiple Like maps then the source account is assigned to the first Like map in the mapping table.

By default, FDM wildcard maps (Like, In, Between) are sorted alphabetically by the Rule Name field. Enable map sequencing in the Locations Form to enables FDM to sort and process maps by a numeric value rather than the sorting alphabetically by rule name.

Copying Mapping Tables

➤ To copy mapping tables:

1 Set the **POV** to the location to be copied.

2 Select **Activities > Maps**.

3 Select **Copy**.

This prompts the user for the target location.

4 Select the location.

5 Click **OK**.

After the target location is selected, FDM only copies the active conversion table from the current location to the target location.

Note:

Only the active dimension conversion table is copied.

Restoring Maps

Each time a file is imported for a category or period, FDM saves a copy of the mapping file associated with the import file.

➤ To restore a map that was used in a previous period/category:

1 Click **Restore**. The **Map Restore** dialog is displayed. The Map Restore Dialog includes mappings for all periods and/or categories used for the current location.

2 Select the map to restore to the current POV.

3 Click **OK**.

Note:

Only the active dimension conversion table is copied.

Recalculating Logic Accounts and Maps

FDM must recalculate a location map whenever a mapping change has been made, or whenever a logic account has been modified. The recalculation process first reprocesses all logic accounts and then remaps all imported source data against the current mapping tables. When FDM detects that a recalculation is required, an orange Calculate icon appears in the lower left corner of the POV bar.

At this point, FDM does not allow the user to load data into the target system until the user validates the ledger data again by clicking the Validate link. You can also click the Calculate icon

directly to force a recalculation. Even after using the Calculate icon, FDM still forces the user to re-validate again before allowing the user to proceed. If the location being recalculated is a parent location, clicking the Calculate icon prompts the user if they would like to force all child locations to be recalculated as well.

A recalculation only occurs for the active category and period set in the POV. This ensures the mapping audit trail remains intact for prior periods and other categories. If a change has been made to a map or logic account that you want applied to other periods or categories then the POV must be changed and a manual recalculation must be performed by clicking Activities>Process Logic/Maps.FDM reprocesses all logic accounts and remap the imported source data against the current map. The recalculation only occurs for the active category and period set in the POV.

Importing Mapping Tables

LedgerLink Conversion Tables

FDM can read and import Hyperion LedgerLink Account and Name conversion tables. LedgerLink Conversion tables must have the file extension (`*.tra`) or (`*.trn`). If ranges are used in the LedgerLink Conversion tables, FDM properly interprets these as wildcards and generate the appropriate conversion table entry. Acceptable field delimiters are pipes (|), semicolon (;), exclamation point (!), and comma (,).

Note:

Because LedgerLink extracts do not include a location or dimension ID, the location in the POV and the dimension for the map to load must be set before importing. FDM imports `.tra` and `.trn` files to the selected POV and dimension.

➤ To import a LedgerLink conversion table:

1. From **Dimension**, select the dimension in which to load the mapping table.
2. Click the **Upload File** tab.
3. Select the mapping table in one of two ways:
   - Browse for the mapping table if it is not already in the application's Inbox.
   - Click Select File from Inbox tab if the map is in the Inbox.
4. Click **Import** to complete the import of the mapping file.

   The imported map is displayed in the mapping table.
5. Repeat steps 1–4 complete loading all the maps for the application.

Hyperion Enterprise Conversion Tables

FDM can read and import Hyperion Enterprise account and entity conversion tables. These files must have the file extension *.asc. Acceptable field delimiters are pipes (|), semicolon (;), exclamation point (!), and comma (,).

Account Conversion Table Format/FDM Equivalent

The word “sub” designates that this account should have its sign changed which is indicated by a check being placed in the Minus column of the account conversion table grid. Other values that can be used to indicate a sign change are Subtract (-), “True,” and “Yes.”

The word “add” designates that this account should not have its sign changed which is indicated by no check being placed in the Minus column of the account conversion table grid. Other values that can be used to indicate a no sign change are Add (+), “False,” and “No.”

Sample Hyperion Enterprise Account Table

```
54300.018!54300.018.USL!SUB 002!002.INP
54300.033!54300.033.US!ADD 005!005.INP
```

LedgerLink TRA / FDM Equivalent

```
0011|CASH.01
```

This line is a one-to-one map

```
0100|-AP.01
```

The (-) sign is used to convert the sign when exported into Hyperion Enterprise.

```
0011>0099|CASH.01
```

The (>) sign specifies a range in LedgerLink and is converted to a “Between” in FDM.

```
0011*|CASH.01
```

The (*) sign specifies a like statement in LedgerLink and is converted to a “Like” in FDM.

```
00110?|CASH.0?
```

This is an Automap account entry and is converted to a “Like” in FDM.

LedgerLink TRN / FDM Equivalent

```
200|DET_INP
```

This line is a one-to-one mapping.

```
220>250|CHI_INP
```

The (>) sign specifies a range in LedgerLink and is converted to a “Between” in FDM.

```
30*|MIL_INP
```

The (*) sign specifies a like statement in LedgerLink and is converted to a “Like” in FDM.

```
40?|DET40?
```

This is an Automap Name entry and is converted to a “Like” in FDM.

Importing Descriptions in LedgerLink format

When importing a map file in the LedgerLink format (`*.tra` or `*.trn`) you can also import account/center descriptions. The description must be placed in the last field within the map file.

```
0011,CASH.01,Bank America Acct
```

Or

```
0011|CASH.01|Bank America Acct
```

Hyperion Enterprise Conversion Tables

FDM can read and import Hyperion Enterprise account and entity conversion tables. These files must have the file extension `*.asc`. Acceptable field delimiters are pipes (|), semicolon (;), exclamation point (!), and comma (,).

Account Conversion Table Format / FDM Equivalent

The word “sub” designates that this account should have its sign changed which is indicated by a check being placed in the Minus column of the account conversion table grid. Other values that can be used to indicate a sign change are Subtract (-), “True,” and “Yes.”

The word “add” designates that this account should not have its sign changed which is indicated by no check being placed in the Minus column of the account conversion table grid. Other values that can be used to indicate a no sign change are Add (+), “False,” and “No.”

Sample Hyperion Enterprise Account Table

```
54300.018!54300.018.USL!SUB 002!002.INP
54300.033!54300.033.US!ADD 005!005.INP
```

Importing Excel maps

Instead of using the Import XLS functionality, you can also import an Excel map using the Maps form by clicking the Import button and selecting the Excel map. The same formatting rules apply within the Excel spreadsheet as it would if exporting by using the Import XLS functionality. Excel map templates with the correct formatting are included in the `Outbox/Templates` directory.

12 Financial Controls

In This Chapter

Use financial control groups to configure financial controls for review locations. Control groups can help to meet certain Sarbanes-Oxley requirements.

Use control groups to maintain and organize certification and assessment information. The Controls Groups form is organized into two major areas—Certification (302) and Assessment (404)—with each area having a question recycle bin. To access the controls groups screen, select Metadata > Controls Groups.

Creating Control Groups

The Certification (302) Group and the Assessment (404) Group can be organized into several different controls groups. Control groups can then be further organized into sections. Each section contains questions.

➤ To create controls groups:

1 Select **Metadata > Controls Groups**.

The Controls Group Properties screen is displayed.

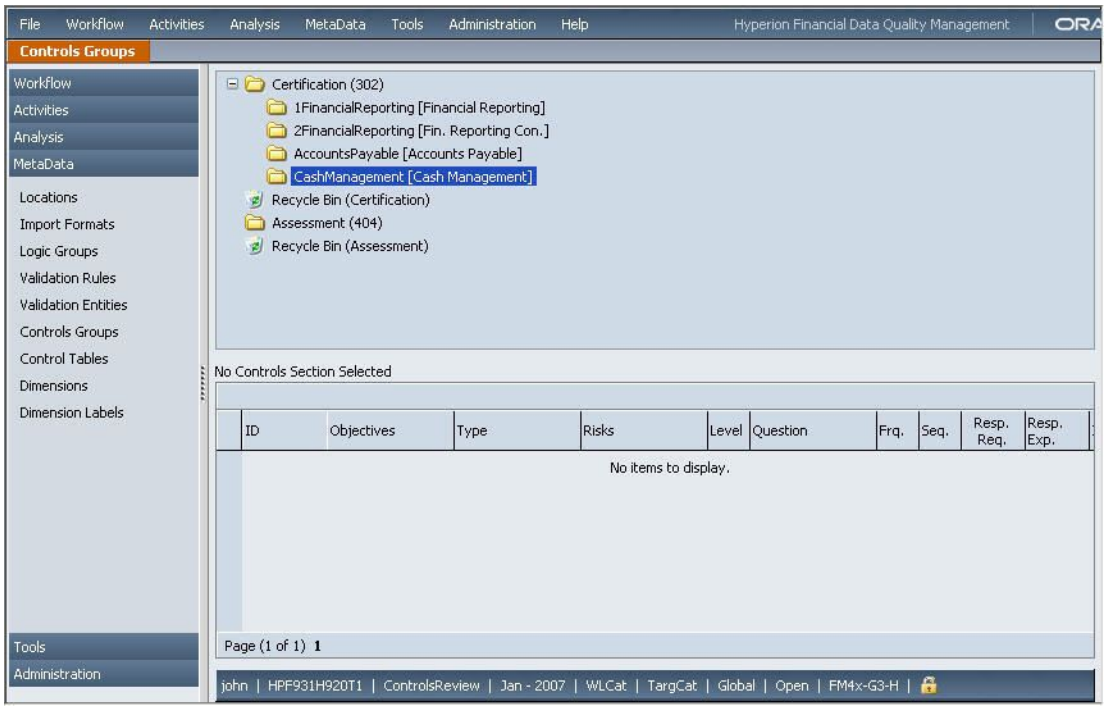
2 Right-click the **Certification or Assessment Group**, and select **Add Controls Group**.

The Add Certification (302) Group or Assessment (404) Group is displayed.

3 Enter the control group name and description.

4 Click **OK**.

A sample of controls groups for Certification (302):



Sections

Sections comprise control groups. Each control group can be organized into as many sections as needed.

➤ To add sections to control groups:

1 Right-click on a control group and select **Add Controls Section**.

The Controls Section Properties dialog box is displayed.

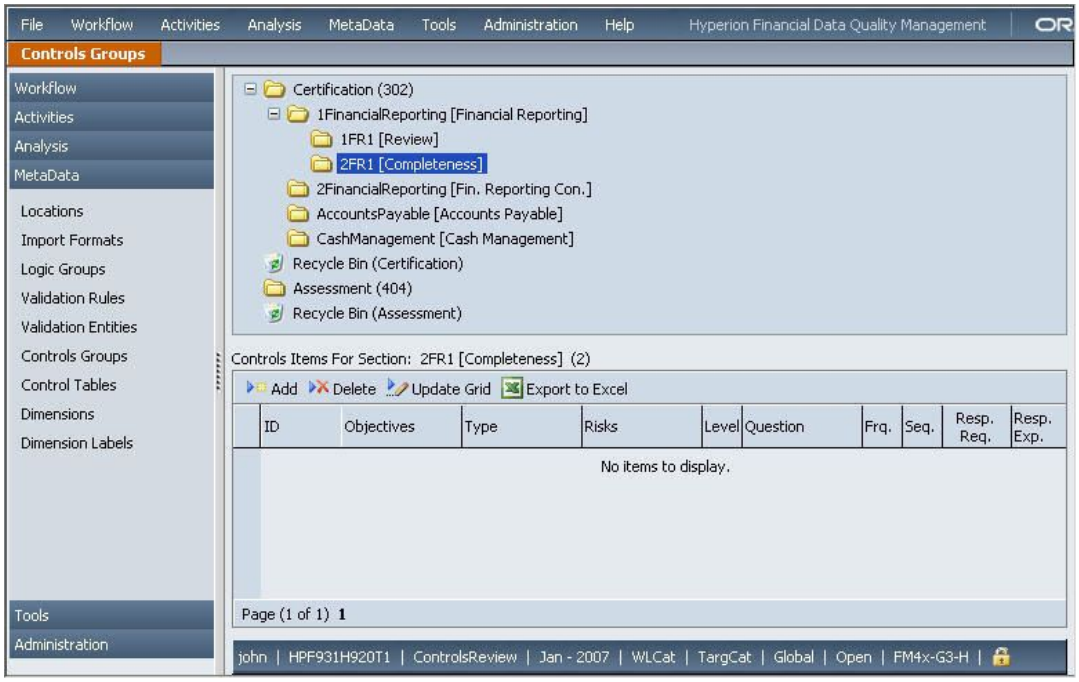
2 Enter the section name and description.

3 Click **OK**.

Note:

All sections must have unique names.

For example, sections for Review and Completeness are contained under 1Financial Reporting as displayed in the following example.



Questions

Sections are composed of questions. Each Section may contain as many questions as required.

➤ To add questions under sections:

1 Highlight a section.

The selected question table is displayed.

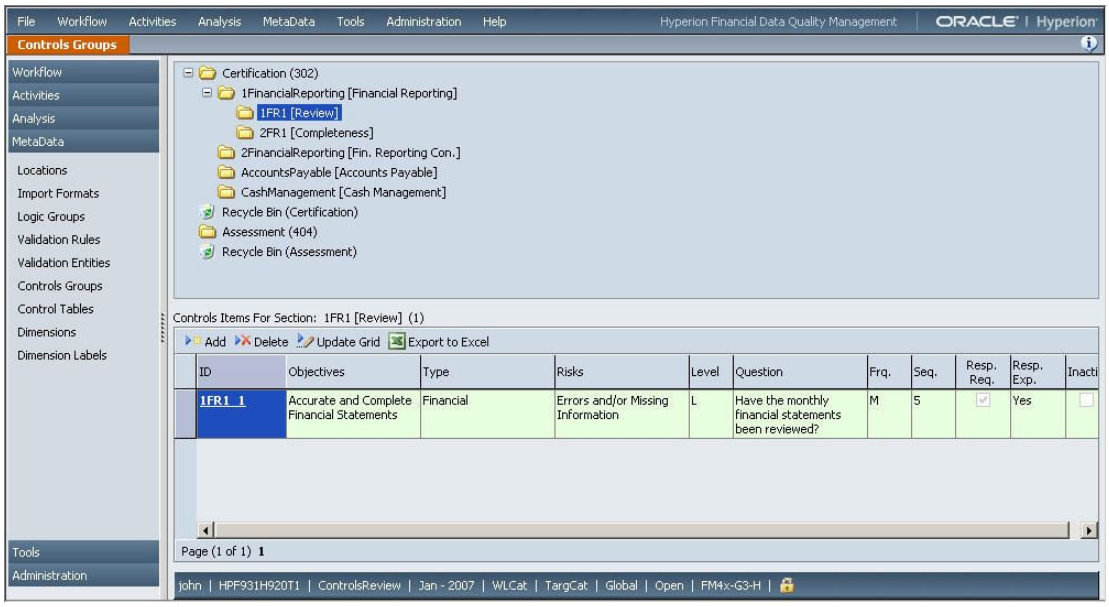
2 Click **Add**.

Enter the question information.

- ID—Each question, regardless of Section MUST HAVE A UNIQUE ID. The ID cannot contain dots.
- Objective—State the desired objective you are trying to obtain with the question.
- Type—Use to specify question type – Financial, Operations or Compliance. Multiple types can be selected.
- Risks—State the risks that could prevent the desired objective from being achieved.
- Level—Use the to specify the level of risk associated with a non-expected response to the question. The risk range is low (L) to high (H).
- Question/Focus Item—State the question or focus item
- Frq. (Frequency)—Specify the frequency that you want the question answered. Use the drop-down to select M – monthly, Q – quarterly, Y – yearly.
- Seq. (Sequence)—Specify the order in which you want to question to be presented

- Resp. Req.—Check if a response to this question is required before the Section review can be completed.
- Resp. Exp.—Use the drop-down to specify the expected response to the question. The expected response may be Yes or No.
- Inactive—Check if you no longer want the question displayed and answered.

For example, a question for the Review Section under 1Financial Reporting is displayed here.



Importing Questions from Excel

You can develop questions in Excel and import them into FDM.

➤ To import questions from Excel:

1 Select **Tools > Import XLS**.

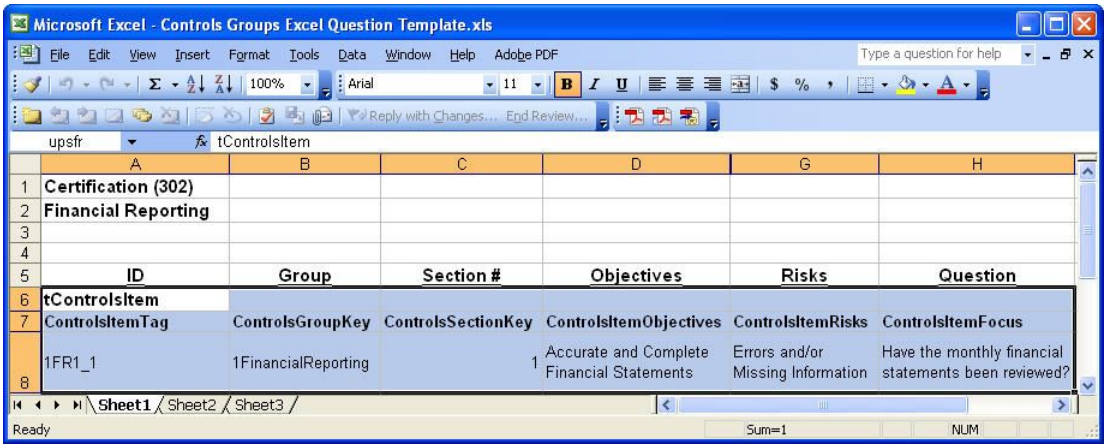
2 Browse to locate the file to import:

- To import from the application's inbox—From Inbox tab, click Select File.
- To import from some other location—From the Upload File tab, click Browse.

3 Select the file and perform an action:

- When importing from Inbox tab—Click OK, then click Import.
- When importing from Select File from Inbox tab—Click OK.

The format for a sample question import template is shown in the following example. The table name is `tControlsItem`. The table is for illustrative purposes and contains only a sample of the available fields. The import range name must be in the format: `upsxxxxx`.



For example, the range name in the following table is `upsfr`.

The available fields include:

- ControlsItemKey—Integer field key (Required)
- ControlsItemTag—Question ID (Required)
- ControlsGroupKey—Group (Required)
- ControlsSectionKey—Section # (Required)
- ControlsItemObjective—Objectives
- ControlsItemType—F (Financial), O (Operations), C (Compliance)
- ControlsItemRisks—Risk associated with question or objective
- ControlsItemRiskLikelihood—1 through 5
- ControlsItemFocus—Questions
- ControlsItemCommentRequired—True or False
- ControlsItemSequence—Order in which question is presented
- ControlsItemHasLinks—True or False (Linked to Certification Question?) (Required)
- ControlsItemInactive—True or False (Required)
- ControlsItemExpectedRespones—Yes or No
- ControlsItemActivationDate—Effective date (Required)
- ControlsItemDeactivationDate—Date Controls item stops being used (Required)

Each field to import must be included in the `upsxxx` range. It is not required that all fields are imported. You may import the general framework of the questions and then update in FDM as needed.

Note:

Format dates as text in the Excel file.

Effective Date

Each question can be set with an effective date. The effective date determines the period in which the question is first presented. The effective date is not required if you want the question to be presented for all periods.

➤ To set the effective date for individual questions:

1 Click the **ID** field link for the question you wish to set an effective date.

2 Enter the effective date.

3 Click **OK**.

Recycle Bin

There are separate recycle bins for certification and assessment questions. Questions that are deleted from a certification section are placed in Recycle Bin (Certification). Questions that are deleted from an assessment section are placed in Recycle Bin (Assessment).

An entire group, section or individual question can be sent to the recycle bin.

Items in the recycle bin can be restored. Items that are deleted from the recycle bin cannot.

Deleting Groups, Sections or Questions

➤ To send groups to the recycle bin:

1 Right-click on a group and select **Delete Controls Group**.

2 Select **OK**.

➤ To send sections to the recycle bin:

1 Right-click on a section and select **Delete Controls Section**.

2 Select **OK**.

➤ To send individual questions to the recycle bin:

1 Highlight the section that contains the question to delete.

2 In the table select a question.

3 Click **Delete**.

The delete icon () appears at the left of the row to indicate that a deletion in pending.

4 Select **Update Grid**.

Restoring Groups, Sections or Questions

➤ To restore groups from the recycle bin:

1 Click the recycle bin (Certification or Assessment) from which to restore a group.

2 Right-click on the group and select **Restore Controls Group**.

3 Click **OK**.

➤ To restore sections from the recycle bin:

1 Click the recycle bin (Certification or Assessment) from which to restore a section.

2 Click the folder named ***Deleted Sections**.

3 Right-click on a section and select **Restore Controls Section**.

4 Select **OK**.

The Select Controls Group dialog box is displayed.

5 From Select Controls Group, choose under which controls group to restore the section.

6 Click OK.

➤ To restore questions from the recycle bin:

1 Click the recycle bin (Certification or Assessment) from which to restore a question.

2 Click the folder named ***Deleted Sections**.

3 Click the folder named ***Delete Questions**.

Deleted questions appear in the lower grid of the Controls Groups screen.

4 From the Restore column, click the recycle icon ().

5 Click **OK**.

The Select Controls Section dialog box is displayed.

6 From Select Controls Section, choose under which controls section to restore the question.

Question Profiles

Question control groups can be organized into question profiles which enable you to assign a different set of questions to each location. One or more control groups can be combined to create profiles. A profile is created as either a Certification profile or an Assessment profile.

➤ To add a profile:

1 Select **Metadata > Locations**.

The Locations screen is displayed.

2 Select the **Financial Controls** tab.

3 Select a location.

4 Click Edit underneath the **Certification (302)** or **Assessment (404)** menu.

The Certification Profile or Assessment Profile dialog box is displayed.

5 Click **Add**.

The Enter Profile Name dialog box is displayed. You can create an individual profile for each location, or use a single profile for multiple locations.

6 Enter the name of the new profile.

7 Click **OK**.

8 From the Certification Profile dialog box, add control groups to the profile:

a. From Available Groups, select a group to add to the profile.

b. Click  to add the group to the Group Profile table.

c. Click OK.

Submitter and Proxy

Each location is assigned a submitter and a proxy (backup submitter). The submitter or proxy submits the certification or assessment after all sections have been reviewed.

If the certification is contingent on data loading, a submitter cannot submit until data has been loaded and all sections have been reviewed.

➤ To assign submitters and proxies to locations:

1 On the **Financial Controls** tab of the **Locations** screen, click the **Submitter** or **Proxy** menu.

2 Select from the available users.

3 Click **Save**.

Users that are configured in FDM security as controls reviewer and submitter are routed directly to the Process Explorer upon logging on.

Reviewers and Proxy

A reviewer and a proxy (backup reviewer) are assigned to each to section. The reviewer or proxy answers the questions in their assigned sections. There are certification reviewers and assessment reviewers.

Each section can have its own reviewer or a single reviewer can be assigned to multiple sections.

➤ To assign reviewers and proxies to sections:

1 In the **Certification Reviewers** or **Assessment Reviewers** grid, double-click the **Reviewer** or **Proxy** column of a section.

2 Select from the available users.

3 Click **Update Grid**.

Users that are set up in FDM security as controls reviewers are routed directly to the Process Explorer desktop upon sign-in.

Risk Flag Level

This field defines when red flags are displayed for questions that are answered with an unexpected response. Questions are assigned a risk level of Low (L) to High (H). Set the Risk Flag Level to indicate when you want an unexpected response to a question to produce a red flag.

For example, a location with a Risk Flag Level set to M (Medium) generates red flags for questions with a risk level of Medium (M) or higher that are answered with an unexpected response. Questions with a risk level of L (Low) to L-M (Low-Medium) do not generate a red flag for an unexpected response on a location with a Risk Flag Level set to M (Medium).

Process Explorer

Access Process Explorer by selecting the Analysis > Process Explorer from the Web client or by clicking on the Notepad icon that is displayed after a location has successfully loaded data. Within Process Explorer, locations are displayed in the left pane, and questions for the selected control area displayed in right pane. The menu above the left frame controls what is displayed on the Process Explorer screen.

Items available in the Process Explorer navigation menu include the following:

- Certification Questions (Review Certification)
- Assessment Questions (Review Assessment)
- Certification Reports
- Assessment Reports
- Pull Assessment Answers from the Prior Period to the Current Period
- Audit Evaluations, Gap Analysis and Action Plans
- Status Reports
- View Journal Entries
- View Map Changes

Adding Memos and Attaching Documentation

Supporting documentation can be attached to any question or focus point. WebLink organizes the documentation into two groups:

- General Documentation—Any user with the appropriate security can attach documents in this group.
- Audit Documentation—Only users who have been granted Auditor status can attach documents in this group. Auditor status is configured in using the FDM security

configuration. The Audit Documentation group is further divided into the following sections:

- Evaluation
- Gap Analysis
- Action Plans

You can add an unlimited number of memos to each question or focus point. Each memo can be assigned ten supporting documents.

➤ To add memos and attach documentation:

1. From the Web client, select **Analysis > Process Explorer**.

   The Process Explorer screen is displayed.

2. In the left pane of Process Explorer, select a Controls Review or Data Load location.
3. From **Section** (right pane) select a section.
4. Click the Memo link (— —) from the memo column ( in the column heading).

   The Memo Item dialog box is displayed.

5. Click **Add** (bottom-left button of Memo Item dialog box).
6. Enter a description of the memo item and click **OK**.
7. In **Memo**, type in the memo.
8. To add attaching documents, click **Add** to the right of an Attachment field, browse to locate the attachment, and click **OK**.
9. Click **Update**.
10. Click **Close**.
11. To close Process Explorer, select **File > Home**.

The Audit group has an additional responsibility area for tracking and reporting on the Audit memo items. You can designate an Assigned To: and Assigned By: user. You can also designate a target date for completion and check when the item is completed. Gap Analysis memo items and Action Plan memo items can be linked to the Evaluation memo item from which they resulted from. A Gap Analysis may be assigned a value for ranking the significance or materiality of the identified gap.

Certification Process

The Certification Process consists of reviewing and submitting user-defined questions at each level in an organization. Each location in a controls structure hierarchy can have a unique group of questions. You can also develop a standard group of questions that is assigned to all locations. Additional question groups can then be assigned to locations that require additional or unique questions.

