Oracle® Health Sciences Clinical Development Analytics

Administrator's Guide

Release 3.1.1 for Standard Configuration

E63311-01

May 2015

Oracle Health Sciences Clinical Development Analytics Administrator's Guide, Release 3.1.1 for Standard Configuration

E63311-01

Contents

A Troubleshooting

Preface

This guide provides information on the configuration of Oracle Health Sciences Clinical Development Analytics (OHSCDA).

This preface contains the following topics:

Audience

This guide is intended for:

- Data Warehouse Administrators, ETL Developers and Operators
- System Administrators
- Deduplication Administrators and Data Stewards

This guide assumes that you have the following general skills:

- Knowledge of Oracle Business Intelligence Enterprise Edition Plus
- Knowledge of Oracle Data Integrator
- Familiarity with Oracle Clinical
- Familiarity with Oracle's Siebel Clinical
- Familiarity with Oracle Healthcare Master Person Index

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc`.

Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info` or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Finding Information and Patches on My Oracle Support

Your source for the latest information about Oracle Health Sciences Clinical Development Analytics is Oracle Support's self-service Web site, My Oracle Support (formerly MetaLink).

Before you install and use an Oracle software release, always visit the My Oracle Support Web site for the latest information, including alerts, release notes, documentation, and patches.

Creating a My Oracle Support Account

You must register at My Oracle Support to obtain a user name and password account before you can enter the Web site.

To register for My Oracle Support:

1. Open a Web browser to http://support.oracle.com.
2. Click the **Register here** link to create a My Oracle Support account. The registration page opens.
3. Follow the instructions on the registration page.

Signing In to My Oracle Support

To sign in to My Oracle Support:

1. Open a Web browser to http://support.oracle.com.
2. Click **Sign In**.
3. Enter your user name and password.
4. Click **Go** to open the My Oracle Support home page.

Searching for Knowledge Articles by ID Number or Text String

The fastest way to search for product documentation, release notes, and white papers is by the article ID number.

To search by the article ID number:

1. Sign in to My Oracle Support at http://support.oracle.com.
2. Locate the Search box in the upper right corner of the My Oracle Support page.
3. Click the sources icon to the left of the search box, and then select Article ID from the list.
4. Enter the article ID number in the text box.
5. Click the magnifying glass icon to the right of the search box (or press the Enter key) to execute your search.

   The Knowledge page displays the results of your search. If the article is found, click the link to view the abstract, text, attachments, and related products.

In addition to searching by article ID, you can use the following My Oracle Support tools to browse and search the knowledge base:

- Product Focus — On the Knowledge page, you can drill into a product area through the Browse Knowledge menu on the left side of the page. In the Browse any Product, By Name field, type in part of the product name, and then select the product from the list. Alternatively, you can click the arrow icon to view the complete list of Oracle products and then select your product. This option lets you focus your browsing and searching on a specific product or set of products.

- Refine Search — Once you have results from a search, use the Refine Search options on the right side of the Knowledge page to narrow your search and make the results more relevant.
- Advanced Search — You can specify one or more search criteria, such as source, exact phrase, and related product, to find knowledge articles and documentation.

Finding Patches on My Oracle Support

Be sure to check My Oracle Support for the latest patches, if any, for your product. You can search for patches by patch ID or number, or by product or family.

To locate and download a patch:

1. Sign in to My Oracle Support at `http://support.oracle.com`.
2. Click the **Patches & Updates** tab.

   The Patches & Updates page opens and displays the Patch Search region. You have the following options:

   - In the Patch ID or Number is field, enter the primary bug number of the patch you want. This option is useful if you already know the patch number.
   - To find a patch by product name, release, and platform, click the Product or Family link to enter one or more search criteria.
3. Click **Search** to execute your query. The Patch Search Results page opens.
4. Click the patch ID number. The system displays details about the patch. In addition, you can view the Read Me file before downloading the patch.
5. Click **Download**. Follow the instructions on the screen to download, save, and install the patch files.

Finding Documentation on Oracle Technology Network

The Oracle Technology Network Web site contains links to all Oracle user and reference documentation. To find user documentation for Oracle products:

1. Go to the Oracle Technology Network at

   `http://www.oracle.com/technetwork/index.html` and log in.
2. Mouse over the Support tab, then click the **Documentation** hyperlink.

   Alternatively, go to Oracle Documentation page at

   `http://www.oracle.com/technology/documentation/index.html`
3. Navigate to the product you need and click the link.

   For example, scroll down to the Applications section and click Oracle Health Sciences Applications.
4. Click the link for the documentation you need.

Related Documents

For more information, see the following documents in the Oracle Business Intelligence Enterprise Edition, Oracle Clinical, Siebel Clinical, Oracle Health Sciences InForm, and Oracle Data Integrator documentation sets:

Oracle Business Intelligence Enterprise Edition Documentation

The Oracle Business Intelligence Suite Enterprise Edition (OBIEE) documentation set includes:

- *Oracle Fusion Middleware User's Guide for Oracle Business Intelligence Enterprise Edition*
- *Oracle Fusion Middleware Metadata Repository Builder's Guide for Oracle Business Intelligence Enterprise Edition*
- *Oracle Fusion Middleware System Administrator's Guide for Oracle Business Intelligence Enterprise Edition*
- *Oracle Fusion Middleware Scheduling Jobs Guide for Oracle Business Intelligence Enterprise Edition*
- *Oracle Fusion Middleware Security Guide for Oracle Business Intelligence Enterprise Edition*
- *Oracle Fusion Middleware Developer's Guide for Oracle Business Intelligence Enterprise Edition*
- *Oracle Fusion Middleware Integrator's Guide for Oracle Business Intelligence Enterprise Edition*

Oracle Clinical Documentation

The Oracle Clinical documentation set includes:

- *Oracle Clinical Administrator's Guide*
- *Oracle Clinical Getting Started*
- *Interfacing from Oracle Clinical*
- *Oracle Clinical Conducting a Study*
- *Oracle Clinical Creating a Study*
- *Oracle Clinical Installation Guide*

Siebel Clinical Documentation

The Siebel Clinical documentation set includes:

- *Siebel Data Model Reference for Industry Applications*
- *Siebel Life Sciences Guide*

Oracle Health Sciences InForm Documentation

The Oracle Health Sciences InForm documentation set includes:

- *Oracle Health Sciences InForm Upgrade and Migration Guide*
- *Oracle Health Sciences InForm Installation Guide*
- *Oracle Health Sciences InForm Study and Reporting Setup Guide*
- *Oracle Health Sciences InForm User Guide*

- *Oracle Health Sciences InForm Reporting and Analysis*
- *Oracle Health Sciences InForm Utilities Guide*
- *Oracle Health Sciences InForm Clinical Data API Guide*
- *Oracle Health Sciences InForm Reporting Database Schema*
- *Oracle Health Sciences InForm Secure Configuration Guide*
- *Oracle Health Sciences InForm Third Party Licenses and Notices*

Oracle Data Integrator Documentation

The Oracle Data Integrator documentation set includes:

- *Release Notes for Oracle Data Integrator*
- *Understanding Oracle Data Integrator*
- *Administering Oracle Data Integrator*
- *Installing and Configuring Oracle Data Integrator*
- *Application Adapters Guide for Oracle Data Integrator*
- *Developing Knowledge Modules with Oracle Data Integrator*
- *Connectivity and Knowledge Modules Guide for Oracle Data Integrator*
- *Oracle Data Integrator Tool Reference*
- *Java API Reference for Oracle Data Integrator*
- *Oracle Data Integrator Online Help*

Oracle Healthcare Master Person Index Documentation

The Oracle Healthcare Master Person Index documentation set includes:

- *Oracle Healthcare Master Person Index Analyzing and Cleansing Data User's Guide*
- *Oracle Healthcare Master Person Index Australia Patient Solution User's Guide*
- *Oracle Healthcare Master Person Index Command Line Reports and Database Management User's Guide*
- *Oracle Healthcare Master Person Index Configuration Guide*
- *Oracle Healthcare Master Person Index Configuration Reference*
- *Oracle Healthcare Master Person Index Data Manager User's Guide*
- *Oracle Healthcare Master Person Index Installation Guide*
- *Oracle Healthcare Master Person Index Loading the Initial Data Set User's Guide*
- *Oracle Healthcare Master Person Index Match Engine Reference*
- *Oracle Healthcare Master Person Index Message Processing Reference*
- *Oracle Healthcare Master Person Index Provider Index User's Guide*
- *Oracle Healthcare Master Person Index Real-time Loader User's Guide*
- *Oracle Healthcare Master Person Index Release Notes*
- *Oracle Healthcare Master Person Index Security Guide*
- *Oracle Healthcare Master Person Index Standardization Engine Reference*
- *Oracle Healthcare Master Person Index United Kingdom Patient Solution User's Guide*

- *Oracle Healthcare Master Person Index United States Patient Solution User's Guide*
- *Oracle Healthcare Master Person Index User's Guide*
- *Oracle Healthcare Master Person Index Working With HPD Profile Application User's Guide*
- *Oracle Healthcare Master Person Index Working With IHE Profiles User's Guide*

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1

Maintaining the OBIEE Repository and Warehouse

This chapter contains the following topics:

1.1 Maintaining the Oracle Health Sciences Clinical Development Analytics OBIEE Repository

Each release of Oracle Health Sciences Clinical Development Analytics (OHSCDA) contains an OBIEE Repository (RPD) file. The Repository is the data store for the Oracle BI Server. It maintains the mapping of the physical tables comprising the data warehouse to the Presentation Layer, which holds the columns and tables available for use in OBIEE Analyses. As shipped, the RPD corresponds to the OHSCDA data warehouse, and can be used without any modification.

Note: OHSCDA ships with more than one Repository (Oracle Data Integrator (ODI) and OBIEE have Repositories). This section, however, is concerned only with the OBIEE Repository that ships with OHSCDA. Therefore, throughout this section, the terms Repository and RPD should be taken to mean the OBIEE Repository shipped with OHSCDA.

However, you might find it desirable to modify the Oracle-supplied OHSCDA Repository file (RPD), for any of the following reasons:

- You want to add a column or table to the data warehouse, and propagate that addition into the layers of the Repository.
- You want to add a calculated column in the Presentation Layer as a function of some set of Physical layer columns.
- You want to modify a Repository variable value, or add a new repository variable, for use in some Presentation Catalog calculation. For instance, you may want to modify the frequency with which the value of the dynamic repository variable CURRENT_DAY is refreshed.

This section describes the procedures you must follow to carry out these types of modifications.

You should be aware that, once you have modified the Oracle-supplied Repository, it is your responsibility to merge these modifications into Repositories supplied by Oracle in subsequent patches and releases of OHSCDA. Details on how to re-apply your modifications are provided below.

> **Caution:** Changes to the Repository should be made with care.
>
> Privileges to make changes in the Repository should be granted only to a limited set of users who need to make such changes and also know how to make them correctly.
>
> Changes should be tested on a side copy of the Repository before being released for production use.

1.1.1 Modifying the Repository

In OHSCDA Standard configuration, maintaining versions of the Repository is the responsibility of the administrator. Oracle recommends that you:

- Make a backup copy of each state of the repository before you modify it
- Attempt all modifications on a side copy of the Repository before putting it into production use
- Verify all changes to the Repository by running requests from the Presentation Service against the modified Repository

Therefore, Oracle recommends that you do not modify the deployed OHSCDA Repository directly.

If you need to modify the Repository, use the OBIEE Administration Tool.

1.1.1.1 Details for Selected Modifications

This section contains details on how to perform certain modifications to the RPD.

To modify the frequency with which CURRENT_DAY is refreshed:

1. In Oracle BI Administration Tool, click **Manage** > **Variables**.
2. Expand Repository and click **Initialization Block** > **ETL_Refresh_Ranges**.
3. In the Repository Variable Init Block - ETL_Refresh_Ranges screen, modify the value of **Refresh interval**.

   Refresh interval indicates how often you want to refresh the value of CURRENT_DAY dynamic repository variable. By default, this value is set to 5 minutes. That is, the CURRENT_DAY dynamic repository variable is refreshed every five minutes. Modify Refresh interval to a suitable value.

See Also:

- *Oracle® Fusion Middleware Metadata Repository Builder's Guide for Oracle Business Intelligence Enterprise Edition 11g Release 1* for more information about modifying the RPD.

1.1.2 Merging Changes into a New Oracle-supplied Repository

Each OHSCDA release, and some patches, includes a copy of the OHSCDA Repository. If you do modify your copy of the OHSCDA Repository, you must merge your changes into the Oracle-supplied Repository each time you receive a release or patch of OHSCDA that includes a repository. At upgrade time, use the OBIEE Merge Repository Wizard (accessed through File > Merge) in the Repository Administration Tool to merge your modified RPD with the Oracle-supplied RPD.

For information on comparing and merging Repositories, refer to Chapter 15, Managing Oracle BI Repository Files, in *Oracle Fusion Middleware Metadata Repository Builder's Guide for Oracle Business Intelligence Enterprise Edition 11g Release 1*.

1.2 Maintaining the Oracle Health Sciences Clinical Development Analytics Data Warehouse

You may need to modify the OHSCDA data warehouse, typically for one of the following reasons:

- *Derivation*: Calculation of a new measure as a function of some supplied measures.
- *Extension*: Adding data that was not delivered with OHSCDA.
- *Substitution*: Swapping data from a different source for a column that was delivered with OHSCDA.

> **Caution:** Exercise caution when you modify the data warehouse. Please conform to the recommendations mentioned in Section 1.2.3, "Modifying Data Warehouse Tables" on page 1-6.

1.2.1 Extensions

An *extension* is a new column added to the data warehouse for data not extracted from the transactional sources by the SDE supplied with OHSCDA.

Example: Adding the study manager's name as an attribute of the study dimension for each study. The following are the assumptions:

- This information is available in a non-Oracle Clinical database, in a table named STUDY_MANAGERS. The column in that table holding the Study Manager name is STUDY_MANAGER_NAME.
- This table has a foreign key to the primary key in Oracle Clinical table OCL_ STUDIES.

> **Note:** When you add extension columns to the OHSCDA data model, the column name must start with "X_". This avoids collisions with names of columns added by Oracle in later releases of OHSCDA.

You can add the column either to the target table, or to an extension table. To minimize the level of effort required when implementing a release with a new repository, Oracle recommends that you add extensions to the warehouse through user-defined extension tables, rather than by adding new columns directly into the relevant staging and target tables.

Note: For each user-defined extension tables, its error table E$_<user-defined extension tables name> must be created. Also, provide the select, insert, update, and delete grants to the ODI warehouse work schema.

Perform the following tasks to add the study manager name to the study dimension for each study:

Important: Create a backup of the Repository and Views before modifying it. Retain a log of changes you make to the Repository to ensure that the Merge Wizard correctly reapplies them to the next repository that Oracle ships.

1. Modify staging table W_RXI_STUDY_DS, adding the X_STUDY_MANAGER_NAME column. To modify the staging table, perform the following tasks in ODI:
   a. Copy the warehouse datastore W_RXI_STUDY_DS into another model.

