# Oracle® Health Sciences Clinical Development Analytics

Administrator's Guide

Release 2.1 for Plus Configuration

**E28551-01**

March 2012

ORACLE®

Oracle Health Sciences Clinical Development Analytics Administrator's Guide, Release 2.1 for Plus Configuration

E28551-01

# Contents

## 3 Extract Transform Load Programs



## 4 Multi-Source Integration

## A Troubleshooting



## Glossary

## Index

# Preface

This guide provides information about how to use Oracle Health Sciences Clinical Development Analytics (CDA).

This preface contains the following topics:



## Audience

The first and second chapters of this guide are intended for the following job classifications:

- Clinical Program/Study Manager, Clinical Data Manager, Clinical Research Associate, Clinical Data Entry Manager, Site Personnel, and Executive Management.

The other chapters in this guide are intended for the following job classifications:

- Data Warehouse Administrators, ETL Developers and Operators
- System Administrator

This guide assumes that you have the following general skills:

- Knowledge of Oracle Life Sciences Data Hub.
- Knowledge of Oracle Business Intelligence Enterprise Edition Plus.
- Knowledge of Informatica PowerCenter.
- Familiarity with Oracle Clinical.
- Familiarity with Oracle's Siebel Clinical.

## 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 accountbefore 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 Clinical Release 4.6* documentation set, the *Oracle Life Sciences Data Hub Release 2.2* documentation set, or the *Oracle Business Intelligence Enterprise Edition 11g Release 1 (11.1.1)* documentation set:

### Oracle Life Sciences Data Hub Documentation

The Oracle Life Sciences Data Hub documentation set includes:

- *Oracle Life Sciences Data Hub Implementation Guide*
- *Oracle Life Sciences Data Hub System Administrator's Guide*
- *Oracle Life Sciences Data Hub Application Developer's Guide*
- *Oracle Life Sciences Data Hub User's Guide*
- *Oracle Life Sciences Data Hub Installation Guide*
- *Oracle Life Sciences Data Hub Adapter Toolkit Guide*
- *Oracle Life Sciences Data Hub Application Programming Interface Guide*
- *Oracle Life Sciences Data Hub Release Notes*
- *Oracle Life Sciences Data Hub Release Content Document*

### Oracle Business Intelligence Enterprise Edition Documentation

The *Oracle Business Intelligence Suite Enterprise Edition Online Documentation Library* documentation set includes:

- *Oracle Fusion Middleware Developer's Guide for Oracle Business Intelligence Enterprise Edition*
- *Oracle Fusion Middleware Enterprise Deployment Guide for Oracle Business Intelligence*
- *Oracle Fusion Middleware Integrator'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 Quick Installation Guide for Oracle Business Intelligence*
- *Oracle Fusion Middleware Security Guide for Oracle Business Intelligence Enterprise Edition*
- *Oracle Fusion Middleware System Administrator's Guide for Oracle Business Intelligence Enterprise Edition*
- *Oracle Fusion Middleware Upgrade Guide for Oracle Business Intelligence Enterprise Edition*
- *Oracle Fusion Middleware User'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 *Oracle Clinical* documentation set includes:

- *Siebel Data Model Reference for Industry Applications*
- *Siebel Life Sciences 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 Repository and Warehouse

This chapter contains the following topics:



## Maintaining the Oracle Health Sciences Clinical Development Analytics Repository

Each release of Oracle Health Sciences Clinical Development Analytics (CDA) contains a 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 Requests. As shipped, the RPD corresponds to the CDA data warehouse, and can be used without any modification.

However, you might find it desirable to modify the Oracle-supplied CDA 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. For more information about why CDA must refresh this variable, refer to the Details for Selected Modifications on page 1-3.
- You want to modify a group, an account, or a privilege maintained through the repository.

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 patches and releases of CDA. 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 CDA 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.

## Modifying the Repository

The CDA Repository is maintained as a versioned object in Oracle LSH. A copy of that Repository is deployed to the application server file system. This *deployed* Repository is the one that the Oracle BI Server uses. All changes to the Repository, however, must be made through a two-step process:

- Modify the versioned Repository object.
- Deploy the latest version of the Repository object.

Therefore, Oracle requires that you do not modify the deployed CDA Repository directly.

If you do need to modify the Repository, perform the following tasks:

1. Check out the Business Area (BA) to a different domain in which you will customize the Repository, or use an existing customized Business Area.

   **Important:** Use the **Copy definition to the local Application Area and check out** option to check out the program to a different domain. This preserves the changes to the definitions in a different domain, and ensures that the changes are not overwritten automatically in the next upgrade of CDA.

2. If the reason you are modifying the RPD is that you have modified the data warehouse:
   a. Verify the mappings of the tables in the BA.
   b. Remap table instances and table descriptors, if necessary.

      If you made any changes to the BA, reinstall the Work Area and check out the Business Area again.

      **Note:** In this checkout, do **not** use the **Copy definition to the local Application Area** option to check out the program.

3. Click **Launch IDE** on the Business Area's Properties screen.

   This downloads the versioned RPD object from the Business Area, and opens the Oracle BI Administration tool. Make the desired modifications to the downloaded RPD. Refer to Details for Selected Modifications for instructions on applying selected modifications.

4. Save the changes; exit the Oracle BI Administration tool.
5. In Oracle LSH, upload the modified RPD back into the Oracle LSH Business Area.

The modified RPD can be found in a location that has been defined for your LSH configuration.

6. Ensure that the OBIEE DP Server is up and running. Otherwise, the next step will indicate success, but the RPD will not be deployed.
7. Install the WorkArea that contains the BA that contains the modified Repository.
8. Launch the Oracle BI Presentation Server to verify the changes.

#### 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 Life Sciences Data Hub Developer's Guide*
- *Oracle Business Intelligence Server Administration Guide* for more information about modifying the RPD.

### Merging Changes Into a New Oracle-supplied Repository

CDA releases include a copy of the Repository. The installation process for each release deploys that Repository. If you do modify your copy of the CDA Repository, you must merge your changes into the Oracle-supplied Repository each time you receive a release or patch of CDA that includes a repository. At upgrade time, use the OBIEE utility File > Merge in the Repository Administrator to merge your modified RPD with the Oracle-supplied RPD.

## Maintaining the Oracle Health Sciences Clinical Development Analytics Data Warehouse

You may need to modify the CDA 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 CDA.
- *Substitution*: Swapping data from a different source for a column that was delivered with CDA.

---

**Caution:** Exercise caution when you modify the data warehouse. Please conform to the recommendations mentioned in the subsequent sections.

---

## Derivations

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

- Calculate the derivation as part of the creation of a request.

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

- Calculate the derivation in the physical or business layer of the RPD file; it is propagated 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 requests. The derived value looks the same as any other Answers column.

- Calculate the derivation in the data warehouse.

  The calculation is run at ETL execution time and not at query time. The derived value looks the same as any other Answers column. In this approach, you must add the result column to the staging and target tables, modify the ETL procedures (both Source Dependent Extract (SDE) and Source Independent Load (SIL)), and then add the column to all the layers of the RPD.

## Extensions

An *extension* is a new column added to the data warehouse for data not available in Oracle Clinical or Oracle's Siebel Clinical.

**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.
- This table has a foreign key to the primary key in Oracle Clinical table OCL_ STUDIES.

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.

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

1. Create a pass-through view of the STUDY_MANAGERS table so that the table is visible in Oracle LSH.
2. Modify staging table W_RXI_STUDY_DS, adding the STUDY_MANAGER column. To modify the staging table, perform the following tasks in Oracle LSH:
   a. Check out the table definition into another domain (to ensure that the changes are not overwritten in the next CDA upgrade) or use an existing customized table definition.
   b. Add the new column and reinstall.

**Important:** Before you reinstall, ensure that the Informatica DP Server is up and running.

3. Modify the SDE that populates W_RXI_STUDY_DS, in two ways:
   - Add the STUDIES_MANAGERS table as a source of the program.
   - Add a mapping of column STUDY_MANAGER from STUDY_MANAGER to W_RXI_STUDY_DS.

   To modify the SDE, perform the following tasks in Oracle LSH:

   a. Check out the SDE into another domain (to ensure that the changes are not overwritten in the next CDA upgrade) or use an existing customized SDE.

   b. Add the new column and reinstall.

   **Important:** Before you reinstall, ensure that the Informatica DP Server is up and running.

4. If it does not already exist to support some other extension, create extension table W_RXI_STDY_DX, containing one column [STDY_WID] to function as a foreign key that joins to the primary key in W_RXI_STDY_D. This table is populated with one row for each row in W_RXI_STDY_D when the Study SIL executes.
5. Add column STUDY_MANAGER to W_RXI_STDY_D to hold the name of the study manager. To add a column, perform the following tasks in Oracle LSH:

   a. Check out the table into another domain (to ensure that the changes are not overwritten in the next CDA upgrade) or use an existing customized table definition.

   b. Add the new column and reinstall.

   **Important:** Before you reinstall, ensure that the Informatica DP Server is up and running.

6. Modify the SIL that populates W_RXI_STDY_D. Add instructions to create a record in W_RXI_STDY_DX for each record in W_RXI_STDY_D, and to copy W_RXI_STUDY_DS. STUDY_MANAGER into W_RXI_STDY_DX.STUDY_MANAGER for each record.

   To modify the SIL, perform the following tasks in Oracle LSH:

   a. Check out the SIL into another domain (to ensure that the changes are not overwritten in the next CDA upgrade) or use an existing customized SIL.

   b. Add the new column and reinstall.

   **Important:** Before you reinstall, ensure that the Informatica DP Server is up and running.

7. Modify the repository:

**Important:** Before you modify the repository, ensure that you check out the Business Area containing the repository (`OCDA_OBIEE_WA`) to a different domain. This preserves the changes to the definitions in a different domain, and ensures that the changes are not overwritten automatically in the next upgrade of CDA. Alternatively, you can use an existing customized Business Area.

a. Import the definition of the extension table, W_RXI_STDY_DX, into the Repository.

b. Using W_RXI_DISCREPANCY_FX as an example, propagate the extension table and its contents to the Business and Presentation layers.

#### Subsequent Releases

- If a subsequent CDA release requires an update of the tables or ETL that you modified in a different domain, CDA does not overwrite such modifications. You can choose to manually upgrade to the new CDA releases by pointing your definitions to Oracle-supplied definitions.
- If a subsequent CDA release requires an update to the Repository that you modified in a different domain, CDA does not overwrite such modifications. You can use the OBIEE Repository merge utility `equalizerpds.exe` to merge your modified RPD with the Oracle-supplied RPD.
- If a subsequent CDA release requires an update to the Web Catalog, the OBIEE Web Catalog merge capability will preserve your changes to the catalog while applying Oracle's changes.

## Substitutions

A substitution occurs if you have a preferred alternative source of data for a column that CDA 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 CDA 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 propagates the data to a staging table already exists. You will have to perform the following tasks:

1. Create a table or pass-through view in Oracle LSH containing the locally-sourced values of the column, and also add whatever keys are needed to join to the Oracle-supplied view.
2. Create a program that joins the two tables and creates a new table, in which the locally-sourced values replace the Oracle-supplied values for the column of interest. Call this the Substitution Table.
3. Modify the SDE to read from the Substitution Table, rather than the Oracle-supplied table.

   To modify the SDE, perform the following tasks in Oracle LSH:

   a. Check out the SDE into another domain (to ensure that the changes are not overwritten in the next CDA upgrade) or use an existing customized SDE.

   b. Modify the definitions and reinstall.

**Important:** Before you reinstall, ensure that the Informatica DP Server is up and running.

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 4-1:

***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 that reads the old table to instead read the new table. |
| Modified table has modified layout | 1. Modify the SDE to read the modified layout.<br>2. Modify the staging table populated by the SDE to include the modified layout.<br>3. Modify the SIL to read the modified layout.<br>4. Modify the target table to include the modified layout.<br>5. Modify the RPD to accept the changed data warehouse table. |
| New table | 1. Add a staging table to accept the new input.<br>2. Add an SDE to read from the new table and write to the staging table.<br>3. Add a warehouse table to make the new data available to the BI Server.<br>4. Add an SIL to populate the new data warehouse table from the new staging table.<br>5. Modify the RPD to accept the new warehouse table. |

## Modifying Data Warehouse Tables

Depending on what changes are required to the data warehouse, it is necessary to modify either the source table in Oracle LSH, or the source, staging, and target tables. In either case, use Oracle LSH capabilities to modify the definition of the relevant tables, or to create new tables.