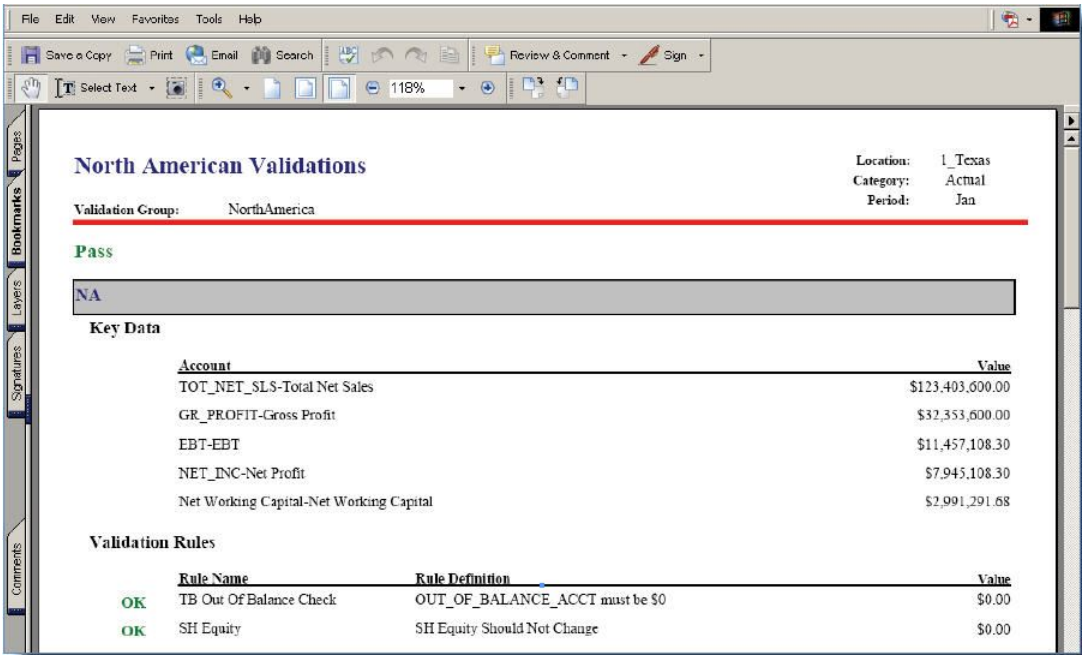
13 Validation Rules and Entities

In This Chapter

Validation rules enable system administrators to define a set of rules and annotations used to enforce data integrity during the data collection process. A set of validation rules are created within a validation rule group. When a validation rule group is assigned to a location, the validation report generates following a data load to the target system. The validation report runs for all entities defined in the validation entities group if also assigned to the location. If no validation entities group is assigned to the location, then the validation report runs for each target entity that was loaded to the target system. The FDM validation reports retrieve values directly from the target system, FDM source data, or FDM converted data.

Validation reports are automatically produced during the data loading process stream, but they can also be run manually. The main function of the reports is to enable reporting locations to review the quality of the information submitted. The results of a validation report for locations are analyzed by FDM and an overall status entry is made in the process monitoring table. In order for a location to show a status of True for validation, each rule on the validation report must pass. If a rule is used for warning purposes only, then no rule logic is assigned to the row.

Click on the Check icon in the FDM toolbar or select Workflow > Check to display the Load Validation report. This report is based on the validation rules that have been assigned to the current location. The following is an example of a validation report.



Creating a Validation Rule Group

➤ To create validation rule groups:

1 From within the Web client, select **Metadata > Validation Rules**.

The Validation Rules screen is displayed.

2 Click **Add** located above the top grid.

A new row is added to the top grid.

3 Double click in the **Validation Group** column, and enter a name for the validation group.

4 Double-click in the **Description** column, and enter a description for the validation group.

5 Click **Update Grid**.

Note:

A date will automatically be added in the Date Created column of the new validation group.

Creating Validation Rules

Each validation rule represents a line on the validation report.

➤ To create validation rules:

1 From within the Web client, select **Metadata > Validation Rules**.

The Validation Rules screen is displayed.

2 In the top grid, select the validation rule group in which to add validation rules.

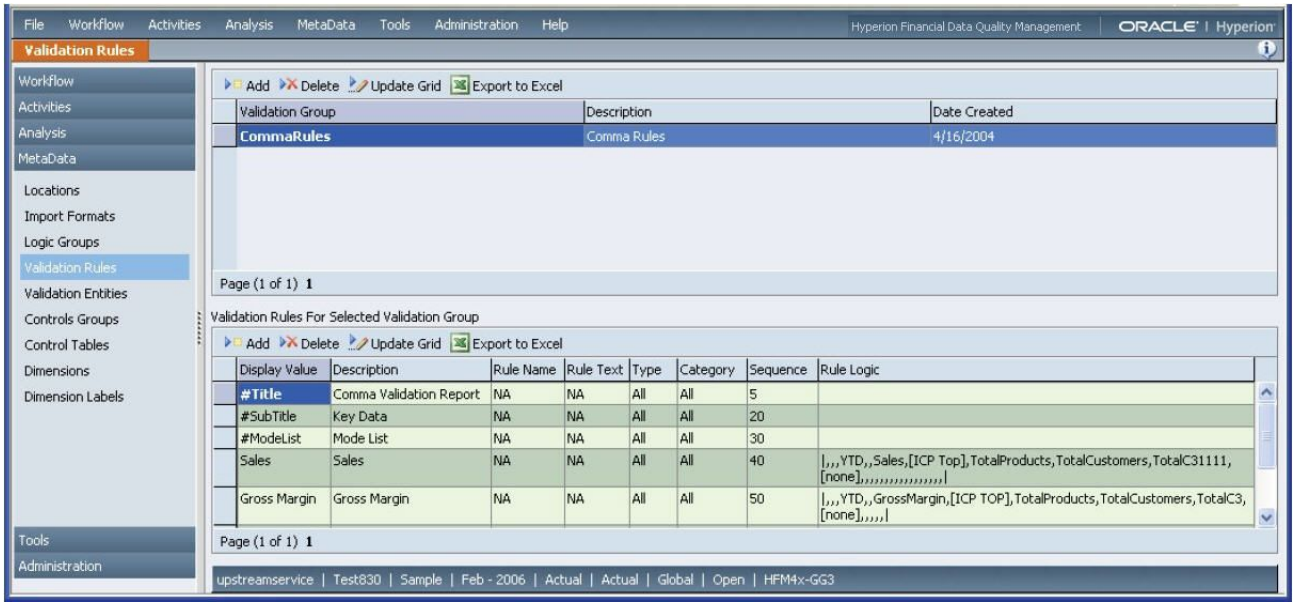
3 Click **Add** located above the bottom grid.

A new row is added to the bottom grid.

4 Enter the validation rule information in each of the fields.

Detailed information regarding specific validation rule information follows this procedure.

5 Click **Update Grid**.



A validation rule consists of the following fields:

- Display Value
- Description
- Rule Name
- Rule Text
- Type
- Category
- Sequence
- (Optional) Logic statement for the rule

Display Value

This column controls how FDM formats each row of data on the validation report. It can be used to select a target account, select a report format code, or create a custom expression.

The contents of this column are processed under the following rules:

- If the field contains a report format code, then no value lookup is attempted.

- If the column contains anything other than a report format code, then the result of the custom expression (rule logic) is displayed in the Value column. This column is limited to 75 characters.

Double-click in the Display Value field to display the menu that contains three items – Zoom, Browse for Target Account, and Select Format Code.

While a rule is in Edit mode, the Browse, Format Code, and Zoom link in the Display Value column are active.

Zoom...

This option opens the cell text in a text editor. This is helpful when entering or editing large amounts of text.

Browse for Target Account...

Select this option to show the Validation Rule Target Account Lookup screen. From there, select the account from the list of target system application accounts to enter it into the Validation Rules grid. In addition to placing the account number in the Display Value field, a custom expression is required to have the value displayed.

Select Format Code...

In addition to selecting target accounts, you can enter format codes in the Target Account column. These codes are used to control the presentation of the validation report. Use this option to select the desired code from the Select Format Code screen. The following table details the format codes that can be used in a validation rule.

Code	Description
`#ModeList`	Sets the report mode to Listing. This mode displays the Display Value, Description, and Amount fields in the report.
`#ModeRule`	(Default) Sets the report mode to Rule. This displays the Rule Name, Rule Text, and Amount in the validation report. The report evaluates the expression in the Rule Logic column and test the true or false condition of the rule. The status of the rule (OK or Error) is displayed on the report for each rule. The Target Account and Description columns are not displayed in this mode.
`#Title`	This code inserts a title line into the report. When `#Title` is used, the text entered in the Description field is displayed as a title.
`#Subtitle`	This code inserts a subtitle line into the report. When `#Subtitle` is entered in the Description field is displayed as a subtitle group. Subtitles are used for report grouping and are inserted into the report navigation tree on the validation report.

Description

This column is used to display account descriptions, titles, or subtitles. This field is only displayed on the validation report if the report is in `#ModeList` mode.

Example Description

`Out-of-Balance Account`

Rule Name

This column is used to store an identifier for a validation rule. This field is only visible in the report if the report is in `#ModeRule` mode. This value should be unique and easy to identify.

Example Rule Name

Out-of-Balance Check

Rule Text

This field is used to define a basis or logic behind a rule. The text in this column is the primary statement shown on the validation report. This field is displayed on the validation report only when the report is in `#ModeRule` mode.

Example Rule Text

This account must be between [+10 and –10]

Type

The Type field corresponds to the Type field in the Validation Entities form. Select All for this rule to be displayed for all entities on the validation report. In the Type field, select whether this rule should only be displayed on the validation report for base entities, parent entities or all entities.

- If the validation rule is designated as Parent, then the validation rule is only displayed in the validation report for entities whose Type field is set to Parent or All on the Validation Entity screen.
- If the validation rule is designated as Base, then this validation rule is only be displayed in the validation report for entities whose Type field is set to Base or All on the Validation Entity screen.
- If a validation rule is designated as All, then the validation rule is displayed in the validation report for all entities, regardless if the entity was flagged as a Base or Parent entity on the Validation Entity screen.

Category

In the Category field, select a FDM category to restrict the validation rule to a specific FDM category. The validation rule is only displayed on the validation report if the FDM category selected in this field is the same as the active FDM category set in the POV. To display the validation rule on the validation report regardless of the category set in the POV, select All.

Sequence

The value in this column controls the order of rule processing. The sequence number determines the order in which the format codes and rules are processed. It is good practice to increment the sequence number by 10 to provide a range for insertion of new format codes and rules.

Rule Logic

The Rule Logic column is used to create a multi-dimension lookup, a validation rule expression, or a more advanced validation rule function. Multi-dimension lookups display a value in the Amount field on the validation report. Validation rule expressions and functions are usually used to validate target system account balances. The Rule Logic field is only processed if the report is in `#ModeRule` or `#ModeList` mode.

Validation rule expressions and validation rule functions use standard expression capabilities of the Microsoft VB Script language. Validation rule expressions and functions test for a true or false condition. After rule logic is processed, FDM evaluates the result of the rule and the rule is flagged in the validation report as either passing or failing.

Validation Rule Expressions

Validation rule expressions are normally used to validate target system account balances. Rule expressions return a True or False result. You can use the Hyperion Enterprise lookup function (when integrating with Hyperion Enterprise) or the multi-dimension lookup function (when integrating with multi-dimension target systems). For more advanced functionality use the advanced rule functions.

Example 1

If the target account “Balance” is greater than $10 and less than $10, then the rule is flagged “OK” on the validation report. If the account balance is outside this range, then the rule is flagged “Error” on the validation report.

```
|Balance| >= -10.00 AND |Balance| <= 10.00
```

Example 2

The following example tests to verify that target account “1000.100” + $100,000 is equal to target account “1000.400”.

```
|1000.100| + 100000 = |1000.400|
```

Enterprise Lookup Function

The following validation rule expression example uses the Hyperion Enterprise Lookup function. If the target account balance is greater than –$10 and less than $10, then the rule is flagged “OK” on the validation report. If this account balance is outside this range, then this rule is flagged “Error” on the validation report.

Example 1

```
|Balance| >= -10.00 AND |Balance| <= 10.00
```

Example 2

This example verifies that target account “1000.100” + $100,000 is equal to target account “1000.400.”

```
|1000.100| + 100000 = |1000.400|
```

Multi-Dimension Lookup Function

The Multi-dimension Lookup function is used to retrieve account values from the target system, FDM source data, or FDM converted data. Multi-dimension lookups can be used in the Rule Logic field and the Display field on the Validation Rules form.

Rule Data Sources

FDM can retrieve data from three sources:

- Target system data values—Retrieves values from the target system.
- FDM source data values—Retrieves values from the data that was loaded into FDM after being mapped to the target members.
- FDM converted data values—Retrieves pull values from the data that was loaded into FDM.

Target System Data

This format enables FDM to retrieve values from the target system for any dimension. This rule type uses the pipe character (|) as the rule beginning and ending character for the rule. The following are the parameters that can be used when referencing a target system account. Unless otherwise specified, all parameters are optional.

```
|Scenario, Period, Year, View, Value, Entity, Account (Required), ICP,
Custom1, Custom2,Custom3, Custom4, Custom5, Custom6, Custom7, Custom8,
Custom9, Custom10, Custom11, Custom12, Custom13, Custom14, Custom15,
Custom16, Custom17, Custom18, Custom19, Custom20|
```

The following examples illustrate ways that values can be retrieved from a target system. Commas must be used as placeholders for dimensions not referenced.

Example 1

Look up the value of the target account “Balance” for the current target period and target scenario (category) set in the POV, and for each entity in the FDM validation entity group that is assigned to the location. In this example, the rule passes validation if the target account is within $10.

```
|,,,,,Balance,,,,,,,,,,,,,,,,,,,,,| > -10.00 AND
|,,,,,Balance,,,,,,,,,,,,,,,,,,,,,| < 10.00
```

Example 2

Look up the value of the target account “Balance” for the supplied dimensions.

```
|Actual,March,2002,YTD,Ohio,Balance,Michigan,Engines,Ford,Trucks,
[None],,,,,,,,,,,,,,,,USD| > 0
```

Example 3

Look up the value of the target account “Balance” for the supplied dimensions and the prior period.

```
|Actual,-1,2002,YTD,Ohio,Balance,Michigan,Engines,Ford,Trucks,
[None],,,,,,,,,,,,,,,,USD| > 0
```

Example 4

Look up the value of the target account “Balance” for the active target scenario (category) set in the FDM POV, the prior target period, and for each entity in the FDM Validation Entity group that is assigned to the location. All missing custom dimensions are defaulted to “[None]”, the ICP dimension defaults to “[ICP-None],” the Year dimension defaults to the current year set in the POV, the Currency dimension defaults to “0”, and the View dimension defaults to “YTD.”

```
|,-1,,,,Balance,,,,,,,,,,,,,,,,,,,,,,| > 0
```

FDM Source Data

Retrieves values from the data that was loaded into FDM after being mapped to the target member. This type of rule uses the tilde character (~) as the rule beginning and ending character for the rule. The following are the parameters that can be used when referencing a target system account. All parameters are optional unless otherwise specified. Parameters designated “UD#” are “User Defined.”

```
~FDM Category, FDM Period, Year (Field Not Applicable), FDM View, FDM
Location, Source Entity, Source Account(Required), Source ICP, Source UD1,
Source UD2, Source UD3, Source UD4, Source UD5, Source UD6, Source UD7,
Source UD8, Source UD9, Source UD10, Source UD11, Source UD12, Source UD13,
Source UD14, Source UD15, Source UD16, Source UD17, Source UD18, Source
UD19, Source UD20~
```

FDM Converted Data

Retrieves pull values from the data that was loaded into FDM. This type of rule uses the grave accent character (`) as the rule beginning and ending character for the rule. The following are the parameters that can be used when referencing a target system account. All parameters are optional unless otherwise specified.

```
`FDM Category, FDM Period, Year (Field Not Applicable), FDM View, FDM
Location, Entity, Account(Required), ICP, Custom1, Custom2, Custom3,
Custom4, Custom5, Custom6, Custom7, Custom8, Custom9, Custom10, Custom11,
Custom12, Custom13, Custom14, Custom15, Custom16, Custom17, Custom18,
Custom19, Custom20`
```

Note:

The beginning and ending characters for each of the rule data sources can be set to any character. Change the characters in the Configuration Settings screen.

Validation Rule Functions

In the Rule Logic field you can use advanced functions that offer additional flexibility when evaluating target system data. FDM validation rules can use VB Script procedures. Therefore, you can set two custom string message fields and two numeric switch fields that can be evaluated by a script or custom report. Also, you can perform validations against data sources other than the target system (use a FDM logic account value, for example) in a rule.

To use a function in a validation rule, the rule must begin with the key word `Fun`. This keyword value triggers the API to interpret the rule as a function rather than an expression. To set the return value of the function, set the keyword `Result` to True or False. The validation rule is flagged "OK" on the report if the function returns a value of True. The validation rule is flagged "Error" on the validation report if the function returns a value of False.

Validation Rule Function Parameters

Validation rule functions use the following parameters that can be evaluated to determine if a particular validation rule should pass for the current POV:

- API—The main FDM API programming object. It allows the use of all FDM data values and internal API functions.
- strEntity—Target Entity
- strTargCat—Target Category
- strTargPer—Target Period
- strTargYear— Target Year
- strTargFreq—Target Frequency

Optional Rule Logic Properties

You can set properties of the RES object which, in turn, sets values fourteen custom fields that reside in the `tDataCheck` table. Access the user-defined fields by using scripts. User-defined field can be used on reports for warning messages and flags.

An example of a Rule Logic function follows.

```
Fun:
If strEntity = "TEXAS.GL" Then
        'Rule not required
        Result = True
        'Set Messages and Switches
        RES.PstrCheckMessage1 = "Rule does not apply to " & strEntity
        RES.PstrCheckMessage2 = "Do not fret!"
        RES.PblnCheckWarning = True
        RES.PblnCheckClearData = True
Else
        'Check Cash
        IF |1000| <= 0 Then
            Result = True
            'Set Messages and Switches
            RES.PstrCheckMessage1 = strEntity & " is Good!"
```

```
            RES.PstrCheckMessage2 = “2”
        Else
            Result = False
        ‘Set Messages and Switches
        RES.PstrCheckMessage1 = strEntity & “ is Bad!”
        RES.PstrCheckMessage2 = “2”
        End If
End If
```

Validation Editor

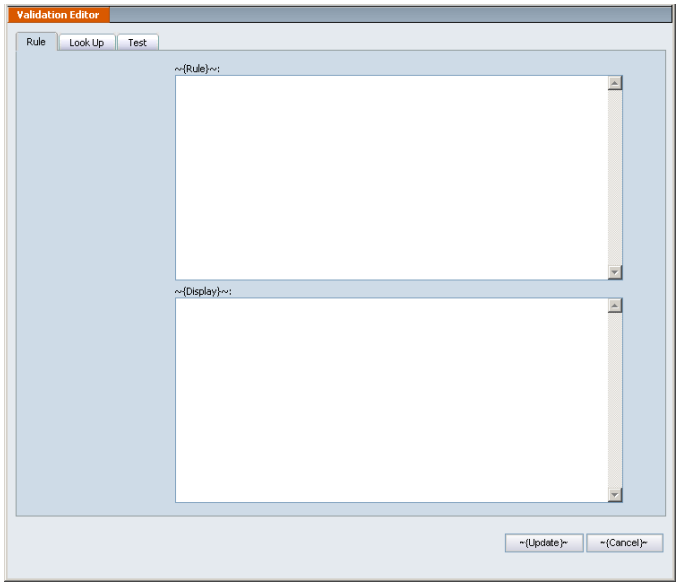
The Validation Rule Editor enables faster and easier creation of validation rules. It assists with development of rule logic associated with items defined under the `#Modelist` or `#Moderule` format codes. You can create validation rules without having to know what each of the FDM retrieve fields represent. Double-click in the Rule Logic field on the Validation Rules form and select Validation Editor to open the Validation Rule Editor.

The Validation Rule Editor is accessible only from the Web client, and consists of three tabs:

- Rule tab
- Look Up tab
- Test tab

Rule Tab

The Rule tab is used to build the validation rule logic expression. The Rule tab displays the following in the Validation Editor:

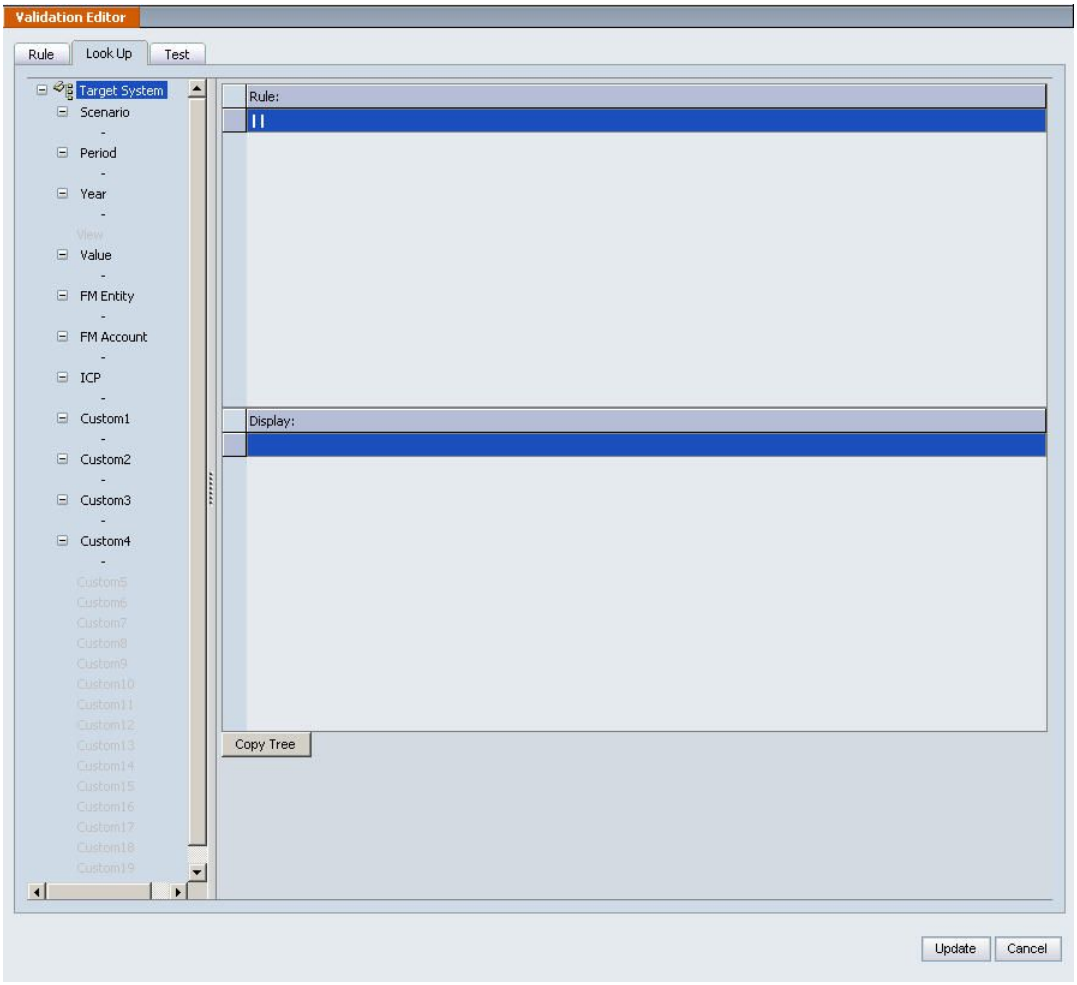


- Rule—Builds the rule to be tested in this area.
- Display—Builds the value to be returned to the validation report. This amount may be different from the value to be tested in the rule.

- Update—Saves the contents of the Rule and Display fields to the validation rules being modified.
- Cancel—Closes the Validation Editor without saving changes.

Lookup Tab

The Lookup tab enables users to select retrieve dimensions directly from the target system. This ensures that each of the required dimensions is entered correctly and in the proper order for the retrieve to function correctly.



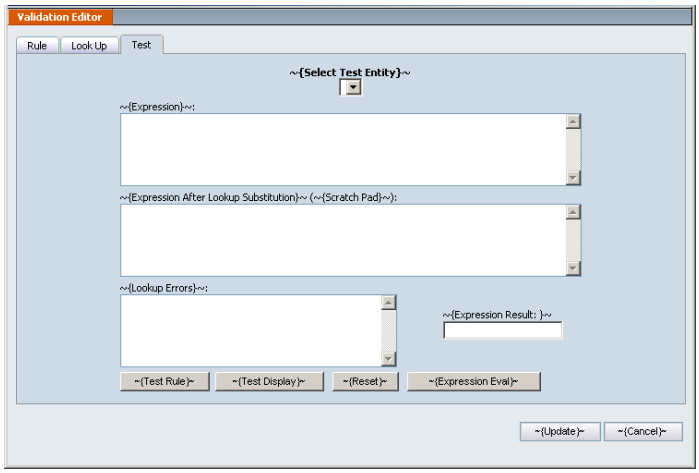
The Look Up tab displays the following in the Validation Editor:

- Rule—Displays the rule that is being created. The rule is composed of Retrieve functions and other VBA script calls. Each Retrieve function is displayed on a separate line from any VBA script functions. This allows individual Retrieve functions to be selected for use with the rule builder.
- Display—Shows the display rule that is being created. The rule is composed of Retrieve functions and other VBA script calls. Each Retrieve function is displayed on a separate line from any VBA script functions. This allows individual retrieve functions to be selected for use with the builder.

- Rule Intersection—When a Retrieve function is selected from the Rule or Display section, the Rule Intersection tree is enabled. This section enables users to enter the values for each dimension in the Retrieve. The text for each active dimension is colored blue while those that are not active are gray. Double-click a dimension to display the list of valid selections.
- Copy Tree/Paste to Editor—Copies the values in all dimensions for the selected rule. After copying, the button text changes to "Paste to Editor." To paste the values to a different rule, select the new rule after copying and then click the Paste to Editor button.

Test Tab

The Test tab is used to test the current rule/display value to ensure that the proper results are be generated before putting the new rule into production. The Test tab shows the following options in the Validation Editor.



- Select Test Entity—Select the entity to test from this list.
- Expression—Displays the rule being tested.
- Expression After Lookup Substitution (ScratchPad)—Returns the results of the expression being tested after the lookup has been performed. This field can also be used as a temporary storage by highlighting text in the field and right-clicking.
- Lookup Errors—Displays any errors that were found in the rule being tested.
- Expression Result—Displays the results of the rule test (True or False). If the location being tested with the current rule passes the validation rule.
- Test Rule—Runs the current rule in the Rule section. When clicked, FDM displays a selection of all available target locations. This selection is then be used to pull values and return the rule results.
- Test Display—Runs the current rule in the display section. When clicked, FDM displays a selection of all available target locations. This selection is then used to pull values and return the display results.
- Reset—Resets all fields in the Rules tab of the Validation Editor screen.
- Expression Eval—Calculates expressions in the Expression After Lookup Substitution (Scratch Pad) field.

Validation Entity Groups

Validation entity groups consist of one or more target system entities that you wish to consolidate and displayed on the validation report. Activate a validation entity group by assigning them to a location.

Open the Validation Entities screen by selecting MetaData > Validation Entities from the FDM Web client.

Defining Validation Entities

Multiple validation entities can be assigned to one validation entity group. See the following topics for descriptions of each validation entity field.

Organization / Parent Entity

For Hyperion Enterprise target systems, select the organization in which the entity is consolidated. For all other target systems, select the parent entity of the entity to be consolidated. This field has no effect if the Consolidate option is not selected.

Entity

The Entity field contains the target entity to be consolidated and displayed on the validation report. If the Consolidate option is selected, this entity is consolidated before being included on the validation report.

Consolidate

This switch is used to signal that a consolidation should be run for the entity prior to being displayed on the validation report.

Start Period

The start period controls how FDM executes the consolidation process within the target system. If the start period is set to "0," then only the current period is consolidated. If the start period is set to a value that is greater than the current period, then only the current period is consolidated. Setting the start period value to "1" forces FDM to consolidate for periods 1 to the current period.