      This is to ensure that the changes are not overwritten in the next OHSCDA upgrade.

   b. Add a new column X_STUDY_MANAGER_NAME and save it.
   c. Navigate to the **Design** > **Tables** subtab and add the new column X_STUDY_MANAGER_NAME under W_RXI_STUDY_DS and save it.
2. Modify the SDE that populates W_RXI_STUDY_DS, as indicated in the following steps:
   a. Copy the SDE into another folder (to ensure that the changes are not overwritten in the next OHSCDA upgrade) or use an existing customized SDE.
   b. Duplicate the SRO view (VW_SDE_STUDY_DIM), used for this SDE as VW_SDE_STUDY_DIM_X.
   c. Add the STUDIES_MANAGERS table in the SQL to select STUDY_MANAGER_NAME.
   d. Grant select on SRO view (VW_SDE_STUDY_DIM_X) to the ODI source work schema.
   e. Reverse-engineer the new customized SRO schema view into another model.
   f. Open the copied SDE mapping in ODI.
   g. Click the **Logical** tab.
   h. Select the source to view its properties.
   i. Under the **General** section, change the Datastore to point to the newly created customized SRO view (VW_SDE_STUDY_DIM_X).
   j. Select the mapping target to view its properties.
   k. Under the **General** section, change the Datastore to point to the customized stage table.
   l. Map the new column STUDY_MANAGER_NAME from source to target.
   m. Validate and save the mapping.

   n. Generate scenario and schedule it in the load plan.

3. If W_RXI_STUDY_D does not already exist to support other extension, create the extension table W_RXI_STUDY_DX, containing one column [STUDY_WID] to function as a foreign key that joins to the primary key in W_RXI_STUDY_D. This table is populated with one row for each row in W_RXI_STUDY_D when the Study SIL executes.
4. Add the extension table W_RXI_STUDY_DX in ODI. To add a table, perform the following task in ODI:
   a. Reverse-engineer W_RXI_STUDY_DX into a new model (to ensure that the changes are not overwritten in the next OHSCDA upgrade) or use an existing customized table definition, if already exists.
5. Modify the SIL view and mapping that populate W_RXI_STUDY_D.
6. Add instructions to create a record in W_RXI_STUDY_DX for each record in W_RXI_STUDY_D and to copy W_RXI_STUDY_DS. STUDY_MANAGER_NAME into W_RXI_STUDY_DX.STUDY_MANAGER_NAME for each record.
7. Generate scenario and schedule it in the load plan.
8. Using W_RXI_DISCREPANCY_FX as an example, propagate the extension table and its content to the Business and Presentation layers.

1.2.2 Substitutions

A substitution occurs if you have a preferred alternative source of data for a column that OHSCDA populates from Oracle Clinical or Siebel Clinical. For example, you have a system for defining what data collection instruments (DCIs) are mandatory for a given study, subject, or subject visit, and you prefer that over the OHSCDA calculation that is based on expected data collection modules (DCMs) and subject visit schedules. In this case, your column will be present in a table, and the SDE that extracts the data to a staging table already exists. Perform the following tasks:

1. Create a view that joins the Oracle Clinical table and the table containing your mandatory DCI information, in which your values replace the Oracle-supplied values for the column of interest. Call this the Substitution view. Also, provide the select grant on Substitution view to the source work schema.
2. Modify the SDE to read from the Substitution view, rather than the Oracle-supplied view.

   To modify the SDE, perform the following tasks in ODI:

   a. Copy the SDE into another folder (to ensure that the changes are not overwritten in the next OHSCDA upgrade) or use an existing customized SDE.
   b. Reverse-engineer the Substitution view into another model.
   c. Select the source to view its properties.
   d. Under the **General** section, change the Datastore so that the mapping reads from the Substitution view.
3. Validate and save the mapping.

If you make changes to a source table, you must propagate that change forward as far as necessary. Some of the scenarios and the related necessary adjustments are described in the Table 1–1:

Table 1–1 Scenarios Requiring Necessary Adjustments

Scenario	Adjustments Required
New table has the same layout as the old table, but is passed through from a different source	Change the SDE view that reads the old table to instead read the new table.
Modified table has modified layout	1. Modify the SDE view to read the modified layout. 2. Modify the staging table populated by the SDE to include the modified layout. 3. Modify the SIL view and mapping to read the modified layout. 4. Modify the target table to include the modified layout. 5. Modify the RPD to accept the changed data warehouse table.
New table	1. Add a new staging table and its error table to accept the new input. 2. Add a new SDE view in the SRO schema. 3. Add a new SDE mapping to read from the new view and write to the new staging table. 4. Add a new warehouse table and its error table to make the new data available to the BI Server. 5. Add a new SIL view in the warehouse. 6. Add an SIL mapping to populate the new data warehouse table from the new view. 7. Modify the RPD to accept the new warehouse table.

1.2.3 Modifying Data Warehouse Tables

Depending on what changes are required to the data warehouse, it is necessary to modify SRO and warehouse views either in SRO and/or warehouse schema and in ODI, or in the source, staging, and target tables.

Managing Indexes

OHSCDA is delivered with a set of indexes. You can create additional indexes to meet your query requirements.

> **Note:** Oracle recommends that you Drop and re-create all indexes only for full load. This should be disabled for incremental load.

It is useful to drop all indexes on the warehouse tables before loading large volumes of data, and to recreate them afterward. ODI can automate this process for you. To drop indexes before a load, and recreate them afterward, perform the following tasks in ODI:

1. Execute the **CDA-Complete Warehouse with Dedup** load plan.
2. Set the variable VAR_OCDA_DWH_LOAD_FLAG value as 1 while running the load plan.

Note: Setting the variable VAR_OCDA_DWH_LOAD_FLAG value as 1, while running load plan, executes for full load, truncates all warehouse tables, drops indexes, loads data, and recreates indexes.

1.3 Derivations

A *derivation* is a calculation of a new measure as a function of some supplied measures. You can use any of the following approaches to calculate derivations:

- Calculate the derivation as part of the creation of an analysis.

  In this approach, only the Presentation Catalog is modified. However, you must specify the calculation for each analysis, and the calculation is executed every time the analysis is executed.

- Calculate the derivation in the physical or business layer of the RPD and propagate it to the presentation layer. This makes the derivation you created appear in Answers as a column.

  Using this approach, you can specify the calculation once and use it for multiple analyses. Once it has been propagated to the Presentation Layer, the derived value looks the same as any other Answers column to an end user.

- Calculate the derivation in the data warehouse.

  In this approach, you add an extension column to the warehouse to hold the derived values. To do so:

  1. Add the result column to the staging and target table of the fact or dimension in which the derivation is being created. It is an extension column, so its name must begin with "X_".
  2. Modify the SDE view and mapping to calculate the derived value during extract of records from the sources.
  3. Modify the SIL view and mapping to transfer the derived value from the Staging to the Target table.
  4. Create and execute a script to provide values for the derived column in rows already present in the Target.
  5. Add the column to all layers of the RPD.

In carrying out these steps, follow the rules described above for changes to the RPD and warehouse.

2 Extract Transform Load Programs

This chapter contains the following topics:

To load data from the source systems to the data warehouse, OHSCDA uses Extract Transform and Load (ETL) programs that

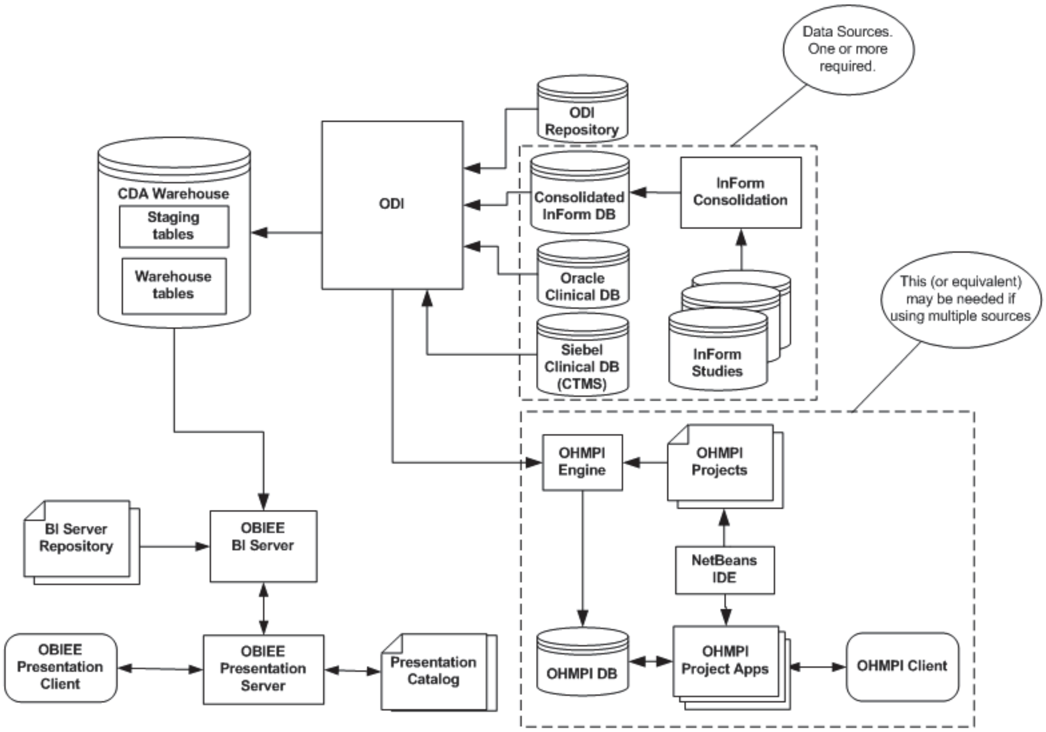
- Identify and read desired data from different data source systems,
- Clean and format data uniformly, and
- Write it to the target data warehouse.

In OHSCDA, Oracle Clinical, Siebel Clinical, and InForm are the source systems for which Oracle provides predefined ETL.

2.1 ETL Architecture

ETL is the process by which data is copied from a source database and placed in warehouse tables, available for analytic queries.

Figure 2–1 ETL Architecture



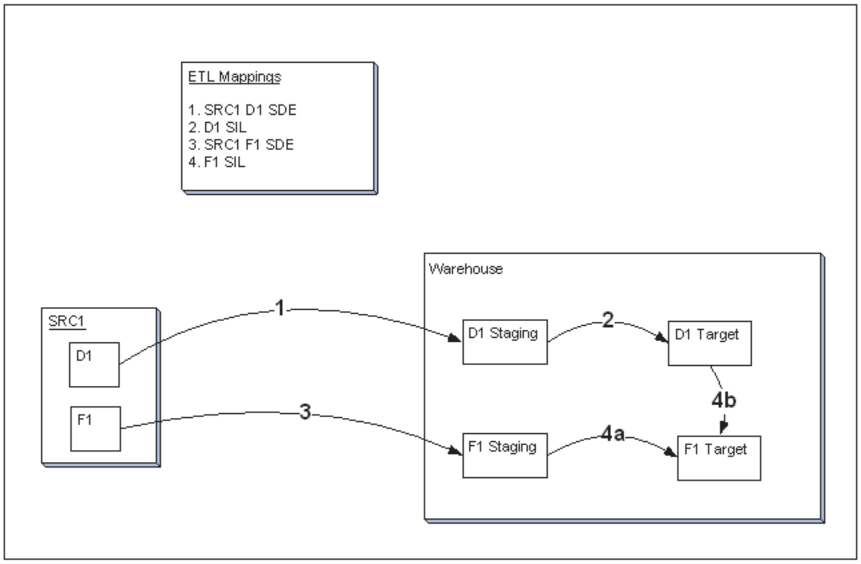
OHSCDA supports the extraction of data from one or more transactional application source databases. In OHSCDA, ETL is performed by execution of ODI Mappings. A mapping is a program that selects data from one or more tables, performs transformations and derivations on the data, and inserts or updates the results in a target table. Mappings are stored in the ODI Repository. Oracle uses ODI to schedule and supervise the execution of the entire set of ETL necessary to populate the OHSCDA warehouse.

Because data can be extracted from databases managed by different applications, the ETL mappings to load any given warehouse table are composed of two parts. The first part is a mapping that reads from a specific application's database and writes to a common warehouse Staging table. This mapping is referred to as SDE, for Source-Dependent Extract. Records in the Staging table have a common format, regardless of the source from which they were extracted. For a given warehouse table, there must be one SDE for each supported source application. The second mapping for the given warehouse table reads from its Staging table and writes to the final warehouse table. This part is called as the SIL, for Source-Independent Load.

If you load data from multiple sources, it may be necessary to deduplicate the dimension data arriving from the various sources. For example, more than one source may supply a description of the same Study. In this case, OHSCDA provides a framework for managing the deduplication, and also provides pre-built programs for deduplicating, using Oracle Healthcare Master Person Index (OHMPI). You can elect to use a program other than OHMPI for deduplication.

Figure 2–2 shows the ETL process for extracting data from two tables (one dimension D1 and one fact F1) in one source application database.

Figure 2–2 ETL for One Source Database

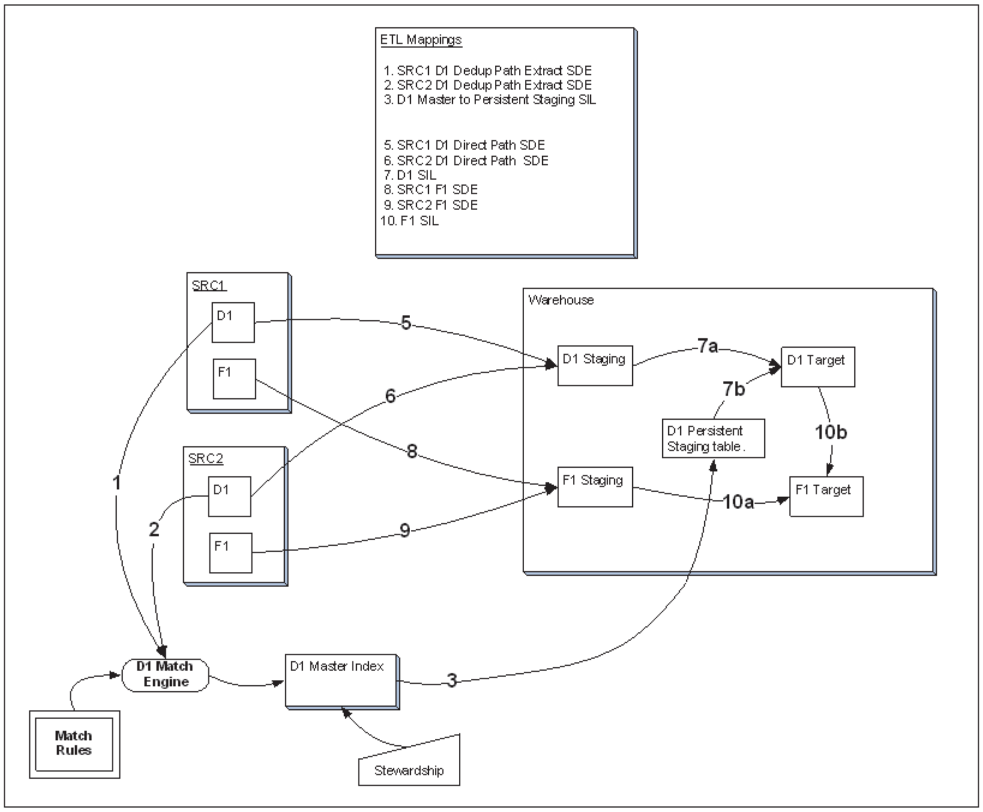


Each numbered arrow represents the execution of a Mapping. The Mappings are executed in the order indicated by the numbers. Mapping 4, the SIL that loads fact F1 into the warehouse target table for the fact, has two components. The first part (4a) represents the copy, transformation, and derivation of the staged data for the fact. The second part (4b) represents the setting of the values of each record's foreign key to dimension D1. The result is that dimension D1 can be queried as one of the dimensions in the star-schema centered on fact F1.