### Managing Indexes

CDA is delivered with a set of indexes. If you wish, you can add appropriate indexes to meet your query requirements. Use Oracle LSH for this purpose.

If all indexes must be dropped and recreated, perform the following tasks in Oracle LSH:

1. Navigate to the Submit Execution Setup screen of the program instance.
2. In the Submission Parameters tabbed page, set the value of Drop and Recreate Index to **Yes**.

   If set to Yes, Oracle LSH drops all indexes on all target Table instances before the Oracle LSH Informatica Program is executed, and recreates them after execution.

If you do not want to drop and recreate indexes for all Table Descriptors, you can call the Oracle LSH API to drop and recreate specific indexes. For more information about

selective index management, refer to the *Oracle Life Sciences Data Hub Developer's Guide, (Selective Index Management).*

**See Also:**

*Oracle Life Sciences Data Hub Developer's Guide, (Defining Table Constraints and Indexes)*

# 2

# Implementing Security

This chapter contains the following topics:



## About Security in Oracle Health Sciences Clinical Development Analytics

Defining security for CDA includes the following tasks

- **CDA Authorization**. In this, you define each CDA user, and determine which OBIEE activities the user is entitled to perform. CDA delegates authentication of its OBIEE user accounts to LSH, so you must define CDA users in LSH.

  You assign user accounts to user groups in Oracle LSH. On login, Oracle LSH passes the authenticated user's user group assignments to OBIEE, where user groups with the same name determine which parts of CDA the user can use. Predefined OBIEE user groups determine the privileges allowed to users and allow access to the shipped CDA dashboards and reports. You must define an Oracle LSH user group with the same name for each OBIEE user group you plan to use. You can create additional user groups as needed in both OBIEE and Oracle LSH. In addition, in Oracle LSH you must define roles, assign the roles to user groups, and assign users to roles in user groups. For more information, refer to Setting Up User Authorization on page 2-4.

- **Data Access Specification**. In this, you determine the study-sites and studies for which each CDA user is permitted to see data. For more information, refer to Setting Up Study and Study Site Data Access for Users on page 2-10.

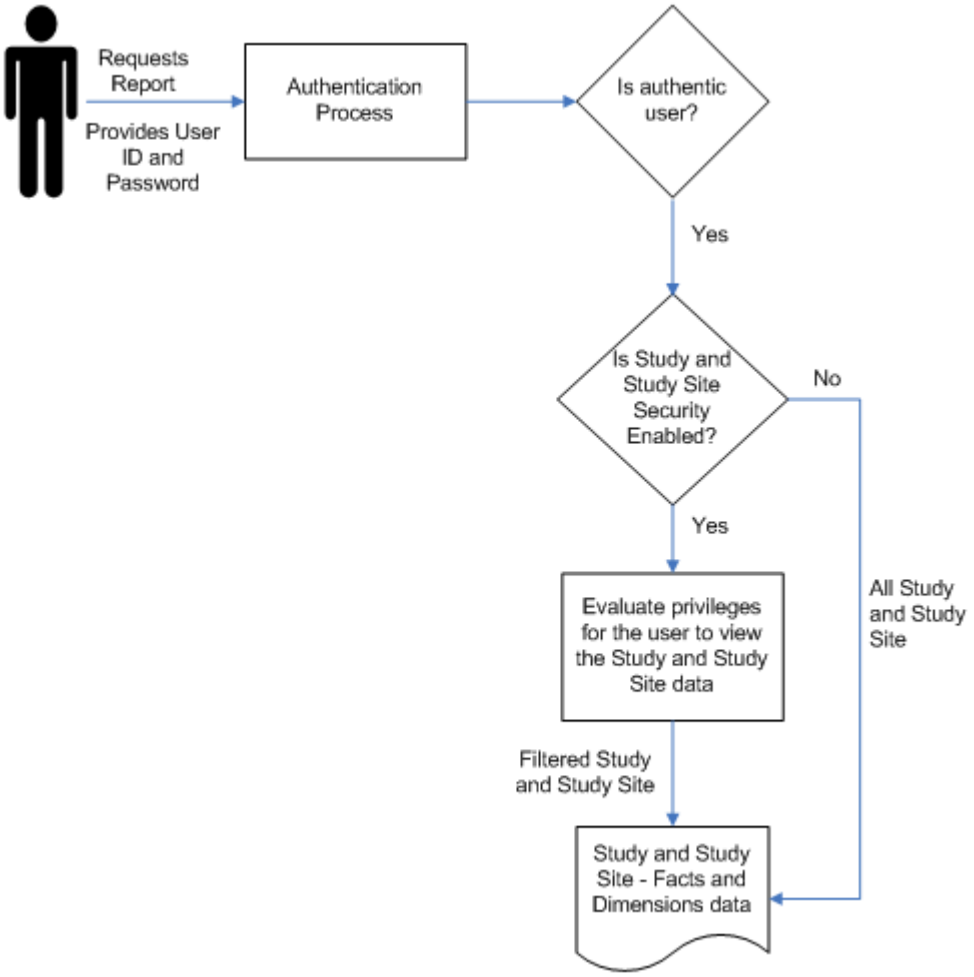
- **ETL Use Authorization**. In this, you determine which users shall have the ability to view, modify, and execute ETL for CDA. This has two parts, one of which is needed only if you are implementing Multi-source Integration.

  – Direct-path ETL is executed through LSH. To use CDA, you must define at least one LSH account with privileges to execute ETL. You may want to enable one or more LSH accounts to modify direct path ETL as well. Security for direct path ETL is controlled through LSH. For information on Direct-path ETL authorization, refer to Roles for LSH Programmers on page 2-7.

  – Deduplication ETL is used only for Multi-source Integration. It is responsible for extracting modified source data for dimension records, and invoking the

deduplication engine to identify duplicates in the dimension data. This ETL is executed through DAC, rather than through LSH. To define and authorize DAC and Informatica users for these purposes, consult the respective product documentation.

***Figure 2–1 Study and Study Site Security Implementation***




## Example

This document describes how to set up security for the following basic types of users as an example. To refine this example for your company's needs, refer to the Oracle LSH System Administrator's Guide chapter on setting up security.

- **CDA End Users** are people who can view Oracle Clinical and Siebel Clinical data in CDA through dashboards and reports. The specific dashboards and reports they can view is determined by the user groups they belong to.
- **CDA Programmers** are people who are authorized to create their own reports in the Answers component of OBIEE/CDA, which does not require any programming skills. You can distinguish between people who can simply create ad hoc reports and those who can save the reports they create to a dashboard so that other people can use them.

- **LSH Programmers** are people who can modify the functionality of CDA by modifying the predefined ETL Programs that CDA uses to transform transactional source data in Oracle LSH for use in CDA. They may also create new ETL Programs to support custom dashboards and reports in CDA.
- **LSH Schedulers** are people who schedule CDA jobs, including the data loading job and the user data access jobs. They need privileges similar to LSH Programmers.
- **LSH Administrators** are people who set up Oracle LSH, including Oracle LSH security, and grant privileges to other users.

Setting up security for these user types is described in the following sections and summarized in Table 2–1, Summary of the Oracle LSH Security Setup Example.

# Setting Up User Authentication

Oracle LSH handles user authentication for CDA through its integration with Oracle Applications UMX. You create and maintain user accounts for CDA in Oracle LSH. When a user logs in, OBIEE passes the user name and password to Oracle LSH, which verifies that they are a valid combination and populates the OBIEE Group session variable with a list of the user groups the user belongs to.

## Creating User Accounts

You can create Oracle LSH user accounts in the following ways:

- Create each user account separately through the Oracle Applications UMX user interface. For more information, refer to the Oracle LSH System Administrator's Guide chapter on setting up security.
- If you have an Oracle LDAP Directory, migrate users to Oracle Applications. For more information, refer to My Oracle Support (ID 1508321.1).

When you create an Oracle LSH user account, you assign one or more application roles to it. These roles are different from the object security roles you create and assign to user groups and users within groups. For more information, refer to the Oracle LSH System Administrator's Guide chapter on setting up security. Different users need different roles:

- **CDA End Users**: Give CDA end users the LSH Consumer role.
- **LSH Administrators**: You must create at least one user with each of these application roles: LSH System Admin role, LSH Adapter Security Admin role, LSH Security Admin role, LSH Function Security Admin role, and LSH Groups Admin role.
- **LSH Programmers**: Give LSH Programmers the LSH Definer application role.
- **CDA Programmers**: Give CDA Programmers the LSH Consumer role.
- **LSH Schedulers**: Give LSH Schedulers the LSH Definer application role.

## Creating Database Accounts

**LSH Programmers** need an Oracle LSH database account linked to their user account. For more information, refer to the Oracle LSH System Administrator's Guide chapter on database accounts for information.

# Setting Up User Authorization

Authorization determines which parts of CDA's OBIEE user interface, and in some cases which parts of Oracle LSH, users can access. The tasks required to set up user authorization are:



You can perform all the Oracle LSH tasks in either of two ways:

- through the Oracle LSH user interface. For more information, refer to the Oracle LSH System Administrator's Guide chapter on setting up security.
- using Oracle LSH public APIs. For more information, refer to the Oracle LSH Application Developer's Guide chapter on using APIs.

For conceptual information on Oracle LSH security, refer to the Oracle LSH Implementation Guide chapter on designing a security system. For detailed instructions on all Oracle LSH security setup tasks, refer to the Oracle LSH System Administrator's Guide chapter on security. For background information on the integration of OBIEE with Oracle LSH, refer to the OBIEE section in the Business Areas chapter of the Oracle LSH Application Developer's Guide.

## Using Predefined User Groups in OBIEE and Creating New Ones

All CDA End Users—people who view Oracle Clinical and Siebel Clinical data in OBIEE—must be associated with one or more OBIEE user groups. The OBIEE groups determine privileges allowed to users and allow access to the shipped CDA dashboards and reports. To associate users with an OBIEE user group, you assign their Oracle LSH user account to an Oracle LSH user group with the same name as the required OBIEE user group.

CDA provides a set of predefined OBIEE user groups. You can create additional groups as needed.

> **Note:** To perform administrative tasks in OBIEE, you must be a member of OBIEE's predefined Administrator group.

### Predefined OBIEE User Groups

CDA includes predefined OBIEE user groups (called *groups* in OBIEE) to allow CDA end users access to predefined dashboards. Each dashboard allows access to a predefined set of reports.

The predefined user groups allow dashboard access as follows:

- CDA-StudyManager: CO - Document Management, CO - Site and Recruitment Overview, CO - Study and Region Overview, and CO - Subject Retention
- CDA-CRA: CRA EDC , CO - Document Management, CO - Site Recruitment Overview, CO - Study and Region Overview, and CO - Subject Retention

- CDA-DataEntryManager: CO - Document Management, CO - Site Recruitment Overview, CO - Study and Region Overview, and CO - Subject Retention
- CDA-DataManager: DM EDC, DM Paper, CO - Document Management, CO - Site Recruitment Overview, CO - Study and Region Overview, and CO - Subject Retention
- CDA-ProjectManager: CO - Document Management, CO - Site Recruitment Overview, CO - Study and Region Overview, and CO - Subject Retention
- CDA-Site: CO - Document Management, CO - Site Recruitment Overview, CO - Study and Region Overview, and CO - Subject Retention

The last two user groups are predefined, but if you want to use them you must associate the groups with dashboards.

For more information, refer to Assigning OBIEE User Groups to Dashboards and Reports on page 2-6.

**Note:** CDA ships with both the Presentation catalog and Repository groups for each predefined user group.

### Creating User Groups in OBIEE

You can create additional user groups in OBIEE as needed; for example:

- If you create new dashboards or reports, you may need new user groups to manage access to them.
- To create new dashboards and reports you must allow some users—CDA Programmers in the example—access to the OBIEE Answers component, for which they need to be in a user group with access to Answers.