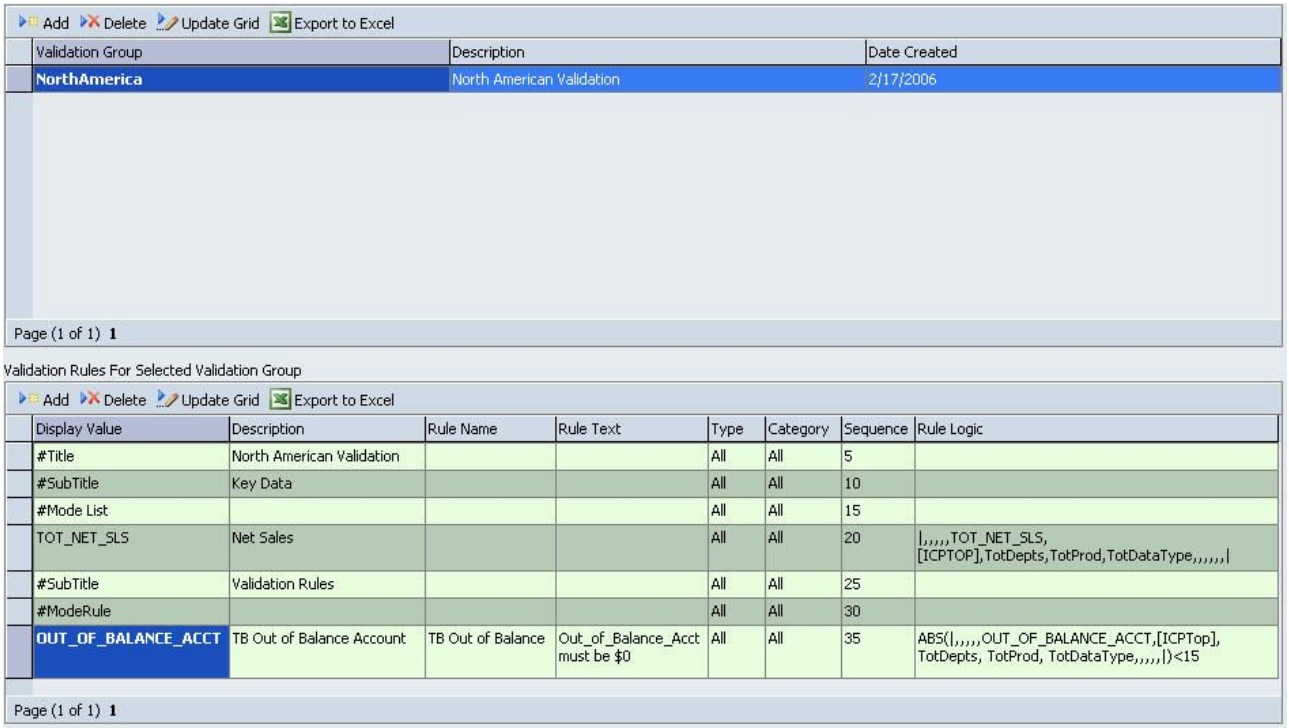
Entity Type

The Type field in the Validation Entity form corresponds to the Validation Rule Type field in the Validation Rules form. In the Type field, select whether the entity is a base entity or parent entity. Select *All* (default) if you do not want to distinguish whether the entity is a base or parent and want this entity to always display in the validation report for each validation rule.

For example, if an entity is designated as Parent, then it is only displayed in the validation report for validation rules whose Type field is set to Parent or All.

If an entity is designated as Base then it is only displayed in the validation report for validation rules whose Type field is set to Base or All.

If an entity is designated as All, then this entity is displayed in the validation report for all validation rules regardless of the validation rule type.



Add Delete Update Grid Export to Excel

Validation Group	Description	Date Created
NorthAmerica	North American Validation	2/17/2006

Page (1 of 1) 1

Validation Rules For Selected Validation Group

Add Delete Update Grid Export to Excel

Display Value	Description	Rule Name	Rule Text	Type	Category	Sequence	Rule Logic
#Title	North American Validation			All	All	5	
#SubTitle	Key Data			All	All	10	
#Mode List				All	All	15	
TOT_NET_SLS	Net Sales			All	All	20	|,,,,,TOT_NET_SLS, [ICPTOP],TotDepts,TotProd,TotDataType,,,,,,|
#SubTitle	Validation Rules			All	All	25	
#ModeRule				All	All	30	
OUT_OF_BALANCE_ACCT	TB Out of Balance Account	TB Out of Balance	Out_of_Balance_Acct must be $0	All	All	35	ABS(|,,,,,OUT_OF_BALANCE_ACCT,[ICPTop], TotDepts, TotProd, TotDataType,,,,,|)<15

Page (1 of 1) 1

The Type field in the Validation Entity form (top grid) corresponds to the Validation Rule Type field in the Validation Rules form (bottom grid).

On Report

Select this option to include this entity on the validation report. If this field is not selected, and the consolidate field is selected, then the entity is consolidated, but not shown on the validation report.

Sequence

This column controls the order of entity processing. The sequence number dictates the order in which the entities are consolidated and included on the validation report.

It is good practice to increment the sequence number by 10 to provide a range for insertion of new entities.

14 Process Flow

In This Chapter

Importing Source Data

After an import format has been defined and assigned to a location, you can then import a source file. From with the FDM Web client, select Workflow > Import to display the Import screen. When this form is open, FDM uses the current POV information to determine the location, category and period that is loaded during a file import.

Load Method (Import Type)

The Import screen features an Import Type menu with two options:

- Replace—(default) Prior to loading a data file, existing data is deleted from the current location, category, and period.
- Append—Appends the new data file to the existing data for the current location, category, and period. A FDM append does not merge the data. Appended lines of data do not affect existing lines.

Import Process Stream

➤ To import source data:

1 From the FDM Workbench, select **Workflow > Import**.

2 Select the import method:

- To upload the file from the application's `Inbox`, click the Select File From Inbox tab.
- To upload the file from a location other than the application's `Inbox`, select the Upload File tab.

3 Click **Select File** or **Browse** (depending on the tab selected in step 1) to locate and select the GL file to import.

4 On the Import form, click **Import**.

FDM uses the import format assigned to the location. This action starts the following processing stream:

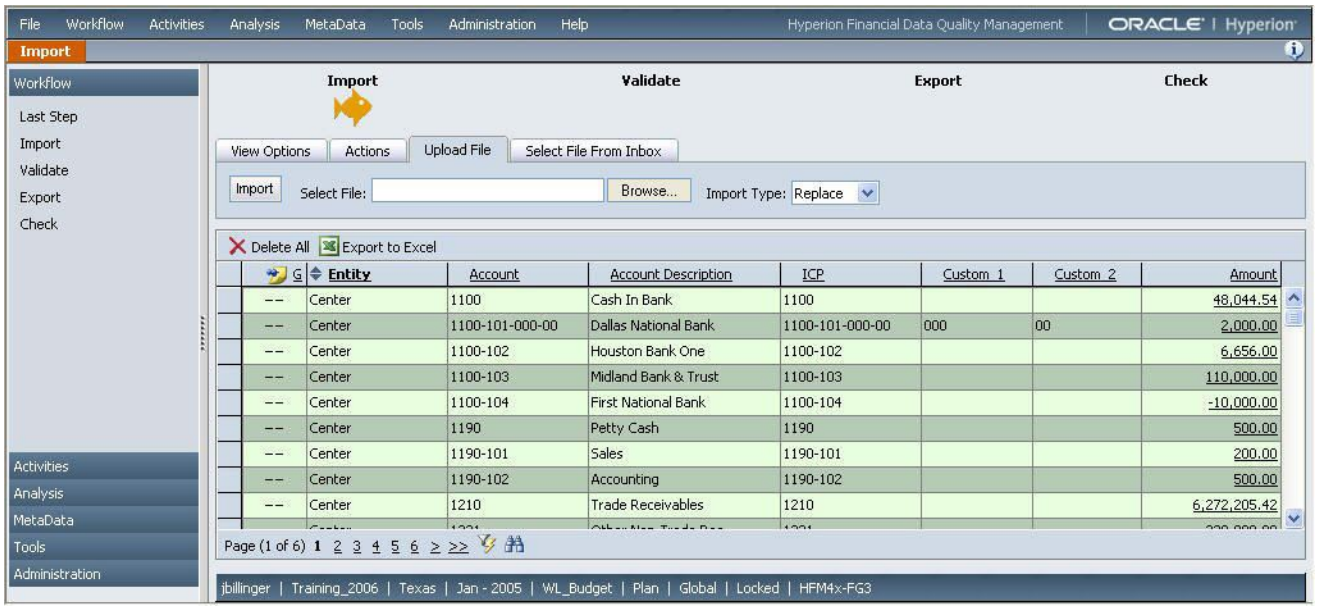
a. If the Replace option is selected, the existing source file is deleted.

b. The new source file is loaded.

c. Logic accounts are processed (if applicable).

d. Source dimensions are mapped to target dimensions.

The following graphic shows an example of the Import form after import of a source file.



Viewing Conversion Rules (Drill-up)

The Import form features a drill-up analysis function. Click on the link in the Amount field of the detail line and select Show Conversion Rules to view where each detail line is mapped.

Data Archiving

FDM archives all imported source files, import logs, journal entries, multiload files, target system load files, and attached memo documents. These files are stored in the application's `Data` directory and are assigned a unique name. These are retrievable from the Import screen and provide an audit trail.

Viewing Source File Information

To view the information about the source file, click the link in the Amount field then select Show Archive Information. This shows information about the source file.

Opening Source Document

To view the source file, click on the link in the Amount field of the detail line and select Open Source Document. The original source ledger file, journal, or multiload template is displayed.

Opening Log

To view the import log that was created when the source ledger file was loaded, click on the link in the Amount field of the detail line and select Open Processing Log. The loaded ledger file is displayed.

Restoring Source File and Log

To restore a source file from the archive directory, right-click on the detail line and select Restore Source Document. The source file and its associated import log is restored to the `Inbox\ArchiveRestore` directory.

Attaching Memos to Detail Lines

You can attach memos and supporting documents to the detail lines in the Import form.

➤ To attach a memo:

1 On the **Import** form, click the first column on a detail line.

The Memo dialog box is displayed.

2 Click **Add**.

3 Enter a memo and click **Add**.

4 (Optional) To attach supporting documents, click **Add**.

a. Browse for the supporting file to attach.

b. Enter a description of the supporting document in the text box to the left of the Add button.

You can add up to 10 supporting documents for each memo ID.

5 Click **Update** to save the changes.

After a memo has been added to detail line, a memo icon is displayed in the first column of the Import form.

All attached documents are stored in the application's `Data` directory.

- To review attached documents or to restore attached documents at a later time, open the Memo dialog by clicking the memo icon on the detail line.

- To display basic information regarding the attachment, click on the attached file name and select Show Archive Information.
- To display the document in its original format, click Show Document.
- To restore the document to the `Inbox\ArchiveRestore` directory, select Restore Document

➤ To delete a memo and remove all associated attachments:

1 Highlight the memo.

2 Select the **General** tab.

3 Click **Delete**.

➤ To remove a single attachment:

1 Highlight the memo.

2 Select the **Supporting Document** tab.

3 Click the **Remove** button located next to the attached document.

Attribute Drill Up

In addition to the custom dimensions UD1 through UD20, FDM also features 14 custom attribute dimensions. These are non-mappable but can be used for custom data warehousing tasks.

Each detail line in the Import form can have attributes that were loaded during the import process. Data loaded to the attribute dimensions can only be viewed from the Import form by opening the Attribute Drill Up form. To view the Attribute Drill Up form, click the link in the Amount field of the detail line and select Show Attributes.

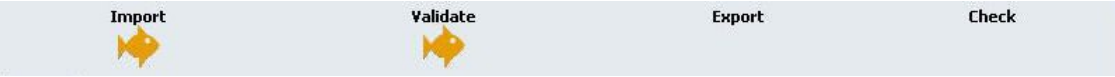
Validating Source Data

FDM forces the validation of the source data against the mapping tables. All source dimensions must be properly mapped before loading the target system. To run the validation process, click the Validation link at the top of the main FDM screen. Display the Validate screen without running the Validate process by clicking on the Validate link in the Workflow menu.

The Validate step compares the dimension mapping to the imported GL file and checks for any unmapped dimensions. If a file has been imported with unmapped dimensions, those dimensions are listed. The process cannot continue until all unmapped dimensions are mapped to a target dimension.

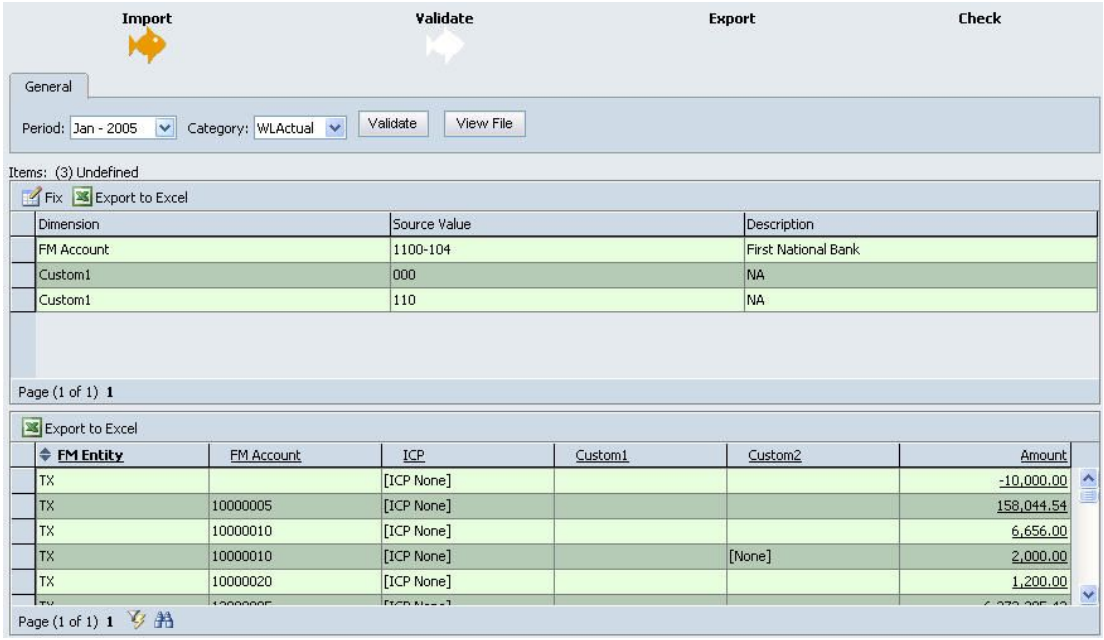
Validation Form No Mapping Errors

When the validation is successful the orange Validate Fish is displayed in the header.



Validation Form Mapping Errors

The addition of new accounts to a GL system can produce a validation error. A new account in a source file is likely to be previously unmapped. If a user has a new account and there is no mapping for the new account, the Validate fish is white in color. The Validation screen shows the number of items that are not mapped, and therefore undefined.



The Validation process displays the Validate form that lists all dimension values missing from their respective conversion tables.

➤ To correct errors in the conversion table:

1 Highlight the row to be corrected.

2 Click **Fix**.

The map form is displayed and the unmapped item is added in the Source column.

3 Click **Browse** and search for the correct target mapping.

4 Repeat steps 2 and 3 to correct all incorrect items.

5 Click **Validate** to refresh the Validation form.

If the source data passes the validation process, you can then load the source data to the target system.

Exporting to Target System

After the source data has passed the validation process, the load file is created.

➤ To export to the target system:

1 Click **Export** on the main form.

FDM creates the export file (in the format required by the target system) and places it in the application's `Outbox` directory.

The Target System Load dialog box is displayed.

Note:

To display the Export screen without running the Export process, click on the Export link in the Workflow menu.

Options that show in the Target System Load Dialog box are adapter-specific (load options of the target system). Refer to the target application documentation for information regarding the export options on the Target System Load dialog.

2 Click **OK**.

Drill-Down Analysis

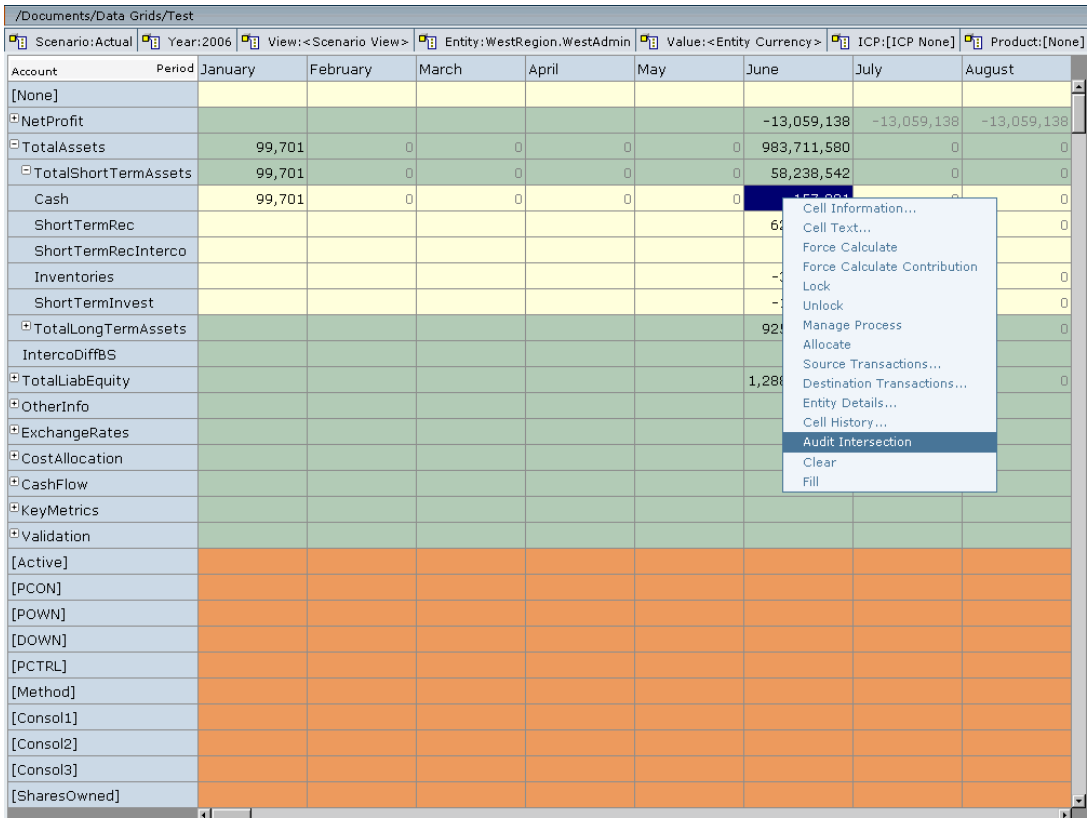
The Export form features drill down analysis functionality. Click on any amount in the Export form grid to display all source accounts that comprise the balance of the target line item.

Attaching Memos and Source Data Retrieval

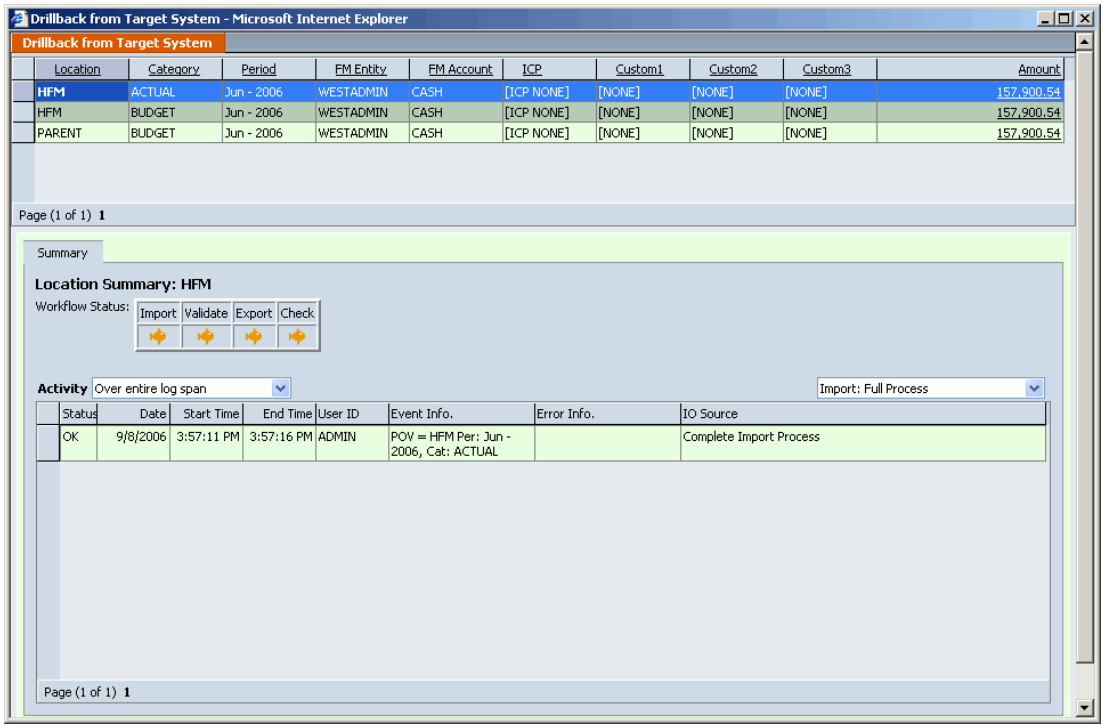
To attach a memo or support document to a detail line, click the first column for any record in the Drill-Down form. View these files by right-clicking on a detail line and selecting from the Drill-down menu. This is the same functionality used in the Import form. See "Data Archiving" on page 130 for details.

Drill-Back from Financial Management

Users of Financial Management can drill back to FDM to view the source used to load a Financial Management intersection. When viewing data in Financial Management, right-click on an amount and select Audit Intersection.



The top pane of the drill-back window shows all sources that loaded to the Financial Management intersection. The bottom pane (Summary tab) provides additional information about the item that is selected in the top pane.



From the drill-back window, access the FDM Drill-down window by clicking on the amount in any of the rows in the top form. See "Drill-Down Analysis" on page 134 for detailed information on using the Drill-down function.

Consolidating Target System

Select Activities > Consolidate to consolidate within the target system. FDM also runs a consolidation after loading the target system (assuming a validation entity group is assigned to the location). The consolidated entities are specified in the validation entity group that is assigned to the active location.

Consolidation Parameters

To enable consolidations, set Enable Consolidations Configuration to ON use Administration > Integration Settings > Enable Consolidation option). FDM uses the validation entity group assigned to the FDM location to determine which entities are consolidated for a given location. The current POV determines the target system category or period that is consolidated.

Consolidation Errors

FDM only consolidates the current period. If prior periods have been impacted for the entity and category that is consolidated, or if another user is currently accessing one of the entities in the consolidation path, an error is returned.

Validating Target System Data (Check)

The FDM validation reports retrieve values directly from the target system, FDM source data, or FDM converted data. Validation reporting is driven by two components– a group of validation rules that comprise the rows of the validation report, and a group of validation entities, which determines which target entities to which the validation rules apply.

Validation reports are automatically produced during the data loading process stream, but they can also be run manually. The main function of the reports is to enable reporting locations to review the quality of the information submitted.

The results of a validation report for locations are analyzed by FDM and an overall status entry is made in the process monitoring table. In order for a location to show a status of *True* for validation, each rule on the validation report must pass. If a rule is used for warning purposes only, then no rule logic is assigned to the row.

15 Importing Other Data Files

In This Chapter

Journals

A journal template is a Microsoft® Excel spreadsheet that has been formatted as a journal entry input screen. The spreadsheet is tagged with a header consisting of metadata that instructs FDM how to interpret the data contained in the template.

FDM journal templates are typically used for the following types of adjustments:

- GAAP adjustments to general ledger files
- Breakdown gross balance accounts into roll-forwards (Beg, Add, Del, End)
- Supplemental data entry (Head Count, Ratios, and so on)

The journal template (`journal.xls`) is located on the Tools > Templates menu. A sample journal template is located in \\Hyperion\FDM\<FDMAppName>\Outbox\Templates directory, where FDMAppName is the directory on the machine where the application was created.

Defining a Journal Template

Define a FDM journal template by defining the metadata that FDM uses to interpret the data contained in the template. Metadata consists of a series of tags that indicate which column contains the account number or what period to which to load.

Metadata tags are required in a specific tabular format. The metadata row order is important, but the column order is not. The first five rows (metadata header) of data must contain the metadata tags for the table of data.

The sample journal template shown contains the metadata header (rows 1–5) and two lines of imported data (rows 6 and 7).

	A	B	C	D
1	ID-Texas100		1_Texas	
2			ACTUAL05	
3			1/31/2005	
4			A	
5	A	C	V	D
6	1100	Texas	500.00	Reclass Cash
7	1210	Texas	(500.00)	Reclass Cash

To define the completed template in Excel, you must create a range name that includes all the metadata and the data value cells. The range name must begin with the prefix "ups." For example, you can create a range name to define a standard template and name it [upsStandardJV (A1 to D7)].

Metadata Structure

The metadata header (Row 1-5) specifies how to find the relevant segments of data from the template. The following describes how each piece of metadata is used by FDM.

Row 1 (Journal ID and Location Tag)

This is used to set the Journal ID and FDM location where the data is loaded. The Journal ID must be placed in row 1 of the Account column. The Location tag must be placed in row 1 of the Amount column.

Note:

The Journal ID is limited to ten characters.

Row 2 (FDM Category Tag)

This sets the FDM category that the journal loads. The category must be a valid FDM category and must be placed in the Amount column.

Row 3 (FDM Period Tag)

This sets the period to where the data is loaded. The period must be a valid FDM period and must be placed in the Amount column.

Row 4 (Load Method Tag)

This sets the journal load method within FDM. Enter "A" to append to an existing journal with the same Journal ID. If a journal exists with the same Journal ID within the same FDM POV, then the new journal is appended to the previously submitted journal.

Enter "R" to replace an existing journal with the same Journal ID. If a journal exists with the same journal ID within the same POV, then the new journal replaces the old journal. This tag must be placed in the Amount column.

The following table lists the metadata tags that can be used in row 4.

Method	Tag
Append Journal	A, Append
Replace Journal (Default)	R, Replace
Append Journal – Zero Suppress	AZ
Replace Journal – Zero Suppress	RZ

Dimension Tags

The tags in row five of a journal template define the dimension where the amounts are loaded. The following table defines dimension tags.

