

---

---

*RELEASE 9.3.1.1 USER'S GUIDE*

---

ORACLE® DATA INTEGRATOR ADAPTER  
FOR HYPERION FINANCIAL  
MANAGEMENT



The Programs (which include both the software and documentation) contain proprietary information; they are provided under a license agreement containing restrictions on use and disclosure and are also protected by copyright, patent, and other intellectual and industrial property laws. Reverse engineering, disassembly, or decompilation of the Programs, except to the extent required to obtain interoperability with other independently created software or as specified by law, is prohibited.

The information contained in this document is subject to change without notice. If you find any problems in the documentation, please report them to us in writing. This document is not warranted to be error-free. Except as may be expressly permitted in your license agreement for these Programs, no part of these Programs may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose.

If the Programs are delivered to the United States Government or anyone licensing or using the Programs on behalf of the United States Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the Programs, including documentation and technical data, shall be subject to the licensing restrictions set forth in the applicable Oracle license agreement, and, to the extent applicable, the additional rights set forth in FAR 52.227-19, Commercial Computer Software--Restricted Rights (June 1987). Oracle USA, Inc., 500 Oracle Parkway, Redwood City, CA 94065.

The Programs are not intended for use in any nuclear, aviation, mass transit, medical, or other inherently dangerous applications. It shall be the licensee's responsibility to take all appropriate fail-safe, backup, redundancy and other measures to ensure the safe use of such applications if the Programs are used for such purposes, and we disclaim liability for any damages caused by such use of the Programs.

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

The Programs may provide links to Web sites and access to content, products, and services from third parties. Oracle is not responsible for the availability of, or any content provided on, third-party Web sites. You bear all risks associated with the use of such content. If you choose to purchase any products or services from a third party, the relationship is directly between you and the third party. Oracle is not responsible for: (a) the quality of third-party products or services; or (b) fulfilling any of the terms of the agreement with the third party, including delivery of products or services and warranty obligations related to purchased products or services. Oracle is not responsible for any loss or damage of any sort that you may incur from dealing with any third party.

---

# Contents

---

<b>Chapter 1. Introduction to Oracle Data Integrator Adapter for Hyperion Financial Management</b> . . . . .	5
Purpose . . . . .	5
Integration Process . . . . .	5
Working with the Adapter . . . . .	6
<b>Chapter 2. Setting Up Environments</b> . . . . .	9
Defining Data Servers . . . . .	9
Defining Physical and Logical Schemas and a Context . . . . .	11
<b>Chapter 3. Reverse-Engineering Financial Management Applications</b> . . . . .	13
What Reverse-Engineering Does . . . . .	13
Using the Hyperion Financial Management RKM . . . . .	13
<b>Chapter 4. Loading and Extracting Data and Metadata</b> . . . . .	19
Data Integration Tasks . . . . .	19
Creating Interfaces . . . . .	19
Loading Metadata . . . . .	21
IKM SQL to Hyperion Financial Management Dimension . . . . .	21
Loading Data . . . . .	22
IKM SQL to Hyperion Financial Management Data . . . . .	22
Extracting Data . . . . .	24
LKM Hyperion Financial Management Data to SQL . . . . .	25
Extracting Members from Member Lists . . . . .	26
LKM Hyperion Financial Management Members to SQL . . . . .	26
Data Store Tables . . . . .	27
HFMDData . . . . .	27
HFMDData_MultiplePeriods . . . . .	28
Account . . . . .	29
Entity . . . . .	30
Scenario . . . . .	31
Currency . . . . .	32
Custom1-4 . . . . .	32

EnumMembersList ..... 33

# 1

# Introduction to Oracle Data Integrator Adapter for Hyperion Financial Management

## In This Chapter

Purpose .....	5
Integration Process .....	5
Working with the Adapter .....	6

## Purpose

Oracle® Data Integrator Adapter for Hyperion Financial Management enables you to connect and integrate Hyperion® System™ 9 Financial Management™ with any database through Oracle Data Integrator. The adapter provides a set of Oracle Data Integrator Knowledge Modules (KMs) for loading and extracting metadata and data and consolidating data in Financial Management applications.

## Integration Process

### Note:

For instructions on installing Oracle Data Integrator Adapter for Hyperion Financial Management, see the *Oracle Data Integrator Adapter for Hyperion Financial Management 9.3.1.1 Readme*, which is delivered with the adapter. You can also download the Readme document from [Oracle E-Delivery](#).

You can use Oracle Data Integrator Adapter for Hyperion Financial Management to perform these data integration tasks on a Financial Management application:

- Load metadata and data
- Extract data
- Consolidate data
- Enumerate members of member lists

Using the adapter to load or extract data involves these tasks:

- Setting up an environment: Importing the Hyperion Financial Management technology and defining data servers and schemas

- See [Chapter 2, “Setting Up Environments.”](#)
- Reverse-engineering a Financial Management application using the Reverse-engineering Knowledge Module (RKM)
  - See [Chapter 3, “Reverse-Engineering Financial Management Applications.”](#)
- Loading metadata and data using Integration Knowledge Modules (IKM)
  - See [Chapter 4, “Loading and Extracting Data and Metadata.”](#)
- Extracting data and members using Load Knowledge Modules (LKM)
  - See [Chapter 4, “Loading and Extracting Data and Metadata.”](#)

## Working with the Adapter

Using Oracle Data Integrator Adapter for Hyperion Financial Management involves these Oracle Data Integrator features:

- Topology Manager—For defining connections to Financial Management applications
  - See [Chapter 2, “Setting Up Environments.”](#)
- Designer—For these tasks:
  - Loading metadata and data into data stores, which are target tables that represent Financial Management dimensions and data tables
  - Extracting data and member lists from data stores, which are source tables that represent Financial Management data tables and member-list tables

Oracle Data Integrator Adapter for Hyperion Financial Management includes the Hyperion Financial Management RKM, which creates the data stores described in these topics:

- [“HFMDData” on page 27](#)
- [“HFMDData\\_MultiplePeriods” on page 28](#)
- [“Account” on page 29](#)
- [“Entity” on page 30](#)
- [“Scenario” on page 31](#)
- [“Currency” on page 32](#)
- [“Custom1–4” on page 32](#)
- [“EnumMembersList” on page 33](#)

In Designer, you use the Hyperion Financial Management RKM to create the data stores.

The adapter includes these other knowledge modules (KM) for loading and extracting data:

- IKM SQL to Hyperion Financial Management Dimension—Loads metadata to an application from the staging area. See [“IKM SQL to Hyperion Financial Management Dimension” on page 21](#) and [“Loading Metadata” on page 21](#).

- [IKM SQL to Hyperion Financial Management Data](#)—Loads data to an application from the staging area. See [“IKM SQL to Hyperion Financial Management Data”](#) on page 22 and [“Loading Data”](#) on page 22.
- [LKM Hyperion Financial Management Data to SQL](#)—Extracts data from an application to a staging area. See [“LKM Hyperion Financial Management Data to SQL”](#) on page 25 and [“Extracting Data”](#) on page 24.
- [LKM Hyperion Financial Management Members to SQL](#)—Extracts members of a member list in the application to a staging area. See [“LKM Hyperion Financial Management Members to SQL”](#) on page 26 and [“Extracting Members from Member Lists”](#) on page 26.