For each new user group you need, you must create identically named user groups in three places:

- Create a new group in the Presentation catalog: Log in to OBIEE, click **Settings** > **Administration** > **Manage Presentation Catalog Groups and Users**. For more information, refer to the *Oracle Business Intelligence Presentation Services Administration Guide*.
- Create a new group in the OBIEE Repository. If you wish, you can use the group to provide increased security at the RPD level. For more information, refer to the *Oracle Business Intelligence Server Administration Guide*.

  Then in Oracle LSH, navigate to **OCDA_domain** > **OCDA_OBIEE_CODE_APP_AREA** > **OCDA_OBIEE_WA** > **OCDA Data Warehouse**. Check out the Business Area, upload the revised RPD file as source code, and reinstall the Work Area to deploy the revised RPD file. For more information, refer to the Oracle LSH Application Developer's Guide Business Area chapter's section on OBIEE.
- Create a new user group in Oracle LSH. For more information, refer to Creating User Groups in Oracle LSH on page 2-6.

**Note:** The OBIEE Presentation catalog and Repository user groups and the corresponding Oracle LSH user group must all have **exactly** the same name.

### Assigning OBIEE User Groups to Dashboards and Reports

To use a group to allow users access to particular dashboards or reports, you must assign the new group to one or more dashboards or reports.

Log in to OBIEE, click **Settings** > **Administration** > **Manage Interactive Dashboards**. For more information, refer to the *Oracle Business Intelligence Presentation Services Administration Guide*.

## Creating User Groups in Oracle LSH

For users to access any part of CDA's OBIEE user interface, they must belong to an Oracle LSH user group that has a corresponding OBIEE user group of the same name. The user can access the parts of the user interface specified for the OBIEE user group.

Some users, such as the LSH Definer, Scheduler, and Administrator in the example, need to work in Oracle LSH. For users to perform most tasks in Oracle LSH, they must belong to an Oracle LSH user group.

You must create one Oracle LSH user group with the same name as each OBIEE user group, including each of the predefined OBIEE user groups—CDA-Site, CDA-CRA, CDA-DataEntryManager, CDA-DataManager, CDA-ProjectManager—that you want to use.

In addition, you must create Oracle LSH user groups for people who need access to Oracle LSH but not necessarily to CDA.

**Example** To support the example user types, create these user groups:

- **CDA End User Groups**: These correspond to the predefined OBIEE user groups or other OBIEE user groups you create to allow access to dashboards and reports.
- **CDA Programmer Group**: This group is for people who have access to the Answers component of OBIEE.
- **LSH Programmer Group**: LSH Programmers and LSH Schedulers can belong to the same user group, with different roles to differentiate what they can do. You might want CDA Programmers to be in the same group so that they can create and modify ETL Programs to support the dashboards and reports they create.
- **LSH Administrator Group**: You do not need this group unless you want to allow LSH Programmers to modify some ETL Programs but not others. For more information, refer to Assigning Roles to Oracle LSH User Groups on page 2-8 and Roles for Administrators on page 2-8.

## Creating Roles in Oracle LSH

You must create roles in LSH that define which actions a user with that role can take on an object (such as a Program) in Oracle LSH. You then assign roles to user groups. When you assign a user to a group, you must assign them to a role in the group at the same time.

**CDA User Role** CDA End Users **must** have a role with the following privileges:

- View operation on Business Area instances of the Default subtype
- Read Data operation on Table instances of the Default subtype

**Note:** Do **NOT** give the role the View operation on Table instances of the Default subtype. If you do, CDA End Users can use Oracle LSH to see all data in the Table instances to which their user group is assigned, even if you limit their access to particular studies and sites through CDA.

**Note:** To give a role operations on an object subtype in Oracle LSH, you work in the **Security** tab. First define the role in the **Roles** subtab. Then go to the **Subtypes** subtab, find the object, then its Default subtype, and then add the role to the appropriate operation. For more information, refer to the Oracle LSH System Administrator's Guide chapter on setting up security.

**Example** To support the basic user types, you can create the following roles for the example user groups:

### Role for CDA End User Groups

Each Oracle LSH user group corresponding to an CDA user group—including groups you create and the predefined CDA user groups—must have the role described above, which we call the **CDA User Role** in the example.

### Role for CDA Programmers

CDA Programmers need the same Oracle LSH role as CDA End Users: the CDA User Role.

### Roles for LSH Programmers

You can customize your installation of CDA. For more information about how to customize the ETL programs, refer to Chapter 3, Extract Transform Load Programs. To support customization, create roles like the following:

**Note:** In all cases, assign the role to the operation on the Default subtype of the object type listed. All predefined CDA objects are created using the Oracle LSH Default subtype.

- **ETL Modifier** is for people who need to modify the shipped Informatica ETL Programs in Oracle LSH. This role requires the View operation on Domains; the View and Modify operations on Application Areas, Execution Setups, Programs, Program instances, Parameters, and Work Areas; and the Install operation on Work Areas.

  In order to modify the data structures, the role also requires the View and Modify operations on Tables and Table instances and the Create Tables and Create Variables operations on Application Areas.

- **ETL Creator** is for people who need to create new Load Sets and Informatica ETL Programs in Oracle LSH in order to load additional data from Oracle Clinical and Siebel Clinical and transform it to the start schema format. This role requires the View operation on Domains; the View and Modify operations on Application Areas, Execution Setups, Load Sets, Load Set instances, Programs, Program instances, Parameters, Tables, Table instances and Work Areas; the Install

operation on Work Areas, and the Create Program, Create Table, and Create Variable operations operation on Application Areas.

- **RPD Modifier** is for people who need to modify the Repository and the corresponding RPD file in the CDA Business Area in Oracle LSH. This role requires the View operation on Domains; the View and Modify operations on Application Areas, Business Areas, Business Area instances, Execution Setups, and Work Areas; and the Install operation on Work Areas.

  In order to modify the data structures, the role also requires the View and Modify operations on Tables and Table instances and the Create Tables and Create Variables operations on Application Areas.

- **ETL Scheduler**. At least one user needs to be able to schedule execution of the program that refreshes Oracle Clinical and Siebel Clinical data in the CDA data warehouse. This role requires the View operation on Domains and Application Areas; the Submit operation on Execution Setups, and the View operation on Program instances and Work Areas.

In addition, programmers who need to create or modify ETL programs need privileges on the Informatica Adapter, and programmers who need to modify the RPD need privileges on the OBIEE adapter. For more information about the operations required, refer to the Oracle LSH System Administrator's Guide chapter on adapters, section on adapter security.

### Roles for Administrators

Most administrator application roles do not need any object-security roles; that is, roles that grant privileges to perform operations on object subtypes. However, if you decide to specify which ETL Programs LSH Programmers can modify, create the **LSH Security Administrator** role with the Apply Security operation on all container and primary object types, Default subtype.

## Assigning Roles to Oracle LSH User Groups

Assign the roles you have created to the appropriate Oracle LSH user group.

> **Note:** Every Oracle LSH user group is automatically assigned a role called Group Administrator. Only a user with this role in a group can add other users to the group and assign roles to them. Each Oracle LSH user group must have at least one group administrator. This user must have the LSH Groups Admin application role.

**Example** To support the basic user types, you can add roles to user groups as follows:

- **CDA End User Groups**: These correspond to the predefined OBIEE user groups or other OBIEE user groups you create to allow access to dashboards and reports. Assign the **CDA User Role** to each of these groups.
- **CDA Programmer Group**: Assign the **CDA User Role** to this group.
- **LSH Programmer Group**: Assign the roles **ETL Modifier**, **ETL Creator**, **RPD Modifier**, and **ETL Scheduler** to this group.
- **LSH Administrator Group**: Assign the role **LSH Security Administrator** to this group.

## Assigning Oracle LSH User Groups to Objects

CDA users cannot do anything in either OBIEE or Oracle LSH if they do not belong to a user group that is assigned to an Oracle LSH object. You can handle this in different ways:

- A user with the LSH Security Bootstrap Admin application role can assign all user groups to the OCDA_domain. All the predefined CDA Programs, Tables, and other objects automatically inherit the user group assignments. The roles you define limit what users in the user groups can do.
- A user with the LSH Security Admin application role can assign selected user groups to selected objects. This has little advantage for controlling the activities of CDA End Users, but does enable you to allow LSH Programmers to only certain ETL Programs, or to a new Application Area for the purpose of creating new ETL Programs without allowing access to the predefined ETL Programs.

## Assigning Users to Oracle LSH User Groups

You must assign at least one user to the Group Administrator role in each group. Each group administrator then assigns users to the group with an appropriate role. Users can belong to more than one user group.

## Example Summary

The following table summarizes the information in the preceding sections. In addition to the setup displayed in Table 2–1, the user groups must be assigned to Oracle LSH objects. For more information, refer to Assigning Oracle LSH User Groups to Objects on page 2-9.

***Table 2–1 Summary of the Oracle LSH Security Setup Example***

| Example User | LSH Application Role | Example LSH User Group | Example Object Security Role |
|---|---|---|---|
| CDA End User | LSH Consumer | One of the following:<br>CDA-StudyManager<br>CDA-Site<br>CDA-CRA<br>CDA-DataEntryManager<br>CDA-ProjectManager<br>CDA-DataManager | CDA User Role |
| CDA Programmer | LSH Consumer | CDA Programmer Group | CDA User Role* |
| LSH Programmer | LSH Definer | LSH Programmer User Group | One or more of the following:<br>ETL Modifier<br>ETL Creator<br>RPD Modifier* |
| ETL Scheduler | LSH Definer | LSH Programmer User Group | ETL Scheduler |
| LSH Administrator | Each of the following roles must be assigned to at least one user:<br>LSH System Admin<br>LSH Adapter Security Admin<br>LSH Security Admin<br>LSH Groups Admin<br>LSH Security Bootstrap Admin | Each user group must have an LSH Group Administrator.<br>The LSH Security Admin belongs to the LSH Administrator user group (optional).<br>Other Admin users do not need to be assigned to a user group. | LSH Administrator |

***Note:** You may want the same person to have both the CDA User Role in the CDA Programmer user group and the RPD Modifier role in the LSH Programmer user group.

# Setting Up Study and Study Site Data Access for Users

This section contains the following topics:



You can set two variables to either allow all users access to data from all studies and study sites or you can require each user to have explicit access to particular studies and study sites. For more information, refer to Setting the Systemwide Access Variables on page 2-10.

In CDA:

- **Study data** means data pertaining to the study as a whole, including planned sites, planned enrollment, and the ratio of actual to planned subjects. It is not a roll-up of all patient data from all study sites. For security purposes, all documents are considered Study data as well, regardless of whether the document pertains to a Study, a Region, or a Study-Site.
- **Study site data** means all other CDA data, including information about discrepancy management, CRF verification and approval, workloads, and more.

This means that if you set the variables to require explicit access:

- If users need access to study-wide data on planned sites, enrollment, or documents, they must have explicit access to study data for that study. Having access to all study sites does not automatically allow access to study data.
- If users need access to study site data from every site in a study, they must have explicit access to each study site. You can set up this access automatically by importing user privileges from Oracle Clinical or Siebel Clinical. For more information, refer to Importing Study and Study Site Data Access Privileges on page 2-12.

**Note:** If a user has access to multiple, but not all, sites in a study, the totals displayed in CDA reports reflect the totals for the sites to which the user has access, not the totals for all sites in the study. For more information, refer to Study-Site Access Example on page 2-14.

## Setting the Systemwide Access Variables

The following static repository variables determine whether explicit access to study or study site data is required for all users:

- **Enable_Study_Access_Sec**: If set to **Y**, all users must have explicit access granted to study-level data for a particular study in order to see that data. If set to **N**, all users can see study-level data for all studies.

- **Enable_Study_Site_Access_Sec**: If set to **Y**, all users must have explicit access to a particular study site in order to see site-level data for that study site. If set to **N**, all users can see site-level data for all study sites.

  The default value for both variables is **N**.

  **Note:** If you set these variables to **Y** you must populate a set of tables with user access data. For more information, refer to Importing Study and Study Site Data Access Privileges on page 2-12.