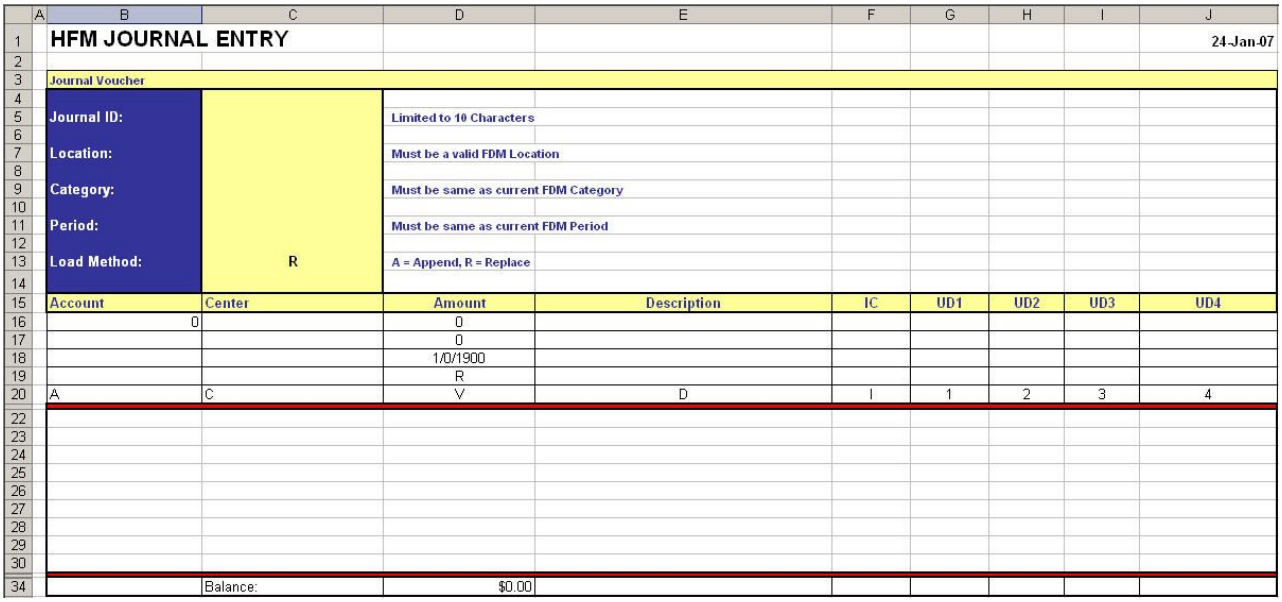
Table 1 FDM Dimension Tags

Dimension	Tag
Account (Required)	A, Account, SrcAcctKey
Center (Required)	C, Center, SrcCenterKey
Description (Optional)	D, Description, SrcAcctDesc
IC Counter Party (Optional)	I, IC, ICCoParty
User Defined 1 (Optional)	1, UD1, UserDefined1
User Defined 2 (Optional)	2, UD2, UserDefined2
User Defined 3 (Optional)	3, UD3, UserDefined3
User Defined 4 (Optional)	4, UD4, UserDefined4
User Defined 5 (Optional)	5, UD5, UserDefined5
User Defined 6 (Optional)	6, UD6, UserDefined6
User Defined 7 (Optional)	7, UD7, UserDefined7
User Defined 8 (Optional)	8, UD8, UserDefined8
User Defined 9 (Optional)	9, UD9, UserDefined9
User Defined 10 (Optional)	10, UD10, UserDefined10
User Defined 11 (Optional)	11, UD11, UserDefined11
User Defined 12 (Optional)	12, UD12, UserDefined12
User Defined 13 (Optional)	13, UD13, UserDefined13
User Defined 14 (Optional)	14, UD14, UserDefined14
User Defined 15 (Optional)	15, UD15, UserDefined15

Dimension	Tag
User Defined 16 (Optional)	16, UD16, UserDefined16
User Defined 17 (Optional)	17, UD17, UserDefined17
User Defined 18 (Optional)	18, UD18, UserDefined18
User Defined 19 (Optional)	19, UD19, UserDefined19
User Defined 20 (Optional)	20, UD20, UserDefined20
Amount (Required)	V, Amount, SrcAmount

The following illustration depicts a journal template. Note that in this template, the metadata are not in rows 1–5, but in rows 16–20. The template has a UPSRange starting from row 16. Therefore, rows 16–20 are the first five rows in the UPSRange.

Rows 4–14 is a simple interface to assist users with creating the metadata header. Metadata information is input here, and referenced by the metadata header.



	A	B	C	D	E	F	G	H	I	J
1		HFM JOURNAL ENTRY								24-Jan-07
2										
3		Journal Voucher								
4										
5		Journal ID:		Limited to 10 Characters						
6										
7		Location:		Must be a valid FDM Location						
8										
9		Category:		Must be same as current FDM Category						
10										
11		Period:		Must be same as current FDM Period						
12										
13		Load Method:	R	A = Append, R = Replace						
14										
15		Account	Center	Amount	Description	IC	UD1	UD2	UD3	UD4
16		0		0						
17				0						
18				1/0/1900						
19				R						
20		A	C	V	D	I	1	2	3	4
22										
23										
24										
25										
26										
27										
28										
29										
30										
34			Balance:	$0.00						

The journal template must contain the following dimensions:

- Center—May be the center from the source or the target system (required).
- Account—May be the account from the source or the target system (required).
- Amount (required)
- Description (optional)
- Intercompany (Optional unless being used by Financial Management)
- Custom UD1-20 Dimensions (optional unless being used by the target system)

Journal Data Validation

UpCheck Function

Prior to loading a template, FDM checks the template for a custom VBA function named "UpCheck." This function forces custom data validations prior to loading (for example, restricting users to a particular accounts or centers that the user can load).

This function should be designed in a new VBA module within the template. If your custom conditions have been met, then set `UpCheck=True`. An example of an UpCheck function follows.

```
----------------------------------------
Public Function UpCheck() as Boolean
'Place your validation code here

If Validate = True Then
    UpCheck = True
Else
    UpCheck = False
End If

End Function
----------------------------------------
```

Multiload Action Events

In the FDM script editor, you can write a validation script within the MultiLoadAction event. This event is executed before and after the journal posting process. You can evaluate the `strEventName` parameter to determine which multiload event is executing and write a validation script to stop the process. See Chapter 19, "Scripting"for information on this event.

Processing a Journal Template

Selecting a Journal

➤ To select journal to process:

1. Select **Activities > Journal**.
2. Click **Browse** and select the journal.
3. Click **Open**.

Checking In a Journal

A journal must be checked in before it is posted. To check in a journal, click Check-in in the upper-right side of the screen. FDM then processes the metadata tags in the template.

FDM first examines the template for all range names that start with the prefix “ups.” It then examines and validates the metadata tags found in each “ups” range name. If any segment of the metadata contained in a given range is invalid then FDM does not check in that segment and the error information is added to the Processing Errors grid.

Posting a Journal

After the journal is checked in, click Post to post the journal. Posting a journal either appends or replaces the data in the Import Screen based on the load method specified in the journal template.

Journal Security

FDM administrators and end-users are restricted to posting journals to the FDM global point of view unless the POV Lock option is turned off.

Excel-Based Trial Balance Files

Overview

An Excel-based trial balance is an Excel spreadsheet formatted to a single period, category, and location through the normal Import screen. The Excel template and text-based file requires header tags that are used to tell FDM how to interpret the data.

Text File Trial Balance Load vs. Excel Trial Balance Load

While importing both standard text files and Excel trial balances, both file types are loaded to the current POV (category and period). Also, Excel-based trial balance files use the same Append and Replace options on the Import form that standard text file imports use. However, while text-based imports must follow the location’s standard import format, each Excel-based trial balance can be configured differently as long as it follows the correct definition for the Excel templates.

Defining an Excel-Based Trial Balance Template

To define an Excel trial balance template, you must define the metadata FDM uses to interpret the data contained in the template. Metadata consists of a series of tags that tell FDM what column has the account number or what period to load.

Template metadata tags are required in a specific tabular format (metadata header). The metadata row order is important, but the column order is not. The first row of data must contain the metadata tags.

The sample template shown has one line of metadata (row 1) and three lines of imported data (rows 5–7).

	A	B	C	D	E	F	G	H	I
1	A	C	D	IC	UD1	UD2	M1	M2	V
2									
3									
4									
5	1100_001	TX	Cash In Bank	[icp none]	[none]	[none]	Memo	Memo Details	48,045
6	1190	TX	Petty Cash	[icp none]	[none]	[none]	Memo	Memo Details	500
7	4200	TX	Sales	[icp none]	[none]	[none]	Memo	Memo Details	(48,545)
8									
9									
10									
14	Must Balance to Zero:								-

Metadata Structure

The metadata header (row 1) determines how to find the relevant segments of data that are used in this template.

Row 5 (Dimension Tags)

The tags in row five define what dimension data is in the column. See Table 1 on page 139for dimension tag definitions.

Multiload Files

A multiload file is an Excel spreadsheet or text file formatted to load multiple periods, categories, and locations. The template requires header tags that instruct FDM how to interpret the data contained in the template.

FDM multiload templates are typically used for the following types of data:

- Plan data collection (budget, forecast, and so on)
- Historical data loading
- Administrator data maintenance

Text File Multiload vs. Excel Multiload

Depending on the amount of data, FDM processes a multiload text file at a much faster rate than an Excel-based multiload file. This is because when processing a text-based file, FDM creates a single load file and consolidates a range of periods, and when processing an Excel-based file, FDM creates separate load files for each period and consolidates one period at a time.

Defining Multiload Templates

To define a FDM multiload template, you must first define the metadata tags that FDM uses to interpret the data contained in the template. The metadata tags define the columns that FDM will import.

Template metadata tags are required in a specific tabular format. The metadata tags row order is important, but the column order is not. The first five rows of data (for Excel-based templates), or the first six rows (for text-based templates) must contain the metadata tags.

The sample Excel template shown has five rows of metadata tags and 4 rows of imported data. Rows 1-5 contain the metadata tags and rows 8 through 11 contain the data values.

upsBudget =

	A	B	C	D	E	F	G	H	I	J	K
1										1_Texas	1_Texas
2										Actual04	Actual04
3										1/31/2004	2/28/2004
4										R,M	R,M
5	**Account**	**Center**	**Description**	**IC**	**UD1**	**UD2**	**UD3**	**UD4**	**DV**	V	V
8	1100	TX	Cash In Bank		MES	GIO		OP	Per	47,564	48,045
9	5000-101	TX	I/C Sales	MI		IOE	TV	OP	Per	1,980	2,000
10	2100-102	TX	I/C Rec	WI		GIO		OP	Per	6,589	6,656
11	3100-103	TX	I/C Pay	NY		TVE		OP	Per	108,900	110,000

In order to define the completed template in Excel, you must create a range name that includes all the metadata and the data value cells. This range name must begin with the prefix "ups." In this example, the range is created and named [upsBudget (A1 to K11)].

The sample text file that follows has six rows of metadata tags and four rows of imported data. Rows 1-6 contain the metadata tags and rows 7 through 10 contain the data values. Whatever delimiter separates the dimensions in Row 6 must be used as the delimiter that separates the data values. Valid delimiters include commas, semicolons, pipes ("|"), tabs, and exclamation points.

1	Texas
2	Budget
3	01/31/2004
4	12
5	R,N,Y
6	A,C,D,V,V,V,V,V,V,V,V,V,V,V
7	1100,TX,Cash In Bank," 43,016 "," 43,451 "," 43,890 "," 44,333 "," 44,781
8	1100-101-000-00,TX,Dallas National Bank," 1,791 "," 1,809 "," 1,827 "," 1
9	1100-102,TX,Houston Bank One," 5,959 "," 6,020 "," 6,080 "," 6,142 "," 6,
10	1100-103,TX,Midland Bank & Trust," 98,487 "," 99,482 "," 100,487 "," 101,
11	1100-104,TX,First National Bank," (8,953)"," (9,044)"," (9,135)"," (9,227
12	1190,TX,Petty Cash, 448 , 452 , 457 , 461 , 466 , 471 , 475 , 480 , 485 ,

MetaData Tags Structure

The metadata header (rows 1-5 for Excel templates, and rows 1–6 for text templates) tells FDM how to find the relevant segments of data within the main body of the template.

Row 1 (Location Tags) - Valid for Amount Columns Only

This tag defines the FDM location to where the data is loaded. The location name must be a valid FDM location.

Row 2 (Category Tags) - Valid for Amount Columns Only

These define the FDM category to which the data is loaded. The category must be a valid FDM category.

Row 3 (Period Tags) - Valid for Amount Columns Only

This defines the FDM period that into which the data is loaded. The period must be a valid FDM period.

Row 4 (Number of Periods Tag) — Text Templates Only

This tag defines the number of periods (data columns) to be loaded. This tag is used in combination with the Starting Period tag to derive the actual period key used in the load. This tag is used for text-based multiload templates only, and must be placed in row 4.

Row 4/Row 5 (Load Control Tags) — Excel and Text Templates

Load Control tags control the load methods for both FDM and the target system. The Load Control tags are dependent upon the target system that FDM is loading. When using an Excel template, these tags must be placed in row 4 and when using a text template, these tags must be placed in row 5. The following table lists the load control tags that can be used by FDM. Use commas to separate load tags.

Load Tag #	Load Control Tag Name	Description
1	FDM Load Method	Method of loading FDM
2	Target System Load Method	Method of loading target system
3	List Box 1	Varies by integration adapter
4	List Box 2	Varies by integration adapter
5	List Box 3	Varies by integration adapter
6	Check Box Calc	Enable calculation (Y/N)
7	Check Box 1	Varies by integration adapter (Y/N)
8	Check Box 2	Varies by integration adapter (Y/N)
9	Check Box 3	Varies by integration adapter (Y/N)
10	Data View	Specifies if data being loaded is YTD, Periodic, Qtr Year-to-Date

Financial Management Load Controls Tags (Multiload Excel File)

The following tables detail the specific load control tags for Financial Management. Financial Management does not use load tags #4, #5, #6, and #9. For these unused tags, you must insert a comma as a placeholder. If any Financial Management tags are missing (#2, #3, #7, #8) then FDM will default to the data load options defined in the Integration Settings. The default value for tag #1 is "Replace" and the default value for tag #10 is "YTD."

Load Tag #1: FDM Load Tag Method	Tag
Append	A
Replace (Default)	R
Append – Zero Suppress	AZ
Replace – Zero Suppress	RZ

Load Tag #2: Financial Management Load Method	Tag
Merge	M, Merge
Replace	R, Replace
Replace by Security	RS
Accumulate	A, Accumulate

Load Tag #3: Financial Management Load Process	Valid Tags
Load	0, Load
Scan	1, Scan

Load Tag #4
Not Used

Load Tag #5
Not Used

Load Tag #6
Not Used

Load Tag #7: Accumulate in File	Tag
Yes	Y, Yes, True, T
No	N, No, False, F

Load Tag #8: File Has Share Data	Tag
Yes	Y, Yes, True, T
No	N, No, False, F

Load Tag #9:
Not Used

Load Tag #10: Data View	Tag
Year-To-Date (Default)	Y, YTD, Year-To-Date
Periodic	P, Per, Periodic
Qtr Year-To-Date	Q, QYTD, Qtr
Half-Year	H, HYTD, Half-Year

Note:

You can create a Data View column in Row 5 (Excel template) or Row 6 (text template) so that each row of data can have it's own Data View tag. The preceding sample template shows a Data View column. If no Data View tag is defined in the header, and no Data View column exists, then the default Data View is set to "YTD."

Financial Management Load Control Tags Example (multiload)

This example shows a tag for each Load Control Tag item. Notice that commas required as placeholders for unused tags (#4, #5, #6, #9).

```
R,M,0,,,,Y,N,,YTD
```

In this example, FDM uses the default Financial Management load settings defined in the Integrations Settings because only the FDM load method tag (Append) is specified. The Data View tag (tag #10) defaults to YTD. You do not need to add trailing commas after the last supplied argument.

```
A
```

In the following example only the FDM load method (Replace) and the Oracle's Hyperion® Financial Management – System 9 load method (Merge) is specified. The other tags default to the option values specified in the Integration Settings.

```
R,M
```

Hyperion Enterprise Load Control Tags (multiload)

The following tables detail the specific load control tags for Oracle's Hyperion® Enterprise®. Enterprise does not use load tags #5, #6, #8, #9, and #10. For these unused tags, you must insert a comma as a placeholder. If any of the Enterprise tags are missing (tags #2, #3, #4, #7) then

FDM defaults to the data load options defined in the Integration Settings. The default value for tag #1 is “Replace.”

Load Tag #1: FDM Load Method	Tag
Append	A
Replace (Default)	R
Append – Zero Suppress	AZ
Replace – Zero Suppress	RZ

Load Tag #2: Enterprise Load Method	Tag
Merge	M, Merge
Replace	R, Replace
Accumulate	A, Accumulate

Load Tag #3: Data View	Tag
Category	C, Cat, Category
Periodic	P, Per, Periodic
Year-To-Date	Y, YTD, Year-To-Date

Load Tag #4: Scale	Tag
None	N, -1, None
Units	U, 0, Units
Tens	T, 1, Tens
Hundreds	H, 2, Hundreds
Thousands	TH, 3, Thousands
Ten Thousands	T-TH, 4, Ten Thousands
Hundred Thousands	H-TH, 5, Hundred Thousands
Millions	ML, 6, Millions
Ten Millions	T-ML, 7, Ten Millions
Hundred Millions	H-ML, 8, Hundred Millions
Billions	B, 9, Billions

Load Tag #5
Not Used

Load Tag #6
Not Used

Load Tag #7: Zero No Data	Tag
Yes	Y, Yes, True, T
No	N, No, False, F

Load Tag #8
Not Used

Load Tag #9
Not Used

Load Tag #10
Not Used

Hyperion Enterprise Load Control Tags Example (multiload)

The example that follows shows a metadata tag for each load control tag item. Notice that commas are required as placeholders for the load tags that are not used (tags #5 and #6). Trailing commas after the last supplied argument are not required.

```
R,M,C,N,,,Y
```

In this example, because only the FDM load method tag (Append) is specified, FDM uses the default Enterprise load settings defined in the Integrations Settings.

```
A
```

In this example, only the FDM load method tag (Replace) and the Enterprise load method tag (Merge) are specified. The other tags default to the option values specified in the Integration Settings.

```
R,M
```

Dimension Tags

These tags define what dimension data can be found in the column. See Table 1for detailed information about each of the load tags that can be used in the metadata header.

A valid delimiter must separate each dimension. Whichever delimiter separates the dimensions in the dimension list must be used as the delimiter that separates the data values. Valid delimiters include: commas, semicolons, pipes (|), tabs or exclamation points (!).

Note:

If loading memos from a multiload file, each month receives the same memo specified for that row.

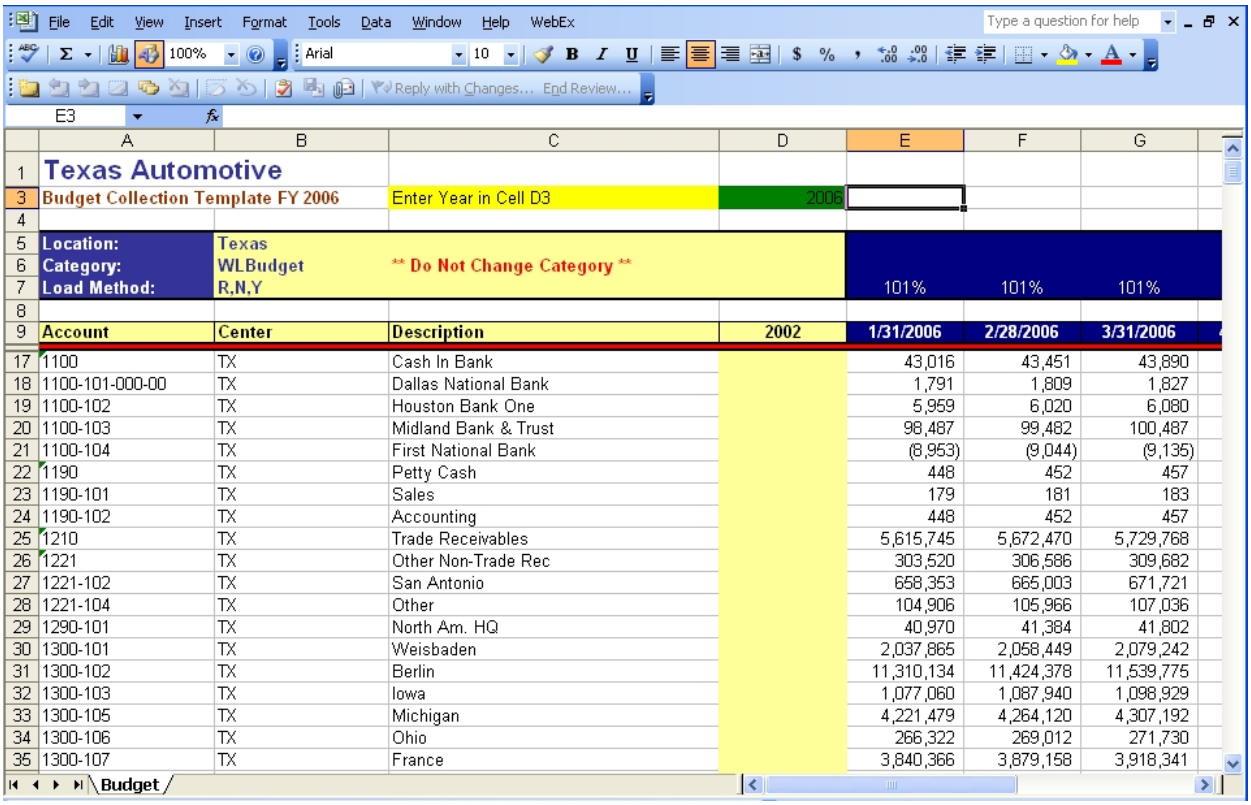
Multiload Excel Template Data Validation

UpCheck Function

Prior to loading a multiload Excel template, FDM searches the template for a custom VBA function named "UpCheck." This function can be used in order to force custom data validations prior to loading (in other words, restricting which categories or periods that users can load). This function should be designed in a new VB module within the template. If your custom conditions have been met, then set `UpCheck=True`. The following is an example of an UpCheck function:

```
-----------------------------------------
Public Function UpCheck() as Boolean

'Place your validation code here

If Validate = True Then
UpCheck = True
Else
UpCheck = False
End If
End Function
-----------------------------------------
```

The following illustration depicts an example of a multiload Excel template.



	A	B	C	D	E	F	G
1	Texas Automotive						
3	Budget Collection Template FY 2006		Enter Year in Cell D3	2006			
4							
5	Location:	Texas					
6	Category:	WLBudget	** Do Not Change Category **				
7	Load Method:	R,N,Y			101%	101%	101%
8							
9	Account	Center	Description	2002	1/31/2006	2/28/2006	3/31/2006
17	1100	TX	Cash In Bank		43,016	43,451	43,890
18	1100-101-000-00	TX	Dallas National Bank		1,791	1,809	1,827
19	1100-102	TX	Houston Bank One		5,959	6,020	6,080
20	1100-103	TX	Midland Bank & Trust		98,487	99,482	100,487
21	1100-104	TX	First National Bank		(8,953)	(9,044)	(9,135)
22	1190	TX	Petty Cash		448	452	457
23	1190-101	TX	Sales		179	181	183
24	1190-102	TX	Accounting		448	452	457
25	1210	TX	Trade Receivables		5,615,745	5,672,470	5,729,768
26	1221	TX	Other Non-Trade Rec		303,520	306,586	309,682
27	1221-102	TX	San Antonio		658,353	665,003	671,721
28	1221-104	TX	Other		104,906	105,966	107,036
29	1290-101	TX	North Am. HQ		40,970	41,384	41,802
30	1300-101	TX	Weisbaden		2,037,865	2,058,449	2,079,242
31	1300-102	TX	Berlin		11,310,134	11,424,378	11,539,775
32	1300-103	TX	Iowa		1,077,060	1,087,940	1,098,929
33	1300-105	TX	Michigan		4,221,479	4,264,120	4,307,192
34	1300-106	TX	Ohio		266,322	269,012	271,730
35	1300-107	TX	France		3,840,366	3,879,158	3,918,341

Multiload Action Events

You can write a validation script within the MultiLoadAction event. This event executes before and after each event during the multiload process. You can evaluate the `strEventName` parameter to determine which multiload event is executing and write a validation script to stop the process. See Chapter 19, “Scripting” for additional information on this event.

Checking In a Multiload File

Checking in a multiload file prompts FDM to process the metadata tags in the template. Check in a template by clicking the Check-In button and then browsing for the multiload Excel file or text file.

The tree on the left contains all the data-sets that are processed. The Processing Details section on the right displays validation errors and validation report errors during the Validate and Check processes.

After a multiload file is checked in, the FDM POV is changed to Local mode. The local POV for that user is changed to each data-set’s POV that is clicked on in the tree. Switching to Local mode enables the user to fix validation errors during the Validate process and view validation reports errors during the Check process. When the multiload form is closed, the FDM POV is changed back to Global mode.

If any segment of the metadata is invalid, then FDM does not check in that segment, and the error information is added to the processing errors grid.

From the UpLoad File Tab, click Browse to locate the multiload file, select the file from the popup window and click Open. The multiload file can either be a text file (.txt) or an Excel file (.xls).

If the file was loaded from the Upload file tab then it must be checked in. Click Check-in in the upper-right side of the screen. If the Multiload file was loaded from the Select File from Inbox screen the check-in step is not required.

Importing a Multiload File

After a multiload template has been checked in, it is imported by clicking the Import button.

Importing a template writes the data to the FDM location table. The  icon appears in the tree on the left if errors occur.

Validating a Multiload File

After a multiload template has been imported, it can be validated by clicking the Validate button. Validating a template verifies that all members within a dimension exist in their respective map tables.

Mapping validation errors are viewed by clicking the  icon. The unmapped members are shown in the Processing Details grid on the right.

All unmapped members from the multiload must be added manually. To add the unmapped members to their proper mapping tables, click on the Maps link under Activities. Proceed to map the invalid member to a target member.

Note:

The mapping table displayed is for the current POV location, this may not be the same location as the location being loaded in multiload. Always verify the current location prior to making mapping changes.

Loading a Multiload File

After a multiload template has been validated, it is exported and loaded to the target system by clicking the Export button. When processing a multiload text file, a single load file is created and loaded into the target system. When processing a multiload Excel file, a separate load file is created and loaded for each period. The  icon appears in the tree on the left if load errors occur.

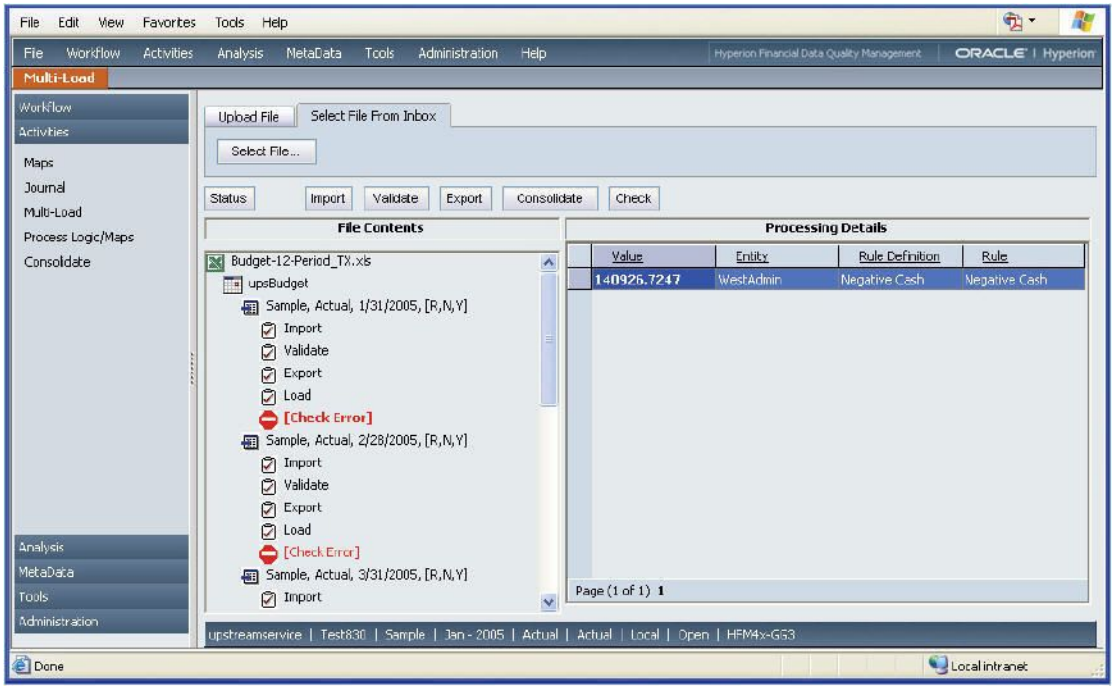
Consolidating a Multiload File

After a multiload template has been exported and loaded to the target system, run a consolidation for each period by clicking the Consolidate button. The icon appears in the tree on the left if any errors occur.