OHSCDA provides SIL for each of its warehouse tables. It provides SDE to stage data for these tables from InForm, Oracle Clinical, and Siebel Clinical database. The execution of the mappings is under the control of ODI.

If data is loaded from more than one data source, it is necessary to integrate the data, by identifying and merging duplicate source records. To address this, OHSCDA provides a multi-source integration capability. Figure 2–3 illustrates how OHSCDA's ETL architecture expands to support multi-source integration. Multi-source integration is described in detail in Chapter 3, "Multi-Source Integration".

Figure 2–3 OHSCDA ETL in Multi-source Integration



The OHSCDA ETL architecture is flexible. It allows you to integrate data from multiple databases instances for one application (for example, multiple Oracle Clinical databases), or databases from different source applications (for example, one Oracle Clinical and one Siebel Clinical), or any combination of these.

The OHSCDA ETL architecture is extensible. OHSCDA provides SDE mappings for InForm, Oracle Clinical, and Siebel Clinical. You can add data from another application to the OHSCDA warehouse by creating the SDE to map its database tables to the OHSCDA Staging tables.

For each source application there is one SDE mapping for each warehouse table. This SDE extracts data from the source system and loads it to the staging tables. SDEs have the following feature:

- **Incremental submission mode**: OHSCDA-supplied ETL uses timestamps and journal tables in the source transactional system to optimize periodic loads.

There is one SIL mapping for each warehouse target table. The SIL extracts the normalized data from the staging table and inserts it into the data warehouse star-schema target table. SILs have the following attributes:

- Concerning changes to dimension values over time, OHSCDA overwrites old values with new ones. This strategy is termed as *Slowly Changing Dimension approach 1*.
- OHSCDA's data model includes aggregate tables and a number of indexes, designed to minimize query time.
- The results of each ETL execution are logged by ODI. The logs hold information about errors encountered, during execution.

Every table in OHSCDA warehouse that is populated through ODI has a corresponding error log table.

The naming convention for error log table is as follows:

```
E$_<TABLE_NAME>
```

If the length of the table name is greater than 26, a different naming convention is used for error log table. The characters until the second instance of underscore (_) in the table name are replaced with E$. The naming convention in this case will be as follows:

```
E$_<remaining characters of table name>
```

For example, if the table name is W_HS_MDM_APPLICATION_USER_DS, the characters W_HS are replaced with E$. Hence, the name of the error log table will be E$_MDM_APPLICATION_USER_DS.

During ETL execution, records which fail to be inserted in the target table (for example, some records violate a constraint) are placed in the corresponding error tables. You can review which records did not make it into the data warehouse, and decide on appropriate action with respect to them.

Adding Data Source Information

As you read data from different database instances, you need to specify the source of the data. OHSCDA provides the OCDA_CONTROL_S, OCDA_CONTROL_LOG_S, and W_RXI_DATASOURCE_S tables (in source-specific SRO schema) that stores information about data sources from which data is extracted for OHSCDA. These tables are populated by procedure trigger or seed data.

- **OCDA_CONTROL_S**: This table is present in the OHSCDA SRO schema. It has only one row per Enterprise ID present in the OHSCDA application. Initial data is populated on full load and subsequent ETL Incremental Runs update this table accordingly for the specific enterprises for which it is run, thereby tracking the last extract date appropriately. The following are some of the columns in these table:
  - **DATASOURCE_NUM_ID**: It is used to identify the data source.
  - **SOURCE_EXTRACTION_START_DT**: It stores the ETL start date.
  - **SOURCE_EXTRACTION_END_DT**: It stores the ETL end date.
  - **PROGRAM_NAME**: It stores information about load plan name.
  - **PROGRAM_RUN_DT**: It stores date on which the last ETL was successfully executed. This value is populated when ETL is executed successfully. Else, it it is NULL.
  - **CONFIG_DAYS**: It stores value for the PRUNE days. By default, its value is 1. You can modify the value of this column. End date for incremental load is derived from this column as SYSDATE - CONFIG_DAYS.
  - **ETL_RUN_ID**: It is generated using a sequence.
- **OCDA_CONTROL_LOG_S**: Before every ETL run the information in the table OCDA_CONTOL_S will be copied to OCDA_CONTROL_LOG_S for auditing via a trigger.
- **W_RXI_DATASOURCE_S**: It is used for InForm source system to get DATASOURCE_NUM_ID and DATASOURCE_NAME. The following are some of the columns in these table:
  - **ROW_WID**: A unique ID for each record in the table.
  - **DATASOURCE_NUM_ID**: The ID for the database.

- **DATASOURCE_NAME**: A meaningful name of the database.
- **DATASOURCE_TYPE**: Application system that manages the database.
- **DESC_TEXT**: Optional text describing the purpose of the database.
- **INTEGRATION_ID**: Set this to the same values as DATASOURCE_NUM_ID
- **INFORM_STUDY_GUID**: Study GUID of the InForm source.

See Also:

- *Oracle Health Sciences Clinical Development Analytics Electronic Technical Reference Manual*, for more information about these tables.
- Section 2.1.1, "Adding a New Data Source" on page 2-7, for more information about how to add a new data source to OHSCDA.

Handling Deletions in Siebel Clinical

OHSCDA provides an optional feature to manage hard deletion of records in Siebel Clinical. You create triggers in the source system to handle deletion of records. To do this:

1. Navigate to the temporary staging location where the OHSCDA installer copies the installation files.
2. Connect to the Siebel Clinical data source and run the `ocda_sc_del_trigger.sql` script delivered with OHSCDA. This script creates the triggers on tables provided as input. The following are the tables in Siebel Clinical for which OHSCDA supports creating triggers:

   - S_CL_PTCL_LS
   - S_PROD_INT
   - S_CL_SUBJ_LS
   - S_CONTACT
   - S_CL_PGM_LS
   - S_PTCL_SITE_LS
   - S_EVT_ACT
   - S_ORG_EXT

   Provide a list of comma separated values of table names for which the triggers need to be created as the script's input. For example, S_CL_PTCL_LS, S_PROD_INT, and S_CL_SUBJ_LS. The tables names that you provide can only be a subset of the tables listed above.

   > **Note:** When you delete a record in the table, the primary key of the deleted record is inserted in the RXI_DELETE_LOG_S table on the Siebel source system.

3. Set the values of the **OHSCDA.DELETE_FLOW** and **DELETE_FLOW** variables to **Y** while executing the load plan in ODI.
4. Execute the ETLs as listed in the Executing the ETL Load Plans section.

   The Siebel Clinical related SDE mappings read the above instance of the RXI_DELETE_LOG_S table.

Note: Records that are deleted in the source system are soft deleted in the data warehouse.

2.1.1 Adding a New Data Source

OHSCDA provides predefined source-dependent extract (SDE) mappings for InForm, Oracle Clinical, and Siebel Clinical. It also provides OCDA_CONTROL_S, OCDA_CONTROL_LOG_S, and W_RXI_DATASOURCE_S tables, and procedure/trigger to populate these tables for each SRO schema.

For InForm source, enter your sources' system-related information in W_RXI_DATASOURCE_S table in Section , "Adding Data Source Information" on page 2-5 where structure of w_rxi_datasource_s table is described. For other source systems, update the procedure which populates OCDA_CONTROL_S with unique DATASOURCE_NUM_ID (set this value to a number greater than 100).

Perform the following tasks to add a new data source:

1. Install OHSCDA in new source system (InForm, Oracle Clinical, or Siebel Clinical). For information, see *Oracle Health Sciences Clinical Development Analytics Installation and Configuration Guide*.

   This creates SRO, work schema, and related objects for the selected source system.

2. In ODI Topology, create a new physical connection pointing to the new source system.

3. Edit the context and change Physical Schemas to point to the new physical connection.

   For example, if the new source system is Oracle Clinical, edit the context CTX_OCDA_OC and change LS_OCDA_OracleClinical to point to the new physical schema.

Important:

- When calling an SDE mapping to read from a particular database instance, ensure that its corresponding SRO view has value of DATASOURCE_NUM_ID that corresponds to that database and ENTERPRISE_ID (normally 0).
- If you write new SDE, ensure that it sets the value of DATASOURCE_NUM_ID in the staging table to which it writes.

4. Make a copy of OCDA_W_RXI_LOV_S_seed.sql.

5. In the copy, modify the value of datasource_num_id to the value that you specified in the procedure that populates the OCDA_CONTROL_S table for the source database that you are adding.

6. Connect to the SRO account of the source.

7. Truncate W_RXI_LOV_S.

8. Run the modified copy of OCDA_W_RXI_LOV_S_seed.sql.

If deduplication is required for this source system, then perform the following steps:

1. Create an extraction program for full load that generates a flat file. Refer to the delivered code.

2. Create an extraction program for incremental load that calls the OHMPI APIs. Refer to the delivered code.
3. Add this new data source in OHMPI Projects file update.xml. For more information, refer to OHMPI documentation.

2.2 Executing the ETL Load Plans

OHSCDA ships the following load plans:

- **CDA - Complete Initial De Dup**: This is used to generate flat file required for OHMPI full load. For information on how to execute the CDA - Complete Initial De Dup load plan, see Section 2.2.1, "Executing CDA - Complete Initial De Dup Load Plan".
- **CDA - Complete Warehouse with Dedup**: This is used to load warehouse with one or all of the three source systems (Oracle Clinical, Siebel Clinical, and InForm) and with or without dedup flow. For information on how to execute the CDA - Complete Warehouse with Dedup load plan, see Section 2.2.2, "Executing CDA - Complete Warehouse with Dedup Load Plan".
- **CDA- Inform Automerge**: This is used to load warehouse only from Inform data source and only when automerge feature is needed. For information on how to execute the CDA- Inform Automerge load plan, see Section 2.2.3, "Executing CDA - Inform Automerge Load Plan".

2.2.1 Executing CDA - Complete Initial De Dup Load Plan

This load plan is applicable if you implement deduplication for your source systems.

To load data from the sources (Oracle Clinical, Siebel Clinical, and InForm) to flat files which are required for OHMPI full load, run the CDA - Complete Initial De Dup load plan packaged with OHSCDA. Perform the following tasks in ODI:

1. Click the **Operator** tab.
2. Select **Load Plans and Scenarios**.
3. Right-click the **CDA - Complete Initial De Dup** load plan and select **Run**.

   The Start Load Plan dialog box appears.
4. Select **CTX_OCDA** from the **Context** drop-down list.
5. Select **OCDA_LA** from the **Logical Agent** drop-down list.
6. Select **<Use Session Task Log Level>** from the **Log Level** drop-down list.
7. Assign startup values for the Load Plan Variables.
   - **OHSCDA.AUTO_MERGE_FLG**: Set value as **N**.
   - **OHSCDA.VAR_EMAIL_SUFFIX**: Set email suffix if any or null.
   - **OHSCDA.VAR_IF**: If InForm is your *one of the or only* data source(s), set value as **Y** to run load plan for the InForm source.
   - **OHSCDA.VAR_INT_CONFIG_DAYS**: Set numeric value (this variable is used to determine the extraction end date for the incremental load).
   - **OHSCDA.VAR_OC**: If Oracle Clinical is your *one of the or only* data source(s), set value as **Y** to run load plan for the Oracle Clinical source.

- **OHSCDA.VAR_SC**: If Siebel Clinical is your *one of the or only* data source(s), set value as **Y** to run load plan for the Siebel Clinical source.

8. Click **OK**.

> **Note:** This load plan is one time execution only. Make sure that the target folder does not have any files before running this load plan.

2.2.2 Executing CDA - Complete Warehouse with Dedup Load Plan

> **Note:** If deduplication for your source is required, then before running this load plan for Full load, run the **CDA - Complete Initial De Dup** load plan and load all flat files generated into the OHMPI projects. Then, run the **CDA - Complete Warehouse with Dedup** load plan in full load mode by assigning startup value of the VAR_OCDA_DWH_LOAD_FLAG variable as 1.

To load data from the source to their target tables in the data warehouse, run the CDA - Complete Warehouse with Dedup load plan packaged with OHSCDA. Perform the following tasks in ODI:

1. Click the **Operator** tab.
2. Select **Load Plans and Scenarios**.
3. Right-click the **CDA - Complete Warehouse with Dedup** load plan and select **Run**.

   The Start Load Plan dialog box appears.

> **Note:** ■The VAR_INT_CONFIG_DAYS variable is used to determine the extraction end date for the incremental load. For full load, set the value of this variable to 1. For incremental load, set the value of this variable to 0. This indicates that the source extraction end date is a day less than the ETL program run date. For example, if the ETL program run date is 28 July, 2010 the source extraction end date is 27 July, 2010.
>
> - The VAR_OCDA_DWH_LOAD_FLAG variable is used to determine full load or incremental load. For Incremental load, set the value of this variable to 0. For full load, set the value of this variable to 1.
> - Setting VAR_OCDA_DWH_LOAD_FLAG value to 1 while executing load plan truncates the complete warehouse drop indexes on the DWH table, loads data, and creates indexes.

4. Select **CTX_OCDA** from the **Context** drop-down list.
5. Select **OCDA_LA** from the **Logical Agent** drop-down list.
6. Select **<Use Session Task Log Level>** from the **Log Level** drop-down list.
7. Assign startup values for the Load Plan Variables.
   - **OHSCDA.AUTO_MERGE_FLG**: Set value as **N**.

- **OHSCDA.DELETE_FLOW**: Set value as **Y** to enable the Siebel Clinical deletion flow.
- **OHSCDA.VAR_IF**: If InForm is your *one of the or only* data source(s), set value as **Y** to run load plan for the InForm source.
- **OHSCDA.VAR_INT_CONFIG_DAYS**: Set numeric value (this variable is used to determine the extraction end date for the incremental load).
- **OHSCDA.VAR_OC**: If Oracle Clinical is your *one of the or only* data source(s), set value as **Y** to run load plan for the Oracle Clinical source.
- **OHSCDA.VAR_OCDA_DWH_LOAD_FLAG**: For full load, set value as **1** and for incremental load set value as **0**.
- **OHSCDA.VAR_OCDA_ERROR_REJECT_LIMIT**: This variable is used to set the number of rows that will be tracked in the respective error tables prior to aborting the ETL in case of errors.

  You must enter positive integer for this variable. For example, 0, 10, 100, 1000, and so on.
- **OHSCDA.VAR_SC**: If Siebel Clinical is your *one of the or only* data source(s), set value as **Y** to run load plan for the Siebel Clinical source.
- **OHSCDA.VAR_De_Dup**: If you have to execute the dedup flow, set value as **Y** to run the load plan with dedup flow.
- **OHSCDA.VAR_LOGFILE**: Set the log file path of SDE Dedup incremental procedures. For example, `/ohmpi_log/OCDALog.log`.
- **OHSCDA.VAR_AUTHFILE**: Set path of the ocda.properties file required for SDE Dedup incremental load. For example, `EE Agent/lib/ocda.properties`.

8. Click **OK**.

Note: Execution of the ETL (specifically the `OCDA_ETL_RUN_S_POP` program) populates W_ETL_RUN_S.LOAD_DT with the timestamp for the execution of the ETL. This ETL execution timestamp is used in the calculation of OHSCDA measures concerning the amount of time that currently open discrepancies have been open.