  Oracle recommends that you set both variables to the same value.

To change the value for either variable:

1. Stop the BI Server and the BI Presentation Server Services.
2. In Oracle LSH, navigate to **OCDA_domain** > **OCDA_OBIEE_CODE_APP_AREA** > **OCDA_OBIEE_WA** > **OCDA Data Warehouse**, and check out the Business Area.
3. Using the OBIEE Administrator tool, edit the Repository:
   a. On the **Manage** Menu, choose **Variables**.
   b. In the **Variable Manager** dialog, choose **Repository**, then **Variables**, then **Static**.
   c. Open the properties of the variable, either by double-clicking it or through the context menu.
   d. Edit the value of Default Initializer for the variable: **Y** enables access control; **N** disables access control.
   e. Exit the Static Repository Variable dialog.
   f. Exit the Variable Manager.
   g. Save the modified Repository.
4. Upload the new RPD file as the Business Area's Source Code.

   For more information, refer to the Oracle LSH Application Developer's Guide chapter on Business Areas for instructions.
5. Check in the Business Area.
6. Install Work Area OCDA_OBIEE_WA.

   For more information, refer to the Oracle LSH Application Developer's Guide for instructions.
7. Start the BI Server and BI Presentation Server Services.

## Data Access Tables

CDA uses three database tables to control users' access to rows of data in the star schema fact tables that pertain to particular studies and study sites. The data access tables are:

- W_HS_APPLICATION_USER_D contains a list of the user accounts that can have data access granted to particular studies and study sites. It must be populated from an external source. CDA includes a sample ETL Program for this purpose.

For more information, refer to Importing Study and Study Site Data Access Privileges on page 2-12.

- W_HS_STUDY_ACCESS_SEC controls which users can see study-level data on which studies.
- W_HS_STUDY_SITE_ACCESS_SEC controls which users can see study site-level data on which study sites.

## Importing Study and Study Site Data Access Privileges

The data access tables must be populated with data. CDA includes a set of template ETL programs for this purpose. The programs are called *template* programs because you will need to adjust them according to your particular configuration, if you are enabling access control. If you are not enabling access control, the template programs can be used as they are. The following list enumerates the degrees to which you may want to modify the template programs:

- If you set the systemwide access variables to **N**, run the template ETL programs as is to populate the tables with a dummy user. All users have access to all study-level and study site-level data for all studies and sites.
- If you set the systemwide access variables to **Y**, modify the ETL programs as necessary to import user access information from Oracle Clinical and Siebel Clinical. If there are people who should be able to use CDA but do not currently have either Oracle Clinical or Siebel Clinical user accounts with privileges for specific studies or sites set, you must up create user accounts with the desired privileges in one of the source transactional systems.
- If you set the systemwide access variables to **Y**, modify the ETL programs as necessary to import user access information from some other source.

### About Oracle Clinical Template Programs

The template ETL programs for Oracle Clinical are:

- OCDA_INFA_Application_User_D_SDE_OC_PRG
- OCDA_INFA_Study_Access_Sec_SDE_OC_PRG
- OCDA_INFA_Study_Site_Access_Sec_SDE_OC_PRG

The Oracle Clinical table OPA.OPA_LEVEL_PRIVS stores study and study site data access information for Oracle Clinical and Oracle Clinical Remote Data Capture (RDC) Onsite users. The Oracle Clinical or RDC administrator sets these privileges in the Maintain Access to Studies and Maintain Access to Sites windows in either Oracle Clinical or the RDC Administration application.

The template OC ETL programs read data from this table and populate the data access tables in the CDA warehouse in Oracle LSH.

CDA uses this data to allow users access to study and study site data in OBIEE. In Oracle Clinical and RDC the concept of study and study site data access is different from CDA's, and you can specify a variety of privileges on studies and study sites, which is not required in CDA where all data access is view-only. The template CDA ETL programs interpret the Oracle Clinical/RDC data as follows:

- If a user has been granted any privileges on a study site in OPA_LEVEL_PRIVS, the programs give the user study site-level access to that study site in CDA.
- If a user has been given any privileges on a study in OPA_LEVEL_PRIVS, the programs give the user:

- Study-level access to that study in CDA
- Study site-level access to all the study sites in that study

The template ETL programs also remove the Oracle Clinical `OPS$` prefix from each user name. You will likely need to alter this translation of Oracle Clinical user name to CDA user name. For more information, refer to Modifying the Data Access Programs on page 2-13.

### About Siebel Clinical Security ETL Programs

The security ETL programs for Siebel Clinical are:

- OCDA_INFA_Application_User_D_SDE_SC_PRG
- OCDA_INFA_Study_Site_Dim_SDE_SC_PRG
- OCDA_INFA_Study_Access_Sec_SDE_SC_PRG
- OCDA_INFA_Study_Hierarchy_SDE_SC_PRG
- "OCDA_INFA_Study_Site_Hierarchy_SDE_SC_PRG

These programs read from the standard tables describing Siebel Clinical users and protocols, and the access that users have to studies. Review the programs, and adjust them to correspond to any changes you have made from the standard Siebel Clinical model.

The other security ETL programs are:

- OCDA_PLS_Application_User_D_SDE_Pool_PRG
- OCDA_PLS_Study_Access_Sec_SDE_Pool_PRG
- OCDA_PLS_Study_Site_Access_Sec_SDE_Pool_PRG
- OCDA_INFA_Application_User_D_SIL_PRG
- CDA_INFA_Study_Access_Sec_SIL_PRG
- OCDA_INFA_Study_Site_Access_Sec_SIL_PRG

## Modifying the Data Access Programs

You may need to modify the data access ETL programs for the following reasons:

**User Name Conversion Modification** You may need to edit the SDE programs to adapt the user name conversion to your input Oracle Clinical or Siebel Clinical user names and your output CDA user names. Be careful; if the following conditions are not met, names will not match up and access control will fail.

- The conversion performed in the all three SDE programs must be identical
- The resultant user name must be the same as the Oracle LSH user name used for CDA purposes. SDE ETL programs that execute the ETL to populate the data access tables have a parameter for entering the email portion of the standard Oracle LSH user name format.

**Interpretation Logic Modification** You may prefer to interpret the Oracle Clinical or Siebel Clinical privileges differently in CDA.

**Source Modification** You may want to import data access information from another source.

For instructions on modifying ETL programs, refer to Customizing an ETL Program on page 3-20.

### Running the Template Data Access Control ETL Programs

You should run your versions of these programs:

- when you first set up CDA
- when new users need access
- when new studies are added
- when new sites are added to studies
- when the systemwide access variable settings are modified

You must run the programs in the order in which they are listed in Importing Study and Study Site Data Access Privileges on page 2-12. For more information, refer to Scheduling an ETL Program on page 3-24.

## Study-Site Access Example

In Study 012345, users U2 and U3 have study-site access defined in the CDA data access table W_STUDY_ACCESS_STUDY_SITE_SEC as follows (note that user U1 is not in the table at all):

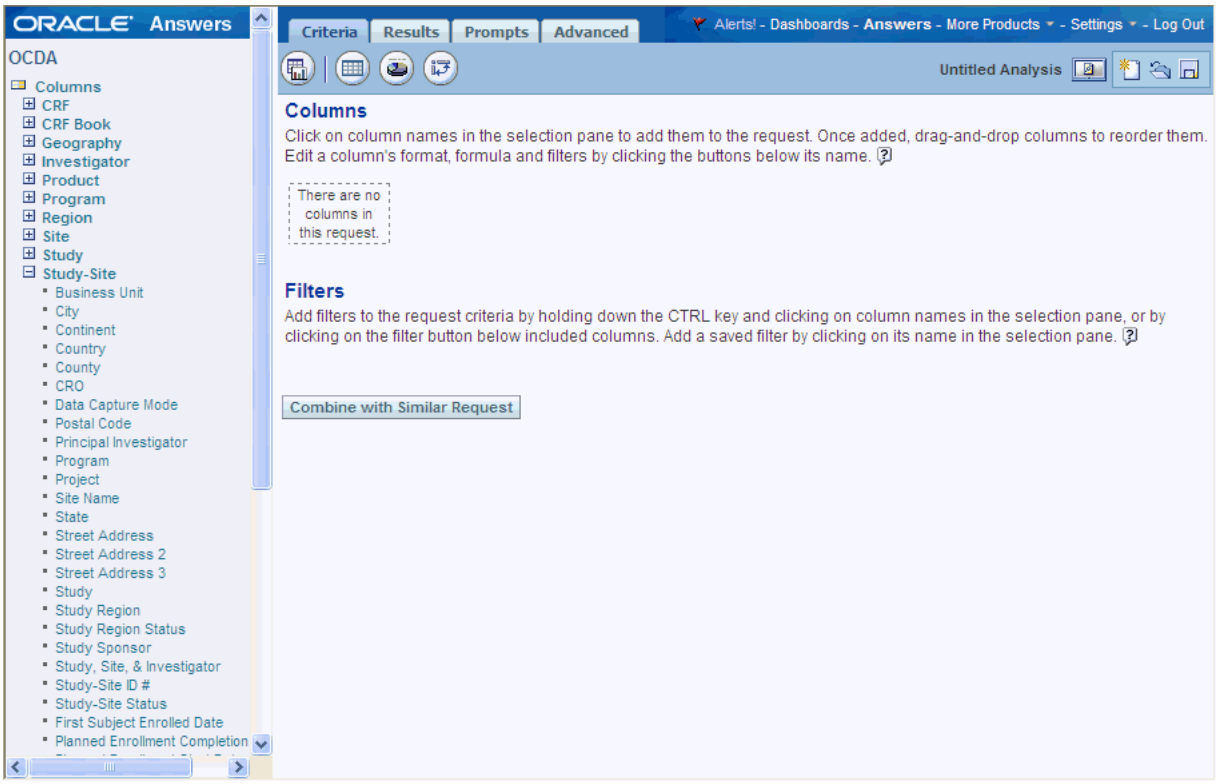
| APPLICATION_USER_WID | STUDY_WID | STUDY_SITE_WID |
|---|---|---|
| U2 | A | A1 |
| U2 | A | A2 |
| U3 | B | B1 |
| U3 | B | B2 |

The distribution of discrepancies by study site, as stored in the discrepancies aggregate table in the warehouse, is:

| Study | Study Site | Number Of Discrepancies |
|---|---|---|
| A | A1 | 20 |
| A | A2 | 15 |
| B | B1 | 30 |
| B | B2 | 10 |
| B | B3 | 20 |

A query on this data has been created and saved as a report:

***Figure 2–2 CDA User Interface***




Users U1, U2, and U3 can run the report. When user U1 runs the report, nothing can be seen. U1 has no access to any study site data.

When user U2 runs the report, U2 sees the following:

| Study | Number of Discrepancies |
|---|---|
| A | 35 |

And U2 drills down within Study A, the following can be seen:

| Study | Site | Number of Discrepancies |
|---|---|---|
| A | A1 | 20 |
| A | A2 | 15 |
| Total | | 35 |

When user U3 runs the report, U3 sees the following:

| Study | Number of Discrepancies |
|---|---|
| B | 40 |

That is, U3 sees the sum of the values for the sites U3 is entitled to see, not the sum for the study. For user U3, it is as if site B3 does not exist. Drilling down shows the same effect:

| Study | Site | Number of Discrepancies |
|---|---|---|
| B | B1 | 30 |
| B | B2 | 10 |
| Total | | 40 |

**Note:** A given document can pertain to study-site, a region, or a study. Ideally, there would be separate security controls for each level. However, in CDA Release 2.0, we are applying the same security to all documents. As every region and study-site belongs to a study, we control documents at the study level.

# 3

# Extract Transform Load Programs

This chapter contains the following topics:



To load data from the source systems to the data warehouse, CDA 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.