Running Validation Reports for a Multiload File

After a multiload template has been consolidated, process the validation reports for each period by clicking the Check button. The validation reports for each period are processed in the background.

You can view validation report accounts that do not pass validation by clicking on the icon for each period. Failed account rules appear in the Processing Details grid on the right.



Multiload Security

FDM end-users can load a multiload file to any FDM period and category but only have rights to load to locations to which they have access. FDM administrators can load a multiload file to any location, period or category.

Multiload Data Fields

All amount data fields must contain a value. Amount fields should have a zero (0) in every field that does not have another value being loaded. If a blank amount field is in the multiload template, the multiload process will not complete successfully.

16 Reports

In This Chapter

Running Reports

➤ To run a report:

1 From the FDM Web client, select **Analysis > Reports**.

The Reports screen is displayed.

2 From **Groups**, select the group of reports to view.

The Reports dialog box displays all reports associated with the current report group.

3 Select the output format for the report by clicking the menu in the upper right corner.

4 Double-click **Publish**.

5 Enter parameter values (if prompted).

Report Viewer

Navigation

You can navigate through report pages by clicking on the arrow buttons. To search for any value in a report, click the Search button, enter the text in the Search field, and click Search.

Display

Zoom in and out by using the Magnification box. Also, the report viewer can be resized to adjust the viewable area.

Printing and Exporting

To print the contents of a tab, click on the Print button. The option to export the report to an external file must be selected prior to running the report. To select the format to export on the

Reports screen, select the menu next to the Publish button, enter any required parameters, and click Publish.

Maintaining Reports

From Workbench, select the Reports tab to change and create new reports. FDM has a built in report creation tool that allows the creation of Active Reports. Creating and integrating new reports requires advanced knowledge of RDBMS Syntax and the FDM Software Data Window Active-X component.

Changing Descriptions

To maintain Report Group Descriptions, Report Descriptions, and Parameter Descriptions from this screen, right-click on the report and select Report Definition Properties to change the report description. This feature allows you to assign meaningful names to these components and does not require advanced knowledge of the FDM Reporting Components.

Process Monitor Reports

From the Reports tab, expand the Process Monitor report group, and select the Process Monitor Report. This report displays a listing of locations and their station within the data conversion process. This is useful if you want to monitor the status of the closing process. This report is time-stamped so that it can be used to determine which locations have loaded data at a specific point in time.

Designing a Report

To view or modify an Active Report, right-click on the report and select Design Report. This feature allows the user to create and modify existing reports. This requires advanced knowledge of the FDM Reporting components.

Setting Report Group Security

Report groups can be assigned a security level that restricts access to users that have a security level that permits access to the report group. Report group security functions like Application Object Security.

➤ To assign security to a report group:

1 From Workbench, click the **Reports** tab.

The reports are displayed in the left pane of the Workbench desktop.

2 Right-click on a report group and select **Properties**.

3 From **Security Level**, select the minimum level for the report group.

4 Click **OK**.

The security level that is assigned to a report group is evaluated against the application security level that is assigned to an end user. Access to reports is then granted to users that have an application security level that is equal to or less than the level assigned to the report.

Security for the Process Explorer reports can be set on individual reports. Security is set in the same manner except that you must right-click the individual report instead of the report group to modify the access for users.

17 Batch Loader

In This Chapter

Overview

Batch Loader is run from the FDM Workbench or the FDM Task Manager. The Batch Processing screen enables you to execute and create batch scripts. Batches executed from the Batch Processing screen can be monitored to view the progress.

Note:

Batch Loader is available at an additional cost. Contact your Hyperion sales representative for information regarding Batch Loader pricing.

Preparing Batch Loader Files

Data files must be placed in specific directories, in a specific format, in order for Batch Loader to locate and load the files. There are two separate directories used for harvesting files. One directory is used for standard text file batch processing. The other directory is used for multiload batch file processing.

File Name Format

Standard processing batch files must be in created in the standard FDM batch file format. The file name is parsed into five different segments (File ID, Location, Category, Period, Load Method).

- File ID—A free form field that can be used to control the order that the batch files are loaded. The batch files load in alphabetic order by file name. To get a specific file to load first, in the batch make the file ID a number that is lower than any other File ID ("a_Texas_Actual04_Jan-2004_RR.txt" vs. "b_Texas_Actual04_Jan-2004_RR.txt" for example)

- Location—The FDM location name to be loaded
- Category—The FDM category name
- Period—The FDM period name
- Load Method—A two-character switch: Character 1 = FDM Append/Replace (Valid values A or R), Character 2 = Target Append/Replace (valid values A or R)

Sample Formatted File Name = “a_Texas_Actual04_Jan-2004_RR.txt”

Batch File Location

All standard files to be loaded using Batch Loader must be placed in the `OpenBatch` directory in the application `Inbox\Batches` subdirectory. After the batch has been processed, a new directory is created and all files in the `OpenBatch` directory are moved into the new directory. This new directory is assigned a unique batch ID that is referenced in the Import log

Multiload File Name Format

Multiload files do not require a special file name format. Because multiload files contain metadata inside the file itself, these types of file need only to be placed into their own separate directory.

Multiload File Batch Directory

All standard files loaded using Batch Loader must be placed in the `OpenBatchML` directory in the application `Inbox\Batches` subdirectory. After the batch has been processed, a new directory is created and all files in the `OpenBatch` directory are moved into the new directory. This new directory is assigned a unique batch ID that is referenced in the Import log and is prefixed with “ML” to indicate that the batch was a multiload batch.

Running Batch Loader

➤ To initiate the batch load process:

1 Create the batch files with the proper file name format and place them in the appropriate FDMdirectory.

2 Start the batch load process in one of two ways:

- From Workbench, select Tools > Batch Processing.
- From Task Manager, select Start > Programs > Hyperion > Financial Data Quality Management > Task Manager > Task Manager.

  The Batch Processing screen is displayed.

3 Set the Batch Controls

Details of Batch Controls follows this procedure.

4 Click Execute or Create Script.

See the section that follows for detailed information regarding these options.

Batch Controls

Batch Type—Determines if the batch being executed is a standard or a multiload batch. Standard batch files are a single-month file that would normally be loaded through the import screen. Multiload batch files are text or Excel files that can contain multiple periods and locations.

Process Level—Determines the level of processing to be performed. You can create a batch to only import files, to process files, to running the check report step, or any step in between.

Process Method—Determines if the batch is loaded in serial or parallel mode. Serial batches processes files sequentially, requiring that one file complete the process prior to the next file starting. Parallel batches process multiple files simultaneously.

No. of Parallel Processes—Determines the number of processes the batch uses concurrently when the Process Method is set to Parallel. Files within each process is executed in a serial batch.

Example:

If there are 20 files in the batch to be processed, and the No. of Parallel Processes is set to 5, then five processes, each with four files, will run sequentially.

Note:

When processing batch files in parallel, FDM groups files that are loading to the same FDM location into the same process. This ensures that a location does not have two processes attempting to load data at the same time, potentially resulting in invalid results.

Load Balance Server—Allows the administrator to select any load balance server that has been configured in the Workbench Load Balances Server group. This is the load balance server used for processing the batch.

Auto-Map Correct—Allows the batch to have auto-mapping enabled or disabled for the selected batch. This applies for all files in the batch being processed. FDM can auto map any item in the batch load files that are not included the dimension maps. Any members that are not mapped have a new map entry created in the location's map for the unmapped member based on a predetermined suspense account. To set up the suspense account, a new location named *AutoMapCorrect* must be added to the FDM application. After adding this location, a new map entry must be created in the Explicit map for each dimension. The source value for this mapping must be named *AutoCorrect* while the Target value can be any valid member for the dimension. These mapping entries are the basis for the auto mapping for any unmapped members of a batch load.

File Name Delimiter—Defines what character is being used in the standard batch files names to separate the five segments of the file name. In standard batch processing, the file name must be set up in a predefined manner to allow FDM to determine the location, category, period, load method, and order of processing. By defining these parameters in the file name, FDM can process

the batch file unattended. For multiload batch files, this option does not need to be set because the parameters for processing multiload files are stored within the file itself.

Execute—Initiates the batch process with the selected parameters. The Batch Monitor section shows all files being processed and updates with the results for each file.

Create Script—Creates a batch processing script with the selected parameters. FDM prompts the user for a script name when this button is selected. This script can then be added to a scheduled task to run automatically.

Batch Processing Screen Results

While processing a batch from the Batch Processing screen, the Batch Monitor field displays the batch results. The results are displayed in a graphical format. Each load that completes displays a symbol that allows the administrator to determine what processes failed and which ones passed.

Processing Statuses

 Auto Corrected—Processes with this status have had an autocorrect applied.

 Successful—Process with this status have completed successfully.

 Failure—Displayed when a process has failed one of more steps.

 Invalid POV—Displayed when the process is being loaded to an invalid period, category or location.

Scripting for Batch Loader

In addition to launching the process within Workbench, you can run Batch Loader by using a scripting environment. To run the batch load process either manually or through Task Manager, you must create a batch load script. You can create the batch load script manually by copying the following scripts, or by using the Batch Processing screen (previous section).

Script for Standard Batch Load

```
Sub BatchLoad()
'------------------------------------------------------------------------
---
' FDM Custom Script:
'
'Created By:      ADMIN
'Date Created:     3/24/2006 09:00
'
'Purpose:        Execute a STANDARD Serial Processing FDM Batch
```

```
‘-------------------------------------------------------------------------
---
‘Declare Local Variables
Dim lngProcessLevel
Dim strDelimiter
Dim blnAutoMapCorrect
‘Initialize Variables
lngProcessLevel = 12  ‘Up-To-Check
strDelimiter = “_”
blnAutoMapCorrect = 0
‘Create the file collection
Set BATCHENG.PcolFiles = BATCHENG.fFileCollectionCreate(CStr(strDelimiter))
‘Execute a Standard Serial batch
BATCHENG.mFileCollectionProcess BATCHENG.PcolFiles, CLng
(lngProcessLevel), , CBool(blnAutoMapCorrect)
End Sub
```

After the script has been created, you can create a FDM Task manager event to run the batch loader on a scheduled basis.

Script for Parallel Batch Loads

FDM supports parallel processing of batch files. This allows multiple files to be processed simultaneously instead of sequentially. FDM can process up to fifty files simultaneously. FDM groups files to be processed in the batch into processes. The files in each process are processed in serial order, requiring that one file complete the process prior to the next file starting, while each of the individual processes run in parallel.

Example: If there are twenty files in the batch to be processed and No. of Parallel Processes is set to “5,” then each process will have four files in it that are processed in a serial manner.

Note:

When processing batch files in parallel , FDM groups files that are loading to the same FDM location into the same processes. This ensures that a location does not have two processes attempting to load data at the same time; potentially yielding invalid results.

```
Sub ParallelBatch()
‘-------------------------------------------------------------------
‘ FDM Custom Script:
‘
‘Created By:        Admin
‘Date Created:      3/23/2006 13:50
‘
‘Purpose:           Execute a FDM Batch Loader with parallel processing
‘-------------------------------------------------------------------
‘Declare Local Variables
Dim lngProcessLevel
Dim strDelimiter
Dim blnAutoMapCorrect
Dim lngParallelProcessCount
Dim strLoadBalanceServerName
‘Initialize Variables
```

```
lngProcessLevel = 12   ‘Up-To-Check
strDelimiter = “_”
blnAutoMapCorrect = 0
lngParallelProcessCount = 5
strLoadBalanceServerName = “LocalHost”
‘Create the file collection
Set BATCHENG.PcolFiles = BATCHENG.fFileCollectionCreate(CStr(strDelimiter))
‘Execute a Standard Parallel batch
BATCHENG.mFileCollectionProcessParallel BATCHENG.PcolFiles, CLng
(lngProcessLevel), CLng(lngParallelProcessCount), CStr
(strLoadBalanceServerName), , CBool(blnAutoMapCorrect)
End Sub
```

18 Additional Tools

In This Chapter

Exporting to XML

FDM enables you to export and entire application to an XML file.

➤ To export applications:

1 Log on to the application using Workbench.

2 Select **File > Export**.

The save Metadata Export File dialog box is displayed.

3 Enter a name for the exported file.

4 Click **Save**.

The Export dialog box is displayed.

5 Select the components to export.

6 Click **Save**.

Note:

If you export locations, you can also export the associated maps by selecting Export Maps with Locations on the Options tab.

System Log

From the Web client, select Analysis > Log to open the system activity log. This log lists all system transactions.

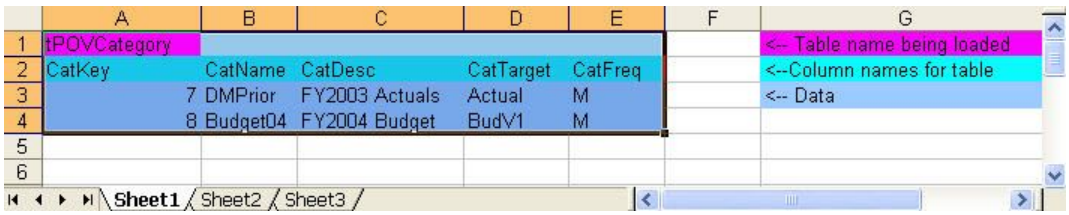
From Transaction Keys, select the transaction type you want to view. The log shows the activities that have occurred , which user performed the activity, and at what time.

There are three fields that present detailed information about each transaction:.

- Event Info—Displays statistics or information about the event that was logged.
- Error Info—Displays a Visual Basic or Microsoft® DAO error code. This information may be required by Hyperion to determine the cause of and to correct errors.
- IO Source (Input / Output)—Lists both RDBMS Statements that were run for the event, and external files that were either created or imported.

Importing Data from Microsoft Excel

The Import XLS tool enables you to import data directly into any table in the current FDM database. Therefore, you can import large amounts of application setup data (metadata) without having to enter each line manually in the Web client setup grids. For an Excel spreadsheet to be used as an import source, it must have a valid Excel import range that contains the table name to be imported along with the column names for each column in the range.



The grid represents a range name in Excel (upsCategory). The first cell of the range must contain the FDM table name in which to import. The second row of the range must contain the field names of the table. Rows 3 and higher contain the data to load to FDM.

Multiple ranges can be identified in one spreadsheet. They must each begin with the FDM import identifier (ups). If the tables you are importing to are related, the parent table must be imported prior to the child table. In this case, name the Excel ranges in alphabetical order. For example, the range name “upsAParent” processes before the range name “upsBChild.”

Note:

Any dates in the data should be formatted as text.

➤ To import data from an Excel spreadsheet:

1 From Workbench or Web client, select **Tools > Import XLS**.

The Import XLS dialog box is displayed.

2 Browse to find the Excel file to import.

3 Click **Open**.

Replacing Data

Create a Visual Basic (VB) function in a module within the Excel file named `UpReplace` and set its return value to "True." This instructs FDM to delete all data from the `TDim` table prior to adding the new data.

Example:

```
Public Function UpReplace() as Boolean
'Set value to true to force delete prior to load
UpReplace = True
End Function
```

Note:

The UpReplace function can only be used in the `tDim` table.

Map Converter

Map Converter is used to triangulate dimension maps when changing to a new target application. For example, if there is a map in the Texas location between the general ledger and the current target system, and in the Michigan locations there is a map between the current target system to the new target system, Map Converter creates a new map with from the general ledger to the new target system.

➤ To use Map Converter:

1 From within Workbench or the Web client, select **Tools > Map Converter**

The Map Converter screen is displayed.

2 Click the **Dimension** tab.

3 Select the dimension with the map to convert.

4 Click the **Source Location** tab.

5 Select the FDM location that contains the new map (between the original target account and the new target account).

6 Click the **Target Location** tab.

7 Select the FDM location that contains the original map (GL > Original Target Account).

8 Click the **Preview and Convert** tab.

9 Click **Preview the Conversion**.

Preview the Conversion enables you to view the map for the original location without performing the change.

10 Click **Perform the Conversion** to convert the map.

The location that had the original map (GL > Original Target Account) is updated to store the converted map.

Table Editor

Table Editor enables you to open any existing table in the current FDM database.

➤ To open a table:

1 From within Workbench or the Web client, select **Tools > Table Editor**.

2 From **Table**, select the table to edit.

This functionality is useful when there are custom tables attached to the FDM application that require updating.

Note:

Use caution when using Table Editor. Changes made here have the potential to cause errors in the application, as well as invalidate the system audit trail.

Data Segments

You can create, delete, recreate, and reassign data segments from within Workbench.

Creating New Segments

The Create New Segments function enables you to add additional data segment tables to the database without recreating the entire database.

➤ To create new data segments:

1 From Workbench, select **Tools > Manage Data Segments > Create New Segments**.

The Create New Segments dialog box is displayed.

2 From **Segments**, select the number of segments to create.

3 Click **Save and Close**.

Note:

This reassigns the locations that have already been created to the new segments. All locations that have been created remain in their original data segment.

Deleting, Recreating, and Reassigning Data Segments

This function enables you to change the number of active data segments. This process also reassigns the existing locations to the new data segments. When this is done, all data associated with the existing locations (including maps) is deleted.

➤ To delete, recreate, and reassign data segments:

1 From Workbench, select **Tools > Manage Data Segments > Delete, Recreate, and Reassign All Segments**.

The Recreate Segments dialog box is displayed.

2 From **Segments**, select the number of segments to create.

3 Click **Save and Close**.

Text Editor

Text Editor enables you to open and edit any text file from within Workbench. Open the editor by clicking selecting Tools > Text Editor.

Backup Application Files Function

The Backup Application Files function automatically compresses all files in the FDM `Application` directory into one `*.zip` file. Files included are:

- Scripts
- Reports
- Excel templates
- Logs
- Archives
- Any other files that have been created or saved in the application directory

This functionality is useful for archiving applications for backup purposes as well as for support issues when needed.

➤ To back up application files:

1 From Workbench, select **File > Backup Application Files**.

2 Enter a name for the archive file.

3 Click **OK**.

19 Scripting

In This Chapter

About Scripting

Note:

This chapter makes references to the API Guide. Contact your Hyperion representative or implementation consultant to obtain a copy of this guide.

FDM uses the Visual Basic scripting engine to enhance the flexibility and power of the product. Scripts are created by using standard VB Script syntax. The internal object model (FDM-specific functions) can also be referenced in a FDM script (see the FDM *API Guide*).

Script Editor

The Script Editor is used to define Visual Basic scripts that can be run in response to a FDM event, a custom menu selection, or during the file import process. Scripts are created by using the script editor, are saved in the FDM application's `Data\Scripts` directory and have a `.uss` extension. Scripts can be copied to other FDM applications. You can also edit scripts by using a text editor or XML editor.

Web Client

The script editor will automatically be launched during certain script-related procedures. To directly access Script Editor from within the Web client, select Tools > Script Editor.

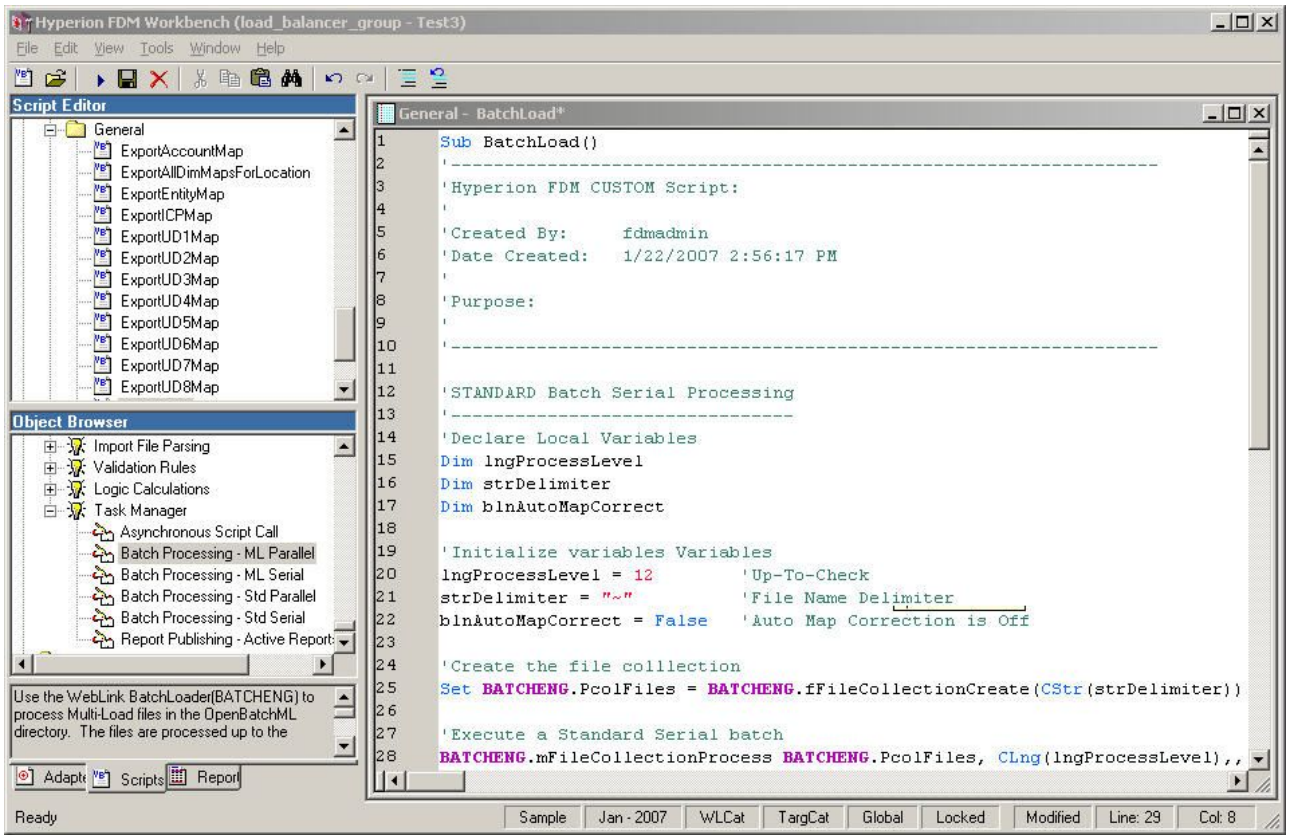
Workbench

The script editor will automatically be launched during certain script-related procedures. To access Workbench Script Editor, activate the Scripts tab. The upper left tree contains directories that contain the three types of FDM scripts. In the bottom tree is a hierarchy of the accessible FDM objects. Each FDM object corresponds to a public class module within a FDM `.dll`. Each object contains sub-objects, methods, functions, and properties that can be accessed in a script.

Double-clicking on a method, function, or property inserts the code directly into the script. All FDM objects can be referenced in any of the three types of scripts, with the exception of the API object, which cannot be referenced in an Import script.

Accelerators

The Script Editor (Workbench only) contains accelerators in the Object Browser section of the script editor page. Accelerators are code modules that simplify the process of creating of scripts. To use an accelerator, double-click the accelerator name. This automatically adds the accelerator code at the cursor position in the open script.



Import Scripts

Import scripts are usually used to manipulate source data upon importing a source file. Import scripts are executed during the import process every time a source line is read by FDM. When FDM reads a source file, it automatically skips every line that does not contain a valid amount, but scripts in the Amount field are still executed. This provides the flexibility to write scripts for lines that FDM would otherwise skip (in other words, the ability to store variables that can be retrieved in scripts assigned to other import fields). All the FDM objects are supported in import scripts except the FDM API object.

Creating an Import Script

An Import script can be assigned to any import field within an import format.

➤ To build an import script:

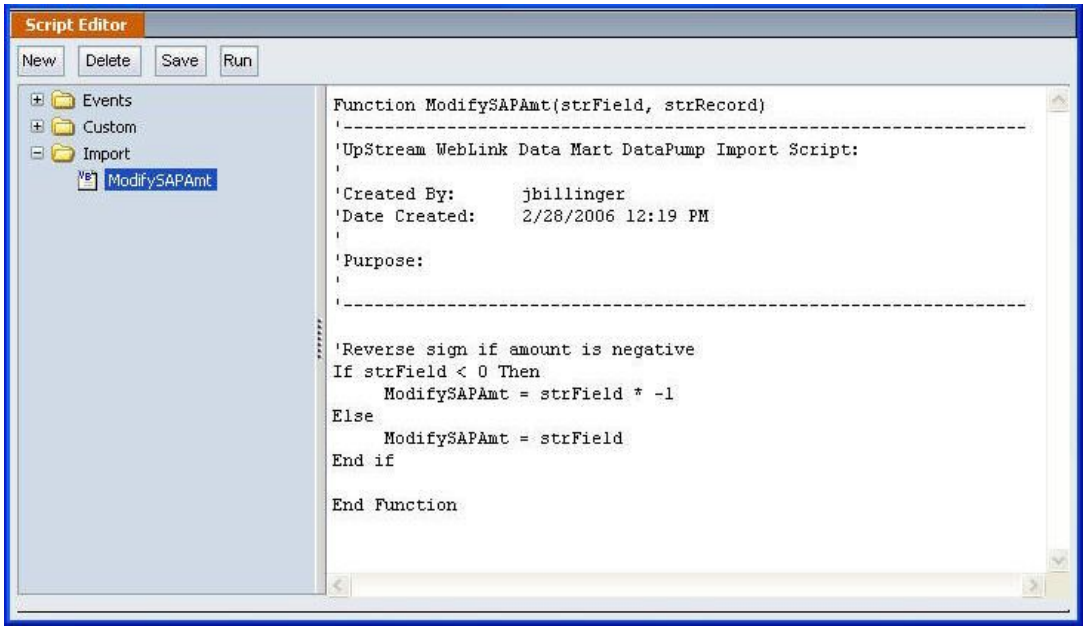
1. Open the Import Format screen by selecting **Metadata > Import Formats**.
2. Right-click in the **Expression** field of the line to add the script and select **Build Script**.

   The Script Editor is displayed.
3. Click **New** to create a new import script.
4. From **Script Type**, select **Import (DataPump)**.
5. In **File Name**, enter the name of the script.
6. Click **OK**.

Import Script Parameters

There are two parameters that are passed into an import script:

- strField—This parameter contains the field value in the source file that was defined in the import format. For example, if the import script is assigned to the Amount field, then this parameter contains the amount. If the import script is assigned to the Account field then this parameter will contain the account.
- strRecord—This parameter contains the entire line (record) of the source file that is being scanned.



strField and strRecord are passed to the script.

Click Save to save the script.

Import Script Function Return Value

In an import script, you must assign a value to the function name. This value is what FDM imports. Referring to the previous example, if the source amount is negative then the sign is reversed and the new value is assigned to the function name. This modified value is what is imported.

Assigning Import Script to Import Format

After the import script has been created and saved in the script editor, add the script to the Import field.