# 2

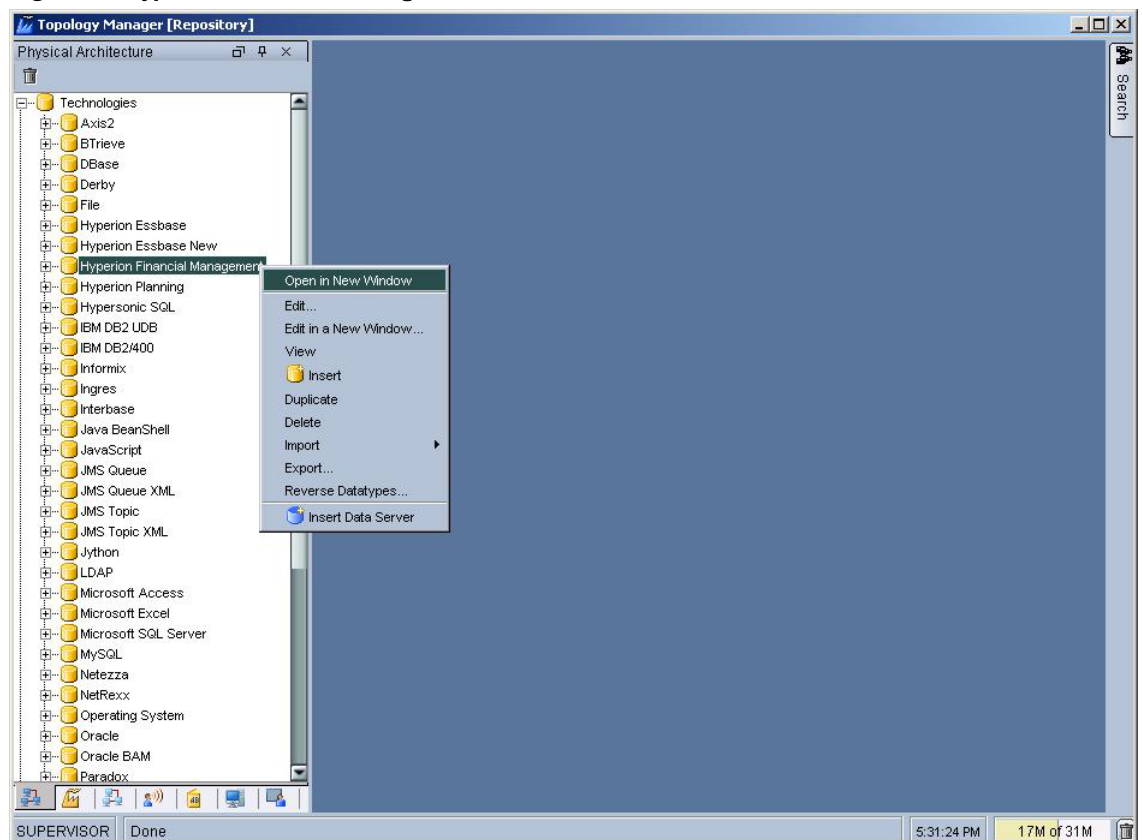
## Setting Up Environments

### In This Chapter

Defining Data Servers .....	9
Defining Physical and Logical Schemas and a Context .....	11

## Defining Data Servers

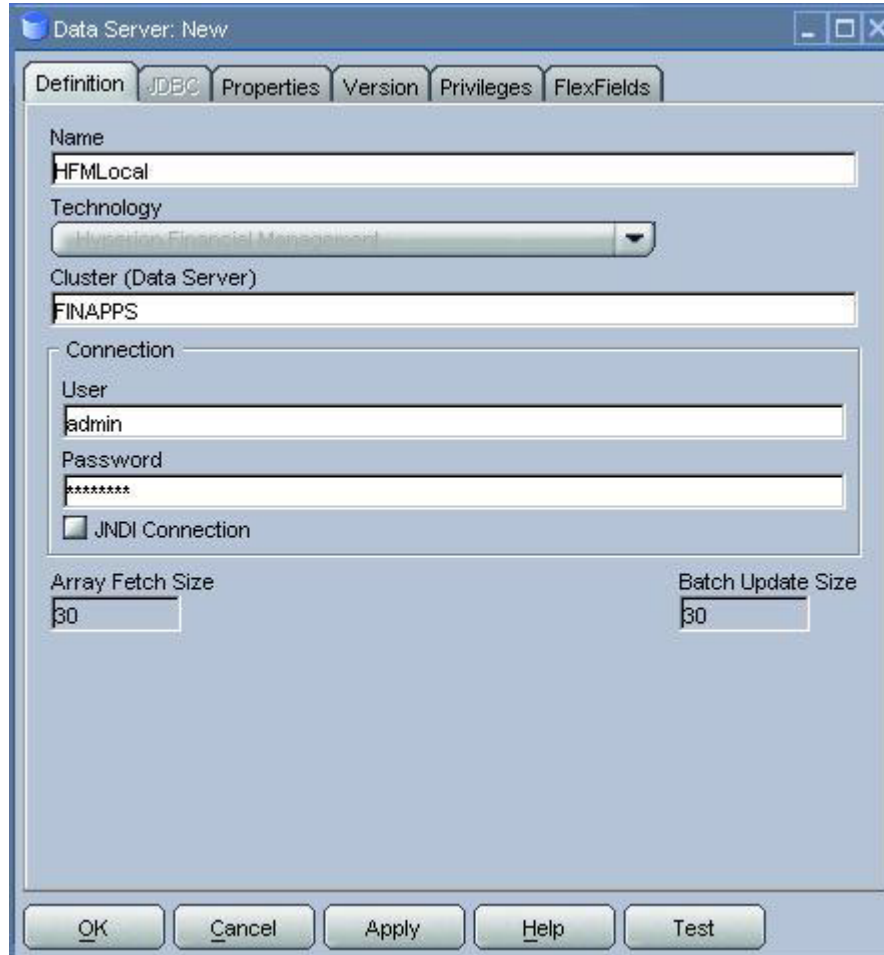
- ▶ To define a data server for connecting to a Financial Management server:
  - 1 In **Topology Manager**, expand **Technologies**.
  - 2 Right-click **Hyperion Financial Management**, and select **Insert Data Server**.



**Note:**

If the Hyperion Financial Management technology is not defined in your master repository, you can import it from the ImpExp folder.

The Data Server window opens.



**3 On the Definition tab:**

- a. Under **Name**, enter a name for the data server definition.
- b. Under **Cluster (Data Server)**, enter the Financial Management cluster name.
- c. Under **Connection**, enter a user name and password for connecting to the Financial Management server.
- d. Click **OK**.

**Note:**

The Test button does not work for a Hyperion Financial Management data server connection; it works only for relational technologies that have a JDBC driver.