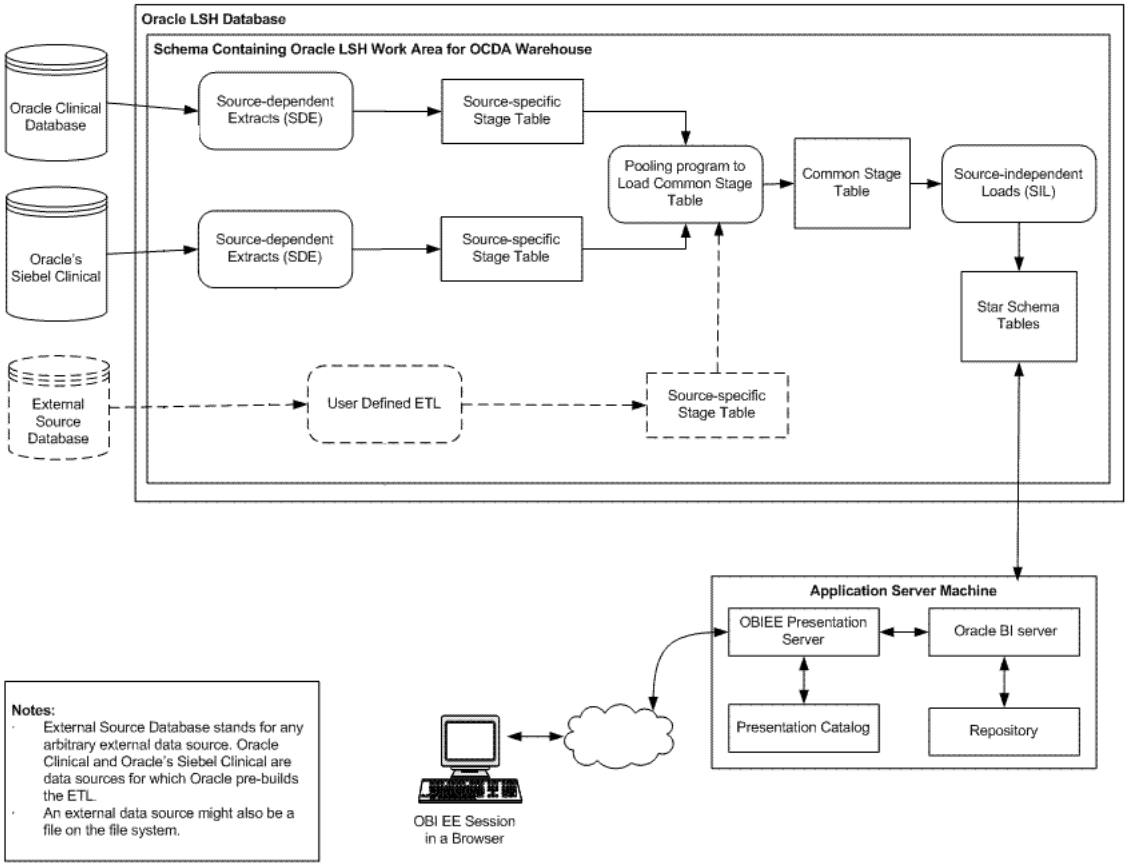
In CDA, Oracle Clinical and Oracle's Siebel Clinical are the source systems for which Oracle provides predefined ETL.

## ETL Architecture

Figure 3-1 displays the ETL process delivered with CDA.

> **Note:** This figure does not cover Multi-source Integration, which is an optional CDA capabililty. For CDA architecture including this option, see Chapter 4, "Multi-Source Integration," on page 1.

***Figure 3–1 The CDA Architecture***




CDA uses Oracle Life Sciences Data Hub (Oracle LSH) to maintain star-schema tables that enable user reporting. Set up as a recurring job, the Oracle LSH Extraction, Transformation, and Load process (ETL) is designed to periodically capture targeted metrics (dimension and fact data) from multiple clinical trial databases, transform and organize them for efficient query, and populate the Oracle LSH star-schema tables.

While the CDA data model supports data extraction from multiple sources, CDA only includes source-dependent extract (SDE) mappings for the Oracle Clinical and Siebel Clinical databases. However, you can also define SDE mappings from additional external sources that write to the appropriate staging tables. Note that you are responsible for resolving any duplicate records that may be created as a consequence. For more information about how to add a new data source to CDA, refer to Adding a New Source System in LSH on page 3-6.

Oracle LSH uses pass-through views to access transactional data from source databases. The SDE programs map the transactional data to source specific staging tables, in which the data must conform to a standardized format, effectively merging the data from multiple, disparate database sources. This is the architectural feature that accommodates external database sourcing.

A pooling program reads the data from all the source specific staging tables, and loads it into the common staging table.

Oracle LSH thence transforms the staged data (in the common staging table) using source-independent loads (SILs) to internal Star-schema tables, where such data are organized for efficient query by the Oracle BI Server.

There is one SDE mapping for each target table, which extracts data from the source system and loads it to the respective source specific staging tables. SDEs have the following features:

- Incremental submission mode: CDA supplied ETL uses timestamps and journal tables in the source transactional system to optimize periodic loads.
- Bulk and normal load: *Bulk load* uses block transfers to expedite loading of large data volume. It is intended for use during initial data warehouse population. Bulk load is faster, if data volume is sufficiently large. However, if load is interrupted (for example, disk space is exhausted, power failure), load cannot be restarted in the middle; you must restart the load.

  *Normal load* writes one record at a time. It is intended to be used for updates to the data warehouse, once population has been completed. Normal load is faster, if data volume is sufficiently small. You can also restart load if the load is interrupted.

  You must set the appropriate target load type for an ETL program in Oracle LSH to indicate bulk and normal load. In CDA, by default, bulk load is enabled for all SDEs.

  **See Also:**

  Setting Up the Target Load Type on page 3-24 for information about setting the appropriate table processing type.

There is one SIL mapping for each target table. The SIL extracts the normalized data from the common 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, CDA overwrites old values with new ones. This strategy is termed as *Slowly Changing Dimension approach 1*.
- CDA's data model includes aggregate tables and a number of indexes, designed to minimize query time.
- By default, bulk load is disabled for all SILs.
- The results of each ETL execution is logged. The logs hold information about errors encountered, during execution.

  Informatica provides the following four error tables:

  – PMERR_DATA
  – PMERR_MSG
  – PMERR_SESS
  – PMERR_TRANS

  During ETL execution, records which fail to be inserted in the target table (for example, some records violate a constraint) are placed in the Informatica PowerCenter 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. CDA provides the W_RXI_DATASOURCE_S table (in RXI schema) that stores all information about all data sources from which data is extracted for CDA. The following are some of the columns in this table:

- ROW_WID - A unique ID for each record in the table.

- DATASOURCE_NUM_ID - The ID for the database. Must be coordinated with the value given to the database when ETL is run.
- 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

See Also:

- *Oracle Health Sciences Clinical Development Analytics Electronic Technical Reference Manual*, for more information about the W_RXI_DATASOURCE_S table.
- Adding a New Source System in LSH, for more information about how to add a new data source to CDA.

### Handling Deletions in Siebel Clinical

CDA 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 CDA installer copies the installation files.
2. Connect to the Siebel Clinical data source and run the `ocda_sc_del_trigger.sql` script delivered with CDA. This script creates the RXI_DELETE_LOG_S table and triggers on tables provided as input. The following are the tables in Siebel Clinical for which CDA 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

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

   Note that when the user deletes 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. Update the remote location of the OCDA_RXI_DELETE_LS load set in OCDA_DELETE_LOG_WA to Siebel Clinical source database connection and install this work area.
   1. Navigate to the OCDA_SOURCES_APP_AREA.
   2. Click **OCDA_DELETE_LOG_WA** work area.
   3. Click **OCDA_RXI_DELETE_LS** loadset.
   4. Click **Check Out.**

5. Click **Apply.**
6. In the Load Set Attributes section, click **Update**.
7. Click the Search icon.
8. Select **OCDA_SC_OLTP_RL/<Connection_Name>.**
9. Click **Apply.**
10. Reinstall the work area containing the load set and passthrough views.

For more information, refer to *Oracle Health Sciences Clinical Development Analytics Installation Guide (Post Installation Tasks)*

4. Modify the value of the DELETE_FLOW submission parameter for the following dimension programs based on the triggers created in step 2:
   - `OCDA_INFA_Study_Dim_SDE_SC_PRG`
   - `OCDA_INFA_Study_Site_Dim_SDE_SC_PRG`
   - `OCDA_INFA_Study_Region_Dim_SDE_SC_PRG`
   - `OCDA_INFA_Program_Dim_SDE_SC_PRG`
   - `OCDA_INFA_Product_Dim_SDE_SC_PRG`
   - `OCDA_INFA_Study_Subject_Dim_SDE_SC_PRG`
   - `OCDA_INFA_Party_Per_Dim_SDE_SC_PRG`
   - `OCDA_INFA_SS_Con_Dim_SDE_SC_PRG`

   Perform the following steps:
   a. Navigate **OCDA_domain** > **OCDA_CODE_APP_AREA** > **OCDA_SDE_SC_WORK_AREA**.
   b. Click the Name hyperlink of the program.
   c. Click **Submit**.
   d. Enter the following information in Submission Details:
      – Submission Type: **Backchain**
      – Force Execution: **Yes**
   e. In Submission Parameters, enter the value of DELETE_FLOW as **Y**. The default value is N, which indicates that CDA does not handle deletion in Siebel Clinical.
   f. Click **Submit**.
5. Execute the ETLs as listed in the Executing the ETL Programs section.

   The Siebel Clinical related SDE mappings reads 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.

**See Also:**

*Oracle Life Sciences Data Hub User's Guide, (Tracking Job Execution)*, for more information about viewing job execution logs.

## Adding a New Source System in LSH

The steps below will consider Oracle Clinical work areas as an example to demonstrate adding new OLTP source system to be read for the ware house load.

The process involves:

1. Creating a remote location for the new source OLTP pass-through views.
2. Creating a clone and configuring the new replica of the OC source Pass through the view's work area.
3. Creating a clone and configuring the new replica of the OC SDE work area and programs.
4. Modifying the pool program to include the new source specific stage tables.

### Creating Remote Location for New Source OLTP Pass-through Views

This section describes how to create a LSH remote location which connects to the new instance of the OC OLTP source system.

Perform the following steps to configure the remote location OCDA_OC_OLTP_RL_Inst2:

1. Click the **Remote Location** subtab under the **Administration** tab. The **Maintain Remote Locations** screen opens.
2. Click **Add Remote Location**. The **Create Remote Location** screen appears.
3. Enter values in the following fields:
   - **Remote Location Name** - `OCDA_OC_OLTP_RL_Inst2`.
   - **Description** - Description of the remote location.
   - **DBLINK Prefix** - The name of the database link. If another DBLINK Prefix with the same name exists in the database, the system adds an additional string to make it unique. The DBLINK_NAME is usually the global name or the TNS name of the remote database.
   - **Connect String** - The name of the string that Oracle LSH must use in the USING clause of the create database link SQL statement. Connect string has following format:

     ```
     ((DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=hostname)(PORT=
     dbportnumber))(CONNECT_DATA=(SID=dbsid))))
     ```

   - **Adapter** - Select **Oracle Tables and Views** from the drop-down list.
4. Click **Apply**.

To configure the connection for location OCDA_OC_OLTP_RL_Inst2:

1. Select the remote location just created.
2. Click **CREATE CONNECTION** and provide the following OPA connection details:
   - **Name** — Enter a name for the connection. For example, RXA_DES.
   - **User Name** — Enter the database username. For example, RXA_DES.
   - **Password** — Enter the database password for above user.
3. Click **Apply**.
4. Repeat steps 1 through 3 for the other two database connections RXC, OPA.

**See Also:**

*Oracle Life Sciences Data Hub System Administrator's Guide (Chapter 6, Registering Locations and Connections)*, for more information on registering locations and connections in Oracle LSH.

### Creating Clone and Configuring New Replica of OC source Pass Through View's Work Area

This sections describes how to replicate a source work area and set the connection of load set.