➤ To add an import script to an Import field:

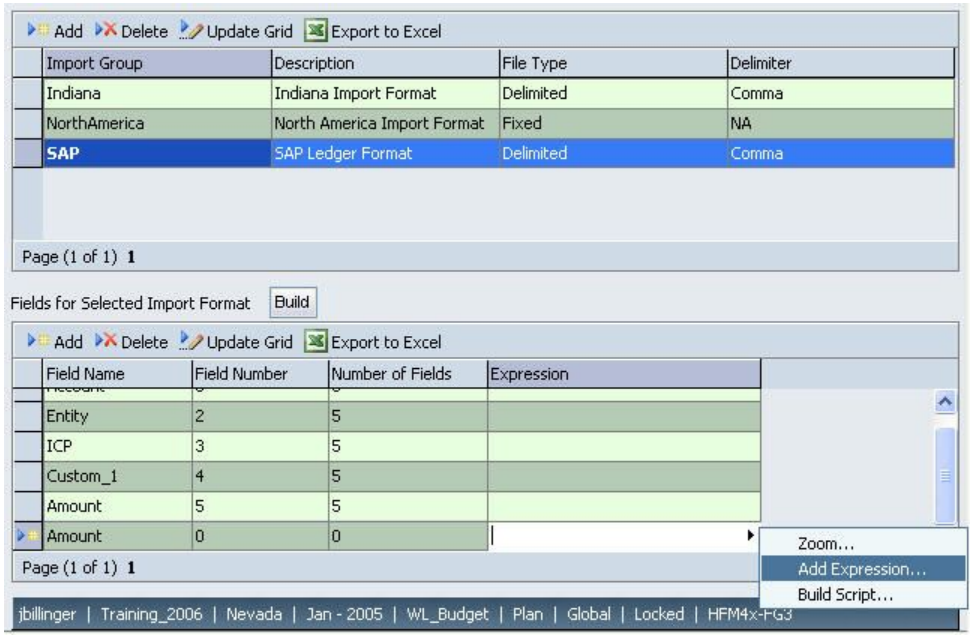
1 Within the Web client, select **Metadata > Import Formats**.

The Import Formats screen is displayed.

2 Select an import format group (top grid).

3 Right-click the **Expression** field of the row to add the script (bottom grid) and select **Add Expression**.

The Script Editor dialog is displayed.

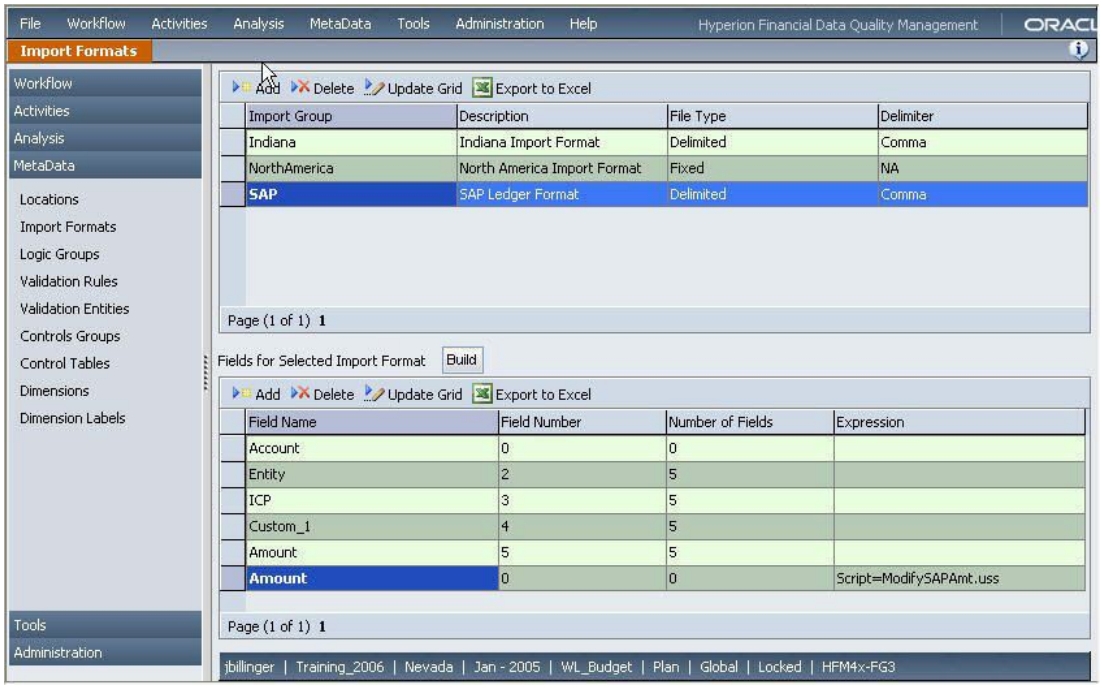


4 From **Expression Type**, choose **Script**.

5 Browse for the import script and select it.

6 Click **OK**.

The script is assigned to the Import field. The name of the import script is displayed in the Expression column.

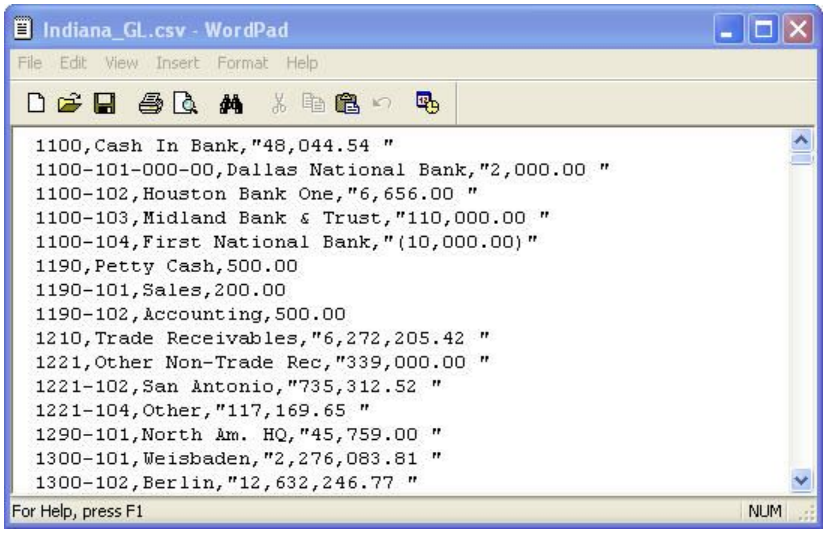


Import Scripting Functions

Left Function

```
Left (string, # of characters)
```

This is the example of the Indiana GL. The mapping for this location only uses the first four digits of the account number. One solution is to change the mapping by adding wildcards. Another solution is to import only the first four characters by using an import script.

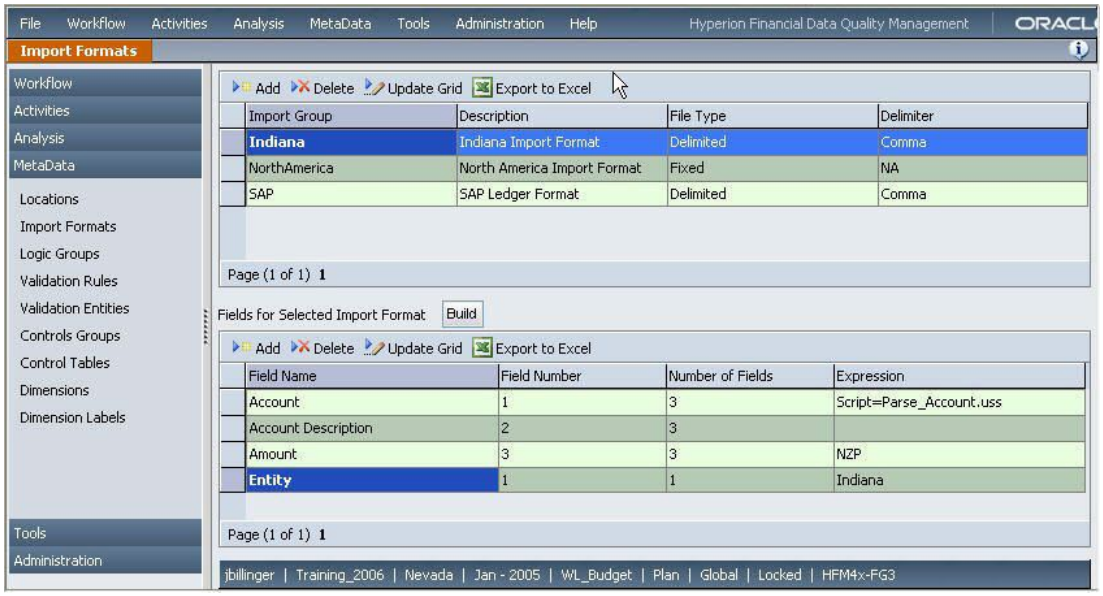


The account consists of the left four characters of the first field. Use the Visual Basic `Left` function to return the first four characters of the field.

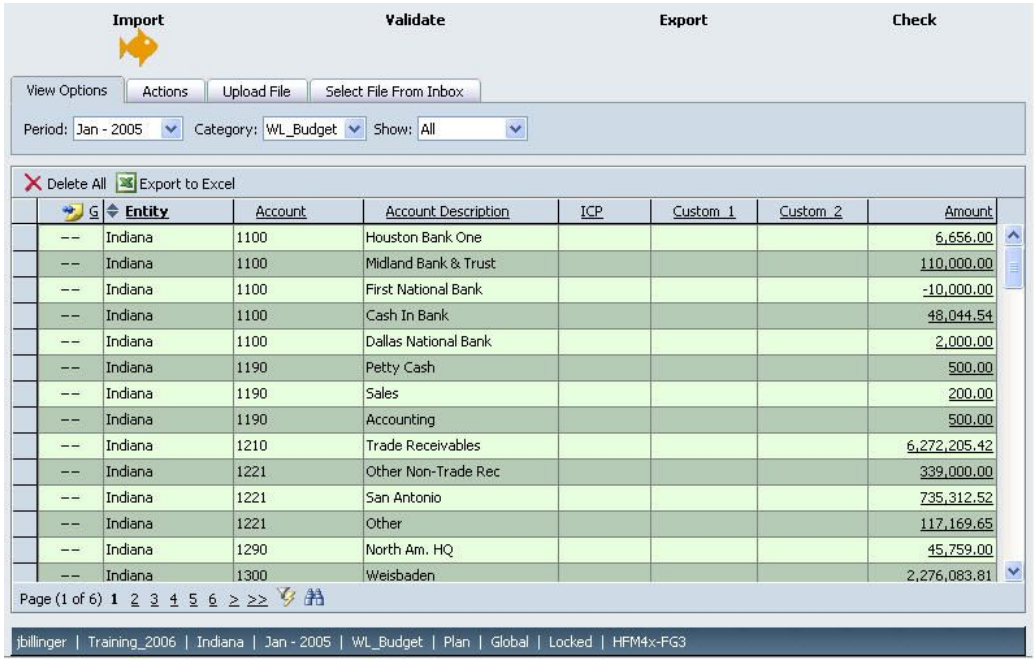
```
---------------------------------------------------------------------
Function Parse_Account [strField, strRecord]
‘-------------------------------------------------------------------
‘ FDM DataPump Import Script:
‘Created by:    FDM_Admin
‘Date created: 2/28/2006
‘-------------------------------------------------------------------
Parse_Account = Left (strField, 4)
End Function
---------------------------------------------------------------------
```

The code assigns the first four digits of the account number to `Parse_Account`.

After adding this script to the Expression column in the Account field of the import format it now runs for each line in the source file. The `Parse_Account` script overrides the field with its result.



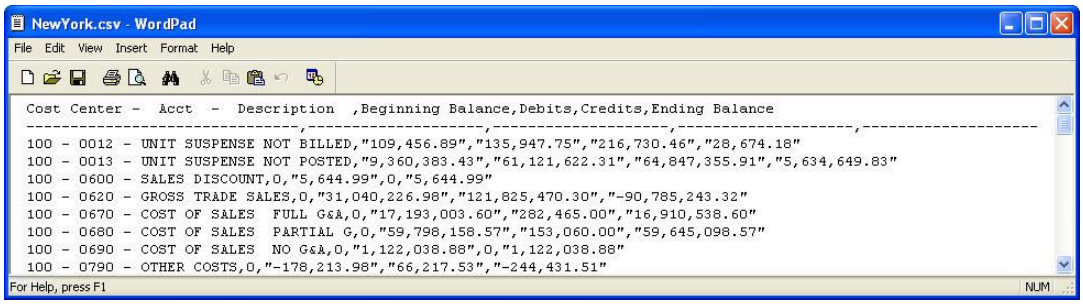
The following example illustrates the import file for the import format using `Parse_Account` script in the Account field.



Mid Function

```
Mid (string, start character, # of characters)
```

The file shown in the following example contains the cost center, account, and the account description as a continuous string in the first field. This string must be separated into separate fields. The original source file cannot be changed but you can write a script to parse the fields.

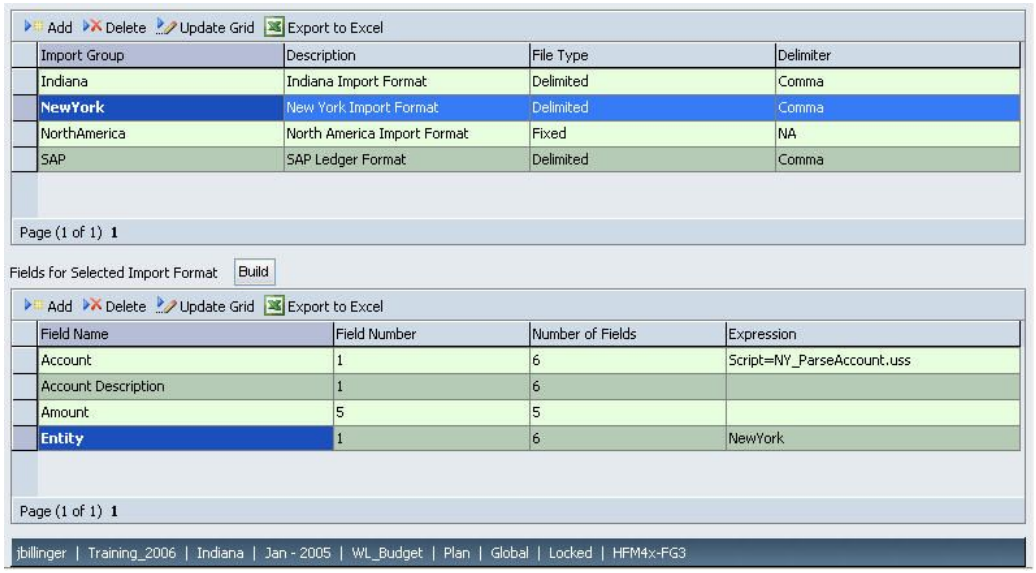


The account starts at Column 7 and is four characters long. Use `Mid` to return the characters in Columns 7 through 10.

```
-------------------------------------------------------------------
Function NY_ParseAccount [strField, strRecord]
'-----------------------------------------------------------------
' FDM DataPump Import Script:
'Created by:    FDM_Admin
'Date created: 2/28/2006
'-----------------------------------------------------------------
NY_ParseAccount = Mid (strField, 7,4)
End Function
-------------------------------------------------------------------
```

`Mid` in this script assigns the string of characters from Columns 7-10 to `NY_ParseAccount`.

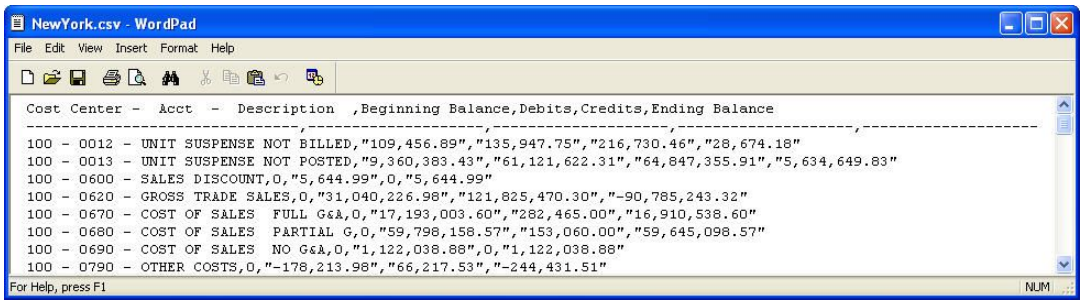
The Import script is assigned to the Account field (next example).



Right Function

`Right (string, # of characters)`

The account number in this file consists of the last four characters of the first field.



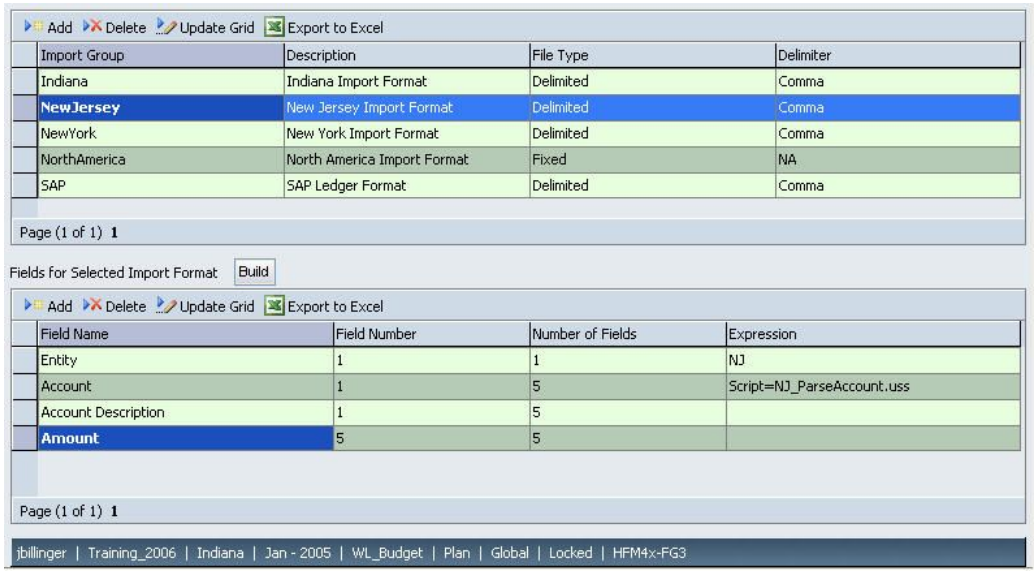
The first field in this comma-delimited file includes the account description, cost center, and account number. The field must be parsed in order to extract each of the elements separately.

Use `Right` to return the last four characters of the first field.

```
---------------------------------------------------------------------
Function NJ_ParseAccount [strField, strRecord]
‘-------------------------------------------------------------------
‘ FDM DataPump Import Script:
‘Created by:    FDM_Admin
‘Date created: 2/28/2006
‘-------------------------------------------------------------------
NY_ParseAccount = Right (strField,4)
End Function
---------------------------------------------------------------------
```

This script assigns the last four characters of the first field in each row of the source file to `NJ_ParseAccount`.

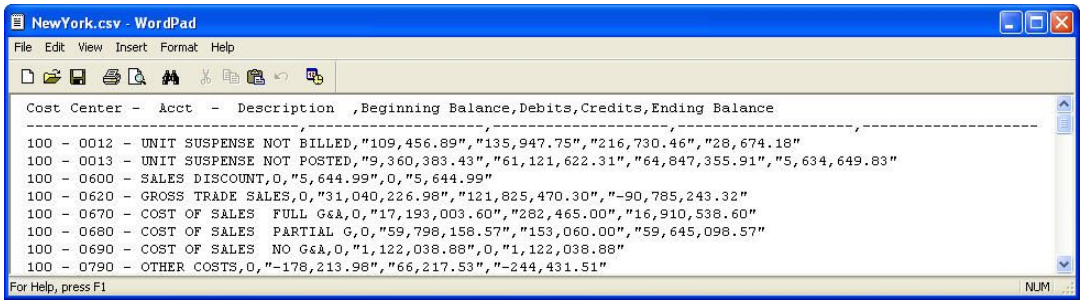
The script is assigned to the Account field.



FDM Parsing Function

`DW.Utilities.fParseString (string, total field count, field # to return, delimiter)`

The `fParseString` function is not a Visual Basic function but a FDM function used to parse strings that contain a delimiter. In he file shown in the following example, the first field is separated by dashes (“-“). By using fParseString, this field can be separated into three distinct fields by specifying the dash as the delimiter. Use `fParseString` to retrieve the entity and description fields.



The entity in the source file is the set of digits before the first hyphen in the first field.

```
-------------------------------------------------------------------
Function NY_ParseCenter [strField, strRecord]
‘-----------------------------------------------------------------
‘ FDM DataPump Import Script:
‘Created by:    FDM_Admin
‘Date created: 2/28/2006
‘-----------------------------------------------------------------
NY_ParseCenter = DW.Utilities.fParseString (strField, 3, 1, “-”)
End Function
-------------------------------------------------------------------
```

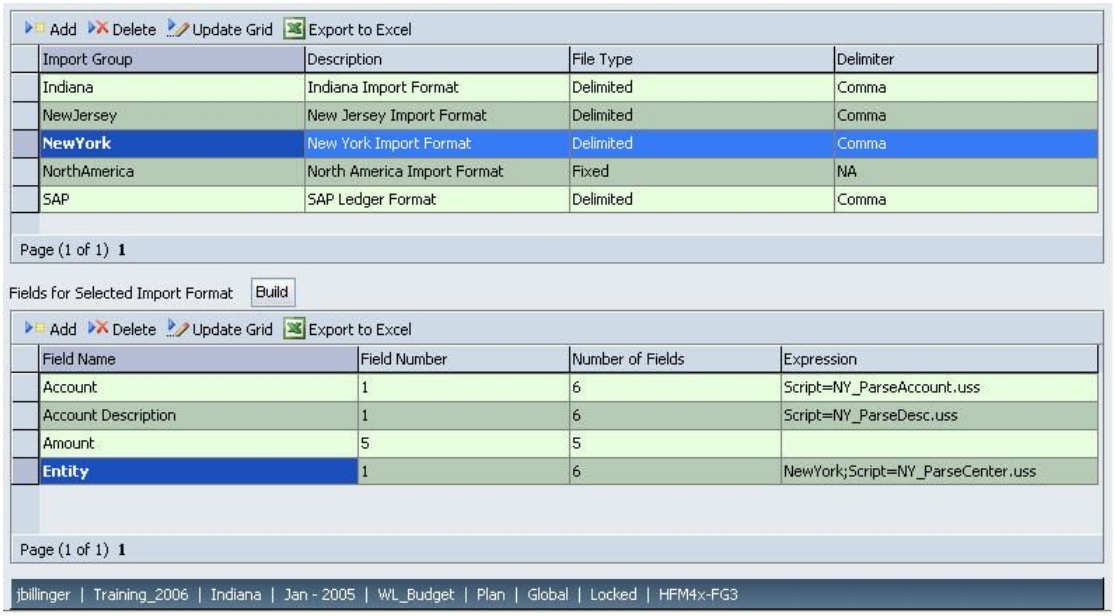
`NY_ParseCenter` returns the entity (first set of numbers before the first hyphen in the first field of the data file).

The description is the last set of characters in the first field of the source file row (after the second hyphen).

```
Function NY_ParseDesc [strField, strRecord]
‘-----------------------------------------------------------------
‘ FDM DataPump Import Script:
‘Created by:    FDM_Admin
‘Date created: 2/28/2006
‘-----------------------------------------------------------------
NY_ParseDesc = DW.Utilities.fParseString (strField, 3, 3, “-”)
End Function
```

`NY_ParseDesc` returns the description (the set of characters after the second hyphen in the first field of the data file).

The scripts are assigned to the Entity and Description fields.



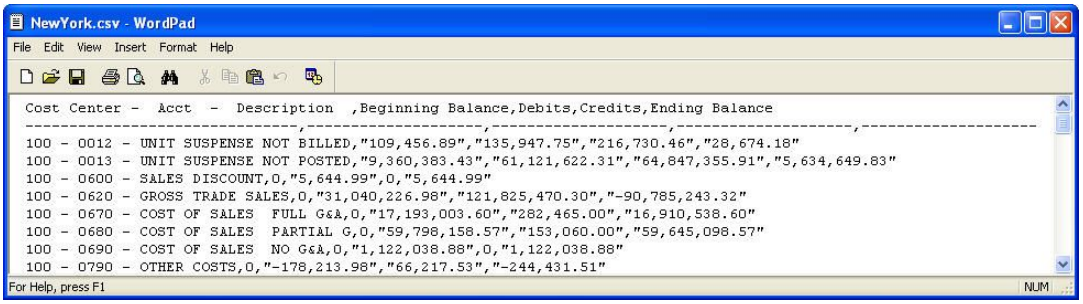
Note:

The Parse_Account script shown here was created in an earlier example.

Skip Function (conditional skip)

`RES.PblnSkip`

You can selectively skip source file lines by using `RES.PblnSkip`. This function skips the entire row so it is not imported into FDM. In this example of the New York source file, any entity that begins with “06” must be skipped.



The following script skips all entities that begin with “06.” The script uses the FDM `fParseString` function to parse the entity field. Also, a temporary variable is used in this script. This is a local variable whose value is lost when the script has finished its execution.

```
'-----------------------------------------------------------------------
Function NY_Skip06Center [strField, strRecord]
'-----------------------------------------------------------------------
' FDM DataPump Import Script:
'Created by:    FDM_Admin
'Date created: 2/28/2006
'-----------------------------------------------------------------------
Dim strEntity
```

```
'Store first entity in a variable
strEntity = DW.Utilities.fParseString (strField, 3, 1, "-")
'Check first two characters of entity
If Left(strEntity, 2) = "06" then
    'Skip line
    Res.PblnSKip = True
Else
    NY_Skip06Center = strEntity
End if
End Function
-----------------------------------------------------------------------
```

This script skips all entities that begin with "06."

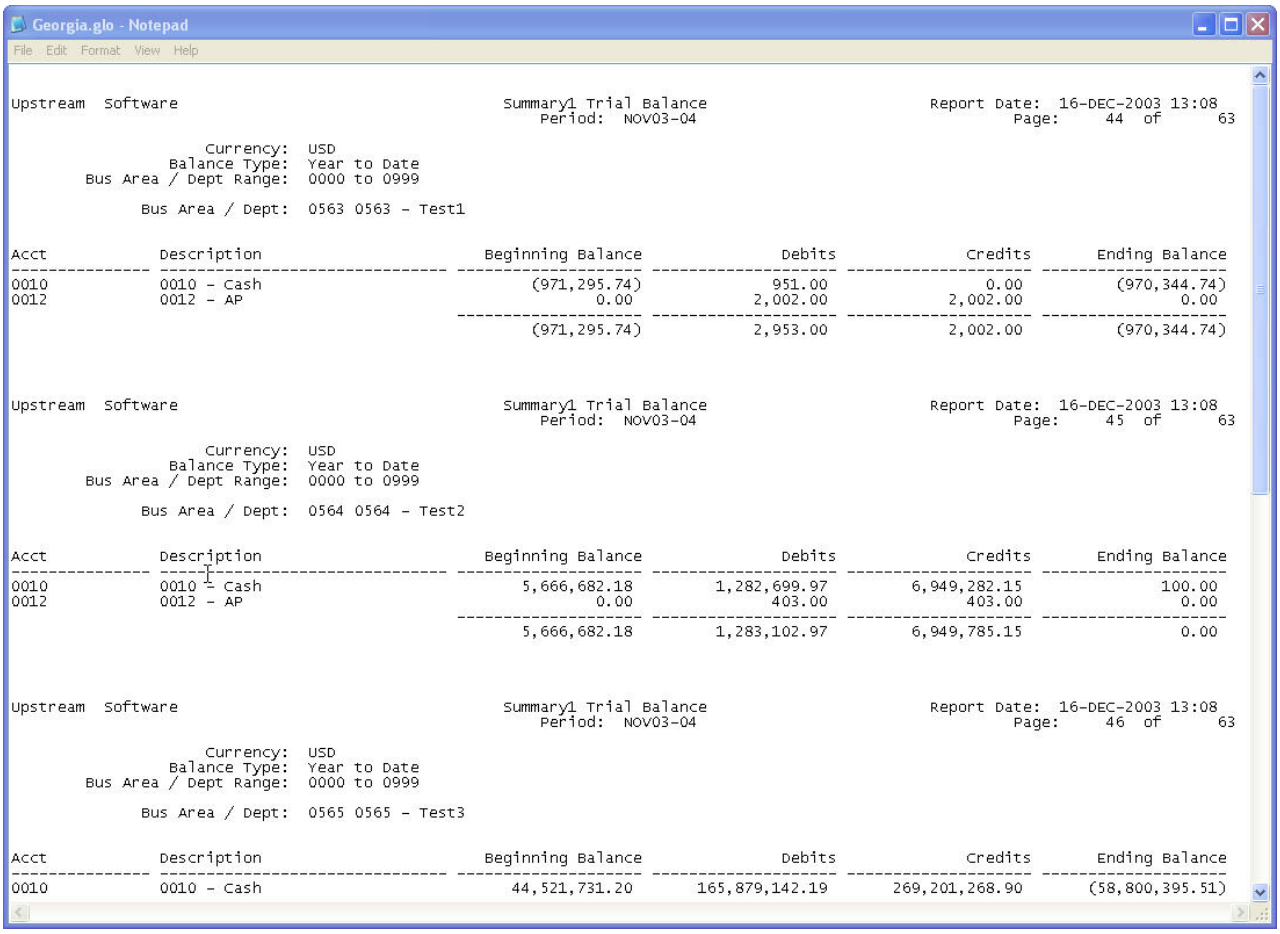
Temporary Variables

Storing Temporary Variables

`RES.PvarTemp1.....RES.PvarTemp5`

Some source files contain fields that do not repeat on every line (Entity or Description, for example). When FDM imports a source file, it automatically skips every line that does not contain a valid amount, but scripts in the Amount field are still executed. This allows the flexibility to write scripts for lines that FDM would otherwise skip. Therefore, you can write scripts in the Amount field to store variables that can be retrieved in scripts assigned to other import fields. FDM can store up to five different temporary variables simultaneously.