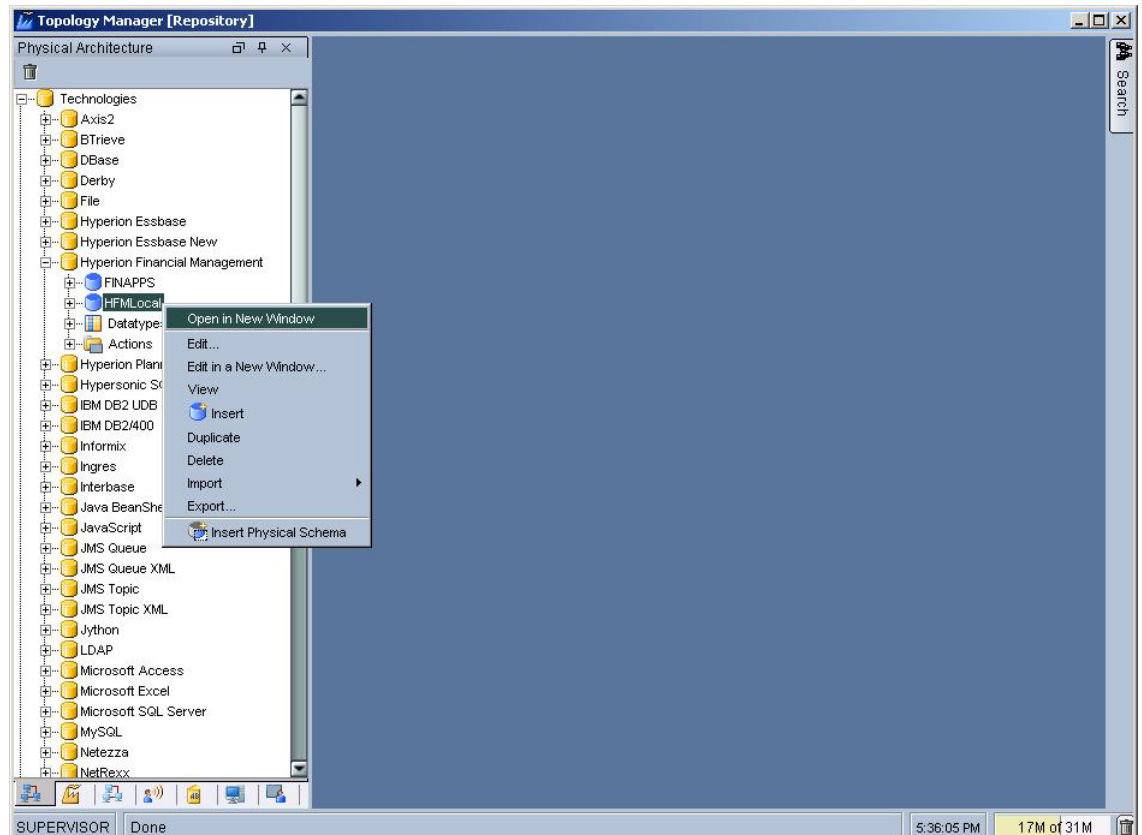
The Topology Manager window opens. See [“Defining Physical and Logical Schemas and a Context” on page 11](#).

# Defining Physical and Logical Schemas and a Context

Under a data server, you define a physical schema corresponding to an application and the logical schemas on which models are based. You work with Oracle Data Integrator and Adapter for Hyperion Financial Management through a logical schema. A context is used to link the logical schemas and the physical schemas.

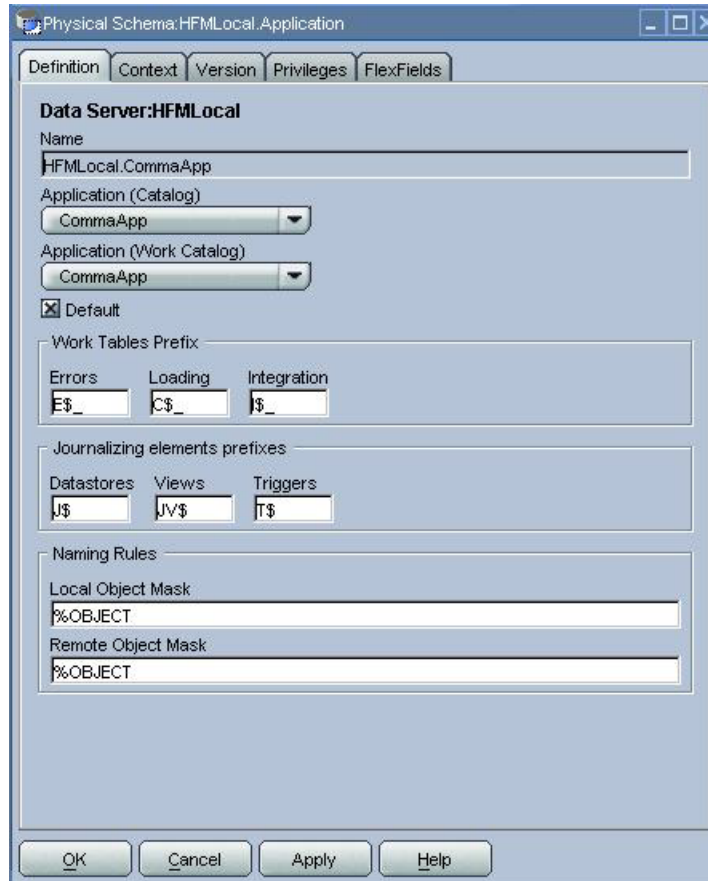
► To create a physical schema:

- 1 In Topology Manager, right-click the Hyperion Financial Management data server, and select **Insert Physical Schema**.



The Physical Schema window opens.

Figure 1 Physical Schema Definition




- 2 In **Physical Schema** (Figure 1), on the **Definition** tab, under **Application (Catalog)**, specify a Financial Management application.

In Figure 1, the specified application is Comma.

- To specify a logical schema and a context for a new physical schema:

- 1 On the **Context** tab:

- If one or more Hyperion Financial Management logical schemas exist, select a context and a logical schema.
- If no Hyperion Financial Management logical schemas exist:
  - a. Click .
  - b. Select a context from the left column.
  - c. Enter a name for a logical schema in the right column.

- 2 Click **OK**.

The logical schema that you selected or created is associated with the physical schema in the selected context.

See the *Oracle Data Integrator User's Guide* for more information about physical schemas, logical schemas, and contexts.

# 3

## Reverse-Engineering Financial Management Applications

### In This Chapter

What Reverse-Engineering Does.....	13
Using the Hyperion Financial Management RKM .....	13