1. Navigate to **OCDA_domain** > **OCDA_SOURCES_APP_AREA** and select the **OCDA_OC_DATA_WA**.
2. Click **CLONE**.
3. On the **Clone Destination** page, select **OCDA_SOURCES_APP_AREA** within OCDA_domain.
4. Provide the desired clone label.
5. Click **Review** and select **Finish**.
6. A replica of New OC Sources work area is created with the name **OCDA_OC_DATA_WA_1**.
7. To rename it, click **OCDA_OC_DATA_WA_1**.
8. Click **Update** and modify the name to **OCDA_OC_DATA_WA_Instance_2**.
9. Click **Apply**.
10. Navigate to **OCDA_domain** > **OCDA_SOURCES_APP_AREA** > **OCDA_OC_DATA_WA_Instance**_2.
11. Check individual Loadset Instances:
    - OCDA_OC_OPA_LS
    - OCDA_OC_RXA_DES_LS
    - OCDA_OC_RXC_LS

    Update their respective Load Set Attributes with **remote location/connections** which was created in Creating Remote Location for New Source OLTP Pass-through Views on page 3-6.

Install the OCDA_OC_DATA_WA_Instance_2 work area in full mode.

### Creating Clone and Configuring New Replica of the OC SDE Work Area and Programs

This section describes how to create a clone of OC SDE work area and configure it to read from the new OC pass-through work created in Creating Clone and Configuring New Replica of OC source Pass Through View's Work Area on page 3-7.

1. Create new application area and clone the SDE OC work area.
    a. Click **OCDA_domain** and select **Add Application Area**.
    b. Create a new application area with name **OCDA_CODE_APP_AREA_NEW**.
    c. Click **Apply**.
    d. Navigate to **OCDA_domain** > **OCDA_CODE_APP_AREA** and select **OCDA_SDE_OC_WORK_AREA**.

e. Click **CLONE**.

f. On the **Clone Destination** page, select **OCDA_CODE_APP_AREA_NEW** within OCDA_domain.

g. Provide the desired clone label.

h. Click **Review** and select **Finish**.

i. A replica of the new OC SDE work area named **OCDA_SDE_OC_WORK_AREA** is created.

j. To rename it, click **OCDA_SDE_OC_WORK_AREA**.

k. Click **Update** and modify the name to **OCDA_SDE_OC_WORK_AREA_Instance_2**.

l. Click **Apply**.

2. Modify the table instances.

a. Navigate to **OCDA_SDE_OC_WORK_AREA_Instance_2**.

b. Click on the Table instance **W_HS_OC_APPLICATION_USER_DS**.

c. Click **Update** and change the fields:

**Name** - For example, W_HS_OC_APPLICATION_USER_DS_1.

**Oracle Name**

**SAS Name**

---

**Note:** Ensure that the **Name**, **Oracle Name** and **SAS Name** above should not be beyond 30 chars. Exceeding this limit will cause errors in any program that reads from this table instance.

---

3. Modify the table descriptors' mapping and set the parameter for each SDE program.

This step describes how to unmap and remap the source table descriptors to the new OC Source instance work area that was created in Creating Clone and Configuring New Replica of OC source Pass Through View's Work Area on page 3-7.

a. Click on **OCDA_INFA_Application_User_D_SDE_OC_PRG** in OCDA_SDE_OC_WORK_AREA_Instance_2.

b. Click **Check out**.

c. Select **Copy definition to the local Application Area and checkout**.

---

**Note:** This will check out the SDE program and create a local copy of it thereby not affecting the definition of the program, which was cloned for the original **OCDA_doman** > **OCDA_CODE_APP_AREA**. When further CDA patches are applied, programs within this work area will remain unaffected.

---

d. Select **Table Descriptors**. Each source table descriptor can be identified by **Is Target = No** in the table that shows all table descriptors.

**Note:** Ignore W_CONTROL_S table as it is not an Oracle Clinical source table.

e. Click the following mapping icon:

f. Click **Update**, then **Unmap** and **Apply**.

g. Repeat step d for all the source table descriptors.

h. To remap all the source table descriptors for a SDE program, select **Automatic Mapping by Nam**" from the **Actions** drop down.

i. To get to OC source pass through view's work area that was created in Creating Clone and Configuring New Replica of OC source Pass Through View's Work Area, select:

**Domain** - OCDA_domain.

**Application area** - OCDA_SOURCES_APP_AREA

**Workarea** - OCDA_OC_DATA_WA_Instance_2

j. Select the source table for current program that is in consideration.

k. Click **Map**. This will map all the table descriptors.

l. Click the **Parameter** tab for the program that is in consideration.

m. Change the default for the parameter **DATASOURCE_NUM_ID** from 1 to 3. This number should be unique the program within this work area and should not be repeated.

n. Repeat steps a through m for all the Programs in the OCDA_SDE_OC_WORK_AREA_Instance_2 work area.

Once the entire program's mapping and parameters are changed install the work area in full mode. Create a new execution setup and submit it in back chain for all the programs.

### Modifying Pool Program to Include the New Source-specific Stage Tables

This section describes how to modify the pool program to include table instances that were created in Creating Clone and Configuring New Replica of the OC SDE Work Area and Programs on page 3-7. By performing the following steps, newly created OC SDE target table instances will propagate data to the target dimension and fact warehouse tables.

1. Navigate to **OCDA_doman** > **OCDA_CODE_APP_AREA** > **OCDA_POOL_WORK_AREA**.
2. Click **OCDA_PLS_Application_User_D_SDE_Pool_PRG**.
3. Click **Check out**.
4. Check out the existing definition and click **Apply**.
5. To add the new table descriptor for this program, select **Create Table Descriptors from Table Instances** from the **Actions** drop down.

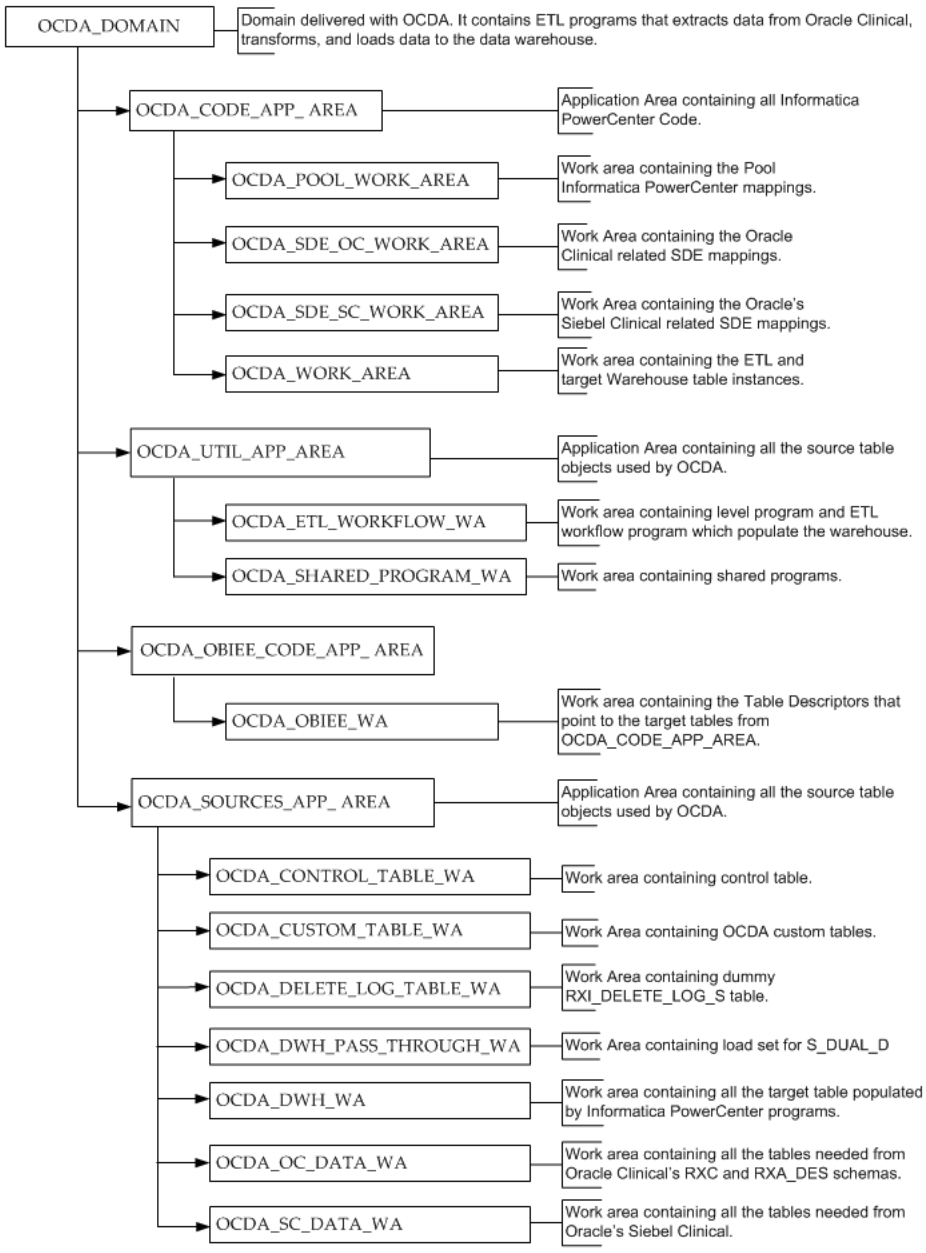
6. To get to the OC source SDE work area that was created in Creating Clone and Configuring New Replica of the OC SDE Work Area and Programs, select the following:
   - **Domain** - OCDA_domain
   - **Application area** - OCDA_SOURCES_APP_AREA_New
   - **Workarea** = OCDA_SDE_OC_WORK_AREA_Instance_2
7. Select the Table Instance **W_HS_OC_APPLICATION_USER_DS_1** that was created in Creating Clone and Configuring New Replica of the OC SDE Work Area and Programs.
8. Click **Create table descriptor**.
9. Repeat steps 2 to 9 for all the programs in the **OCDA_POOL_WORK_AREA** work area. This will read all OC SDE Target table instance data corresponding to their SDE's.

After the entire program's table descriptors are added into their respective pool programs, install the work area in full mode. Create a new execution setup and submit them in back chain for each of programs in OCDA_POOL_WORK_AREA.

## Oracle Health Sciences Clinical Development Analytics Domain Structure in Oracle Life Sciences Data Hub

Figure 5-2 displays the CDA domain hierarchy in Oracle LSH:

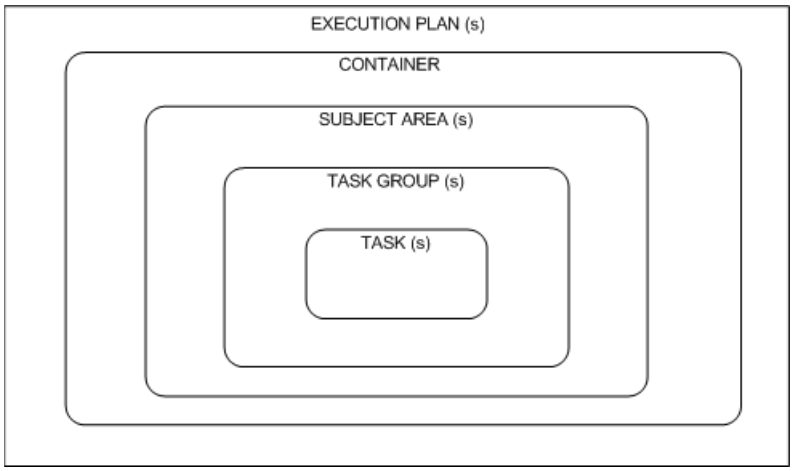
***Figure 3–2 CDA Domain Hierarchy in Oracle LSH***




## ETL Mapping Hierarchy

This section describes the ETL mapping in DAC. Figure 3-4 displays the  hierarchy:

***Figure 3–3 ETL Mapping Hierarchy***




Following is the ETL mapping hierarchy:

- CONTAINER (CDA_Warehouse) - A single container that holds all objects used for OHSCDA. For deduplication, however, there is a container for every deduplicated dimension that holds all the objects involved in deduplication.
- EXECUTION PLAN - A data transformation plan defined on subject areas that needs to be transformed at certain frequencies of time. An execution plan is defined based on business requirements for when the data warehouse needs to be loaded. Single Execution Plan to Load Complete Warehouse.
- SUBJECT AREAS - A logical grouping of tables related to a particular subject or application context. It also includes the tasks that are associated with the tables, as well as the tasks required to load the tables. Subject areas are assigned to execution plans, which can be scheduled for full or incremental loads.
- TASK GROUPS - This is a group of tasks that should be run in a given order.
- TASKS - A unit of work for loading one or more tables. A task comprises the following: source and target tables, phase, execution type, truncate properties, and commands for full or incremental loads. Each task is a single Informatica workflow.