While the timestamp is captured in CURRENT_ETL_LOAD_DT, it is only available for calculation of discrepancy intervals through the OBIEE Dynamic Repository Variable CURRENT_DAY. CURRENT_DAY is refreshed from LOAD_DT at a fixed interval, by default 5 minutes, starting each time the Oracle BI Service is started. Between the time that the ETL is run, and the time that CURRENT_DAY is refreshed, calculations of intervals that currently open discrepancies have been open will be inaccurate.

The following are two remedies:

- Restart the Oracle BI Server after every execution of the ETL. This will cause CURRENT_DAY to be refreshed to the correct value.
- If this is inconvenient, you can modify the intervals between refreshes of the value of CURRENT_DAY. For more information on how to modify the refresh interval for CURRENT_DAY, see Maintaining the Oracle Health Sciences Clinical Development Analytics OBIEE Repository on page 1-1.

Tip: You can schedule the jobs to execute at regular intervals. For more information on scheduling jobs, refer to Scheduling an ETL Load Plan on page 2-13.

2.2.3 Executing CDA - Inform Automerge Load Plan

To load data from the source to their target tables in the data warehouse, run the CDA - Inform Automerge load plan packaged with OHSCDA. Perform the following tasks in ODI:

1. Click the **Operator** tab.
2. Select **Load Plans and Scenarios**.
3. Right-click the **CDA - Inform Automerge** load plan and select **Run**.

   The Start Load Plan dialog box appears.
4. Select **CTX_OCDA** from the **Context** drop-down list.
5. Select **OCDA_LA** from the **Logical Agent** drop-down list.
6. Select **<Use Session Task Log Level>** from the **Log Level** drop-down list.
7. Assign startup values for the Load Plan Variables.
   - **OHSCDA.AUTO_MERGE_FLG**: Set value as **Y**.
   - **OHSCDA.VAR_INT_CONFIG_DAYS**: Set numeric value (this variable is used to determine the extraction end date for the incremental load).
   - **OHSCDA.VAR_OCDA_DWH_LOAD_FLAG**: For full load, set value as **1** and for incremental load set value as **0**.
   - **OHSCDA.VAR_OCDA_ERROR_REJECT_LIMIT**: This variable is used to set the number of rows that will be tracked in the respective error tables prior to aborting the ETL in case of errors.

     You must enter positive integer for this variable. For example, 0, 10, 100, 1000, and so on.
8. Click **OK**.

2.3 Customizing an ETL Load Plan

When you customize an ETL Load Plan, it is your responsibility to maintain version control over changes to ETL mappings.

Oracle recommends that you carefully track the changes you make to Oracle-supplied ETL so that you can re-apply these changes in subsequent releases.

2.4 Creating an ETL Load Plan

Though OHSCDA includes ETL Load Plans for extracting data from InForm, Oracle Clinical, and Siebel Clinical to OHSCDA data warehouse, you may want to create your own ETL to extract data from other data sources.

Note: The value of DATASOURCE_NUM_ID is set to *1* for Oracle Clinical, *2* for Siebel Clinical, and *3* for InForm. If you want to add your own data sources, set this value to a number greater than 100.

See Also:

- *Oracle Data Integrator Online Help*

2.5 Modifying an ETL Load Plan

You may also want to modify an existing ETL to meet your reporting requirements.

See Also:

- *Oracle Data Integrator Online Help*

To modify an ETL load plan without any changes to the associated tables or columns, perform the following tasks:

1. Identify the load plan that needs to be modified in ODI.
2. Open and modify the ETLs.
3. Test and save the changes in repository.
4. Re-generate the scenario with the same scenario version.
5. Navigate to the load plan.
6. Regenerate and execute ETL to verify the changes.

> **Note:** The ETL Load Plans that extract data for the warehouse fact tables assume that the dimensions to which each fact is related are up-to-date at the time the fact ETL Load Plans are executed. This assumption is the basis for certain fact calculations that would provide erroneous results if the assumption were not true. For example, in the *received CRFs* fact, the value of the *pCRF entry complete measure* depends on whether or not the study requires second pass entry. But that piece of information -- second pass entry required -- is obtained from an attribute of the Study dimension. So, if the second-pass requirement for a study changes, and the change is not applied to the Study dimension, the Received CRF fact attributes will contain incorrect values.
>
> As shipped, OHSCDA ETL load plan ensure this interlock by executing the ETL for related dimensions immediately before running the ETL for a fact. This is standard warehouse management practice, but especially important given the interdependence of the dimensions and the fact. The need to execute dimension ETL immediately before corresponding fact ETL, and the danger of not doing it, is emphasized here because it is possible (though discouraged) to modify these shipped load plan.

To modify one or more tables or columns without any changes to the associated ETL programs (typically to widen a column):

1. Change the table properties as needed.
2. Refresh and validate the mapping and regenerate the scenario of the existing version.
3. Navigate to the load plan and verify the scenario to which it is referred.

Note: Ensure that changes to the tables or columns are compatible with the table that is installed in the data warehouse schema. For example, if you are reducing the length of a number column from 15 to 10, ensure that the change is compatible with the existing data in the table.

2.6 Scheduling an ETL Load Plan

When you submit a Load Plan for execution in ODI, you can schedule it to execute at regular intervals. To schedule a Load Plan, perform the following tasks:

1. Select **Load Plans and Scenarios**.
2. Expand the **CDA - Complete Warehouse with Dedup** load plan.
3. Right-click **Scheduling** and select **New Scheduling**.
4. Select **Definition**.
5. Select the required values for the following:
   - **Context**
   - **Logical Agent**
   - **Log Level**
   - **Status**
   - **Execution**
6. Click **Save**.

3
Multi-Source Integration

This chapter contains the following topics:

3.1 Overview

Multi-source integration provides a mechanism for identifying duplicate dimension value records, merging them into a single record, and adjusting fact record foreign keys accordingly. All of this can be done when loading the OHSCDA warehouse from multiple transactional databases.

The purpose of multi-source integration is to permit data to be loaded into the OHSCDA warehouse from two or more source databases, while providing means to ensure that duplicate records across the sources are represented by single records in the warehouse. For instance, every database used as a source for OHSCDA will have a Studies table. If you load data from two source databases, and both have an entry for the same investigator (for example, Joseph Smith), it is desirable that this investigator be represented only once in the Investigator dimension in the warehouse.

There are two reasons why deduplicating dimension data is important:

- If it is not done, the duplicated data will be displayed as multiple separate rows in OBIEE prompts, which are used to dynamically filter the data to be displayed in reports. For example, there would be two rows in the Prompt drop down list for Joseph Smith.

Investigators
Andy Jones

Investigators
Claudine Roberts
.....
Joseph Smith
Joseph Smith
.....

A user wanting to see all the data for that Investigator would have to select both rows. Multiple selections might not always be possible).

- The duplicated data will result in multiple rows in reports where there should only be one row. For instance, suppose a report asks for Number of Queries by Investigator. Assume that database 1 records 20 queries for Joseph Smith, and database 2 records an additional 30 queries for Joseph Smith. Presume that these are distinct queries. Then, if no deduplication was done, the result of the query would be

Investigator	Number of Queries
Joseph Smith	20
Joseph Smith	30
.....	.....

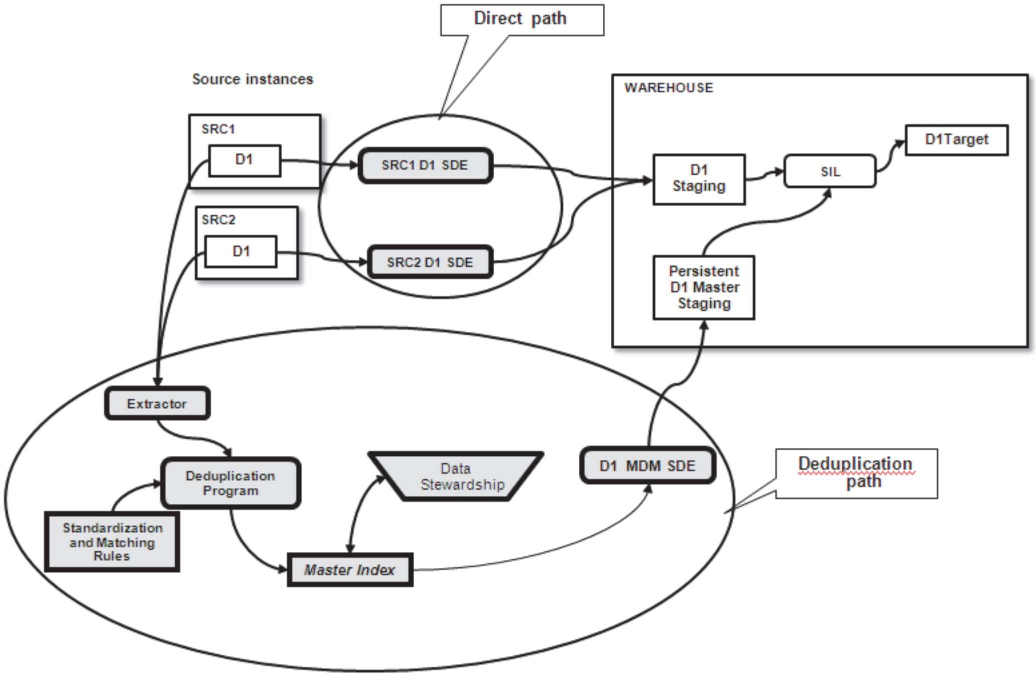
To arrive at the final number of queries, you will have to take the sum of the two rows.

Deduplication of dimension data eliminates both these problems from the presentation of the data in the dashboard.

Note: It is necessary to identify which records are duplicates, before deduplication can be performed. For instance, determining whether Joe Smith and Joseph Smith are two different people, or are the same person, is a matter of identification. This must be performed by a person with the necessary knowledge and to a certain extent this process can be embedded in rules. The details of the process are described below.

Figure 3–1 illustrates how one dimension is loaded when using OHSCDA's multi-source integration capability.

Figure 3–1 OHSCDA Multi-Source Integration Paths



Data for the dimension flows into the warehouse by following two paths:

- Direct path - This is the path by which data is always loaded, whether or not it needs deduplication.
- Deduplication path - This is a supplementary path that supports the identification and confirmation of matches, and the loading of that information into the warehouse.

The two paths converge during the source independent load (SIL) execution of the dimension. The SIL applies the results of deduplication that arrived through the deduplication path to the complete set of data that arrives through the direct path. When SIL execution completes, the deduplication information has been applied to the warehouse table for the dimension. For every set of duplicates that has been identified, there is only one record in the warehouse that will be accessible by queries from OBIEE.

> **Note:** OHSCDA retains all the records in the duplicate set, but marks them as merged. OHSCDA creates a single best record in lieu of them, and it is this record which is accessible to OBIEE queries.

Following is the sequence of the loading process when it includes deduplication:

1. The deduplication system performs all deduplication, provides all attributes, and captures linkages between source records and result records.
2. The Data Steward makes decisions about potential matches not automatically resolved by rules. Until this happens, such records are treated as singletons.
3. An SDE mapping that knows how to read the dimension from the Master Index does so, writing records to a persistent staging table.

4. On the Direct Path, the source-specific SDE mappings for the dimension executes, populating the dimension staging table with all the contributor records (that is, the raw materials for deduplication).
5. On the Direct Path, the SIL writes the contributor records to the dimension target table.
6. Turning its attention to the Deduplication Path, the SIL reads the Persistent Master Staging table. If the SIL finds any new records there (telling what records are to be considered duplicates), it adjusts the dimension target table so that those sets of duplicates already in the target table are reduced to single records.

> **Note:** All the records stay in the target, but the ones composing a group of duplicates are marked as merged, while a new Single Best Record representing the whole set is created in the target table. The merged records are excluded from OBIEE queries, so in effect there is only one record in the warehouse for the set.

3.1.1 Foreign Key Adjustment

If duplicate dimensions are merged into a single representative record in the warehouse, the ETL process must also adjust the foreign keys of fact records in the warehouse so that they point at the correct dimension record.

In the source databases, records that contribute to warehouse facts have foreign keys to source dimension table records. For instance, suppose that in source 1 there is a record describing a query sent to an investigator. The identity of the investigator will be specified by the value of the investigator foreign key in the queries table. The value will match the value of the investigator table primary key for the relevant investigator.

If no deduplication is applied to the investigator while extracting records to the warehouse, the following sequence occurs:

1. Records from the investigator table are loaded into the warehouse dimension table. Each record is given a new, warehouse-specific, primary key value in the Row_wid column. The primary key value of the record in the source database is retained in the record's integration_id column. So the Investigator table in the warehouse will be as follows:

Row_wid	Investigator	Integration_id
1	Andy Jones	101
2	Claudine Roberts	102
...	...	...
13	Joseph Smith	113

2. Fact records are loaded next. As each fact record is entered into its warehouse target, its foreign key value is set for each dimension. A query fact table would start out like this:

Deduplicating a dimension involves reducing each set of duplicates across the various databases to single representations of each unique entity.

3.1.2 Unit of Work

Deduplication is performed separately for each dimension depending on whether they are ascertained to contain duplicates. For example, you might have two instances of Oracle Clinical, one for studies for Product A and another for studies for Product B. Then you could load the Study dimension from both databases into OHSCDA without need for deduplication. However, if some of the same Investigators were used for studies in both databases, then you would have duplicates on the Investigator dimension across the databases. In this case, you will have to deduplicate the Investigator dimension.

3.1.3 Necessity of Deduplication

In general, deduplication of a dimension is required if you know (or suspect) that there are duplicates in the dimension, that is, there are two records standing for the same entity instance.

In general, a dimension loaded from two or more databases is a candidate for deduplication. However, this cannot be assumed. Deduplication of a particular dimension is not needed if you are confident that there are no duplicate records in the source tables for that dimension in the source databases. For example, if one database is used only for Product A, and another only for product B, there is no need to deduplicate the Product dimension when loading data from those two databases.

Likewise, if you are loading data from only one database, you probably do not require multi-source integration. However, there is a possible exception to consider - if your single source database itself contains duplicates on a dimension, you could use multi-source integration to manage those duplicates. For example, suppose your list of investigators includes both Joe Smith and Joseph Smith, both the same person. These will give rise to multiple rows in prompts and reports, when there should be only one. You can clean this up by correcting the source database, which entails redirecting all foreign key references from Joe Smith to Joseph Smith, and then deleting the entry for Joe Smith. Or you can use OHSCDA's multi-source integration, which allows you to produce the same effects.

3.1.4 Coordinated Dimensions

You may have undertaken to ensure that values for a dimension are coordinated across source databases. Coordination typically means that there is no Investigator in Source 2 that is not also present in Source 1, and that the name of each investigator is spelled identically in both databases. Depending on the nature of the coordination procedure, it may also be the case that the set of investigators in both databases is identical - every investigator in Source 1 also appears in Source 2, and vice versa.

This coordination can be accomplished by several means:

- A standard operating procedure that is carefully followed when creating Investigator names.
- A procedure under which you create new Investigators only in Source 1, and they are programmatically propagated to Source 2 (for example, by Oracle AIA).
- Both databases being populated from a third table that is designated as the single gold source, for example, through use of a Master Data Management tool.