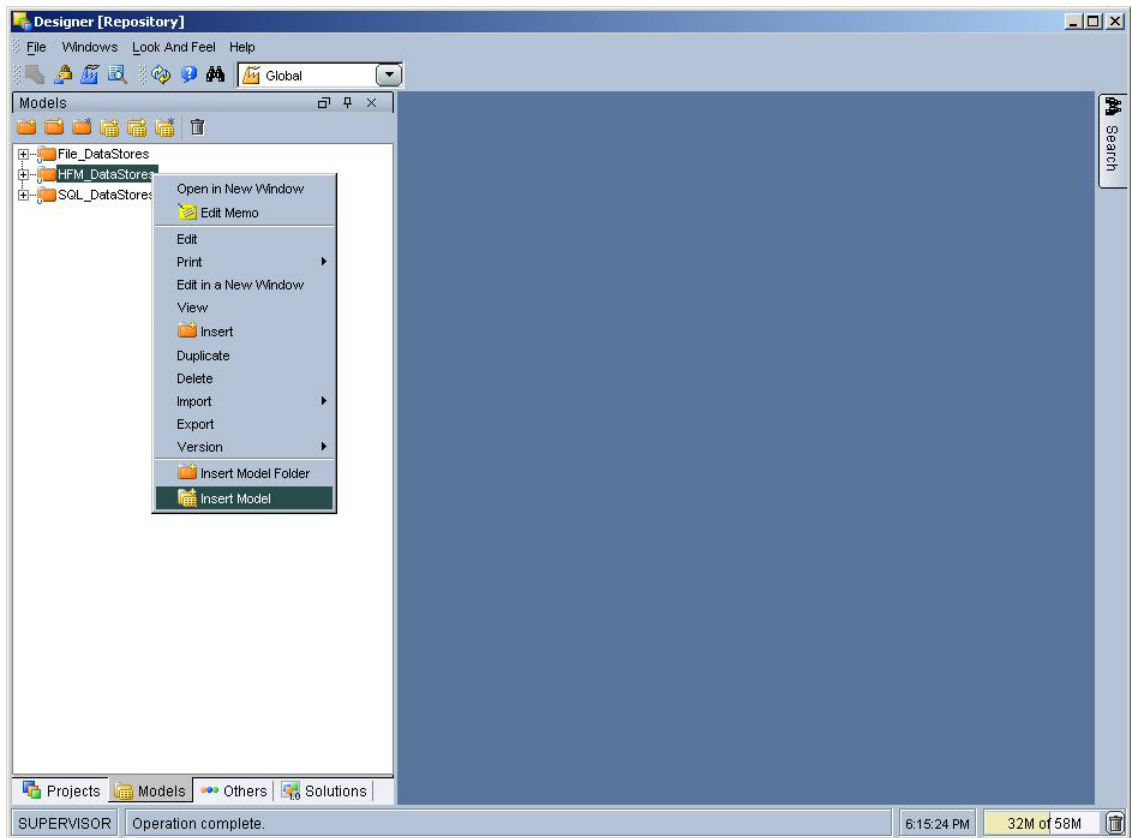
### What Reverse-Engineering Does

Reverse-engineering a Financial Management application creates an Oracle Data Integrator model that includes a data store for each dimension in the application, a data store for data, an optional data store for data with multiple periods, and an EnumMemberList data store. For more information about reverse-engineering, models, and data stores, see the *Oracle Data Integrator User's Guide*.

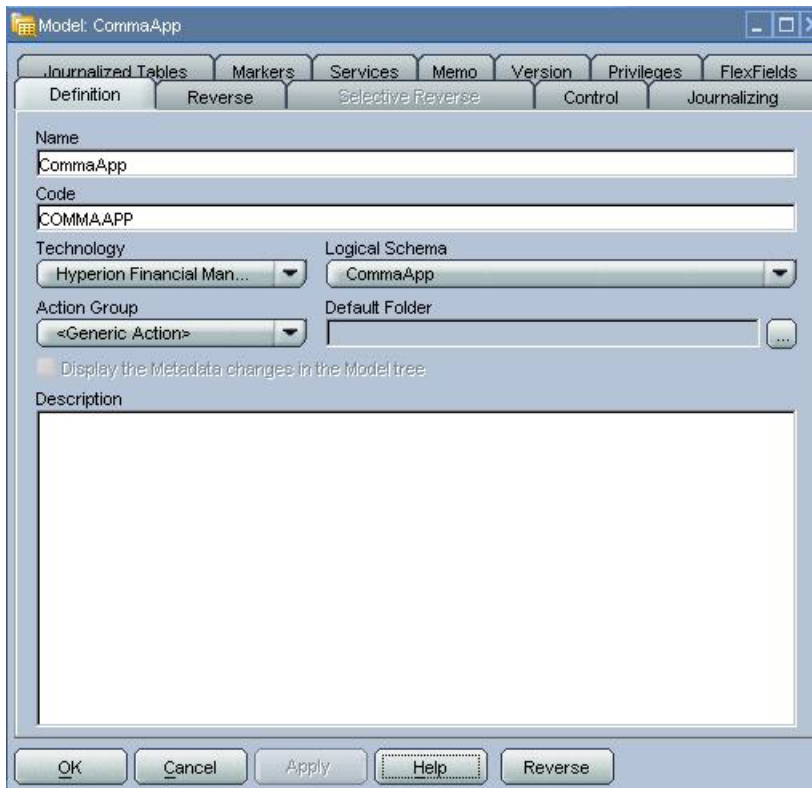
### Using the Hyperion Financial Management RKM

Use Oracle Data Integrator Designer to reverse-engineer applications. For more information about Designer, see the *Oracle Data Integrator User's Guide*.

- To reverse-engineer a Financial Management application:
  - 1 In **Designer**, in the left pane, select **Models > Insert Model**.

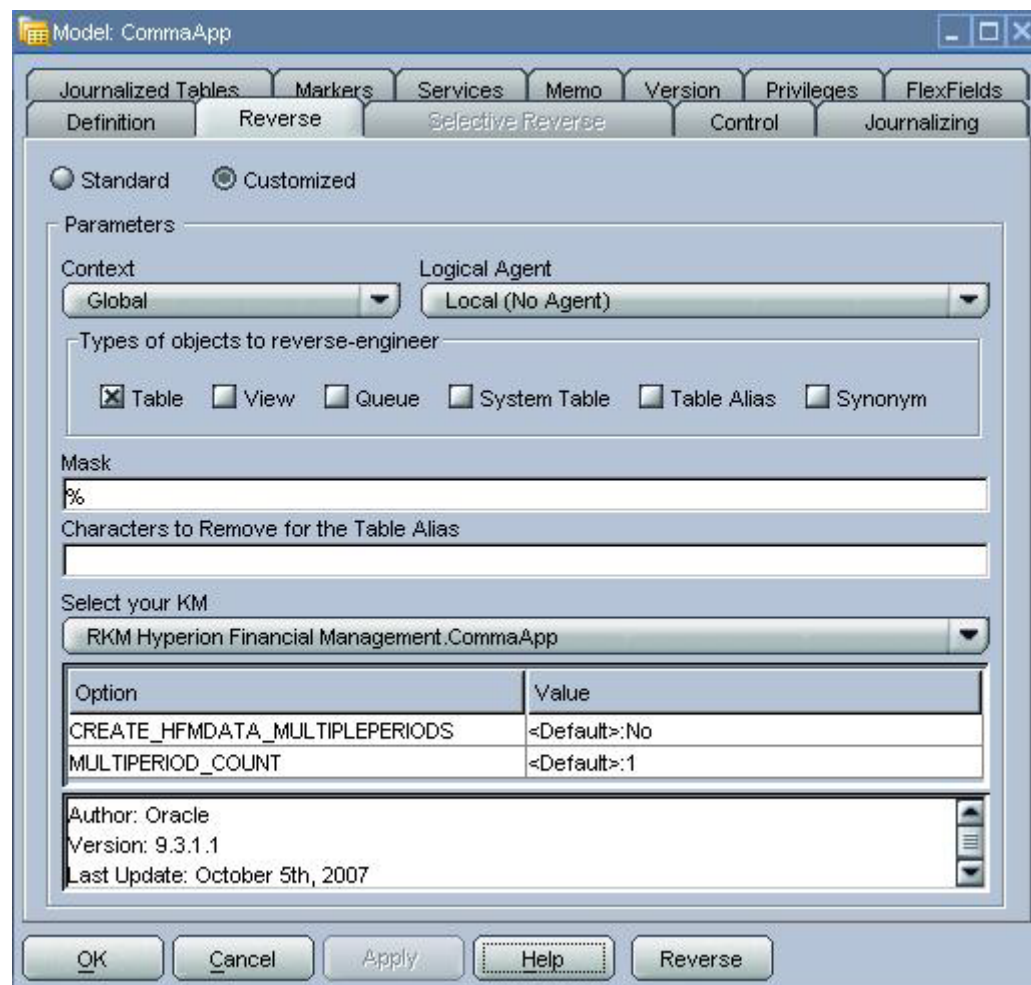


The Model window opens.



- 2 On the **Definition** tab, enter a name for the new model, select the **Hyperion Financial Management** technology, and select a logical schema on which to base the model.
- 3 On the **Reverse** tab (Figure 2):
  - a. Select **Customized**.
  - b. In **Context**, make a selection.
  - c. In **Select your KM**, select **RKM for Hyperion Financial Management**.
  - d. In **Option**, select RKM options:
    - Create HFMDATA\_MULTIPLEPERIODS—Valid values: Yes or No  
If set to Yes, an additional data store is created for data with multiple periods, and the number of periods for that model is specified by the MULTIPERIOD\_COUNT option. Default: No.
    - MULTIPERIOD\_COUNT—Number of periods for the HFMDATA\_MULTIPLEPERIODS data store.
  - e. Click **Reverse**.