## Executing the ETL Programs

ETL Programs are responsible for bringing the warehouse target tables up to date with respect to the current state of the source application database tables. You run ETL programs as either a Full load or an Incremental Load.

You perform a Full load when you want to completely refresh the target tables. This occurs when you first load the warehouse, and at any subsequent time when you need to do a complete refresh. Full loads drop the indexes on the warehouse target tables, truncate the tables, load all the appropriate records from the source into the target tables, and rebuild the indexes on the target tables.

Once a full load has been done, you run periodic Incremental loads to update the warehouse target tables with changes that have occurred in the source tables since the last prior load.

CDA has an optional capability, called Multi-Source Integration, which enables you to perform deduplication of data coming from multiple source databases. This capability

is described in Section Multi-Source Integration below. If you use it, Deduplication requires the execution of additional ETL; additionally, the method for triggering the execution of the ETL differs if you use Deduplication.

The processes for executing CDA's ETL is described below in four sections, as outlined here:



## ETL Execution for Full Data Warehouse Load

**Important:** Before you reinstall, ensure that:

- Informatica DP Server is up and running
- LSH Job Queue is running

If you are running Deduplication ETL, you must also ensure that the LSH Message Queue is running.

Refer to *Oracle Life Sciences Data Hub System Administrator's Guide* for information on these components.

This section gives steps for full load of the warehouse target tables. You will need to do a full load after the initial install of the CDA software. This is referred to as an Initial Load. You may subsequently need to do additional full loads; these are referred to as Reloads. A Reload would be necessary, for example, if you initially load from a test database, and then shift to a production database.

If this is the first time you are executing the ETL programs after the initial install, or you need to reload the complete warehouse, follow the steps given below.

Certain steps are required only if doing a reload, rather than an initial load. They are called out accordingly.

### Full Load Without Deduplication

This section gives information on setting up and executing the LSH program to perform a full load to load, in the case where you are not running the Deduplication ETL.

1. Navigate to **OCDA_domain** > **OCDA_SOURCES_APP_AREA** > **OCDA_DWH_PASS_THROUGH_WA**

**Note:** Ensure that you have installed the domain, Application Area, and Work Area in Oracle LSH before you can perform the subsequent steps.

For more information, refer to *Oracle Health Sciences Clinical Development Analytics Installation Guide (Post Installation Tasks)*.

2. If you are executing the ETL programs for the first time after installing CDA, submit the OCDA_PLS_S_DUAL_D_PRG program before submitting any fact in backchain.
3. Navigate to **OCDA_domain** > **OCDA_CODE_APP_AREA** > **OCDA_WORK_AREA**.

   **Note:** Ensure that you have installed the domain, Application Area, and Work Area in Oracle LSH before you can perform the subsequent steps.

   For more information, refer to *Oracle Health Sciences Clinical Development Analytics Installation Guide (Post Installation Tasks)*.

4. If you are executing the ETL programs for the first time after installing CDA, submit the following programs in the given order before submitting any fact in backchain:
   a. OCDA_PLS_DUAL_PRG
   b. OCDA_INFA_DayDimension_SIL_PRG
   c. OCDA_INFA_MonthDimension_SIL_PRG

   **Important:** Ensure that you submit the above programs every time you modify them.

5. Perform this step *only* if you are reloading the complete warehouse. In this step, you clear all records from the W_CONTROL_S table.
   1. Navigate to **OCDA_domain** > **OCDA_SOURCES_APP_AREA** > **OCDA_CONTROL_TABLE_WA**.
   2. Click the OCDA_CONTROL_TABLE_POPULATE_PRG hyperlink.
   3. Click **Submit**.
   4. Enter the following values for submission parameters:

      **Config_days:** Accept the default. This parameter is ignored when Delete_ mode is selected.

      **Delete_mode:** ALL

      **Input_values:** Leave this parameter blank.
   5. In Submission Type, select **Immediate**.
   6. In Force Execution, select **Yes**.
   7. Click **Submit**.

      Navigate to my home and monitor the job. This will clear all the entries from the control table
6. Submit the program to populate the control table in backchain.
   1. Navigate to **OCDA_domain** > **OCDA_SOURCES_APP_AREA** > **OCDA_CONTROL_TABLE_WA**.
   2. Click the OCDA_CONTROL_TABLE_POPULATE_PRG hyperlink.
   3. Click **Submit**.

4. Enter the following values for submission parameters:

   **Config_days:** Offset relative to the beginning of the current day, used to determine the latest record timestamp that will be loaded. See the following note for considerations in setting Config_days.

   **Delete_mode:** None

   **Input_values:** Leave this parameter blank.

   **Considerations for Config_days parameter**

   The Control Table (W_CONTROL_S) stores the time span that determines which source records for base dimensions and facts are extracted from the database during each ETL execution. That is, source records for base facts are extracted if their creation or modification timestamp is between the start and end timestamps specified in the Control Table record for a given ETL execution.

   These endpoints are defined before the execution of the ETL mapping begins. The start timestamp is the end timestamp of the previous execution of the same ETL mapping. The end timestamp is calculated as follows:

   a. Start with the current time at the warehouse.

   b. Truncate that time, yielding the midnight that started the current day

   c. Subtract the value of the CONFIG_DAYS parameter. CONFIG_DAYS is expressed in days, or fractions of days, and defaults to a value of 1.

   The truncation is performed in case the source is in a later timezone than the warehouse. This ensures that no record will fail to load because it falls into a gap between the end of one load's timespan and the start of the next.

   CONFIG_DAYS can be used to ensure that there are no temporarily "orphaned" fact records. An orphaned fact record will be created in the warehouse if a dimension, and a fact that depends on it, are created in the source database during the period after a parent dimension has been loaded but before the fact load starts. This is avoided by setting CONFIG_DAYS to a value greater than zero.

   It is possible, for example, that after the data for the study-site dimension have been loaded, but before the received CRF fact has been loaded, that a new study site would be created in the source database, and some received CRFs recorded. With config_days at zero, the received CRFs would be loaded, but their parent study-site would not be.

   Lacking a proper parent for these received CRF records , CDA would set their foreign key to the Unknown study-site. Reports would reflect that allocation; users would not be able to tell where the orphaned CRFs came from.

   Orphaned records remain orphaned only until the next ETL execution; at that time the absent parent records are extracted, and the foreign keys are adjusted. But the CONFIG_DAYS parameter is provided to enable you to avoid having any records be temporarily orphaned.

   By setting it to one, you instruct CDA to not load any base fact records from a source, if they were created or modified less than a day before loading from that table commences. As long as your entire load takes less than a day, you will have no orphaned fact records. But the other side of the coin is that your warehouse will reflect the state of the database as of a day before the ETL ran.

If reporting that approaches real-time is valued more than avoiding temporary orphans, you will want to set CONFIG_DAYS to a value close to zero. You also should adjust CONFIG_DAYS to reflect the actual frequency of creation of new dimension records, and fact records dependent on them, in your source database. The expected frequency may be less than once/day, in which case you would be safe to set CONFIG _DAYS to less than 1.

5. In Submission Type, select **Backchain**.
6. In Force Execution, select **Yes**.
7. Click **Submit**.

7. Adjust Table Descriptors depending on source applications used
   1. Navigate to **OCDA_domain** > **OCDA_UTIL_APP_AREA** > **OCDA_ETL_WORKFLOW_WA**.
   2. If Oracle Clinical is your *only* data source, remove the following Table Descriptors from `OCDA_PLS_LEVEL1_FACT_PRG`:

      `W_ACTIVITY_F`

      `W_RXI_RGN_ENRLMNT_PLN_F`
   3. If Siebel Clinical is your *only* data source, remove the following Table Descriptors from `OCDA_PLS_LEVEL1_FACT_PRG`:

      `W_RXI_DISCREPANCY_F`

      `W_RXI_DISCRPNCY_STATUS_F`

      `W_RXI_RECEIVED_CRF_F`
   4. Install the program `OCDA_PLS_LEVEL1_FACT_PRG`.
   5. If Siebel Clinical is your *only* data source, remove the following Table Descriptors from `OCDA_PLS_LEVEL1_AGG_FACT_PRG`:

      `W_RXI_DISCREPANCY_A`

      `W_RXI_DISCRPNCY_STATUS_A`

      `W_RXI_RECEIVED_CRF_A`
   6. Install the program `OCDA_PLS_LEVEL1_AGG_FACT_PRG`.
8. Submit the job to execute the Full Load:
   1. Navigate to **OCDA_domain** > **OCDA_UTIL_APP_AREA** > **OCDA_ETL_WORKFLOW_WA**.
   2. Click the `OCDA_PLS_ETL_WORKFLOW_PRG` hyperlink.
   3. Click **Submit**.
   4. Enter the following values for submission parameters:

      **Config_days:** Must be the same value as you specified for the OCDA_CONTROL_TABLE_POPULATE_PRG program in Step 6 of this section.

      **MATCH_MERGE_FLOW:** N

      **FULL_LOAD:** Y
   5. In Submission Type, select **Immediate**.
   6. In Force Execution, select **Yes**.
   7. Click **Submit**.

**Note:** Execution of the ETL (specifically the `OCDA_ETL_RUN_S_POP_PRG` 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 CDA 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.

There are two remedies: (i) restart the Oracle BI Server after every execution of the ETL. This will cause CURRENT_DAY to be refreshed to the correct value. (ii) 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, refer to Chapter 1, "Maintaining the Repository and Warehouse," on page 1-1

## Full Load With Deduplication

This section gives information on executing the ETL programs for full load when the ETL includes both the direct path ETL and also the Deduplication ETL.

This section assumes that:

- You have two or more source application databases. This can be one OC and one SC database, multiple OC databases, multiple OC databases and an SC database, or (unlikely) multiple SC databases along with zero or more OC databases.
- You are using OHMPI for deduplication of dimension data from the source databases.
- You have installed OHMPI and loaded the pre-defined Projects for identifying matches on dimensions.

### Prerequisite

Execute steps 1 to 6 from Full Load Without Deduplication on page 3-13. This loads confirmed dimensions.

Refer to Chapter 4, "Multi-Source Integration" on page 4-1and the OHMPI documentation, for details of how to set up and use OHMPI for deduplication of CDA dimension data.

### Execute the initial load of each dimension into the Master Index.

1. For each source database, perform the following steps:
    1. Determine the execution plan for the source applications from which you are deduplicating data, according to the following table:

       **CDA - Oracle Clinical Initial De Dup**: Oracle Clinical database or databases

       **CDA - Siebel Clinical Initial De Dup**: Oracle Siebel database or databases

**CDA - Complete Initial De Dup**: One Oracle Clinical and one Siebel Clinical database

2. In DAC, define the source database with the selected Execution Plan. If you have multiple instances of a source application database, add a copy of the appropriate execution plan for the subsequent instances.
3. Run the selected Execution Plan. This generates a flat file for each dimension extracted from the database.

2. Clean up each flat file. If you fix errors by modifying the source database, use the Initial DeDup execution plans to re-extract the data to flat files.
3. For each dimension, combine the flat files that have been generated for that dimension into a single flat file.
4. For each dimension, run the dimension's project to identify matches. Review the results, adjust the project's rules and re-execute the project until they generate the optimum set of non-matches, potential matches, and matches for your data. When satisfied run the project in Bulk Load mode so that it places its results in the dimension's Master Index.
5. For each dimension, perform data stewardship.

   This completes the initial load of the dimensions into their Master Indexes.
6. Confirm that the LSH Job Execution Message Queue is running.