If you are loading the Investigator dimension in OHSCDA from two source databases where Investigator has been coordinated, it is necessary to deduplicate the dimension when loading its values from the two (or more) source databases. Even though there is no uncertainty about whether Joseph Smith in Source 1 is the same person as Joseph

Smith in Source 2, if you simply load the Investigator dimension from both sources into the OHSCDA warehouse, you will end up with two entries in the Investigators table for Joseph Smith, with the resulting problem -- multiple rows in prompts and reports.

If you want to forestall these problems, deduplication is needed. However, deduplication is a two-step process: first, rules are followed to identify actual and potential matches; then a data steward determines whether potential matches are to be treated as actual matches or non-matches. If data for a dimension has been coordinated, then the rules will never identify any potential matches that require stewardship. This will substantially simplify the effort to perform initial and ongoing deduplication.

> **Note:** If you have used a Master Data Management (MDM) system to coordinate values in a dimension, you may want to use that MDM system in place of the OHMPI project supplied by Oracle for that dimension. For more information, refer Section 3.3, "Processes for Using Oracle Healthcare Master Person Index Deduplication Projects" on page 3-15.

All descriptions of deduplication in this document describe the process for one dimension. Your decisions about whether to use multi-source integration, OHMPI or another deduplication program, and what identification rules suit your data, will have to be made for each of the dimensions for which deduplication multi-source integration is supported. Table 3-1 lists the dimensions for which OHSCDA provides multi-source integration support.

Table 3–1 Warehouse Dimensions Supported by Multi-Source Integration

Warehouse Table
W_EMPLOYEE_D
W_GEO_D
W_HS_APPLICATION_USER_D
W_LOV_D
W_PARTY_D
W_PARTY_ORG_D
W_PARTY_PER_D
W_PRODUCT_D
W_RXI_CRF_BOOK_D
W_RXI_CRF_D
W_RXI_PROGRAM_D
W_RXI_SITE_D
W_RXI_STUDY_D
W_RXI_STUDY_REGION_D
W_RXI_STUDY_SITE_D
W_RXI_STUDY_SUBJECT_D

Table 3–1 (Cont.) Warehouse Dimensions Supported by Multi-Source Integration

Warehouse Table
W_RXI_VALDTN_ PROCEDURE_D
W_USER_D

3.1.5 Layering and Options

OHSCDA's multi-source integration capability is layered on top of its Direct Path loading capability. The Direct Path loads all records from the source and does not have any knowledge of whether records are duplicates of one another.

The purpose of the Deduplication Path is to provide a way to communicate to the dimension's SIL that certain records loaded through the Direct Path are to be treated as duplicates. The SIL then applies that information to the records that it has loaded into the target warehouse table, reducing the designated duplicates to a single representative, and adjusting fact foreign keys accordingly.

The Deduplication Program allows you to use a combination of stored rules and human judgment to identify which records are duplicates and to determine what values should go into the warehouse record that consolidates those duplicates.

All dimensions have a Direct Path, but you can choose which of your dimensions are to be passed through the Deduplication Path.

3.1.5.1 Oracle Clinical Data Analytics and Oracle Healthcare Master Person Index

OHSCDA has been designed to work with Oracle Healthcare Master Person Index (OHMPI) as its deduplication program. OHSCDA provides all the files required to integrate OHMPI into its Deduplication Path. You can, however, use another deduplication program to serve this role.

3.1.6 Intersection of Deduplication Paths

The Direct Path and the Deduplication Path intersect at the SIL for the dimension. This is the program that reads the data from the Staging table, does the necessary transformations on it, and writes the dimension data to its warehouse table.

In OHSCDA, the SIL for each dimension does an additional task, which is to incorporate data from the Deduplication Path for that dimension. The SIL always checks to see if there is anything new in the Persistent Staging table for the dimension. If the SIL finds anything new in the Persistent Staging table, it applies this deduplication information to the dimension's target table. If you are not using deduplication for the dimension, nothing will show up in the Persistent Staging table, and the Direct proceeds unchanged.

3.1.7 Initial Load and Incremental Load

Deduplication applies to the initial load and to subsequent incremental loads.

In the Direct Path, there is little difference between the initial load and incremental loads. The differences are that indexes on warehouse tables are dropped, tables are truncated, and the starting date is set to the earliest date for which data is to be loaded, before an initial load. In incremental loads on the Direct Path, the ETL loads information only from records that have been created, changed, or deleted since the last ETL execution.

In the Deduplication Path, there is a marked difference between initial and incremental loads. For the initial load, you must initially create the Master Index for each dimension. To create a Master Index, perform the following steps:

1. Extract the dimension data from the various source databases into a flat file. OHSCDA provides an ODI Execution load plan for doing this.
2. Profile the data for the dimension. This gives insight into patterns and groupings in the data. This may lead you to adjust the pre-defined rules for identifying matches.
3. Cleanse the dimension data by passing it through filters that identify records which, left unchanged, and would fail to be processed by the OHMPI deduplication program.
4. Run the Bulk Match by passing the data through the deduplication program, and get a report that indicates which records would be considered assumed matches. This again may lead you to adjust the pre-defined rules for identifying matches.
5. Run the Bulk Load. In this step, the rules are applied and the results are placed in the Master Index. This completes the initial load for the dimension.

Incremental load on the Deduplication Path is more automatic. If you've configured OHMPI and OHSCDA to run the Deduplication Path for a dimension, then OHSCDA will perform the following tasks whenever a job is executed for an incremental load.

1. OHSCDA will gather information about any new duplicates from the Master Index for the dimension.
2. OHSCDA runs the incremental load along the Direct Path.
3. OHSCDA adjusts the dimension target table so that the newly identified sets of duplicates already in the target table are merged into single records. If you have elected to unmerge records since the last execution of the SDE for the dimension, OHSCDA will adjust the dimension target table accordingly.

Note: Merging records implies that a new Single Best Record is created, as indicated by your decisions in the OHMPI deduplication program, and the contributing records are marked as merged, meaning that they are invisible to queries from OBIEE.

Incremental deduplication has an additional activity. If your match rules are set up so that the deduplication program can create potential matches, and such potential matches are identified, those potential matches have no immediate effect on the OHSCDA warehouse tables. Each of the records in a potential match is treated as if it were unique. A Data Steward must use the OHMPI Master Index Data Manager (MIDM) for the dimension to inspect each potential match, and decide how to deal with it. If the Steward has decided that it is indeed a match, then the erstwhile potential-match records become part of an assumed match. The next time the SIL for the dimension is executed, OHSCDA learns of the new assumed match, and makes the appropriate changes to the warehouse table.

3.1.8 Oracle Healthcare Master Person Index Deduplication Process

The OHMPI Deduplication process consists of a Match Engine, which carries out Matching Rules to decide which input records are duplicates. Results of the decisions are stored in the Master Index. This section provides a conceptual introduction to the Match Engine, Master Index, and the Matching Rules.

3.1.8.1 Match Engine and Master Index

The Match Engine is the part of OHMPI that applies the Match rules to evaluate whether incoming source records are duplicates of records already in the Master Index. For more information, refer to the *Oracle Healthcare Master Person Index Data Manager's Guide*.

The Master Index consists of all source records that have already been matched against one another. Each source record in the Master Index is a member of a Profile. Each Profile is a set of source records that have been determined to represent the same entity in the dimension. A Profile may consist of one or more source records. In addition to its source records, each Profile also has an additional record called its Single Best Record (SBR). The SBR is the representative for the Profile. Its attribute values are set to be the best available from across the contributing source records in the Profile.

The Match Engine receives each new source record from a queue. Each queued record is compared against the Master Index, as follows:

The Match Engine looks among the Profiles in the Master Index for the closest match. Matching is based on the values of the key fields that have been defined for the dimension. The Engine identifies which Profile (if any) in the Master Index is most similar to the new source record. This comparison is done by computing a weight for the similarity of the incoming record's key values to the corresponding values of the Profile's SBR, and then summing those weights. The greater the similarity, the higher the summed weights.

While the actual algorithm for doing it is more efficient, the Engine behaves as if it compares each record against every Profile in the Master Index. At the end of this, it will have identified one existing Profile that is most similar to the incoming record, unless the record under consideration is the initial one loaded into the Master Index.

Having identified the likeliest-match Profile, the Engine places the incoming record into one of three categories relative to this likeliest-match Profile, based on the summed similarity weight. The summed weight for the comparison is compared against two thresholds. These thresholds are *Match* Threshold and the *Duplicate* Threshold. The following three scenarios are possible:

- If the summed weight is equal to or greater than the Match threshold, the incoming record is considered to represent the same entity as the likeliest-match Profile does. In this case the record is an assumed match, and is added to the Profile in the Master Index.
- If the sum of the weights is equal to or greater than the Duplicate threshold, and less than the Match threshold, the Match Engine can only conclude that the incoming record *may* describe the same entity as the likeliest-match Profile does, but human judgment is needed to make a determination. In this case, the record is marked as a potential match to the Profile.
- If the sum of the weights is less than the Duplicate threshold, the incoming record is deemed to not describe the same entity as the likeliest-match Profile. Therefore it is a non-match to that Profile (and a non-match to all other Profiles, since they had already been determined to be less similar to it that the likeliest-match Profile). Therefore, the incoming record represents a new entity, and it becomes the first source record in a new Profile.

3.1.8.2 Matching Rules Using Project Configuration

The Match Engine makes its decisions based on several Configuration parameters. These are collectively called the Matching Rules for the dimension. There are two global configuration parameters in a Project:

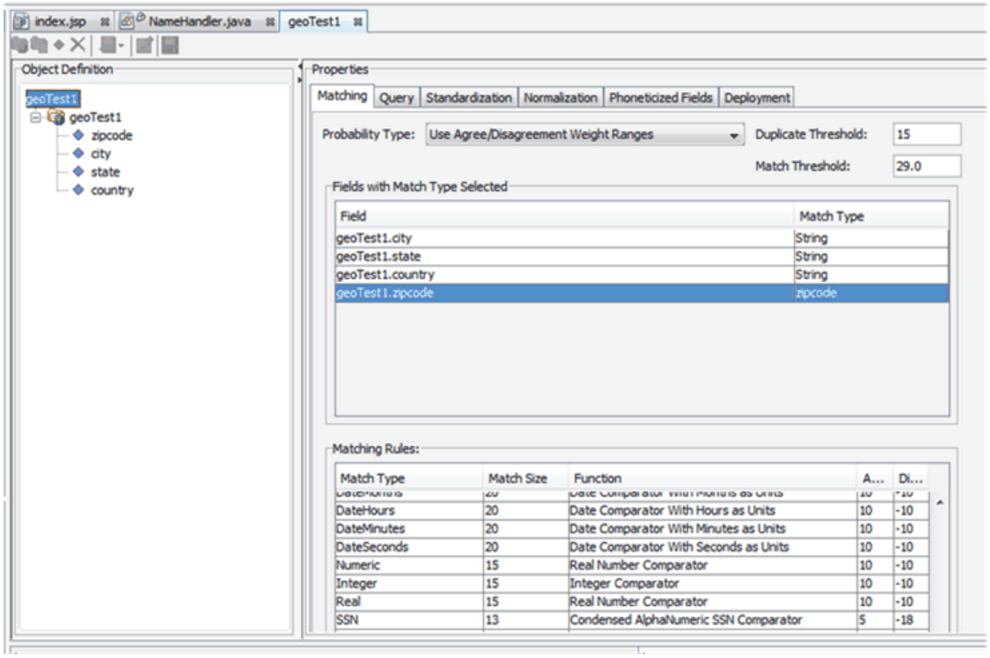
- Duplicate Threshold - a record must have a sum of weights greater than this to be considered a potential match to a Profile in the Master Index.
- Match Threshold - a record must have a sum of weights greater than this value to be considered an assumed match to a Profile in the Master Index.

The Configuration also identifies which fields in the incoming records are the keys. For each key, the Configuration determines the MatchType to be used to compare the value of the incoming record to the corresponding value in the likeliest-match SBR. The MatchType in turn determines:

- The algorithm used to compare the values
- The range of weights to be given, depending on how different or similar the values are. This range is bounded by a Disagreement weight and an Agreement weight. The weight given to a particular comparison can fall between these endpoints, depending on the MatchType's algorithm
- The weight to be given to the comparison in the event that one or both of the fields being compared is null or an empty string

While OHMPI provides numerous pre-built MatchTypes, it is also possible to define new ones as needed for particular match fields. All the parameters in a project's Configuration can be adjusted through the Project Configuration Screen. Figure 3–2 shows the configuration screen for a project.

Figure 3–2 Project Configuration Screen



Probability Type: Use Agree/Disagreement Weight Ranges

Duplicate Threshold: 15

Match Threshold: 29.0

Fields with Match Type Selected

Field	Match Type
geoTest1.city	String
geoTest1.state	String
geoTest1.country	String
geoTest1.zipcode	zipcode

Matching Rules:

Match Type	Match Size	Function	A...	Di...
DateMonths	20	Date Comparator With Months as Units	10	-10
DateHours	20	Date Comparator With Hours as Units	10	-10
DateMinutes	20	Date Comparator With Minutes as Units	10	-10
DateSeconds	20	Date Comparator With Seconds as Units	10	-10
Numeric	15	Real Number Comparator	10	-10
Integer	15	Integer Comparator	10	-10
Real	15	Real Number Comparator	10	-10
SSN	13	Condensed AlphaNumeric SSN Comparator	5	-18

For more information, refer to the *Oracle Healthcare Master Person Index Match Engine Reference*.

3.1.8.2.1 Special Handling for Null Value in Key Fields

Note: Special handling is required if either of the records being matched has a null value for any of these fields.

Normally, the keys used in matching not null columns in the source database are mandatory. Geography represents an exception to this rule. Because you draw GEO records from information about investigators and sites, the source systems do not require zipcode, city, state, and country for each investigator and site. This requires special handling for deduplication of the geography dimension.

If there is a match on zipcode and any other field in the preferred record is null, OHMPI processing results in a discrepancy between the integration ID of the record loaded on the direct path and the integration ID of the record loaded through the deduplication path. This is because OHMPI, while creating the SBR, automatically replaces null with the value of the key from the contributor record. This is contrary to our general rule that no attribute merge of integration ID keys is permitted. When the rule is broken, the integration key of the SBR is no longer identical to that of the preferred source record, and adjustment of foreign keys to point to the SBR fails.

You must prevent OHMPI from merging attribute of Geo key values.

1. Clean up the source data in the preferred source so that it contains as few null values as possible for the key Geography fields.

   This forestalls any attribute merge of key values. It is not essential to clean up nulls in the non-preferred source.

2. Put into effect an SOP that requires that when a new address is added to a source table in the preferred source, the values of all four keys must be entered.
3. Override OHMPI's automatic swapping of values from the contributor source if null values still remain in the preferred source. To do so, Data Steward must be involved in MIDM.