Figure 2 Model Window Reverse Tab



#### 4 Click **Yes** when prompted to confirm your entries.

The RKM connects to the application (which is determined by the logical schema and the context) and imports some or all of these data stores, according to the dimensions in the application:

- HFMDData—For loading and extracting data
- HFMDData\_MultiplePeriods—For data with the number of periods specified by the option MULTIPERIODS\_COUNT

**Note:**

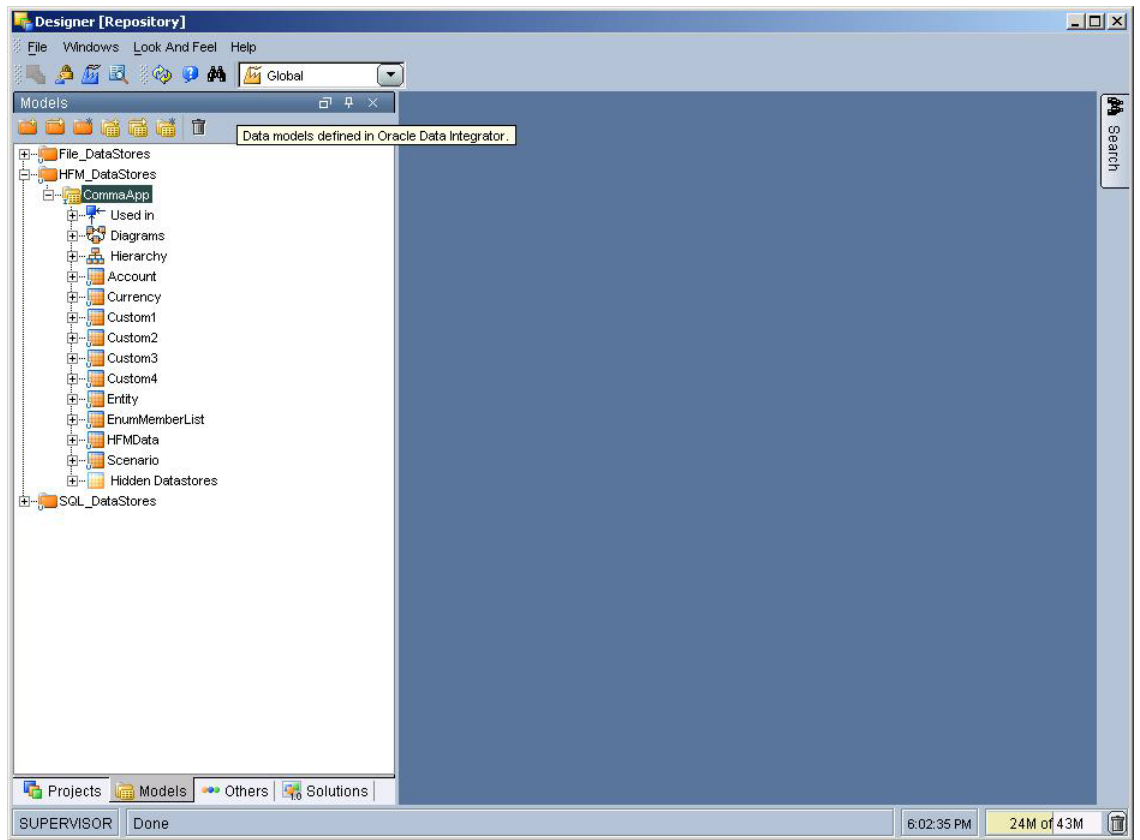
This data store is imported only if the CREATE\_HFMDATA\_MULTIPLEPERIODS option is set to Yes in the model definition.

- Account—For loading the Account dimension  
See [“Account” on page 29](#).
- Entity—For loading the Entity dimension  
See [“Entity” on page 30](#).
- Scenario—For loading the Scenario dimension  
See [“Scenario” on page 31](#).
- Currency—For loading the currency dimension  
See [“Currency” on page 32](#).
- Custom1–4— For loading the Custom1–4 dimensions  
See [“Custom1–4” on page 32](#).
- EnumMembersList—For extracting a members list  
See [“EnumMembersList” on page 33](#).

Any errors that occur in the reverse-engineering are listed in the Oracle Data Integrator Operator. For information about Operator, see the *Oracle Data Integrator User's Guide*.

The following figure shows a model after a successful reverse-engineering.







# 4

## Loading and Extracting Data and Metadata

### In This Chapter

Data Integration Tasks.....	19
Creating Interfaces.....	19
Loading Metadata.....	21
IKM SQL to Hyperion Financial Management Dimension .....	21
Loading Data .....	22
IKM SQL to Hyperion Financial Management Data .....	22
Extracting Data .....	24
LKM Hyperion Financial Management Data to SQL.....	25
Extracting Members from Member Lists.....	26
LKM Hyperion Financial Management Members to SQL .....	26
Data Store Tables .....	27

## Data Integration Tasks

In Oracle Data Integrator, loading or extracting Financial Management application metadata or data involves these tasks:

- Creating interfaces for loading and extracting data and metadata
- **(Optional)** Chaining interfaces into packages so that you can run the interfaces in one process
- Using interfaces

See “[Creating Interfaces](#)” on page 19.

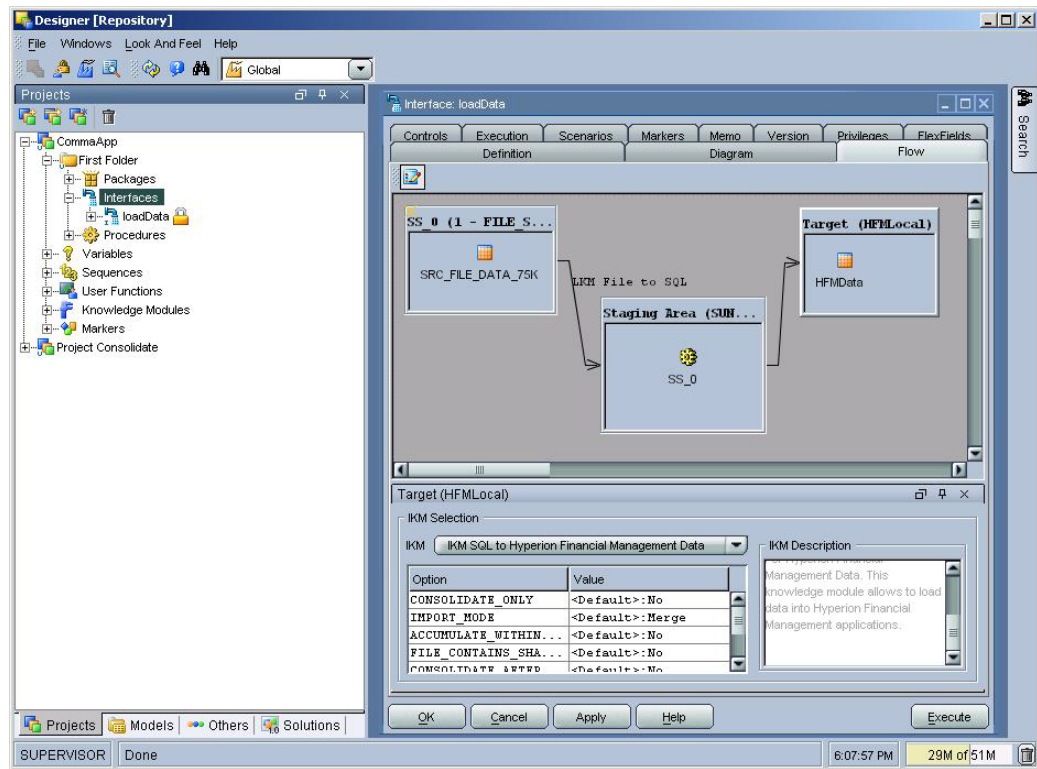
For instructions on creating interfaces and chaining them into packages, see the *Oracle Data Integrator User's Guide*.