The example GL that follows shows that the entity is not repeated in every row. The entity is contained in the header for each section of the report and appears after the "Bus Area/Dept" label. The account, description, and ending balance can be easily identified. The entity must be stored in a temporary variable and assigned to every row in the export file.



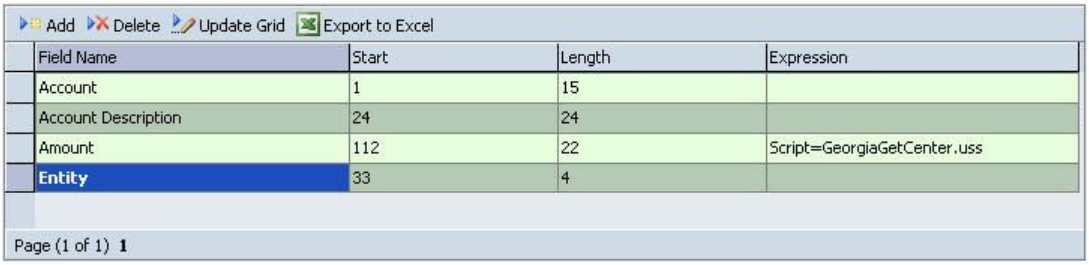
The following script uses an `If...Then` statement and the `Mid` function to check if the line contains the label “Bus Area/Dept:” If the line contains the label, then the script will store the entity in temp variable. The center begins at position 33 and is four characters long. If a line that does not include the center is read, then `strField` is assigned to the function name (in this example, GeorgiaGetCenter=strField).

The `RES.PvarTemp1` through `RES.PvarTemp5` variables are global—unlike local variables that lose their value when the current script closes, these temporary variables do not lose scope while the current FDM session remains open. Therefore, FDM cam store values from within one script and retrieve the values from within another script.

```
------------------------------------------------------------------------
Function GeorgiaGetCenter [strField, strRecord]
‘------------------------------------------------------------------------
‘ FDM DataPump Import Script:
‘Created by:    FDM_Admin
‘Date created: 2/28/2006
‘------------------------------------------------------------------------
If Mid(strRecord, 15,16) = “Bus Area / Dept:” then
    RES.PvarTemp1 = Mid(strRecord,33,4)
End if
GeorgiaGetCenter = strField
End Function
------------------------------------------------------------------------
```

If the data file row starts with “Bus Area / Dept:” then `Mid` is used to store the entity in `Temp Variable 1`.

Because the Amount field is the only rule that executes even if the line in the source code does not contain a valid amount, the script in the following example must be assigned to the Amount field in the import format .



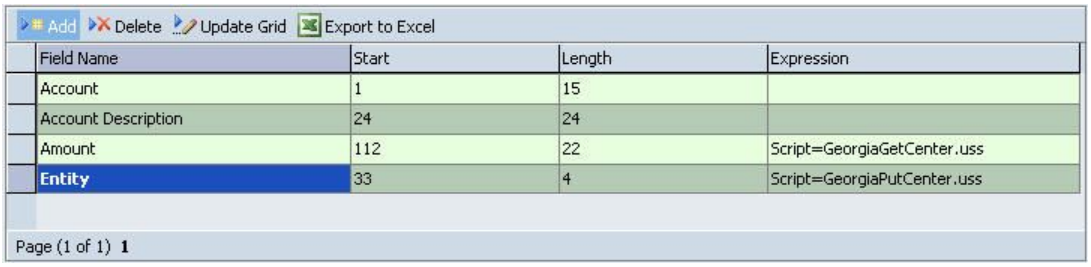
Retrieving Temporary Variables

After the entity is stored in a temporary variable, a script is required to assign the temporary variable to the Entity field. The following script assigns the temporary variable to the function name.

```
--------------------------------------------------------------------
Function GeorgiaPutCenter [strField, strRecord]
‘-------------------------------------------------------------------
‘ FDM DataPump Import Script:
‘Created by:    FDM_Admin
‘Date created: 2/28/2006
‘-------------------------------------------------------------------
GeorgiaPutCenter = RES.PvarTemp1
End Function
--------------------------------------------------------------------
```

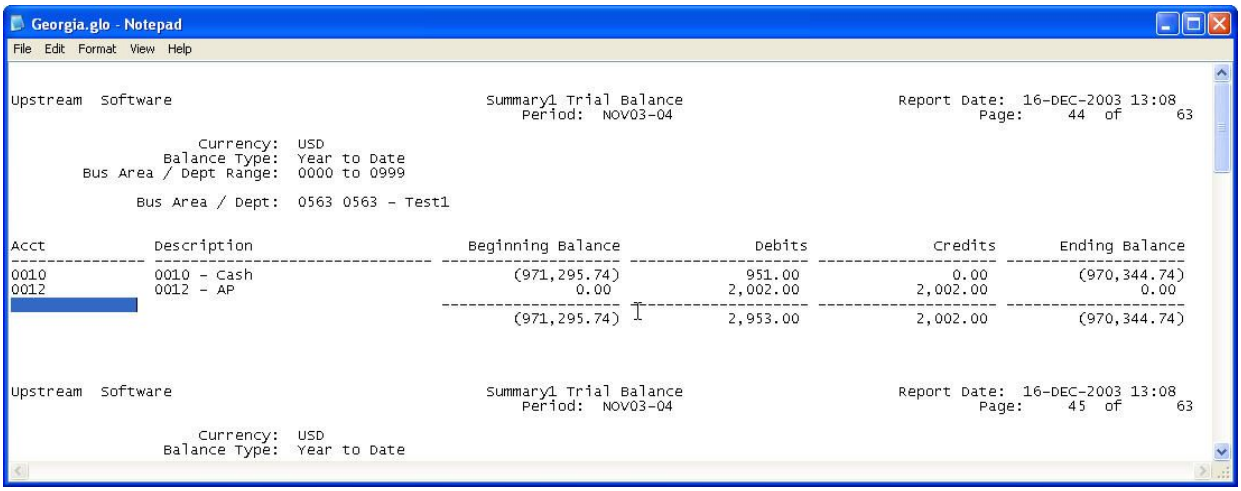
The temporary variable `PvarTemp1` is assigned to `GeorgiaPutCenter`.

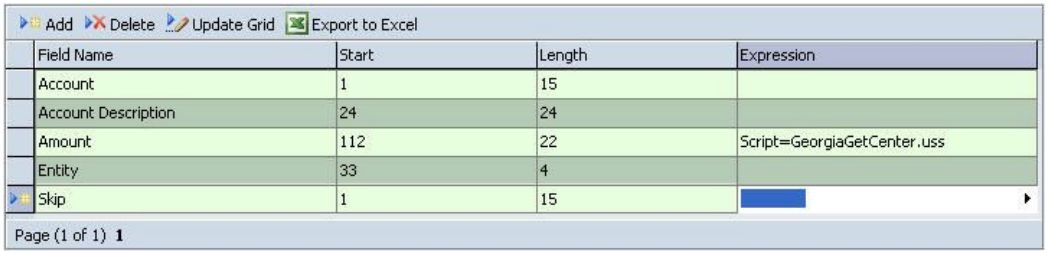
The script is then assigned to the Entity field.



With the `GeorgiaPutCenter` script assigned to the Entity field, the entity is assigned the value of the temporary variable instead of reading the entity from the source file.

This format also requires the use of `Skip` function because it includes subtotal rows that must not be imported. Configure the skip line to be triggered by recognizing a blank Account column (15 blank spaces). Use the drag-and-drop function to define the Start and Length fields for the expression. FDM now skips every line with no account number.





Use the `Skip` function to skip the subtotal rows (identified by the blank Account column) to complete the import format for Georgia.

Integration Import Scripts

Integration import scripts must be used when importing data directly from an ODBC/OLEDB-compliant data source (rather than an ASCII file). This integration script is responsible for connecting to the source data and importing data directly from the source system into the FDM database. This script is run when the user clicks the Import button on the Import screen. All FDM objects are supported in the import integration scripts except the FDM API objects.

Creating an Integration Import Script

When defining the import group, select Script as the file type. Do not specify individual fields in the bottom table like a normal import format. Instead, there is only a single field where name of the integration script is input.

➤ To create an import integration script:

1. From the Web client, select **Metadata > Import Formats**.
2. From the top grid, select an import format group.

   The format group must be configured with Script as the file type.
3. On the bottom grid, click the arrow to the right of the row and select **Build Script ...**from the menu.

The Script Editor dialog box is displayed.

4 Click **New**.

5 From **Script Type**, select **Import (Integration)** script.

6 Enter in the name of the script.

7 Click **OK**.

Import Integration Script Parameters

There are four parameters that are passed into an Integration Import script:

- strLoc—Contains the active FDM location key. Reference this parameter when updating the FDM worktable with the source data.
- lngCatKey—Contains the active FDM category key. Reference this parameter when updating the FDM worktable with the source data.
- dblPerKey—Contains the date serial key of the active FDM period. Reference this parameter when updating the FDM worktable with the source data.
- strWorkTableName—Contains the name of the worktable that the source data is imported into for this FDM location.

FDM Worktable Fields

When creating an Integration Import script, you must populate the FDM working table with the source values retrieved from the data source. The following table lists all of the available FDM working table fields.

Table 2 FDM Worktable Fields

Field Name	Field Info	Note
PartitionKey	[Type=Long, Size=0, Required=Yes]	WL location name. Use the strLoc parameter
CatKey	[Type=Long, Size=0, Required=Yes]	WL category key. Use the lngCatKey parameter
PeriodKey	[Type=TimeStamp, Size=0, Required=Yes]	WL period key. Use the dblPerKey parameter
DataView	[Type=VarWChar, Size=5, Required=Yes]	Load Frequency. Defaults to "YTD"
Amount	[Type=Double, Size=0, Required=Yes]	Source amount
Desc1	[Type=VarWChar, Size=75, Required=No]	Source account description
Account	[Type=VarWChar, Size=75, Required=Yes]	Source account
Entity	[Type=VarWChar, Size=75, Required=Yes]	Source entity
ICP	[Type=VarWChar, Size=75, Required= No]	Source ICP
UD1	[Type=VarWChar, Size=75, Required= No]	Source UserDefined 1 dimension

Field Name	Field Info	Note
UD2	[Type=VarWChar, Size=75, Required= No]	Source UserDefined 2 dimension
UD3	[Type=VarWChar, Size=75, Required= No]	Source UserDefined 3 dimension
UD4	[Type=VarWChar, Size=75, Required= No]	Source UserDefined 4 dimension
UD5	[Type=VarWChar, Size=75, Required= No]	Source UserDefined 5 dimension
UD6	[Type=VarWChar, Size=75, Required= No]	Source UserDefined 6 dimension
UD7	[Type=VarWChar, Size=75, Required= No]	Source UserDefined 7 dimension
UD8	[Type=VarWChar, Size=75, Required=No]	Source UserDefined 8 dimension
UD9	[Type=VarWChar, Size=75, Required=No]	Source UserDefined 9 dimension
UD10	[Type=VarWChar, Size=75, Required=No]	Source UserDefined 10 dimension
UD11	[Type=VarWChar, Size=75, Required=No]	Source UserDefined 11 dimension
UD12	[Type=VarWChar, Size=75, Required=No]	Source UserDefined 12 dimension
UD13	[Type=VarWChar, Size=75, Required=No]	Source UserDefined 13 dimension
UD14	[Type=VarWChar, Size=75, Required=No]	Source UserDefined 14 dimension
UD15	[Type=VarWChar, Size=75, Required=No]	Source UserDefined 15 dimension
UD16	[Type=VarWChar, Size=75, Required=No]	Source UserDefined 16 dimension
UD17	[Type=VarWChar, Size=75, Required=No]	Source UserDefined 17 dimension
UD18	[Type=VarWChar, Size=75, Required=No]	Source UserDefined 18 dimension
UD19	[Type=VarWChar, Size=75, Required=No]	Source UserDefined 19 dimension
UD20	[Type=VarWChar, Size=75, Required=No]	Source UserDefined 20 dimension
Attr1	[Type=VarWChar, Size=20, Required=No]	Source Attribute 1 dimension
Attr2	[Type=VarWChar, Size=20, Required=No]	Source Attribute 2 dimension
Attr3	[Type=VarWChar, Size=20, Required=No]	Source Attribute 3 dimension
Attr4	[Type=VarWChar, Size=20, Required=No]	Source Attribute 4 dimension
Attr5	[Type=VarWChar, Size=20, Required=No]	Source Attribute 5 dimension
Attr6	[Type=VarWChar, Size=20, Required=No]	Source Attribute 6 dimension
Attr7	[Type=VarWChar, Size=20, Required=No]	Source Attribute 7 dimension
Attr8	[Type=VarWChar, Size=20, Required=No]	Source Attribute 8 dimension
Attr9	[Type=VarWChar, Size=20, Required=No]	Source Attribute 9 dimension

Field Name	Field Info	Note
Attr10	[Type=VarWChar, Size=20, Required=No]	Source Attribute 10 dimension
Attr11	[Type=VarWChar, Size=20, Required=No]	Source Attribute 11 dimension
Attr12	[Type=VarWChar, Size=20, Required=No]	Source Attribute 12 dimension
Attr13	[Type=VarWChar, Size=20, Required=No]	Source Attribute 13 dimension
Attr14	[Type=VarWChar, Size=20, Required=No]	Source Attribute 14 dimension
ArchiveID	[Type=Bigint, Size=8, Required=No]	If there is an archive to be associated then enter the archive ID# from the data directory.

Assigning Import Integration Script to Import Formats

After the Import Integration script has been created and saved in the script editor, you must add the script to the Import format just as you would with a regular Import script by clicking the Add Expression… link and browsing for the script file.

Import Integration Script Example

The script that follows is an example of an import integration script that opens a SQL server source database (sample Northwind database) and writes the data directly into FDM. Notice that a value of True is assigned to the function if the import was successful. Assign a value of False to the function if any errors are raised in the script.

```
---------------------------------------------------------------------
Function SQLIntegration(strLoc, lngCatKey, dblPerKey, strWorkTableName)
‘---------------------------------------------------------------------
‘ FDM Integration Import Script:
‘Created By:      Admin
‘Date Created: 04/19/2004 2:18:39 PM
‘Purpose: This import integration script connects to the sample
‘        Northwind SQL Server database and imports sample
‘        source data from the Orders table into FDM.
‘---------------------------------------------------------------------
Dim cnSS        ‘ADO connection object
Dim strSQL     ‘SQL string
Dim rs       ‘Source system recordset
Dim rsAppend     ‘ FDM recordset
‘Initialize ADO objects
Set cnSS = CreateObject(“ADODB.Connection”)
Set rs = CreateObject(“ADODB.Recordset”)
‘Open  FDM work table recordset for appending
Set rsAppend = DW.DataAccess.farsTableAppend(strWorkTableName)
‘Connect to Northwind SQL Server database (our data source)
Dim strConn
strConn=”Provider=SQLOLEDB.1;Integrated Security=SSPI;Persist Security
Info=False;”
strConn=strConn & “Initial Catalog=Northwind;Data Source=LocalHost;”
cnss.open strConn
```

```
‘Create source query string
strSQL = “Select * “
strSQL = strSQL & “FROM Orders “
‘Open source recordset
rs.Open strSQL, cnSS
‘Check for data in source system
If rs.bof AND rs.eof Then
‘Give error message
    RES.PlngActionType = 2
RES.PstrActionValue = “No records to load!”
    ‘Assign Return value of function
    SQLIntegration = False
    Exit Function
End If
‘Loop through source records in Northwind database (Orders table) and
append to FDM work table
If Not rs.bof And Not rs.eof Then
    Do While Not rs.eof
        rsAppend.AddNew
        rsAppend.Fields(“PartitionKey”) = RES.PlngLocKey
        rsAppend.Fields(“CatKey”) = lngCatKey
        rsAppend.Fields(“PeriodKey”) = dblPerKey
        rsAppend.Fields(“DataView”) = “YTD”
        rsAppend.Fields(“Amount”) = rs.fields(“Freight”).Value
        rsAppend.Fields(“Account”) = rs.fields(“CustomerID”).Value
        rsAppend.Fields(“Entity”) = rs.fields(“ShipCountry”).Value
        rsAppend.Fields(“Desc1”) = rs.fields(“ShipName”).Value
        rsAppend.Update
        rs.movenext
    Loop
End If
‘Give success message
RES.PlngActionType = 2
RES.PstrActionValue = “SQL Import successful!”
‘Assign Return value
SQLIntegration = True
End Function
-----------------------------------------------------------------------
```

Custom Scripts

Custom scripts can be executed by using a custom link on the Task Flow menu. Custom scripts allow users or administrators to manually execute a script. All the FDM objects are supported in custom scripts.

Creating a Custom Script

➤ To create a new custom script:

1 From Workbench, select **File > New Script**.

The New Script dialog box is displayed.

2 From **Script Type**, select **Custom (Web)**.

3 In **Script Name**, enter a name for the script.

4 Click **OK**.

There are three options for custom script types—General, Plug-in, and Web. General custom scripts are intended for use only in Workbench. Web scripts are scripts that are intended for use within the FDM Web Client. Plug-in scripts are generally custom integration solutions provided by Hyperion.

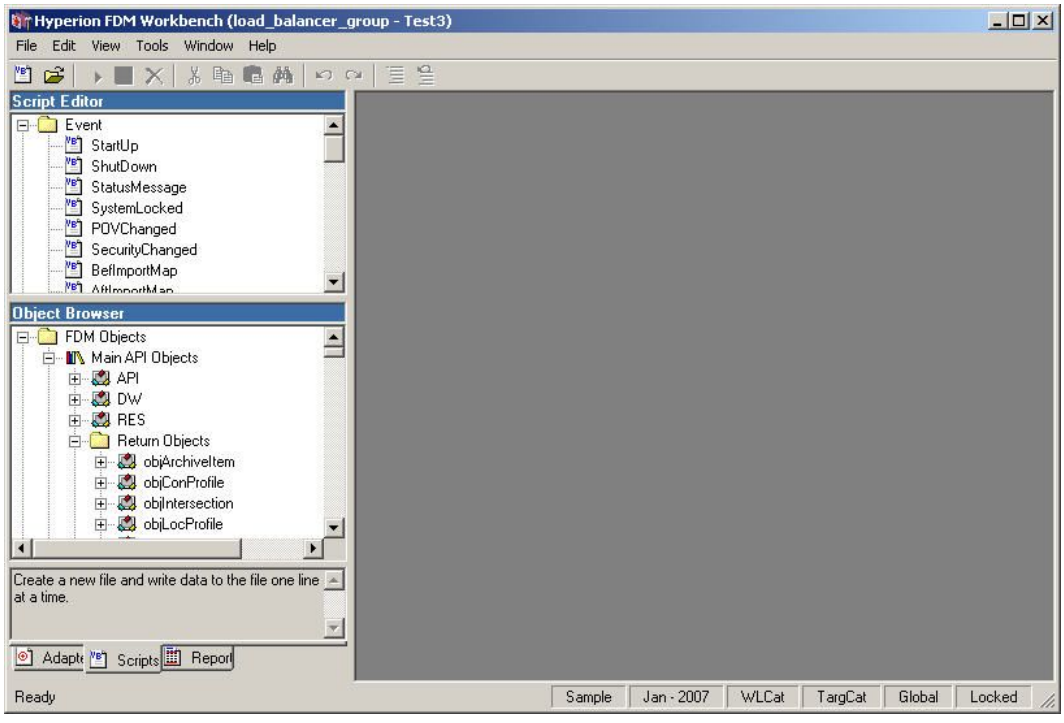
After adding the script, FDM automatically adds a “web” prefix to the name of the script.

Event Scripts

Event scripts are executed in response to specific FDM events. A list of trigger events is located in the Event directory within the script editor. See the API Guide for a description of all the FDM events. All the FDM objects are supported in the event scripts.

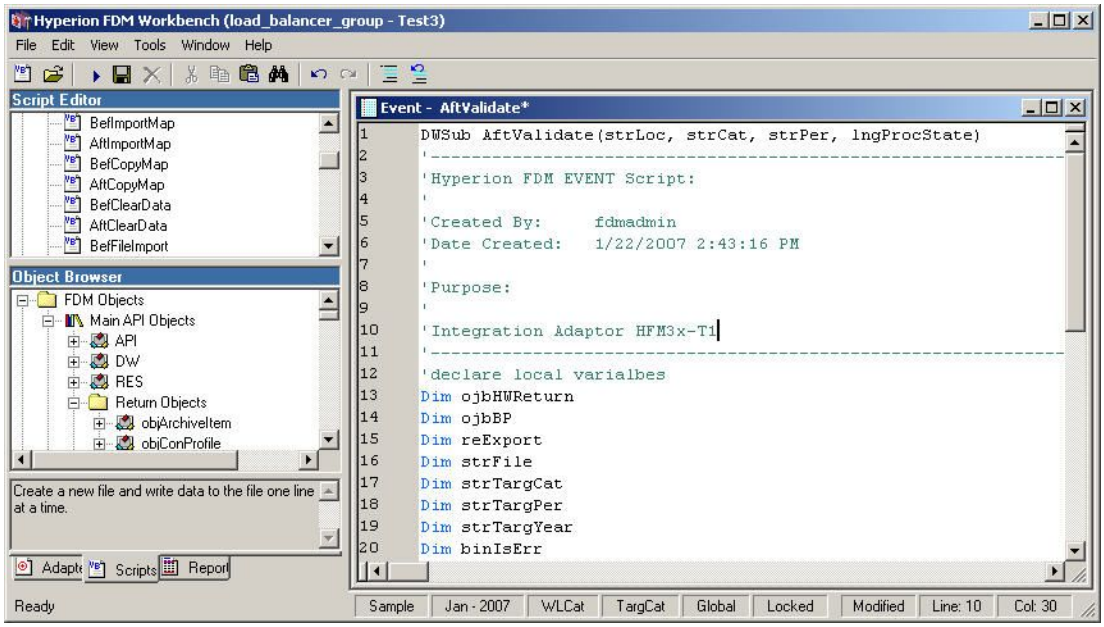
Creating an Event Script

To create a new event script, open the Script Editor in the FDM WorkBench and double-click the event for which to develop a script.



Event Script Parameters

Each FDM Event script has its own set of parameters. These parameters can be used within the script instead of calling a FDM function, method, or property that returns the same value. For example, the parameter `strLoc` contains the name of the active FDM location. This is more efficient than using the `API.POVMgr.PPOVLocation` property to return the name of the FDM location. See the FDM API guide for descriptions of all the event script parameters.



Dynamically Changing Import Formats

You can dynamically change the import format for a location using the `BefFileImport` event. In the example that follows, the script changes the import group based on the file name. To implement this script, you must create two import formats and assign one of them to the FDM location. The script evaluates the file name and changes the import format if necessary.

```
-------------------------------------------------------------------
Sub BefFileImport(strLoc, strCat, strPer, strFile)
‘-------------------------------------------------------------------
‘Hyperion EVENT Script:
‘
‘Created By:      Admin
‘Date Created:     10/28/2004 4:29:54 PM
‘
‘Purpose: Change the import group if importing the B/S
‘
‘-------------------------------------------------------------------
‘Check if the file name contains “BS”
If InStr(strFile,”BS”)>0 Then
    ‘Override default import group with B/S import group
    RES.PblnImportGroupOverride=True
    RES.PstrImportGroup=”BSImportFormat”
End If
End Sub
```

```
-----------------------------------------------------------------
```

Dynamically Changing Other Location Attributes

You can use event scripts to dynamically change validation groups, validation rules, and logic groups before they are processed. Refer to the following table for a description of the events and properties to change a location attribute.

Table 3 Events and properties to change a location attribute

Event Name	FDM Properties	Purpose
`BefFileImport`	`RES.PblnImportGroupOverride=True` `RES.PstrImportGroup=”MyNewImportFormat”`	Change Import Format before import
`BefProcLogicGrp`	`RES.PblnLogicGroupOverride=True` `RES.PstrLogicGroup=”MyNewLogicGroup”`	Change Logic Group before processing logic.
`BefConsolidate`	`RES.PblnValEntGroupOverride=True` `RES.PstrValEntGroup=”MyNewEntityGroup”`	Change Validation Entity Group before consolidating.
`BefCheck`	`RES.PblnValRuleGroupOverride=True` `RES.PstrValRuleGroup=”MyNewValRuleGroup”`	Change Validation Rule Group before running Validation report.

Using the File System Object in an Event Script

Use the Visual Basic File System Object to process files and folders. The following example uses the File System Object to create a new file and copies the contents of an existing file to the new file.

```
-----------------------------------------------------------------
Sub AftExportToDat(strLoc, strCat, strPer, strTCat, strTPer, strFile)
‘-----------------------------------------------------------------
‘ FDM EVENT Script:
‘
‘Created By:      Admin
‘Date Created:     3/18/2004 4:17:58 PM
‘
‘Purpose: This script loops through the newly created Hyperion
‘     Enterprise .dat file And creates a new .dat file with
‘     a different category and multiplies the amount by .75
‘-----------------------------------------------------------------
‘Declare local variables
Dim strline
```

```
Dim fso
Dim f1
Dim f2
Dim strNewFileName
Dim strEntity
Dim strAcct
Dim dblAmt
‘Declare file system object
Set fso = CreateObject(“Scripting.FileSystemObject”)
‘Open existing dat file for reading
Set f1 = fso.OpenTextFile(strFile, 1)
‘Create new .dat file with a “-FAS” suffix
strNewFileName=Left(strFile,Len(strFile)-4) & “-FAS” & Right(strFile,4)
Set f2=fso.CreateTextFile(strNewFileName,True)
‘Write category and beginning and ending periods to new file
f2.writeLine “FAS”
f2.writeLine strTPer
f2.writeLine strTPer
‘Skip first 3 header lines of existing dat file
f1.SkipLine
f1.SkipLine
f1.SkipLine
‘Loop through existing .dat file
Do While f1.AtEndOfStream <> True
‘Store line in a variable
     strline = f1.ReadLine
     ‘Parse the entity from the line
     strEntity = DW.Utilities.fParseString(strline, 3, 1, “,”)

     ‘Parse the account from the file
     strAcct = DW.Utilities.fParseString(strline, 3, 2, “,”)
     ‘Parse the amount from the file
     dblAmt = DW.Utilities.fParseString(strline, 3, 3, “,”)

     ‘Write out line to new .dat file but multiply amt by .75
     f2.writeline strEntity & “,” & strAcct & “,” & dblAmt * .75
Loop
‘Close the files
f1.Close
f2.Close

‘Destroy file system object
Set fso=Nothing
End Sub
----------------------------------------------------------------------
```

See Appendix A for more advanced information about scripting in FDM.