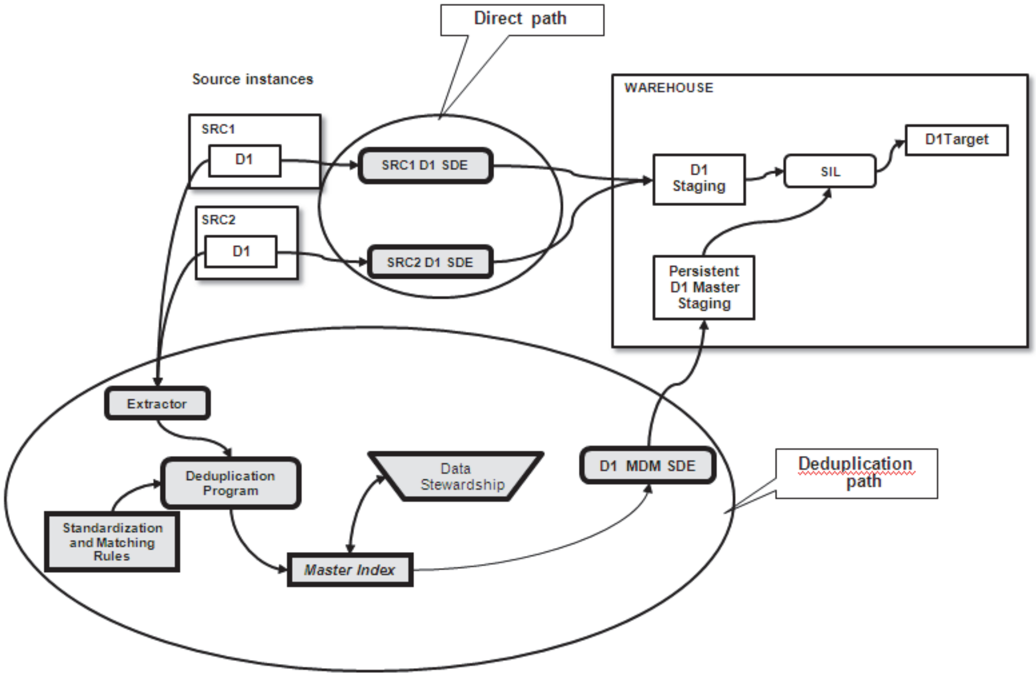
You can use one approach for few records, and another approach for other records. The values of Zipcode, City, State, and Country in the SBR must be same as their values in the preferred record for the SBR, including null values, if any.

If you perform these actions, there will be no geography dimension SBRs with invalid integration IDs.

3.1.9 Components of the Deduplication Path

This section defines each of the components in the deduplication path for one dimension. Each dimension will have its own copy of these components.

Figure 3–3 Deduplication Path



3.1.9.1 Bulk Extracting, Cleansing, and Loading

This is used during Initial load only. It represents a set of activities that you must perform to prepare your existing data, and then call the deduplication program to load them into the Master Index. For more information, refer to Section 3.3.6, "Initial Load Processes" on page 3-17.

3.1.9.2 Extractor

The extractor is used during Incremental loads only. This program determines which dimension record is new, has changed, or has been deleted since the last execution of the SDE. For each such record, it calls the API provided by the deduplication program, asking it to apply the match rules to this record, comparing it to all records already present in the Master Index for the dimension. The deduplication program makes the comparison, and takes one of the following actions:

1. If the record is determined to be a duplicate of a record in the Master Index (if it is an assumed match), the record is marked as a contributor to the SBR for that set of duplicates.
2. If the record is deemed to potentially match one or more records in the Master Index, it is marked as a potential match, awaiting a final decision by the Data Steward.
3. If the record is determined to not match any of the records in the Master Index, it is placed in the Master Index as a non-duplicate.

3.1.9.3 Deduplication Program

The Deduplication Program is the dimension-specific body of code (an OHMPI Project) that provides the capability to match an input record against the contents of the dimension's Master Index. It consults its match rules for the dimension, and estimates whether the record is an assumed match, a potential match, or a non-duplicate.

3.1.9.4 Matching Rules Specification

Matching Rules determine which records are assumed matches, which are potential duplicates, and which are non-duplicates. OHMPI provides an interface for specifying these rules.

3.1.9.5 Data Stewardship

This represents the activity of using the Master Index Data Management (MIDM) web application to review potential matches and decide their fate. Each dimension that is deduplicated has its own MIDM application.

3.1.9.6 Master Index

The Master Index is a database schema that holds all of the records for the dimension that have been processed by the deduplication program. It has attributes by which each record is identified as being either part of match, a potential match, or a non-duplicate.

> **Note:** Every record in the Master index is a member of an Object Profile. Each Object Profile represents one dimension entity, and has a Single Best Record to represent the Profile. Some Profiles have one contributor record, others have multiple contributor records. For more information on of Object Profile concepts, refer to the *Oracle Healthcare Master Person Index Data Manager's Guide*.

3.1.9.7 Dimension MDM SDE

The Dimension MDM SDE is a query that reads from the Master Index table for the dimension, and writes to the Persistent Master Staging table for the dimension. This query selects only records describing assumed matches and potential matches confirmed by the Data Steward. Of these, the query selects only the records that have been created since the last execution of the SDE. As a result, it updates the Persistent Master Staging table with new deduplication information that must be acted upon by the SIL.

3.1.9.8 Persistent Master Staging Table

The Persistent Master Staging Table accumulates all merge decisions defined by the deduplication program for the dimension. This information is used by the dimension's SIL to apply that merge information to the data in the dimension warehouse table, thus reducing each set of duplicates to a representative Single Best Record.

3.2 Preliminaries to Using Oracle Healthcare Master Person Index Deduplication Projects

This section describes how to carry out the processes required using the deduplication path and the starting point from which you begin those processes. That starting point is the state of the OHSCDA system after the Installation (or Upgrade) process has completed.

3.2.1 Installation Results

The process of installing or upgrading OHSCDA 3.1.1 provides all the pre-built objects required for performing deduplication with OHMPI.

> **Note:** Some of these items are installed by you, as pre-requisites to the OHSCDA installation. Others are installed as part of the OHSCDA installation itself.

Some of the items are packaged per-dimension. These are:

- Two common Execution Plans one for full load and other for Incremental load. This includes extraction logic for all dimensions.
- OHMPI Project files structure for the dimension. This includes the match rules, configurations, and the files needed to install the MIDM application in WebLogic. Each dimension's file structure is a folder under a root NetBeansProjects folder.
- Persistent Staging table for the dimension
- SIL that reads both Direct and Deduplication Paths
- A database schema to hold the Master Index for the dimension

You can enable or disable the De-dup loading in the CDA - Complete Warehouse with Dedup load plan.

This set of components represents the starting point for carrying out the processes described below.

3.2.2 Oracle Healthcare Master Person Index File Structure

In order to carry out these processes, it is helpful to have an understanding of where OHMPI files will be found.

The files pertaining to a dimension are maintained in a Project. Each Project is rooted in a folder with a name corresponding to the dimension it loads. These are called project folders. All project folders are contained within a folder named NetBeansProjects.

A project has a fairly deep and complex folder structure. However, for normal purposes, you need to open files in only a few of these folders. These are:

Table 3–2 OHMPI Folders

Folder	Description
cleanser	Work in this directory to do cleansing of the extracted, profiled data prior to bulk load
loader	Work in this directory to do bulk matching and loading of the cleansed data. Also known as the Master IBML Tool home directory.
mdm	This directory is also used by the Bulk Loader. It is referred to in the OHMPI documentation as the working directory. *This directory is not automatically created when you create a Project. You must create it, and place it in the location specified in the loader-config.xml file.*
profiler	Work in this directory to do profiling of the extracted data prior to cleansing.

Table 3–2 (Cont.) OHMPI Folders

Folder	Description
src\Configuration	This directory contains the configuration files for the Project. These files can be edited or viewed here, but it is preferable to view and modify them under the Configuration folder in the NetBeans IDE representation of the Project.

3.3 Processes for Using Oracle Healthcare Master Person Index Deduplication Projects

This section provides descriptions of processes that must (or in the case of optional processes, may) be carried out when performing deduplication using OHMPI. This section describes processes in terms of what you do for one dimension. You will need to repeat these processes for each dimension that you elect to deduplicate using OHMPI. If you use a different deduplication system, the names of the processes will vary, but the tasks will essentially remain the same.

There are three classes of processes: project configuration, initial load, and incremental loads.

3.3.1 Adding Sources to the Project

For each source database, add a processing code for the database. See *Oracle Healthcare Master Person Index User's Guide*.

3.3.2 Adjusting Project Configuration

Project configuration sets the values of the parameters that determine how the project processes the data for the dimension. For more information, refer to Section 3.1.8.2, "Matching Rules Using Project Configuration" on page 3-10.

Oracle provides pre-defined configuration settings for each dimension project we ship.

> **Note:** These pre-defined configurations are intended to be a starting point only. They represent a set of assumptions about how you might want the dimensions processed. However, they are generic, and not tuned to your specific data. It is essential that you review and adjust these configuration settings so that they are appropriate to your data.

You should adjust the configuration so that they provide the right balance of assumed, potential, and non-duplicates, given what you know about your data. You should adjust the configuration so that assumed matches occur only for real duplicates, and avoid false assumed matches. To do this, adjust weights and the match threshold such that only records you know to be identical will have weights equal to or greater than the match threshold.

You should also adjust the configuration to avoid creating non-matches where the records are really duplicates. Do this by adjusting weights and the duplicate threshold such that only records you know to be unique will fall below the duplicate threshold.

Oracle urges you to carefully read the OHMPI documentation on this topic, and to work carefully on adjusting each dimension's configuration so as to avoid mis-identifications.

OCDA's deduplication capability has been designed so that you can make some adjustments after the fact: you can merge two profiles into one, and CDA will make the corresponding adjustment in the next execution of the dimension's SIL.

While OHSCDA lets you make these adjustments, the effort needed to accomplish them in the MIDM is complex. The only adjustment that is simple in the MIDM is that of resolving a potential match, either to a merge or to two separate profiles. Therefore, Oracle strongly recommends that you configure your projects such that, when there's any doubt about the right disposition, it creates potential matches.

In pre-defining configurations for projects, we have tried to define rules in this manner, but you must review and refine them in light of your knowledge of your data.

3.3.3 Promoting an Attribute to Being a Match Field

One configuration change that you may want to perform is to take an attribute that is part of the SBR for a record, and add it to the set of Match Fields for the dimension's Project.

For example, the shipped Project for the Study dimension has only one match field, STDY_NM. If your study naming conventions let you use the same study name in different Programs, you would want to add Program as a match field in the Study Project.

Since Program is an attribute of the Study dimension, it is already included in the SDE that extracts the Study dimension data. It can serve as a match field. Perform the following to define it as a match field:

1. Create a new Match Type for Program if none of the pre-defined strings have the correct size, comparator, and agreement and disagreement weights.
2. In the field's properties, change its Match Type from None to the desired Match Type.
3. Include the Program Name as part of block query in query.xml.
4. Adjust the duplicate and match threshold values to take the new match field's weight into consideration. The simplest change is to increase both thresholds by the agreement weight of the new match field. You should also consider what impact of agreement or disagreement on this field you want to have on the match outcome.
5. Clean and build the Project.
6. Save a copy of the current Cleanser directory and regenerate the Cleanser.
7. Add Program to the cleansing rules.
8. Regenerate the Loader.
9. Empty the Master Index tables, if they have been populated.
10. Run Bulk Match in Analysis mode to confirm that Program now differentiates.
11. Deploy the Project on your Application Server.

3.3.4 Preparing the Master Index Database Schema

Refer to *Master Person Index Database* in the *Oracle Healthcare Master Person Index User’s Guide*. Follow the instructions in this guide to create and seed the database schema that will hold the Master Index for the dimension.

3.3.5 Generating and Deploying the Master Data Index Manager Application

The MIDM is the web application that allows the Data Steward to evaluate potential matches, and inspect the properties of all Profiles in the Master Index. Refer to *Oracle Healthcare Master Person Index WebLogic User's Guide* and *Oracle Healthcare Master Person Index User's Guide, Generating the Master Person Index Application* to generate and deploy the application to the WebLogic Application Server.

3.3.6 Initial Load Processes

3.3.6.1 Extract

This process extracts data from the source databases for use in initial load. The output of this process is a flat file that conforms to the OHMPI specification. This file is used as input during the Profile, Cleanse, Bulk Match and Load Processes.

For more information on the format of the file, refer to *Oracle Healthcare Master Person Index User's Guide*.

Use the ODI load plan, CDA - Complete Initial De Dup, for extracting the flat files for cleansing.

3.3.6.2 Profile

This process is optional. Consult the reference noted below to determine if profiling is useful for your data. Profiling gives insight into patterns and groupings in the data. It takes an OHMPI-conformant flat file as input, and yields reports on patterns and groupings. For more information, see *Oracle Healthcare Master Person Index User's Guide*.

The basic steps of this process are:

1. Generate and unzip the profiler directory for the project.

> **Note:** You must grant recursive write privileges on the profiler directory to the current user.

2. Make a copy of sampleConfig.xml.
3. Adjust your config.xml to specify the desired reports.
4. Extract data from the source databases into an OHMPI-conformant flat file.
5. Edit run.bat to execute your sampleConfig.xml.
6. Run run.bat.
7. Review reports. Determine what changes in matching rules, or addition to matching rules, are necessary given the profile of the data for the dimension.
8. Modify the source data based on conclusions drawn from reports.
9. Repeat steps 5-8 until data is satisfactory.

The output of the Profile process is an OHMPI-conformant flat file that satisfies all profiling requirements.

3.3.6.3 Cleanse

This process is optional, but strongly recommended. Cleansing identifies those records in the OHMPI-conformant flat file that will fail to pass successfully through the bulk loader. The cleansing process takes an OHMPI-conformant flat file as input, and yields two output files. By default they are named good.txt and bad.txt. Records in good.txt will be acceptable to the bulk loader engine; Records in bad.txt will fail to be processed by the bulk loader. The goal of the cleansing process is to iteratively clean the source data until cleansing the extracted flat file produces no rejected records. For more information, refer to Rule three in Section 3.5.1, "Rules" on page 3-22.

The basic steps of this process are:

1. Generate and unzip the cleanser directory for the project. You can create this directory from NetBeans IDE.

   > **Note:** You must grant recursive write privileges on the cleanser directory to the current user.

2. Make a copy of sampleConfig.xml.
3. Adjust your config.xml to specify the desired reports.
4. Extract data from the source databases into an OHMPI-conformant flat file.

   All hash (#) character occurrences which are not prefixed by tilde (~) should be prefixed by tilde (~) in the flat file.

   All new line characters within a given record should be removed from the flat file.

5. Edit run.bat to execute your sampleConfig.xml.
6. Run run.bat.
7. Review the bad.txt file. For each record that it contains, determine why it was rejected by the match engine. Then either:

   a. clean the source data to fix the error (for example, correct typos such as alphabetic characters in fields declared to be numeric)

   b. modify the configuration rules in the Project so that the record will pass (for example, modify data type)

   > **Note:** The OHMPI guide suggests that you can use the Cleansing process to modify data that otherwise would be rejected. For use with OHSCDA, Oracle recommends that you do not use the cleansing process to modify dimension data. Instead, you should modify the data in the source database. Refer to Rule one in Section 3.5.1, "Rules" on page 3-22.

8. Modify the source data based on conclusions drawn from reports.
9. Repeat steps 5-8 until data is satisfactory.

The output of the Cleanse process is an OHMPI-conformant flat file that will produce no rejects if passed through the cleanser again.

For more information, refer to *Oracle Healthcare Master Person Index User's Guide* and *Oracle Healthcare Master Person Index Analyzing and Cleansing Data User's Guide*.

3.3.6.4 Running Bulk Match in Analysis Mode and Adjusting Match Rules

This process is optional, but highly recommended. The reason is that its output also includes a report on which input records get treated as assumed matches under the current set of match rules. By running the Bulk Match process standalone, you can use the report to tune the match rules until you get the set of assumed matches you deem correct for your input data.

1. Generate the project's loader.zip file. Expand the zip file.

> **Note:** You must grant recursive write privileges on the loader directory to the current user.