## Creating Interfaces

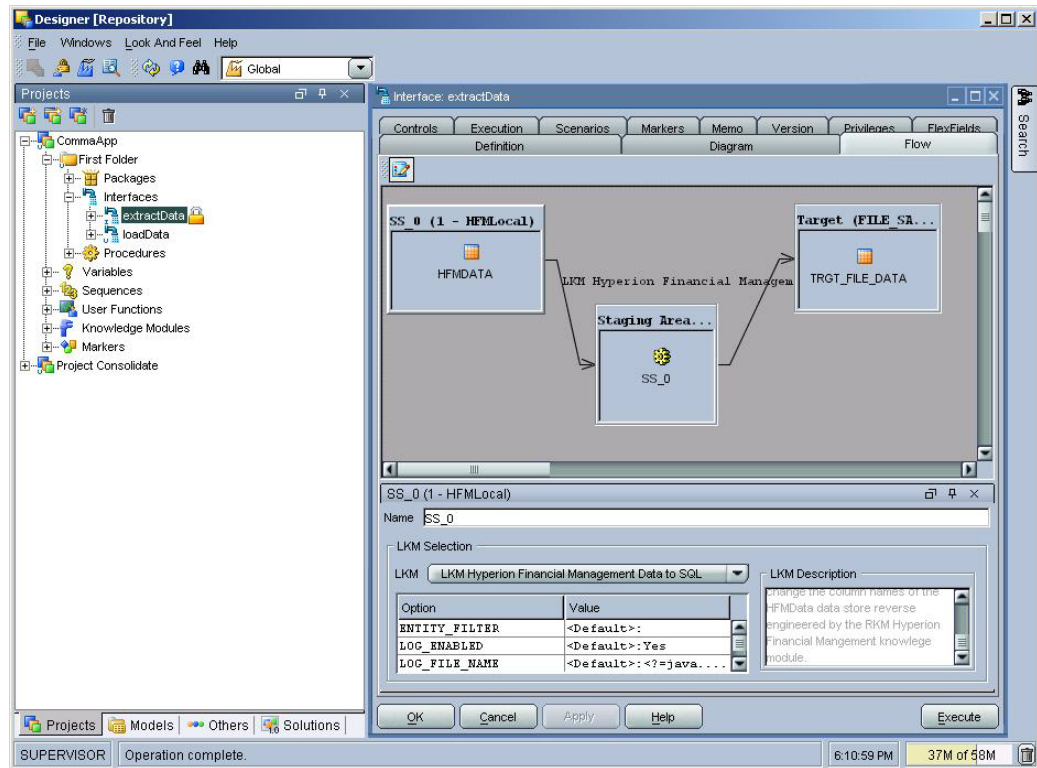
After reverse-engineering a Financial Management application as a model, you can use the data stores in this model in these ways:

- As targets of interfaces for loading data and metadata into the application

The following figure shows the flow of an interface targeting Financial Management.



- As sources of interfaces for extracting data and member lists from the application
- The following figure shows the flow of an interface with a Financial Management source.



## Loading Metadata

Metadata comprises dimension members. You must load members, or metadata, before you load data values for the members.

You can load members only to existing Financial Management dimensions. You must use a separate interface for each dimension that you load. You can chain interfaces to load metadata into several dimensions at once.

► To load metadata into a Financial Management application:

**1 Create an interface for loading metadata.**

You can give the interface any name. See the *Oracle Data Integrator Users Guide*.

**2 On the diagram, drag the target dimension data store from a Hyperion Financial Management model**

**3 Define the source data stores.**

**4 Define the mapping between source and target data.**

**5 On the **Flow** tab:**

- a. Ensure that **IKM SQL to Hyperion Financial Management Dimension** is selected.
- b. Specify load options. See [“IKM SQL to Hyperion Financial Management Dimension” on page 21](#).

**6 Click **Execute**.**

The metadata is loaded into the application.

**7 Check the Operator to verify that the interface ran successfully.**

## IKM SQL to Hyperion Financial Management Dimension

IKM SQL to Hyperion Financial Management Dimension supports these options for defining how the adapter loads metadata into a Financial Management application:

- **REPLACE\_MODE**

Valid values: Yes or No (default)

If set to Yes, metadata is replaced in the application (Replace); if set to No, metadata is overwritten in the application (Merge).

- **CLEAR\_ALL\_METADATA\_BEFORE\_LOADING**

Valid values: Yes or No (default)

If set to Yes, all metadata is cleared before loading.

---

**Caution!**

If you set this option to Yes, you lose any active data or journals in the application.

---

- **LOG\_ENABLED**

Valid values: Yes or No (default)

If set to Yes, logging is done during the load process to the file specified by the LOG\_FILE\_NAME option.

- LOG\_FILE\_NAME

The name of the file where logs are saved; default: *Java temp folder/dimension.log*

## Loading Data

You can load data into selected dimension members that are already created in Financial Management. You must set up the Financial Management application before you can load data into it.

Before loading data, ensure that the members (metadata) exist in the Financial Management relational database. A data load fails if the members do not exist.

► To load data into a Financial Management application:

**1 Create an interface for loading data.**

You can give the interface any name. See the *Oracle Data Integrator Users Guide*.

**2 On the diagram, drag and drop the target data store (HFMDData or HFMDData\_MultiplePeriods) from a Hyperion Financial Management model.**

**3 Define the source data stores.**

**4 Define the mapping between source and target data.**

**5 On the Flow tab:**

- a. Ensure that **IKM SQL to Hyperion Financial Management Data** is selected.
- b. Specify load options. See [“IKM SQL to Hyperion Financial Management Data”](#) on page 22.

**6 Click Execute.**

The data is loaded into the application.

**7 Check the Operator to verify that the interface ran successfully.**

## IKM SQL to Hyperion Financial Management Data

IKM SQL to Hyperion Financial Management Data supports these options for defining how the adapter loads and consolidates data in a Financial Management application:

- IMPORT\_MODE

Determines how data in the application cells is handled during data load. Valid values:

- Merge (default)—For each unique point of view that exists in the load data and in the application, the load data overwrites the data in the application. For each unique point

of view that is in the load data but not in the application, the load data is loaded into the application.

- Replace—For each unique point of view in the load data, the system clears corresponding values from the application, and then the data is loaded.

**Note:**

Unless the connected user has full access rights to all specified cells, no data is changed.

- Replace by Security—For each unique point of view in the load data to which the user has full access rights, the system clears corresponding values from the application, and then the data is loaded. Cells to which the user lacks full access are ignored.
- Accumulate—For each unique point of view that exists in the load data and in the application, the value from the load data is added to the value in the application.

- ACCUMULATE\_WITHIN\_FILE

Valid values: Yes or No (default)

If set to Yes, multiple values for the same cells in the load data are added before they are loaded into the application.

- FILE\_CONTAINS\_SHARE\_DATA

Valid values: Yes or No (default)

Set to Yes if the load file contains ownership data, such as shares owned.

---

**Caution!**

If ownership data is included in the file and this option is set to No, an error occurs when you load the file.

---

- CONSOLIDATE\_AFTER\_LOAD

Valid values: Yes or No (default)

If set to Yes, data is consolidated after being loaded.