Menu Maker

The Menu Maker is used to assign custom scripts to new menu options. and enables scripts to be assigned a meaningful menu name that allows end users to invoke a script. After the script has been created in the Script Editor, a link to the script can be added to the Web Client interface to allow users to manually run the script.

➤ To add the script to the Task Flows screen:

1 Within the Web Client, select **Tools > Menu Maker**.

The Menu Maker screen is displayed.

2 Click **Add**.

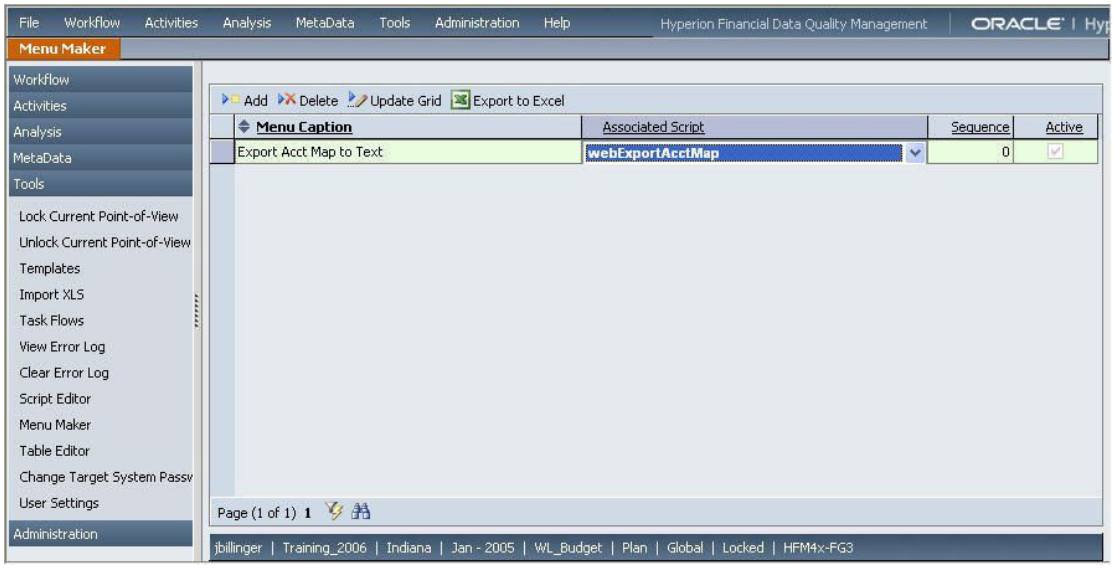
A new row is added to the grid.

3 In the **Menu Caption** column, Enter a caption for the menu.

4 Double-click in the **Associated Script** column and select the Web Script from the menu.

The sequence field controls the order of all the custom web scripts as they appear on the Task Flows screen.

5 Select the **Active** checkbox to display the script on the Task Flows screen in the Web client.



➤ To execute a script from the Task Flows screen:

1 From within the Web client, select **Tools > Task Flows**.

2 Click the link of the script to execute.

The script runs immediately.

Asynchronous Script Execution

FDM can execute scripts in parallel using a command line interface object. Run any selected script from within an asynchronous script shell and the script will execute in parallel mode.

```
Sub AsynchScript()
'-------------------------------------------------------------------
'Oracle's Hyperion® Financial Data Quality Management Custom Script:
'
'Created By:      Admin
'Date Created:     3/23/2006 13:57
'
```

```
‘Purpose:        Asynchronous Script Processing
‘
‘-----------------------------------------------------------------
‘Declare Local Variables
Dim strScriptToRun
Dim strLoadBalanceServerName
‘Initialize variables Variables
strScriptToRun = “YourScriptName”  ‘Name of the script to execute (Type
Custom General Only)
strLoadBalanceServerName = “LocalHost”     ‘Load balance server for
Asynchronous process to use
‘Execute the script
If API.DataWindow.Utilities.fExecuteCustomScriptAsync(CStr
(strScriptToRun), CStr(strLoadBalanceServerName)) = True Then
    ‘Async Script Started, you can make another call while it is
executing
Else
    ‘Async process failed to start
End If
End Sub
-----------------------------------------------------------------
```

This functionality is also supported in the Task Manager which allows scripts to execute at the same time within a task or to allow individual tasks to behave asynchronously.

A

Common Visual Basic Scripting Expressions

In This Appendix

Overview

This appendix includes a list of common Visual Basic scripting functions. For a complete reference to VB scripting refer the following Microsoft web site:

http:\\www.msdn.microsoft.com\\vbscript

InStr

Returns the position of the first occurrence of one string within another string.

```
InStr([start, ]string1, string2[, compare])
```

Arguments

start

Numeric expression that sets the starting position for each search. If omitted, search begins at the first character position. If start contains `Null`, an error occurs. The start argument is required if compare is specified (optional).

string1

String expression being searched (required).

string2

String expression searched for (required).

compare

Numeric value indicating the kind of comparison to use when evaluating substrings. See the following section for values. If omitted, a binary comparison is performed (optional).

Settings

The compare argument can have the following values:

Constant	Value	Description
`vbBinaryCompare`	0	Perform a binary comparison
`vbTextCompare`	1	Perform a textual comparison

Return Values

The `InStr` function returns the following values:

If	InStr Returns
string1 is zero-length	0
string1 is Null	Null
string2 is zero-length	start
string2 is Null	Null
string2 is not found	0

If	InStr Returns
string2 is found within string1	Position at which match is found
start > Len(string2)	0

Remarks

The following examples use InStr to search a string:

```
Dim SearchString, SearchChar, MyPos
SearchString =”XXpXXpXXPXXP”  ‘ String to search in.
SearchChar = “P”  ‘ Search for “P”.
MyPos = InStr(4, SearchString, SearchChar, 1)  ‘ A textual comparison
starting at position 4. Returns 6.
MyPos = InStr(1, SearchString, SearchChar, 0)  ‘ A binary comparison
starting at position 1. Returns 9.
MyPos = InStr(SearchString, SearchCharb  ‘ Comparison is binary by default
(last argument is omitted). Returns 9.
MyPos = InStr(1, SearchString, “W”)  ‘ A binary comparison starting at
position 1. Returns 0 (“W” is not found).
```

Note:

The `InStrB` function is used with byte data contained in a string. Instead of returning the character position of the first occurrence of one string within another, `InStrB` returns the byte position.

InStrRev

Returns the position of an occurrence of one string within another, from the end of string.

```
InStrRev(string1, string2[, start[, compare]])
```

Arguments

string1

String expression being searched (required).

string2

String expression being searched for (required).

start

Numeric expression that sets the starting position for each search. If omitted, -1 is used, which means that the search begins at the last character position. If start contains `Null`, an error occurs (optional).

compare

Numeric value indicating the kind of comparison to use when evaluating substrings. If omitted, a binary comparison is performed. See the following section for values (optional).

Settings

The compare argument can have the following values:

Constant	Value	Description
`vbBinaryCompare`	0	Perform a binary comparison
`vbTextCompare`	1	Perform a textual comparison

Return Values

`InStrRev` returns the following values:

If	InStrRev returns
string1 is zero-length	0
string1 is `Null`	Null
string2 is zero-length	start
string2 is `Null`	Null
string2 is not found	0
string2 is found within string1	Position at which match is found
start >`Len(string2)`	0

Remarks

The following examples use the `InStrRev` function to search a string:

```
Dim SearchString, SearchChar, MyPos
SearchString =”XXpXXpXXPXXP”  ‘ String to search in.
SearchChar = “P”  ‘ Search for “P”.
MyPos = InStrRev(SearchString, SearchChar, 10, 0)  ‘ A binary comparison
starting at position 10. Returns 9.
```

```
MyPos = InStrRev(SearchString, SearchChar, -1, 1)  ‘ A textual comparison
starting at the last position. Returns 12.
MyPos = InStrRev(SearchString, SearchChar, 8)  ‘ Comparison is binary by
default (last argument is omitted). Returns 0.
```

Note:

The syntax for the `InStrRev` function is not the same as the syntax for the `InStr` function.

IsNumeric

Returns a Boolean value indicating whether an expression can be evaluated as a number.

```
IsNumeric(expression)
```

The expression argument can be any expression.

Remarks

`IsNumeric` returns True if the entire expression is recognized as a number; otherwise, it returns False. `IsNumeric` returns False if expression is a date.

The following example uses the `IsNumeric` function to determine whether a variable can be evaluated as a number:

```
Dim MyVar, MyCheck
MyVar = 53  ‘ Assign a value.
MyCheck = IsNumeric(MyVar)  ‘ Returns True.
MyVar = “459.95”  ‘ Assign a value.
MyCheck = IsNumeric(MyVar)  ‘ Returns True.
MyVar = “45 Help”  ‘ Assign a value.
MyCheck = IsNumeric(MyVar)  ‘ Returns False.
```

Len

Returns the number of characters in a string or the number of bytes required to store a variable.

```
Len(string | varname)
```

Arguments

string

Any valid string expression. If string contains `Null`, Null is returned.

varname

Any valid variable name. If varname contains `Null`, Null is returned.

Remarks

The following example uses the `Len` function to return the number of characters in a string:

```
Dim MyString
MyString = Len("VBSCRIPT") ' MyString contains 8.
```

Note:

The `LenB` function is used with byte data contained in a string. Instead of returning the number of characters in a string, `LenB` returns the number of bytes used to represent that string.

Left

Returns a specified number of characters from the left side of a string.

```
Left(string, length)
```

Arguments

string

String expression from which the left-most characters are returned. If string contains `Null`, Null is returned.

length

Numeric expression indicating how many characters to return. If 0, a zero-length string ("") is returned. If greater than or equal to the number of characters in string, the entire string is returned.

Remarks

To determine the number of characters in string, use the `Len` function.

The following example uses the `Left` function to return the first three characters of `MyString`:

```
Dim MyString, LeftString
MyString = "VBScript"
LeftString = Left(MyString, 3) ' LeftString contains "VBS".
```

Note:

The `LeftB` function is used with byte data contained in a string. Instead of specifying the number of characters to return, length specifies the number of bytes.

LCase

Returns a string that has been converted to lowercase.

LCase(string**)**

The string argument is any valid string expression. If string contains Null, Null is returned.

Remarks

Only uppercase letters are converted to lowercase; all lowercase letters and non-letter characters remain unchanged.

The following example uses the LCase function to convert uppercase letters to lowercase:

```
Dim MyString
Dim LCaseString
MyString = “VBScript”
LCaseString = LCase(MyString) ‘ LCaseString contains “vbscript”.
```

LTrim and RTrim

Returns a copy of a string without leading spaces (LTrim) , trailing spaces (RTrim), or both leading and trailing spaces (Trim).

```
LTrim(string)
RTrim(string)
Trim(string)
```

The string argument is any valid string expression. If string contains Null, Null is returned.

Remarks

The following example uses the LTrim, RTrim, and Trim functions to trim leading spaces, trailing spaces, and both leading and trailing spaces, respectively.

```
Dim MyVar
MyVar = LTrim(“  vbscript “)  ‘ MyVar contains “vbscript “
MyVar = RTrim(“  vbscript “)  ‘ MyVar contains “  vbscript”
MyVar = Trim(“  vbscript “)  ‘ MyVar contains “vbscript”
```

Mid

Returns a specified number of characters from a string.

Mid(string, start[, length])

Arguments

string

String expression from which characters are returned. If string contains `Null` ,Null is returned.

start

Character position in string at which the part to be taken begins. If start is greater than the number of characters in string, `Mid` returns a zero-length string (“”).

length

Number of characters to return. If omitted or if there are fewer than length characters in the text (including the character at start), all characters from the start position to the end of the string are returned.

Remarks

To determine the number of characters in string, use the `Len` function.

The following example uses the `Mid` function to return six characters, beginning with the fourth character, in a string:

```
Dim MyVar
MyVar = Mid(“VBScript is fun!”, 4, 6) ‘ MyVar contains “Script”.
```

Note:

The `MidB` function is used with byte data contained in a string. Instead of specifying the number of characters, the arguments specify numbers of bytes.

MsgBox

Displays a message in a dialog box, waits for the user to click a button, and returns a value indicating which button the user clicked.

```
MsgBox(prompt[, buttons][, title][, helpfile, context])
```

Note:

The MsgBox function does not work within the Web Client. Messages using the Web Client must be displayed in a new window. Use the Show Message accelerator provided in the Workbench Object Browser for this task.

Arguments

prompt

String expression displayed as the message in the dialog box. The maximum length of the prompt is approximately 1024 characters, depending on the width of the characters used. If the prompt consists of more than one line, they can be separated by inserting a carriage return character (`Chr(13)`), a linefeed character (`Chr(10)`), or carriage return–linefeed character combination (`Chr(13)`) and (`Chr(10)`) between each line.

buttons

Numeric expression that is the sum of values specifying the number and type of buttons to display, the icon style to use, the identity of the default button, and the modality of the message box. If omitted, the default value for buttons is 0.

title

String expression displayed in the title bar of the dialog box. If you omit title, the application name is placed in the title bar.

helpfile

String expression that identifies the Help file to use to provide context-sensitive Help for the dialog box. If helpfile is provided, context must also be provided. This is not available on 16-bit platforms.

context

Numeric expression that identifies the Help context number assigned by the Help author to the appropriate Help topic. If context is provided, helpfile must also be provided. This is not available on 16-bit platforms.

Remarks

When both helpfile and context are provided, the user can press F1 to view the Help topic corresponding to the context.

If the dialog box displays a Cancel button, pressing ESC has the same effect as clicking Cancel. If the dialog box contains a Help button, context-sensitive Help is provided for the dialog box. However, no value is returned until one of the other buttons is clicked.

When the MsgBox function is used with Microsoft Internet Explorer, the title of any dialog presented always contains “VBScript:” to differentiate it from standard system dialogs.

The following example uses the MsgBox function to display a message box and return a value describing which button was clicked:

```
Dim MyVar
```

```
MyVar = MsgBox (“Hello World!”, 65, “MsgBox Example”)
  ‘ MyVar contains either 1 or 2, depending on which button is clicked.
```

Now

Returns the current date and time according to the setting of your computer’s system date and time.

```
Now
```

Remarks

The following example uses the Now function to return the current date and time:

```
Dim MyVar
MyVar = Now ‘ MyVar contains the current date and time
```

Replace

Returns a string in which a specified substring has been replaced with another substring a specified number of times.

```
Replace(expression, find, replacewith[, start[, count[, compare]]])
```

Arguments

expression

String expression containing substring to replace (required).

find

Substring being searched for (required).

replacewith

Replacement substring (required).

start

Position within expression where substring search is to begin. If omitted, 1 is assumed. Must be used in conjunction with count (optional).

count

Number of substring substitutions to perform. If omitted, the default value is -1, which means make all possible substitutions. Must be used in conjunction with start (optional).

compare

Numeric value indicating the kind of comparison to use when evaluating substrings. See the following section for values. If omitted, the default value is 0, which means perform a binary comparison (optional).

Settings

The compare argument can have the following values:

Constant	Value	Description
`vbBinaryCompare`	0	Perform a binary comparison
`vbTextCompare`	1	Perform a textual comparison

Return Values

Replace returns the following values:

If	Replace Returns
expression is zero-length	Zero-length string (“”)
expression is `Null`	An error
find is zero-length	Copy of expression
`replacewith` is zero-length	Copy of expression with all occurrences of find removed
start > Len(expression)	Zero-length string
count is 0	Copy of expression

Remarks

The return value of the `Replace` function is a string, with substitutions made, that begins at the position specified by start and concludes at the end of the expression string. It is not a copy of the original string from start to finish.

The following example uses the `Replace` function to return a string:

```
Dim MyString
‘ A binary comparison starting at the beginning of the string.
```

```
MyString = Replace(“XXpXXPXXp”, “p”, “Y”)
‘ Returns “XXYXXPXXY”.
‘ A textual comparison starting at position 3.
MyString = Replace(“XXpXXPXXp”, “p”, “Y”, 3, -1, 1)
‘ Returns “YXXYXXY”.
```

Right

Returns a specified number of characters from the right side of a string.

```
Right(string, length)
```

Arguments

string

String expression from which the right-most characters are returned. If string contains `Null`,Null is returned.

length

Numeric expression indicating how many characters to return. If 0, a zero-length string is returned. If greater than or equal to the number of characters in string, the entire string is returned.

Remarks

To determine the number of characters in string, use the `Len` function.

The following example uses the `Right` function to return a specified number of characters from the right side of a string:

```
Dim AnyString, MyStr
AnyString = “Hello World”   ‘ Define string.
MyStr = Right(AnyString, 1)  ‘ Returns “d”.
MyStr = Right(AnyString, 6)  ‘ Returns “ World”.
MyStr = Right(AnyString, 20)  ‘ Returns “Hello World”.
```

Note:

The `RightB` function is used with byte data contained in a string. Instead of specifying the number of characters to return, length specifies the number of bytes.

UCase

Returns a string that has been converted to uppercase.

```
UCase(string)
```

The string argument is any valid string expression. If string contains `Null`,Null is returned.

Remarks

Only lowercase letters are converted to uppercase; all uppercase letters and non-letter characters remain unchanged.

The following example uses the `UCase` function to return an uppercase version of a string:

```
Dim MyWord
MyWord = UCase(“Hello World”)  ‘ Returns “HELLO WORLD”.
```

Do...Loop

Repeats a block of statements while a condition is True or until a condition becomes True.

```
Do [{While | Until} condition]
  [statements]
  [Exit Do]
  [statements]
Loop
```

Or, you can use this syntax:

```
Do
  [statements]
  [Exit Do]
  [statements]
Loop [{While | Until} condition]
```

Arguments

condition

Numeric or string expression that is True or False. If condition is `Null`, condition is treated as False.

statements

One or more statements that are repeated while or until condition is True.

Remarks

The `Exit Do` can only be used within a `Do...Loop` control structure to provide an alternate way to exit a `Do...Loop`. Any number of `Exit Do` statements may be placed anywhere in the

`Do...Loop`. Often used with the evaluation of some condition (for example, `If...Then`), `Exit Do` transfers control to the statement immediately following the `Loop`.

When used within nested `Do...Loop` statements, `Exit Do` transfers control to the loop that is nested one level above the loop where it occurs.

The following examples illustrate use of the `Do...Loop` statement:

```
Do Until DefResp = vbNo
  MyNum = Int(6 * Rnd + 1)  ‘ Generate a random integer between 1 and 6.
  DefResp = MsgBox (MyNum & “ Do you want another number?”, vbYesNo)
Loop
Dim Check, Counter
Check = True: Counter = 0  ‘ Initialize variables.
Do              ‘ Outer loop.
  Do While Counter < 20   ‘ Inner loop.
   Counter = Counter + 1  ‘ Increment Counter.
   If Counter = 10 Then  ‘ If condition is True...
     Check = False    ‘ set value of flag to False.
     Exit Do       ‘ Exit inner loop.
   End If
  Loop
Loop Until Check = False   ‘ Exit outer loop immediately.
```

If...Then...Else

Conditionally executes a group of statements, depending on the value of an expression.

```
If condition Then statements [Else elsestatements ]
```

Or, you can use the block form syntax:

```
If condition Then
  [statements]
[ElseIf condition-n Then
  [elseifstatements]] . . .
[Else
  [elsestatements]]
End If
```

Arguments

condition

One or more of the following two types of expressions:

A numeric or string expression that evaluates to True or False. If condition is `Null`, condition is treated as False.

An expression of the form `TypeOf` objectname Is `objecttype`. The objectname is any object reference and objecttype is any valid object type. The expression is True if `objectname` is of the object type specified by `objecttype`; otherwise it is False.

statements

One or more statements separated by colons; executed if condition is True.

condition-n

Same as condition.

elseifstatements

One or more statements executed if the associated condition-n is True.

elsestatements

One or more statements executed if no previous condition or condition-n expression is True.

Remarks

You can use the single-line form (first syntax) for short, simple tests. However, the block form (second syntax) provides more structure and flexibility than the single-line form and is usually easier to read, maintain, and debug.

Note:

With the single-line syntax, it is possible to have multiple statements executed as the result of an If...Then decision, but they must all be on the same line and separated by colons, as in the following statement:

```
If A > 10 Then A = A + 1 : B = B + A : C = C + B
```

When executing a block `If` (second syntax), condition is tested. If condition is True, the statements following `Then` are executed. If condition is False, each `ElseIf` (if any) is evaluated in turn. When a True condition is found, the statements following the associated `Then` are executed. If none of the `ElseIf` statements are True (or there are no `ElseIf` clauses), the statements following `Else` are executed. After executing the statements following `Then` or `Else`, execution continues with the statement following `End If`.

The `Else` and `ElseIf` clauses are both optional. You can have as many `ElseIf` statements as you want in a block `If`, but none can appear after the `Else` clause. Block `If` statements can be nested.

What follows the `Then` keyword is examined to determine whether or not a statement is a block `If`. If anything other than a comment appears after `Then` on the same line, the statement is treated as a single-line `If` statement.

A block `If` statement must be the first statement on a line. The block `If` must end with an `End If` statement.

File System Object

The FileSystemObject (FSO) object model allows you to use the familiar object.method syntax with a rich set of properties, methods, and events to process folders and files.

Creating Files

There are three ways to create an empty text file (sometimes referred to as a “text stream”).

The first way is to use the `CreateTextFile` method. The following example demonstrates how to create a text file by using the `CreateTextFileMethod` method.

```
[VBScript]
Dim fso, f1
Set fso = CreateObject(“Scripting.FileSystemObject”)
Set f1 = fso.CreateTextFile(“c:\testfile.txt”, True)
```

The second way to create a text file is to use the `OpenTextFile` method of the FileSystemObject object with the `ForWriting` flag set.

```
[VBScript]
Dim fso, ts
Const ForWriting = 2
Set fso = CreateObject(“Scripting.FileSystemObject”)
Set ts = fso.OpenTextFile(“c:\test.txt”, ForWriting, True)
```

A third way to create a text file is to use the `OpenAsTextStream` method with the `ForWriting` flag set.

```
[VBScript]
Dim fso, f1, ts
Const ForWriting = 2
Set fso = CreateObject(“Scripting.FileSystemObject”)
fso.CreateTextFile (“c:\test1.txt”)
Set f1 = fso.GetFile(“c:\test1.txt”)
Set ts = f1.OpenAsTextStream(ForWriting, True)
```

Adding Data to the File

Once the text file is created, add data to the file by following three steps:

1. Open the text file
2. Write the data
3. Close the file

To open an existing file, use either the `OpenTextFile` method of the `FileSystemObject` object or the `OpenAsTextStream` method of the file object.

To write data to the open text file, use the `Write`, `WriteLine`, or `WriteBlankLines` methods of the TextStream object, according to the tasks outlined in the following table.

Task	Method
Write data to an open text file without a trailing newline character.	`Write`
Write data to an open text file with a trailing newline character.	`WriteLine`
Write one or more blank lines to an open text file.	`WriteBlankLines`

To close an open file, use the `Close` method of the TextStream object.

Note:

The newline character contains a character or characters (depending on the operating system) to advance the cursor to the beginning of the next line (carriage return/line feed). Be aware that the end of some strings may already have such non-printing characters.

The following example demonstrates how to open a file, use all three write methods to add data to the file, and then close the file:

```
[VBScript]
Sub CreateFile()
  Dim fso, tf
  Set fso = CreateObject("Scripting.FileSystemObject")
  Set tf = fso.CreateTextFile("c:\testfile.txt", True)
  ' Write a line with a newline character.
  tf.WriteLine("Testing 1, 2, 3.")
  ' Write three newline characters to the file.
  tf.WriteBlankLines(3)
  ' Write a line.
  tf.Write ("This is a test.")
  tf.Close
End Sub
```

Reading Files

To read data from a text file, use the `Read`, `ReadLine`, or `ReadAll` method of the TextStream object. The following table describes which method to use for various tasks.

Task	Method
Read a specified number of characters from a file	`Read`
Read an entire line (up to, but not including, the newline character)	`ReadLine`
Read the entire contents of a text file	`ReadAll`

If you use the `Read` or `ReadLine` method and want to skip to a particular portion of data, use the `Skip` or `SkipLine` method. The resulting text of the read methods is stored in a string which can be displayed in a control, parsed by string functions (such as `Left`, `Right`, and `Mid`), concatenated, and so forth.

The following example demonstrates how to open a file, write to it, and then read from it:

```
[VBScript]
Sub ReadFiles
  Dim fso, f1, ts, s
  Const ForReading = 1
  Set fso = CreateObject("Scripting.FileSystemObject")
  Set f1 = fso.CreateTextFile("c:\testfile.txt", True)
  ' Write a line.
  Response.Write "Writing file <br>"
  f1.WriteLine "Hello World"
  f1.WriteBlankLines(1)
  f1.Close
  ' Read the contents of the file.
  Response.Write "Reading file <br>"
  Set ts = fso.OpenTextFile("c:\testfile.txt", ForReading)
  s = ts.ReadLine
  Response.Write "File contents = '" & s & "'"
  ts.Close
End Sub
```

Moving, Copying, and Deleting Files

The FSO object model has two methods each for moving, copying, and deleting files, as described in the following table.

Task	Method
Move a file	`File.Move or FileSystemObject.MoveFile`
Copy a file	`File.Copy or FileSystemObject.CopyFile`
Delete a file	`File.Delete or FileSystemObject.DeleteFile`

The following example creates a text file in the root directory of drive C, writes some information to it, moves it to a directory named `\tmp`, makes a copy of it in a directory named `\temp`, then deletes the copies from both directories.

To run the following example, create directories named `\tmp` and `\temp` in the root directory of drive C:

```
[VBScript]
Sub ManipFiles
  Dim fso, f1, f2, s
  Set fso = CreateObject("Scripting.FileSystemObject")
  Set f1 = fso.CreateTextFile("c:\testfile.txt", True)
  Response.Write "Writing file <br>"
  ' Write a line.
  f1.Write ("This is a test.")
  ' Close the file to writing.
  f1.Close
  Response.Write "Moving file to c:\tmp <br>"
  ' Get a handle to the file in root of C:\.
  Set f2 = fso.GetFile("c:\testfile.txt")
```

```
  ‘ Move the file to \tmp directory.
  f2.Move (“c:\tmp\testfile.txt”)
  Response.Write “Copying file to c:\temp <br>”
  ‘ Copy the file to \temp.
  f2.Copy (“c:\temp\testfile.txt”)
  Response.Write “Deleting files <br>”
  ‘ Get handles to files’ current location.
  Set f2 = fso.GetFile(“c:\tmp\testfile.txt”)
  Set f3 = fso.GetFile(“c:\temp\testfile.txt”)
  ‘ Delete the files.
  f2.Delete
  f3.Delete
  Response.Write “All done!”
End Sub
```