2. Edit...\loader\conf\.

   a. Set the `/loader/system/properties/property/@matchAnalyzerMode` property to `true`, instructing the loader to perform analysis, rather than generating a load file.

   Iterate on this cycle:

3. Run Bulk Match.

4. Review the Bulk Match report. Compare the outcome to your expectations.

> **Note:** The Bulk Match report provides the sum of the weights for each pairing of input records. It is reported as the weight for the Systemcode and LocalID attributes of the records.

5. If the outcomes are not what you want:

   - Adjust the match rules. For more information, refer to Section 3.1.8.2, "Matching Rules Using Project Configuration" on page 3-10.
   - Clean and build the Project.
   - Run cluster-truncate.sql in the project database schema.
   - Execute steps 3-5 again.

For more information, refer to *Oracle Healthcare Master Person Index User's Guide* and *Oracle Healthcare Master Person Index Loading the Initial Data Set User's Guide*.

3.3.6.5 Bulk Load

When the matching rules are giving the results you want, that is, all records are correctly directed to the appropriate category, use Bulk Match to generate a set of load files. You can choose to have it carry out the load into the dimension's master index tables as part of the generation, or do it as a separate step. Use the **BulkLoad** property in Edit ...\loader\conf\loader-config.xml to determine this behavior. For more information, refer to the *Oracle Healthcare Master Person Index Loading the Initial Data Set User's Guide*.

3.3.7 Incremental Load Processes

3.3.7.1 Using MIDM Steward Loaded Data

OHMPI project provides generates a web application for inspecting the results of testing incoming source records against the Master Index according to the dimension's matching rules. This is the Master Index Data Manager (MIDM). In MIDM, a Data Steward can review potential matches, and determine whether the records should be treated as duplicates or not. If they are to be treated as duplicates, the Steward can also override the rule-based decisions about which data source provides the value for each attribute in the result record.

At any time, the Master Index for the dimension contains a cumulative set of records that result from the evaluation of source records. Use the MIDM for the following purposes:

- Decide on the fate of potential matches
- Merge attributes into single best record
- Unmerge records from profiles
- Merge currently separate profiles

3.4 Handling Fact Data after Dimension Deduplication

For most fact tables, the consequence of deduplication is limited to adjustment of foreign keys. Adjustment occurs when a dimension record to which the fact had a foreign key is identified as a contributor to an SBR in the dimension. The SBR gets a new ROW_WID in the dimension table; the foreign key in the fact is correspondingly updated so that it points to the SBR, rather than to the contributor record.

There are certain situations in which Dimension Deduplication has additional effects on Fact tables in the warehouse. These are discussed in the following subsections.

3.4.1 Merged Fact Records Consequent to LOV Dimension Merge

One of the dimensions that can be deduplicated in OHSCDA is the LOV dimension. This dimension holds sets of values for different codelists. Values from a particular codelist share a common value on R_TYPE. With deduplication, two values in an R_TYPE may be identified as referring to the same value. The consequence would be that an SBR is created, using the codelist value from the preferred source.

For instance the source for the Subject_Status codelist might include both Enroll and Enrolled. Both of these records would be loaded into the LOV dimension in the warehouse. During deduplication, however, the records will have been identified as a potential match, and the Data Steward will have selected a value for the VAL column in the SBR.

Now, if the subject status fact table happens to contain two records about Subject 101, one which pointed to an LOV value of Enroll, and another record which pointed to an LOV value of Enrolled, these records are now discovered to be duplicates of one another. This is because, after foreign key adjustment, they both will be found pointing to the same record in the LOV table. The Subject Status table is constrained to hold only one instance of a particular status for a subject; therefore one of the two fact records will have to be treated as the winner, and the other suppressed.

In the case outlined here, both come from the same database, so it is not possible to choose based on which contributor dimension record came from the preferred source for the dimension. Instead, for LOV, OHSCDA uses the original status code of the preferred LOV source record, i.e. the LOV record that was selected as the winner during deduplication. So, if the LOV record with the value Enrolled was chosen as the preferred source during deduplication of the LOV dimension, then the Subject Status fact record that originally pointed to that dimension record will be the winner.

While this seems obscure, it has a very practical consequence. Each of the subject status fact records has a date on which the subject achieved the status. Since only one of them survives, the date it carries is the date that is displayed for the subject achieving the status.

3.4.2 General Case: Discovered Duplicate Fact Records

Driving Dimensions

3.4.2.1 Impact of Dimension Deduplication on Fact and Target Tables

This section applies to the following:

Fact Tables

- Study-site Enrollment Plan (Table W_RXI_SITE_ENRLMNT_PLN_F)
- Region Enrollment Plan (Table W_RXI_RGN_ENRLMNT_PLN_F)
- Study Enrollment Plan (Table W_RXI_STDY_ENRLMNT_PLN_F)
- Subject Participation (Table W_RXI_SUBJECT_PRTCPTN_F)
- Subject Status (Table W_RXI_SUBJECT_STATUS_F)
- Study (Table W_RXI_STUDY_F)
- Study Region (Table W_RXI_STUDY_REGION_F)
- Study-site (Table W_RXI_STUDY_SITE_F)

Target Tables

- Study Region (Table W_RXI_STUDY_REGION_TGT)
- Program (Table W_RXI_PROGRAM_TGT)
- Study-site (Table W_RXI_STUDY_SITE_TGT)
- Study (Table W_RXI_STUDY_TGT)
- Region (Table W_RXI_REGION_TGT)
- Therapeutic Area (Table W_RXI_THERAPEUTIC_AREA_TGT)

For each of these tables, deduplication of the dimension that is at the grain of the fact table (the *driving* dimension for the fact) can cause multiple records in the fact table to point to the same dimension record, which will violate uniqueness requirements. To resolve this, OHSCDA merges fact records where necessary, to maintain the correct grain in the fact. The retained fact record is the one that comes from the same source as the SBR in the driving table.

To understand this better, consider the following example. Table 3–3 displays the results of deduplication of the Study-Site dimension. Data sources 1 and 2 have a Study-Site identified as SS-1. During deduplication, it has been established that they are duplicates since they refer to the same actual Study-Site. A Single Best Record, with ROW_WID = 103, has been defined for the set of duplicates. The preferred source for the SBR has been set as Datasource 1.

Table 3–3 Study-Site Dimension Table After Deduplication

ROW_WID	Study Site Identification #	Integration_ID	Datasource_ num_ID	Winner_ Row_Wid	Merge_Flag	SBR_Flag
101	SS-1	SS-1	1	103	Y	N
102	SS-2	SS-2	2	103	Y	N
103	SS-1	SS-1	1	(blank)	N	Y

Let us consider a fact that has Study-Site as its grain (and therefore has the Study-Site dimension as its driving dimension). Table 3–4, " Study-Site Enrollment Plan Fact After Key Adjustment and Dimension-driven Deduplication" displays records in the Study-Site Enrollment Plan Fact table.

As shown in the Orig_Study_wid column, these two fact records, originally pointed to Study-Site dimension records 101 and 102. But Study-Site dimension records 102 and 103 have been merged into the SBR, record 103. Neither the Enrollment Plan fact records cannot point to Study-site rows 102 and 103 nor can both of them point to the SBR Study-Site record 103. If they both did, there would be two records in the Study-Site Enrollment Plan Fact, both claiming to represent Study-Site SS-1.

Since the grain of the Study-Site Enrollment Plan Fact is one record per Study-Site, there can only be one fact record for each Study-Site. One of the fact records has to be suppressed (its Merge_Flag is set to Y), and the other gets its Merge_Flag set to N, marking it as the only record describing the enrollment plan for Study-Site SS-1.

Table 3–4 Study-Site Enrollment Plan Fact After Key Adjustment and Dimension-driven Deduplication

ROW_WID	Study_wid	Orig_Study_ wid	Study_site_ wid	Orig_ Study_ Site_wid	Planned_ Subject_ Enrolled_ Cnt	Datasource_ num_ID	Merge_ Flag
1001	1	1	103	101	100	1	N
1002	1	1	103	102	120	2	Y

OHSCDA uses this method to establish which record is suppressed and which is retained. The setting of the value of Merge_Flag in these fact tables depends on the integration of SBR record in *Driving* dimension table. The fact table record, related to integration_id selected in SBR record of Driving dimension, will be treated as the survivor record. It will be marked as Merge Flag = N and the other records will be marked as Merge Flag = Y. The fact table record coming from the same data-source as the data-source of the SBR of the Driving dimension will be treated as survivor record (Merge Flag = N), while the other record coming from the non-SBR data-source gets marked as Merge Flag = Y. In the example, the data-source for the Study-Site SBR (record 103 in the Study-Site dimension table) is source 1. For two records in the fact table pointing to Study-Site row 103, the one that gets Merge_Flag set to N is the one that shares the same data-source as the Study-Site SBR, and that is fact row 1001.

3.5 Rules and Recommendations

This section lists the rules that you must follow when deduplicating data, and adds some recommendations on best practices.

3.5.1 Rules

1. When merging attributes into an SBR, do not blend values of the dimension's integration ID from multiple sources.
2. Do not use the Cleansing process to modify the data. Make any needed changes in the original data source. Doing otherwise will lead to inconsistencies between data loaded via the Direct and Deduplication paths. Also, since cleansing is typically done only during initial load, changes that you make via cleansing will be lost if a cleansed record is subsequently reloaded.
3. When cleansing data, keep modifying source data (or altering matching configuration) until there are no rejected records. Any record that is rejected during cleansing is a record that will not be included in the Master Index, and therefore will not take part in deduplication.
4. When merging attributes into an SBR, do not blend the unique key attributes.
5. In Geography dimension, match cannot be left null.
6. If there is more than one database for either application, it is the user's responsibility to choose which of them is to be the preferred source for that application.

3.5.2 Recommendations

1. Oracle strongly recommends that you carefully review and revise the configuration of each project. Oracle has supplied pre-defined configurations, but these are intended only as starting points for your tuning. They definitely should not be used blindly. See 4.1 above for details on this recommendation.
2. OHMPI has a configuration parameter, SameSystemMatch that determines whether two profiles from the same source database are allowed to be programmatically merged into one profile. A setting of true prevents merging records from the same source database, regardless of whether the summed weights are greater than the Match threshold. A setting of false allows merger of records from the same source if the summed weights are greater than the Match threshold. In the Projects shipped by Oracle, this parameter is set to false. You should consider whether this setting is correct for your data. Refer to the *Oracle Healthcare Master Person Index Configuration Guide*.
3. OHMPI has a configuration parameter, OneExactMatch that determines the number of source records that can be incorporated into a Profile. If OneExactMatch is set to true and there is more than one record above the match threshold, then none of the records are considered an assumed match and all are flagged as potential duplicates. If OneExactMatch is set to false and there is more than one record above the match threshold, then all matching records are considered an assumed match. In the Projects shipped by Oracle, this parameter is set to false. You should consider whether this setting is correct for your data. Refer to the *Oracle Healthcare Master Person Index Configuration Guide*.
4. If new data is going to be loaded into a dimension, and you have any reason to be concerned that the current rules for the dimension will not properly categorize the new records, extract the new records into a flat file, and use the cleanser and Bulk Match Analyzer to see how they will be handled by the match engine. If necessary, adjust the matching rules so they categorize the records properly. Then load the records into the source database, and allow them to be processed by the next incremental load.

3.6 Oracle Health Sciences Clinical Development Analytics' Match Rules

This section provides information about the matching rules that are shipped with each OHSCDA dimension's Project. These matching rules are intended as a starting point only. it is imperative that you at least review these rules to determine whether they meet your needs. It is very likely that they will not suffice without modification.

3.6.1 Policies for Creating Shipped Match Rules

This section describes how OHSCDA will define the rules it ships for identifying duplicates in its dimensions. For each dimension it gives:

The general policies are as follows. However, they may be overridden for particular dimensions.

1. To be above the match threshold, the match fields being compared must be strictly identical.
2. OHSCDA's default match rules attempt to identify records as non-duplicates only if it is clear that there is no possibility that they could be part of a potential match. When in doubt, the default rules lean toward marking records as potential matches. It is easy for the steward to push data from the potential category into either assumed match or non-match. It is possible, but more work, to take non-matches and turn them into matches.
3. In creating an assumed or potential match SBR, use the values from the preferred source. See Table 3–5, " Preferred Source for Each Deduplicated Dimension" on page 3-24 for the preferred sources defined in the Projects shipped by OHSCDA.
4. If two or more records match, OHSCDA's default match rules set the values of the attributes of the Single Best Record (SBR) to be the attributes of the preferred source. For preferred source, refer Table 3–5, " Preferred Source for Each Deduplicated Dimension" on page 3-24
   a. If the preferred record has an attribute corresponding to an attribute required by the SBR, but its value is null, leave it null in the SBR. That is, if the preferred record contains an attribute that is needed for the BR, always take the value supplied by the preferred record, even if that value is NULL.
   b. If the preferred record does not have an attribute corresponding to a given target record attribute, but that attribute is available in a donor record, use the attribute value from the highest ranking donor record. For example, if Oracle Clinical is the preferred source for Study, since OC lacks an attribute for EUDRA_NUMBER, take the value of EUDRA_NUMBER from SC.