- CONSOLIDATE\_ONLY

Valid values: Yes and No

If set to Yes, data is consolidated but not loaded.

- CONSOLIDATE\_PARAMETERS—Specifies the parameters for consolidation as comma-separated values in this order: Scenario (required), Year, Period, Parent.Entity, and Type; default: an empty string.

Valid Type parameter settings:

- “I” = Consolidate
- “D” = Consolidate All with Data
- “A” = Consolidate All
- “C” = Calculate Contribution

- “F” = Force Calculate Contribution

Example: Actual,1999,2,EastRegion.EastSales,A

- LOG\_ENABLED

Valid values: Yes or No (default)

If set to Yes, logging is done during the load process to the file specified by the LOG\_FILE\_NAME option.

- LOG\_FILE\_NAME

The name of the file where logs are saved; default: *Java temp folder/HFMDData.log* or *HFMDData\_MultiplePeriod.log*

## Extracting Data

You can extract data for selected dimension members that exist in Financial Management. You must set up the Financial Management application before you can extract data from it.

Before extracting data, ensure that the members (metadata) exist in the Financial Management relational database; no records are extracted for members that do not exist (including the driver member and the members specified in the point of view.)

- To extract data from a Financial Management application, in Oracle Data Integrator:

- 1 Create an interface for extracting data.**

You can give the interface any name. See the *Oracle Data Integrator User's Guide* for instructions on creating interfaces.

- 2 On the diagram, drag and drop the source data store (HFMDData) from a Hyperion Financial Management model.**

- 3 Define the target data store.**

- 4 Define the mapping between source and target data.**

- 5 On the **Flow** tab:**

- Ensure that **LKM Hyperion Financial Management Data to SQL** is selected.
- Specify extract options.

See [“LKM Hyperion Financial Management Data to SQL” on page 25](#).

- 6 Click **Execute**.**

The data is extracted from the application.

- 7 Check the Operator to verify that the interface ran successfully.**



## LKM Hyperion Financial Management Data to SQL

LKM Hyperion Financial Management Data to SQL supports these options for defining how Oracle Data Integrator Adapter for Hyperion Financial Management extracts data:

- **SCENARIO\_FILTER**—The Scenario dimension members for which you are exporting data  
You can specify comma-delimited Scenario members or one scenario. If you do not specify scenarios, the system exports data for all scenarios.
- **YEAR\_FILTER**—The Year dimension members for which you are exporting data  
You can specify comma-delimited years or one year. If you do not specify years, the system exports data for all years.
- **PERIOD\_FILTER**—The set of Period dimension members for which you are exporting data  
Specify a range of members using the ~ character between start and end period numbers; for example, 1~12. If you do not specify periods, the system exports data for only the first period.
- **ENTITY\_FILTER**—The Entity dimension members for which you are exporting data  
You can specify comma-delimited entities or one entity. To specify the parent and child, separate them with a period; for example, I.Connecticut. If you do not specify entities, the system exports data for all entities.
- **ACCOUNT\_FILTER**—The Account dimension members for which you are exporting data  
You can specify comma-delimited accounts or one account. If you do not specify accounts, the system exports data for all accounts.
- **VIEW\_FILTER**—The View dimension member for which you are exporting data  
Possible values: Periodic, YTD, or <Scenario\_View> (default)
- **LOG\_ENABLED**  
If set to Yes, logging is done during the extract process to the file specified in LOG\_FILE\_NAME.
- **LOG\_FILE\_NAME**  
The name of the file where logs are saved
- **DELETE\_TEMPORARY\_OBJECTS**  
If set to Yes (default), tables, files, and scripts are deleted after integration.

**Tip:**

Temporary objects can be useful for resolving issues.

## Extracting Members from Member Lists

You can extract members from selected member lists and dimensions in a Financial Management application. You must set up the Financial Management application and load member lists into it before you can extract members from a member list for a dimension.

Before extracting members from a member list for a dimension, ensure that the member list and dimension exist in the Financial Management relational database. No records are extracted if the top member does not exist in the dimension.

- To extract members from a member list in a Financial Management application, in Oracle Data Integrator:

- 1 Create an interface for extracting member list.**

You can give the interface any name. See the *Oracle Data Integrator User's Guide* for instructions.

- 2 On the diagram, drag and drop the source data store (**EnumMembersList**) from a Hyperion Financial Management model.**

- 3 Define the target data store.**

- 4 Define the mapping between source and target data.**

- 5 On the **Flow** tab:**

- Ensure that **LKM Hyperion Financial Management Members to SQL** is selected.
- Specify extract options. See “LKM Hyperion Financial Management Members to SQL” on page 24.

See “LKM Hyperion Financial Management Members to SQL” on page 26.

- 6 Click **Execute**.**

The member list is extracted from the application.

- 7 Check the Operator to verify that the interface ran successfully.**

## LKM Hyperion Financial Management Members to SQL

LKM Hyperion Financial Management Members to SQL supports these options for defining how Oracle Data Integrator Adapter for Hyperion Financial Management extracts members of member lists:

- **DIMENSION\_NAME**—The name of the dimension for which you are creating a member list; required
- **MEMBER\_LIST\_NAME**—A label for the member list; required
- **TOP\_MEMBER**—The top member of the member list
- **LOG\_ENABLED**

Valid values: Yes and No (default)

If set to Yes, logging is done during the extract process to the file specified by the LOG\_FILE\_NAME option.

- LOG\_FILE\_NAME  
The name of the file where logs are saved
- DELETE\_TEMPORARY\_OBJECTS  
If set to Yes (default), tables, files, and scripts are deleted after integration.

**Tip:**

Temporary objects can be useful for resolving issues.

## Data Store Tables

IKM SQL to Hyperion Financial Management loads columns in tables to create data stores. The following sections describe the columns in each data store.

**Note:**

In the following topics, the column types are String unless the column descriptions specify otherwise.

### HFMDData

Column	Description
Scenario	A Scenario dimension member; example: Actual
Year	A Year dimension member; example: 2000
Entity	An Entity dimension member, in <i>parent.child</i> format; example: United States.NewYork to specify member NewYork as a child of member United States
Account	An Account dimension member; example: Sales
Value	A Value dimension member; example: USD
ICP	An Intercompany Partner dimension member; example: [ICP Entities]
Custom1	A Custom1 dimension member; example: AllCustomers
Custom2	A Custom2 dimension member
Custom3	A Custom3 dimension member
Custom4	A Custom4 dimension member
Period	A Period dimension member
Data Value	The value associated with the intersection

Column	Description
	This value is passed as a Double.
Description	A description of the data value

**Note:**

If custom dimensions have aliases, the aliases (rather than CustomN) are displayed as column names.

## HFMDData\_MultiplePeriods

Column	Description
Scenario	A Scenario dimension member; example: Actual
Year	A Year dimension member; example: 2000
Entity	An Entity dimension member, in <i>parent.child</i> format; example: United States.NewYork to specify member NewYork as a child of member United States
Account	An Account dimension member; example: Sales
Value	A Value dimension member; example: USD
ICP	An Intercompany Partner dimension member; example: [ICP Entities]
Custom1	A Custom1 dimension member; example: AllCustomers
Custom2	A Custom2 dimension member
Custom3	A Custom3 dimension member
Custom4	A Custom4 dimension member
Period1..n	For every data value being loaded, a period must be specified. The number of periods to be loaded for each intersection is specified when the Hyperion Financial Management model is reversed. A period column is created for each specified period.
Data Value1..n	Data values to be loaded The number of periods to be loaded for each intersection is specified when the Hyperion Financial Management model is reversed. A data value column is created for each specified period. This value is passed as a Double.
Description1..n	A description for each data value

**Note:**

If custom dimensions have aliases, the aliases (rather than CustomN) are displayed as column names.