Table 3–5 Preferred Source for Each Deduplicated Dimension

Warehouse Table	Preferred Source
W_EMPLOYEE_D	Siebel Clinical
W_GEO_D	Siebel Clinical
W_HS_APPLICATION_USER_D	Siebel Clinical
W_LOV_D	Siebel Clinical
W_PARTY_D	Siebel Clinical
W_PARTY_ORG_D	Siebel Clinical
W_PARTY_PER_D	Siebel Clinical
W_PRODUCT_D	Siebel Clinical

Table 3–5 (Cont.) Preferred Source for Each Deduplicated Dimension

Warehouse Table	Preferred Source
W_RXI_CRF_BOOK_D	Oracle Clinical
W_RXI_CRF_D	Oracle Clinical
W_RXI_PROGRAM_D	Siebel Clinical
W_RXI_SITE_D	Siebel Clinical
W_RXI_STUDY_D	Siebel Clinical
W_RXI_STUDY_REGION_D	Siebel Clinical
W_RXI_STUDY_SITE_D	Siebel Clinical
W_RXI_STUDY_SUBJECT_D	Oracle Clinical
W_RXI_VALDTN_PROCEDURE_D	Oracle Clinical
W_USER_D	Siebel Clinical

3.6.2 Configurations

Table describes the configuration of the OHMPI Projects shipped with OHSCDA. For each Project, it lists:

- Dimension Name - the name of the dimension in the warehouse
- Project Name - the name given to the OHMPI Project for that dimension
- Duplicate Threshold - the minimum summed weight that will cause a record to be considered a potential match
- Match Threshold - the minimum summed weight that will cause a record to be considered an assumed match
- Attributes of the key fields for the dimension:
  - Match Attribute - the name of the attribute
  - Match Type - the MatchType used for determining the similarity of the field between the records being compared. Note that all fields sharing a MatchType in a Project use the same Disagree and Agree weights
  - Customized/Built-in - Where an existing MatchType would not serve, OHSCDA created a new MatchType. Typically this was done to use the same comparator function, but different weights than another field
  - Comparator Function - an algorithm for determining similarity of fields
  - Disagree Weight - the field's contribution to the summed weight if the values being compared differ completely (per the Comparator function)
  - Agree Weight - the field's contribution to the summed weight if the values being compared agree completely (per the Comparator function)
  - Null Field - the impact of the field on the summed weight if one or both records being compared have a null or empty string for the field

Table 3–6 Project Weights and Thresholds

Dimension Name	Project Name	Duplicate Threshold	Match Threshold	Match Attribute	Match Type	Customized or Built-in	Comparator Function	Disagree Weight	Agree Weight	Null Field
Validation Procedure	OCDA_Valdtn	25	30	STUDY_NAME	StudyName	Customized	Condensed String Comparator	0	10	Zero Weight
Validation Procedure	OCDA_Valdtn	25	30	VALDTN_PROC_NAME	String	Built-in	Condensed String Comparator	0	20	Zero Weight
User	OCDA_User	10	10	LOGIN	String	Built-in	Condensed String Comparator	-10	10	Zero Weight
Study Site	OCDA_Study_Site	20	40	SITE_NAME	String	Built-in	Condensed String Comparator	0	10	Zero Weight
Study Site	OCDA_Study_Site	20	40	STUDY_NAME	String	Built-in	Condensed String Comparator	0	10	Zero Weight
Study Site	OCDA_Study_Site	20	40	STUDY_SITE_IDENTIFICATION	String	Customized	Condensed String Comparator	0	20	Zero Weight
Study Subject	OCDA_Study_Subject	20	20	SUB_IDENTIFICATION	String	Built-in	Condensed String Comparator	-10	10	Zero Weight
Study Subject	OCDA_Study_Subject	20	20	STUDY_NAME	String	Built-in	Condensed String Comparator	-10	10	Zero Weight
Product	OCDA_Product	10	10	PROD_NAME	String	Built-in	Condensed String Comparator	-10	10	Zero Weight
Site	OCDA_Site	80	160	SITE_NAME	String	Built-in	Condensed String Comparator	0	80	Full Agreement Weight
Site	OCDA_Site	80	160	ADDRESS_StName	StreetName	Built-in	Condensed String Comparator	0	10	Full Agreement Weight
Site	OCDA_Site	80	160	ADDRESS_HouseNo	House Number	Built-in	Advanced Jaro Adjusted for HouseNumbers	0	10	Full Agreement Weight
Site	OCDA_Site	80	160	ADDRESS_StDir	StreetDir	Built-in	Advanced Jaro String Comparator	0	10	Full Agreement Weight
Site	OCDA_Site	80	160	ADDRESS_StType	StreetType	Built-in	Advanced Jaro String Comparator	0	10	Full Agreement Weight

Table 3–6 (Cont.) Project Weights and Thresholds

Dimension Name	Project Name	Duplicate Threshold	Match Threshold	Match Attribute	Match Type	Customized or Built-in	Comparator Function	Disagree Weight	Agree Weight	Null Field
Site	OCDA_Site	80	160	SITE_COUNTRY	CountryStateCityZip	Customized	Condensed String Comparator	0	10	Full Agreement Weight
Site	OCDA_Site	80	160	SITE_STATE	CountryStateCityZip	Customized	Condensed String Comparator	0	10	Full Agreement Weight
Site	OCDA_Site	80	160	SITE_CITY	CountryStateCityZip	Customized	Condensed String Comparator	0	10	Full Agreement Weight
Site	OCDA_Site	80	160	SITE_ZIPCODE	CountryStateCityZip	Customized	Condensed String Comparator	0	10	Full Agreement Weight
Program	OCDA_Program	13	13	PROGRAM_NAME_Name	PrimaryName	Built-in	Condensed String Comparator	-2	13	Zero Weight
Program	OCDA_Program	13	13	PROGRAM_NAME_OrgType	OrgTypeKeyword	Built-in	Condensed String Comparator	-6	8	Zero Weight
Program	OCDA_Program	13	13	PROGRAM_NAME_AssocType	AssocTypeKeyword	Built-in	Condensed String Comparator	-3	5	Zero Weight
Program	OCDA_Program	13	13	PROGRAM_NAME_Sector	IndustrySectorList	Built-in	Condensed String Comparator	-4	5	Zero Weight
Program	OCDA_Program	13	13	PROGRAM_NAME_Industry	IndustryTypeKeyword	Built-in	Condensed String Comparator	-4	7	Zero Weight
Program	OCDA_Program	13	13	PROGRAM_NAME_Url	Url	Built-in	Condensed String Comparator	-4	8	Zero Weight
Lov	OCDA_Lov	18	20	R_TYPE	String	Built-in	Condensed String Comparator	-10	10	Zero Weight
Lov	OCDA_Lov	19	20	VAL	String	Built-in	Condensed String Comparator	-10	10	Zero Weight
Geography	OCDA_Geography	25	85	CITY	OCDA CityString	Customized	Condensed String Comparator	0	25	Zero Weight
Geography	OCDA_Geography	25	85	COUNTRY	OCDA CountryString	Customized	Condensed String Comparator	-6	10	Zero Weight

Table 3–6 (Cont.) Project Weights and Thresholds

Dimension Name	Project Name	Duplicate Threshold	Match Threshold	Match Attribute	Match Type	Customized or Built-in	Comparator Function	Disagree Weight	Agree Weight	Null Field
Geography	OCDA_Geography	25	85	STATE_PROV	OCDA StateString	Customized	Condensed String Comparator	0	20	Zero Weight
Geography	OCDA_Geography	25	85	ZIPCODE	OCDA ZipString	Customized	Condensed String Comparator	-30	30	Zero Weight
Study	OCDA_Study	7	10	STDY_NM	String	Built-in	Condensed String Comparator	-10	10	Zero Weight
Application User	OCDA_APP_USER	8	10	APP_USR_NM	String	Built-in	Condensed String Comparator	-10	10	Zero Weight
CRF	OCDA_CRF	20	20	CRF_NAME	String	Built-in	Condensed String Comparator	-10	10	Zero Weight
CRF	OCDA_CRF	20	20	STUDY_NAME	String	Built-in	Condensed String Comparator	-10	10	Zero Weight
CRF_BOOK	OCDA_CRF_BOOK	20	20	CRF_BOOK_NAME	String	Built-in	Condensed String Comparator	-10	10	Zero Weight
CRF_BOOK	OCDA_CRF_BOOK	20	20	STUDY_NAME	String	Built-in	Condensed String Comparator	-10	10	Zero Weight
Party_Per	OCDA_Investigator	160	235	Last_Name_Std	LastName	Built-in	Advanced Jaro Adjusted for Last Names	0	80	Full Combination Weight
Party_Per	OCDA_Investigator	160	235	FULL_ADDRESS_StName	StreetName	Built-in	Condensed String Comparator	0	5	Full Combination Weight
Party_Per	OCDA_Investigator	160	235	FULL_ADDRESS_HouseNo	House Number	Built-in	Advanced Jaro Adjusted for HouseNumbers	0	5	Full Combination Weight
Party_Per	OCDA_Investigator	160	235	FULL_ADDRESS_StDir	StreetDir	Built-in	Advanced Jaro String Comparator	0	5	Full Combination Weight
Party_Per	OCDA_Investigator	160	235	FULL_ADDRESS_StType	StreetType	Built-in	Advanced Jaro String Comparator	0	5	Full Combination Weight

Table 3–6 (Cont.) Project Weights and Thresholds

Dimension Name	Project Name	Duplicate Threshold	Match Threshold	Match Attribute	Match Type	Customized or Built-in	Comparator Function	Disagree Weight	Agree Weight	Null Field
Party_Per	OCDA_Investigator	160	235	ORIG_FST_NAME	OrigNameWegt	Customized	Condensed String Comparator	0	5	Full Combination Weight
Party_Per	OCDA_Investigator	160	235	ORIG_LAST_NAME	OrigLastName	Customized	Condensed String Comparator	0	5	Full Combination Weight
Party_Per	OCDA_Investigator	160	235	ORIG_MIDDLE_NAME	OrigNameWegt	Customized	Condensed String Comparator	0	5	Full Combination Weight
Party_Per	OCDA_Investigator	160	235	FST_NAME_Std	FirstName	Built-in	Advanced Jaro Adjusted for First Names	0	80	Full Combination Weight
Party_Per	OCDA_Investigator	160	235	MID_NAME	MiddleNMString	Customized	Condensed String Comparator	0	20	Full Combination Weight
Party_Per	OCDA_Investigator	160	235	STATE	String	Built-in	Condensed String Comparator	0	5	Full Combination Weight
Party_Per	OCDA_Investigator	160	235	ZIP	String	Built-in	Condensed String Comparator	0	5	Full Combination Weight
Party_Per	OCDA_Investigator	160	235	COUNTRY	String	Built-in	Condensed String Comparator	0	5	Full Combination Weight
Party_Per	OCDA_Investigator	160	235	CITY	String	Built-in	Condensed String Comparator	0	5	Full Combination Weight

3.7 User-supplied Deduplication System

If you want to use a deduplication program other than OHMPI, the program must plug into the OHSCDA deduplication path at two points: it must read from the source databases for the dimension, and it must write to the Persistent Staging tables.

Therefore, to implement a non-OHMPI deduplication program, you must supply the following:

- a set of match rules
- an Extractor
- a program for processing extracted records according to the match rules
- a warehouse schema for the dimension's Master Index
- the SIL to read the dimension's Master Index, and write to the relevant Persistent Staging tables. Each dimension has a dimension-specific Persistent Staging table, with a prefix W_MDM_. For each set of duplicates identified, this table must receive copies of the Single Best Record for the set, and all the members of the set. Additionally, the SIL must write to a common table, W_HS_MDM_MAPPING_S that maps each Single Best Record to its contributors.

The process for carrying this out cannot be detailed here, since it depends on the nature of the non-OHMPI deduplication program and its Master Index.

3.8 Extending the Warehouse

Suppose that you have a column M in a transactional source (S1) for OHSCDA. Column M is not part of the OHSCDA warehouse and is an attribute of a dimension (D1) that is deduplicated. If you want column M to be available in the OHSCDA presentation layer, then following is the overall list of the tasks that have to be accomplished:

Table 3–7 Extension Tasks by Load Path

Task	Path
Add M to the Sites Staging table	Direct
Add M to the Sites warehouse table	Direct
Add M to the Persistent staging table	Deduplication
Add M to the SIL that reads from the Master Index for dimension D1, and writes to the Persistent Staging Tables	Deduplication
Add M to the OHMPI Project definition	Deduplication
Add M to S1 Sites SDE	Direct
Add M to Sites SIL	Direct
Add M to OBIEE Physical, Business and Presentation Layer	Direct

The modifications required for the Direct path are discussed in the following sections.

3.8.1 Adding a Column to the Persistent Staging Table

To add column M to the persistent staging table, add column X_ M to the warehouse table. You must prefix the column name with "X_" since this will prevent a collision if Oracle later adds column M to the shipped dimension table.

3.9 ODI Mappings used in Multi-Source Integration

Table 3–8 shows the ODI mappings that have a role in the deduplication path.

For each mapping the table indicates its location in the deduplication path (Direct path SDE and SIL are included for completeness, although they do not act on the deduplication path). It also shows the source and target of the mapping, and briefly describes what the mapping does.

Table 3–8 ODI Mappings used by Multi-Source Integration

Typical Mappings	Path	Segment ID	Segment	Source	Target	Initial/Incremental	Description
SDE_<Source_App>_<Dimension>_Dim_Init	Dedup	Dedup 1	Bulk Load	Source database	Flat file	Initial	Full extract to generate a flat file for use with the bulk loader in initial load
SDE_<Source_App>_<Dimension>_Dim_Inc	Dedup	Dedup 2	Extractor	Source database	Master Index	Incremental	Incremental extract for call to OHMPI API.
SIL_MPI_<Dimension>_DIM	Dedup	Dedup 3	Convertor	Master Index	Persistent Staging	Both	Populate Persistent Staging from Master Index
SIL_CDA_PS_<Dimension>_Dim	Dedup	Dedup 4a	Apply Merge to Target	Persistent Staging	Dimension table	Both	Extract SBR from Persistent Staging, insert/update it in the Target dimension table
SIL_CDA_PS_<Dimension>_Dim_Match_Merge	Dedup	Dedup 4b	Apply Merge to Target	Persistent Staging	Dimension table	Both	Merge records in Dimension target table into their SBR.
SDE_<Source_App>_<Dimension>_Dim	Direct	Direct 1	SDE	Source database	Staging table	Both	Extract source records for both initial and incremental loads
SIL_<Dimension>_Dim	Direct	Direct 2	SIL	Staging table	Dimension table	Both	Transform Staged data to Target data for both initial and incremental loads

A

Troubleshooting

This appendix contains the following topics:

- Sorting and Displaying of Null Values in Reports on page A-1
- Aborting a Load Plan on page A-2

A.1 Sorting and Displaying of Null Values in Reports

In order to understand results shown in OBIEE reports, it may be necessary to understand how null values are sorted and displayed in reports.

Oracle uses NULL as a pseudo-value for a table cell when there is no actual value. For example, if the number of documents awaiting completion for a site is unknown, the column containing that attribute of the site will be set to null in the database.

As null values can appear in among data, OBIEE has rules that determine how to display the null values. And as OBIEE supports sorting of data in a column, it has rules for how nulls should be sorted.

The following are the rules:

- Oracle's sorting order cause a null value to be treated as greater than any non-null value.
- In table views, OBIEE generally displays null values as empty cells.

  The exception is when the request designer has specified that the user can navigate to a different request by clicking on a value in the column that contains null. In that case, in order to give the user something to click on, OBIEE displays the null value as a zero.

These rules can produce unexpected results. This following section describes how to interpret such unexpected results. It also describes actions you can take in creating OBIEE requests to override OBIEE's default rules.

The results of these rules are:

- If the data in a column contain nulls and non-nulls, and the column is sorted, and navigation is not enabled from cells in the column, then:
  - Nulls will display as blank cells
  - Blank cells will sort as larger than the largest non-null value
- If the data in a column contain nulls and non-nulls, and the column is sorted, and navigation is enabled from cells in the column, then:
  - Nulls will display as zeros

  - Cells representing nulls (but now displaying as zeros) will sort as larger than the largest non-null value. If there are actual zeros in the column as well, they will sort as smaller than the smallest positive value in the column. So, if you have both real zero values and null values, and cell-based navigation is enabled, and you sort the column, you will get two clumps of zeros - one representing the nulls, the other representing the actual zeros - separated by the non-negative actual values.

- OBIEE does have a capability that can be used to make it easier to identify null values. In requests, you can use the IFNULL function to specify that NULL should be replaced by a large negative value that could not be a real value for the column. For instance, if "# Documents Outstanding" could be null in your data, and you want to include it in a request, you could change the functional definition of the column in the request from "# DocumentsOutstanding" to IFNULL("# Documents Outstanding", -99). This would cause nulls to sort and display as if their value was -99.

If you use IFNULL, it is important that you:

- Choose a value that could not also be a legitimate value (this may vary from column to column, though it is preferable to use the same IFNULL replacement across all columns).
- Communicate to your end users the meaning of the IFNULL values.

A.2 Aborting a Load Plan

Perform the following steps in ODI to abort a load plan:

1. Select **Load Plans and Scenarios** in the **Operator** dialog box.
2. In **Agents** section, expand **Agent used for Execution**.
3. Right-click the Load Plan you want to abort.
4. Click **Stop Normal** or **Stop Immediate**, as required.
5. Select **Physical Agent**.
6. Click **OK**.

See Also:

Oracle Data Integrator Online Help