## Account

Column	Description
Member	An account label; required
Description	A description for the account; required
Parent Member	The parent account member
Account Type; required	Valid account types: <ul style="list-style-type: none"> <li>● ASSET</li> <li>● LIABILITY</li> <li>● REVENUE</li> <li>● EXPENSE</li> <li>● FLOW</li> <li>● BALANCE</li> <li>● BALANCERECURRING</li> <li>● CURRENCYRATE</li> <li>● GROUPLABEL</li> <li>● DYNAMIC</li> </ul>
Is Calculated	Whether the account is calculated. Valid values: Y if the account is calculated, or N (default) if it is not calculated and manual input is enabled
Is Consolidated	Whether the account is consolidated into a parent account Valid values: Y if the account is consolidated into a parent, or N (default) if it is not
Is ICP	Whether intercompany transactions are allowed for this account. Valid values: <ul style="list-style-type: none"> <li>● Y if ICP transactions, including self-ICP transactions, are allowed</li> <li>● N (default) if ICP transactions are not allowed</li> <li>● R if ICP transactions are allowed but the account is restricted from having ICP transactions with itself</li> </ul> <p>If you specify Y or R, enter the name of the ICP TopMember. If you do not enter the top member, the default, [ICP TOP], is used.</p>
Plug Account	The name of the account used for identifying discrepancies in intercompany transactions; required if intercompany transactions are allowed for this account.
Custom 1...4 TopMember	The top member in the hierarchy of a Custom dimension that is valid for the account. The specified member, including all of its parents and descendants, is valid for the account. All other members of the Custom dimension are not valid for the account. These columns required if intercompany transactions are allowed for this account.
Number of Decimal Places	The number of digits to display to the right of the decimal point for the account values; required Specify an integer from 0 (default) to 9.

Column	Description
Uses Line Items	Whether the account can have line items. Valid values: Y if the account uses line items, or N (default) if it does not
Aggr Custom 1...4	Whether aggregation is enabled for intersections of the account and the Custom dimensions. This column is used for special totals, not summing. Valid values: Y (default) if the account is allowed to aggregate with Custom dimensions, or N if it is not
User Defined 1...3	Optional custom text for the account
XBRL Tag	Optional XBRL tag for the account
Security Class	The name of the security class that defines users who can access the account data. Default: DEFAULT security class
ICP Top Member	The top member of the ICP group assigned to the account
Enable Data Audit	Whether data auditing is enabled for the account. Valid values: Y (default) to enable auditing, or N to disable auditing
Description 2...10	Optional additional descriptions for the account

## Entity

Column	Description
Member	An entity label; required
Description	A description for the entity; required
Parent Member	The parent entity member
Default Currency	The default currency for the entity; required
Allow Adj	Valid values: Y if journal postings are permitted, or N (default) if journal entries are not permitted
Is ICP	Valid values: Y if the entity is an intercompany entity, or N (default) if it is not <b>Note:</b> An intercompany entity is displayed in the POV in the ICP dimensions under [ICP Entities].
Allow Adj From Child	Valid values: Y if journal postings from children of this parent entity are permitted, or N (default) if they are not
Security Class	The name of the security class that defines users who can access the entity's data. Default: DEFAULT security class
User Defined 1...3	Optional custom text for the entity

Column	Description
Holding Company	The holding company for the entity. Valid values: Any valid entity or blank (default)
Description 2...10	Optional additional descriptions for the entity

## Scenario

Column	Description
Member	A scenario label; required.
Description	A description for the scenario; required
Parent Member	The parent Scenario member
Default Frequency	Period types for which data input is valid for the scenario; required
Default View	Whether the view is YTD or Periodic; required
Zero View Non Adj	Whether the view is YTD or Periodic when missing, nonadjusted data values exist; required
Zero View Adj	Whether the view is YTD or Periodic when missing, adjusted data values exist; required
Consol YTD	The view for consolidations; required Valid values: Y for YTD, or N for Periodic
Support PM	Whether Process Management command is enabled in Data Explorer; required. Valid values: Y to enable Process Management, or N to disable Process Management
Security Class	The name of the security class that defines users who can access the scenario data. Default: DEFAULT security class
Maximum Review Level	The maximum process management review level for the scenario. Enter an integer from 1 to 10.
Uses Line Items	Valid values: Y if the scenario can accept line items, or N (default) if it cannot
Enable Data Audit	Valid values: Y to enable auditing, or N (default) to disable auditing
Def Freq For IC Trans	The default frequency for intercompany transactions. Enter a string that identifies a valid frequency for the application. The default value is an empty string, representing no default frequency.
User Defined 1...3	Optional custom text for the scenario
Description 2...10	Optional additional descriptions for the scenario

## Currency

Column	Description
Member	A currency label; required
Description	A description for the currency; required
Scale	<p>The unit in which amounts are displayed and stored for the currency, which identifies where the decimal point is placed; required</p> <p>Must be one of the following valid integer values:</p> <ul style="list-style-type: none"> <li>● Blank = None</li> <li>● 0 = Units</li> <li>● 1 = Tens</li> <li>● 2 = Hundreds</li> <li>● 3 = Thousands</li> <li>● 4 = Ten Thousands</li> <li>● 5 = Hundred Thousands</li> <li>● 6 = Millions</li> <li>● 7 = Ten Millions</li> <li>● 8 = Hundred Millions</li> <li>● 9 = Billions</li> </ul>
Translation Operator	<p>Whether conversions for the currency are calculated by multiplying or dividing the translation rate.</p> <p>Valid values: D to divide (default) or M to multiply</p>
Description 2...10	Optional additional descriptions

## Custom1-4

Column	Description
Member	The label of a custom dimension member; required
Description	A description for the custom dimension member; required
Parent Member	The parent custom member; required
Is Calculated	<p>Whether the base-level custom account is calculated.</p> <p>If a base-level custom account is calculated, you cannot manually enter values.</p> <p>Valid values: Y if the account is calculated, N if it is not calculated</p>
Switch Sign	<p>Whether the sign is changed (Debit/Credit) for FLOW accounts using the following rules:</p> <ul style="list-style-type: none"> <li>● ASSET to LIABILITY</li> <li>● LIABILITY to ASSET</li> <li>● EXPENSE to REVENUE</li> </ul>



Column	Description
	<ul style="list-style-type: none"> <li>● REVENUE to EXPENSE</li> <li>● BALANCE to FLOW</li> <li>● FLOW to BALANCE</li> </ul> <p>Valid values: Y if the account type is switched, or N if it is not switched</p>
Switch Type	<p>The account type change for FLOW accounts, following these rules:</p> <ul style="list-style-type: none"> <li>● ASSET to EXPENSE</li> <li>● EXPENSE to ASSET</li> <li>● LIABILITY to REVENUE</li> <li>● REVENUE to LIABILITY</li> <li>● BALANCE to FLOW</li> <li>● FLOW to BALANCE</li> </ul> <p>Valid values: Y if the account type is switched, or N if it is not switched</p>
Security Class	<p>The name of the security class that defines users who can access the custom dimension data</p> <p>Security access applies only to data.</p> <p>Default: DEFAULT security class</p>
User Defined 1...3	Optional custom text for the scenario
Aggr Weight	<p>The aggregation weight for the custom dimensions; passed as Double</p> <p>Default: 1</p>
Description 2...10	Optional additional descriptions for the scenario

## EnumMembersList

Column	Description
Members	The members of the member list