#### Use DAC to trigger the execution of Deduplication and Direct Path ETL

Since you are using Deduplication, you must submit the program via through DAC console. This is necessary to properly coordinate the execution of the ETL on deduplication path before executing the ETL on the direct path. Perform the following steps to start the combined ETL through the DAC console:

1. Select the CDA_Employee_De_Dup container.
2. Navigate to the Design Task. Select P**LP_Start_LSH_MasterProgram** task and navigate to the **Task Level Parameters** tab.
3. Set parameters to the following values:

   **PIN_DOMAIN**: OCDA_domain

   **PIN_APP_AREA**: OCDA_UTIL_APP_AREA

   **PIN_WORKAREA**: OCDA_ETL_WORKFLOW_WA

   **PIN_PROGRAM**: OCDA_PLS_ETL_WORKFLOW_PRG

   **PIN_EXESETUP**: OCDA_ES

   Enable The triggered option for the OCDA_ES execution setup.

   **PIN_FL**: Y

   **PIN_USERID**: CDRMGR@ORACLE.COM

   This is LSH application user who has access to OCDA_domain and has privileges to execute the ETL.

   Ensure that you have a LSH User Database Account for the user.
4. Navigate to the **Execute** view and select one of the sources specific execution plans. Selection of the execution plan is based on the source system that you own.

   **CDA - Oracle Clinical Initial De Dup**: Oracle Clinical database or databases

**CDA - Siebel Clinical Initial De Dup**: Oracle Siebel database or databases

**CDA - Complete Initial De Dup**: One Oracle Clinical and one Siebel Clinical database

5. Set the parameter values on the **Parameter** tab.
6. Build the execution plan.
7. Click **Run**.

## ETL Execution for Incremental Data Warehouse Load

This section lists steps for incremental load of warehouse tables.

The first subsection describes incremental loads if you are using direct path loads only (no deduplication).

The second subsection describes how to perform incremental loads if you are using deduplication in addition to direct-path loading.

> **Tip:** You can schedule the jobs to execute at regular intervals. For more information on scheduling jobs, refer to Scheduling an ETL Program on page 3-24.

### Incremental Load Without Deduplication

To perform incremental loads without deduplication, you need to submit an LSH Job. Perform the following steps to do so:

1. Navigate to **OCDA_domain** > **OCDA_UTIL_APP_AREA** > **OCDA_ETL_WORKFLOW_WA**.
2. Click the `OCDA_PLS_ETL_WORKFLOW_PRG` hyperlink.
3. Click **Submit**.
4. Enter the following values for submission parameters:

   **Config_days:** Must be the same value as you specified for the OCDA_CONTROL_TABLE_POPULATE_PRG program.

   **MATCH_MERGE_FLOW:** N

   **FULL_LOAD:** N

   > **Caution:** You must set FULL_LOAD parameter to N else full load will be executed.

5. In Submission Type, select **Immediate**.
6. In Force Execution, select **Yes**.
7. Click **Submit**.

### Incremental Load With Deduplication

To perform incremental loads including deduplication, you need tosubmit the Job through the DAC console. This is necessary to properly coordinate the execution of the ETL on deduplication path before executing the ETL on the direct path. Perform the following steps to start ETL through the DAC console:

1. Confirm that the LSH Job Execution Message Queue is running.

2. In DAC, select the CDA_Employee_De_Dup container.
3. Navigate to the Design Task. Select **PLP_Start_LSH_MasterProgram** task and navigate to the **Task Level Parameters** tab.
4. Set parameters to the following values:

   **PIN_DOMAIN**: OCDA_domain

   **PIN_APP_AREA**: OCDA_UTIL_APP_AREA

   **PIN_WORKAREA**: OCDA_ETL_WORKFLOW_WA

   **PIN_PROGRAM**: OCDA_PLS_ETL_WORKFLOW_PRG

   **PIN_EXESETUP**: OCDA_ES

   Enable The triggered option for the OCDA_ES execution setup.

   **PIN_FL**: N

   This is important. The setting of 'Y' will trigger a full reload.

   **PIN_USERID**: CDRMGR@ORACLE.COM

   This is LSH application user who has access to OCDA_domain and has privileges to execute the ETL.

   Ensure that you have a LSH User Database Account for the user.
5. Navigate to the **Execute** view and select one of the sources specific execution plans. Selection of the execution plan is based on the source system that you own.

   **CDA - Oracle Clinical Initial De Dup**: Oracle Clinical database or databases

   **CDA - Siebel Clinical Initial De Dup**: Oracle Siebel database or databases

   **CDA - Complete Initial De Dup**: One Oracle Clinical and one Siebel Clinical database
6. Set the parameter values on the **Parameter** tab.
7. Build the execution plan.
8. Click **Run**.

## Customizing an ETL Program

The following rules apply when you customize an ETL program:

- You can customize ETL programs in different ways. You can create a new domain, new Application Area within same domain, or a new Work Area within the same Application Area. Creating a new domain, and storing customized definitions in the new domain will ensure that the definitions are not overwritten on the next CDA upgrade.

  Oracle recommends that you set up a single domain for all customization to CDA. This will ensure that all the customized object definitions are available in the same top-level container.
- After you create your own Work Area, clone the Oracle-supplied Work Area on to your own Work Area. This creates copies of object instances inside your Work Area, but they point to the object definitions inside the Oracle-supplied Application Area.

**Caution:** To correctly track the timing of CDA ETL execution, Oracle recommends that no Program in the CDA Domain, other than the SIL provided with CDA, read from any CDA staging table. The staging tables, at any given time, hold only transient records that were loaded during the most recent ETL execution. Ideally, you may not read from them. If it is necessary to read from an CDA staging tables, Oracle recommends that you define the Program in a Domain other than the CDA Domain containing the staging table, and execute it in that separate Domain.

## Creating an ETL Program

Though CDA includes ETL programs for extracting data from Oracle Clinical and Siebel Clinical to CDA 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 and *2* for Siebel Clinical. If you want to add your own data sources, set this value to a number greater than 100.

**See Also:**

- *Oracle Life Sciences Data Hub Application Developer's Guide (Defining Programs)*
- *Informatica PowerCenter Online Help*

To add one or more tables or columns along with the associated ETL programs to populate data into these table, perform the following tasks:

1. Create the new source and target table metadata inside your Work Area.

   If the tables exist outside Oracle LSH in some remote schema, flat file, or SAS file, you can upload the table structure into Oracle LSH using an Oracle LSH Load Set.

2. Create a Program in Oracle LSH and specify that the Program is an Informatica-type Program.
3. Add these tables as sources or targets.
4. Install the new Oracle LSH Program to ensure that the new tables are created in the Work Area schema.
5. Check out your new Oracle LSH program.

   **Important:** Do not use the **Copy definition to the local Application Area and check out** option when checking out the program.

6. In the Program's screen, click **Launch IDE**.

   This launches Informatica PowerCenter client installed on your machine.

7. Work in Informatica PowerCentre and create the ETL components (transformation or workflow) used by this Oracle LSH Program.
8. Go back to Oracle LSH and upload the ETL file from Informatica PowerCenter to Oracle LSH.
9. Install and run the Program in Oracle LSH.

**Important:** Before you reinstall, ensure that the Informatica DP Server is up and running.

**Tip:** If the target table you added relies on new source tables, and if you want the source tables to be automatically populated when you trigger the ETL program for the final target table, enable backchaining for the Oracle LSH Program that populates the source tables.

If there are no new source tables in your customization, Oracle LSH will automatically trigger the population of the source tables when the ETL program to populate the final target table is executed. Note that the backchain submissions are not cloned to the Work Area, and you have to manually create them.

For more information on backchaining, refer to *Oracle Life Sciences Data Hub Application Developer's Guide (Execution and Data Handling)*.

## Modifying an ETL Program

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

**See Also:**

- *Oracle Life Sciences Data Hub Application Developer's Guide (Defining Programs)*
- *Informatica PowerCenter Online Help*

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

1. Install your Work Area and run the ETL to ensure that you can see data populated in the target data warehouse tables.
2. Identify the Oracle LSH program that contains the metadata for the ETL that needs to be modified.
3. Check out the Oracle LSH program that contains the metadata for that ETL.

   **Important:** Use the **Copy definition to the local Application Area and check out** option to check out the program. This ensures that you do not modify the definitions inside the domain shipped with CDA.

4. In the Program's screen, click **Launch IDE**.

   This launches Informatica PowerCenter client installed on your machine.
5. Modify the ETLs (transformation and/or workflow) used by the Oracle LSH Program.
6. Test and upload the ETL from Informatica PowerCenter to Oracle LSH.
7. Install the program in Oracle LSH, and run it to verify the changes.

**Note:** The ETL programs 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 programs 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, CDA ETL workflows 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 workflows.

To modify one or more tables or columns without any changes to the associated ETL programs:

1. Install your Work Area and run the ETL to ensure that you can see data populated in the target data warehouse tables.
2. Check out the Oracle LSH program that contains the metadata for that ETL.

   **IMPORTANT:** Use the **Copy definition to the local Application Area and check out** option when checking out the program. This ensures that you do not modify the definitions inside the domain shipped with CDA.

3. Change the table properties.

   To change the underlying columns and variables, check out the variables that the columns are pointing to. Ensure that you use the **Copy definition to the local Application Area and check out** option when checking out the variable.

   **Note:** If the changes to the tables or columns are not compatible with the table that is installed in the data warehouse schema, you will get a warning while making the change. For example, if you are reducing the length of a number column from 15 to 10, the change is not compatible with the data existing in the table. Such changes will not let you perform an Upgrade install on the table. You will have to drop and create the table using Partial or Full install.

4. Install the changed table or column, and run the ETL program that populates it.

# Scheduling an ETL Program

Scheduling can be categories as

- Scheduling without Deduplication
- Scheduling with Deduplication

### Scheduling without Deduplication

When you submit a Program for execution in Oracle LSH, you can schedule it execute at regular intervals. In the appropriate Work Area, navigate to the installed executable instance you want to submit and click **Submit**.

For more information on how to submit an Execution Setup, refer to *Oracle Life Sciences Data Hub Application Developer's Guide (Submitting Jobs for Execution).*

To schedule a Program, perform the following tasks:

1. Navigate to **OCDA_domain** > **OCDA_UTIL_APP_AREA** > **OCDA_ETL_WORKFLOW_WA**
2. Click **OCDA_PLS_ETL_WORKFLOW_PRG**.
3. Click **Submit**.
4. Enter the following values for submission parameters:
   a. **Config_days** - Must be the same value as you specified for the OCDA_CONTROL_TABLE_POPULATE_PRG program.
   b. **MATCH_MERGE_FLOW** - N
   c. **FULL_LOAD** - N
   d. **Submission Type** - Select Immediate.
   e. **Force Execution** - Select Yes.
5. In the **Submission Details** section, select **Submission Type** as Scheduled.The **Schedule Submission** section is displayed.
6. Enter the required details and click **Submit**.

### Scheduling with Deduplication

DAC Execution plans as discussed in incremental load can be scheduled for execution. For more information, refer to the *Oracle® Business Intelligence Data Warehouse Administration Console User's Guide* section on Scheduling an Execution Plan.

# Setting Up the Target Load Type

When you submit a Program for execution, perform the following tasks to specify the table processing type:

1. In the appropriate Work Area, navigate to the installed executable instance you want to submit.
2. In the Program's screen, click **Launch IDE**.

   This launches Informatica PowerCenter client installed on your machine.
3. In the Workflow Manager, modify the Target load type setting to **Bulk** or **Normal**.
4. Reinstall the program in Oracle LSH.

5. Navigate to the installed executable instance you want to submit, and click **Submit**.

   The Submit Execution Setup screen is displayed.

   For more information on how to submit an Execution Setup, refer to *Oracle Life Sciences Data Hub Application Developer's Guide (Submitting Jobs for Execution).*

6. In the Submission Parameters section, select the Parameter Value for Bulk Load based on what you have set up in Step 3. Select **Yes** if the table processing type is bulk, and **No** if it is normal.

7. Enter the required details and click **Submit**.

# 4

# Multi-Source Integration

This chapter contains the following topics:



## 4.1 Overview

Multi-source integration is an optional capability introduced in Oracle Health Sciences Clinical Development Analytics (CDA) Release 2.1 for Standard Configuration. It 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 CDA warehouse from multiple transactional databases.

The purpose of multi-source integration is to permit data to be loaded into the CDA 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 CDA 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 |
| 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 |
| ..... | ..... |

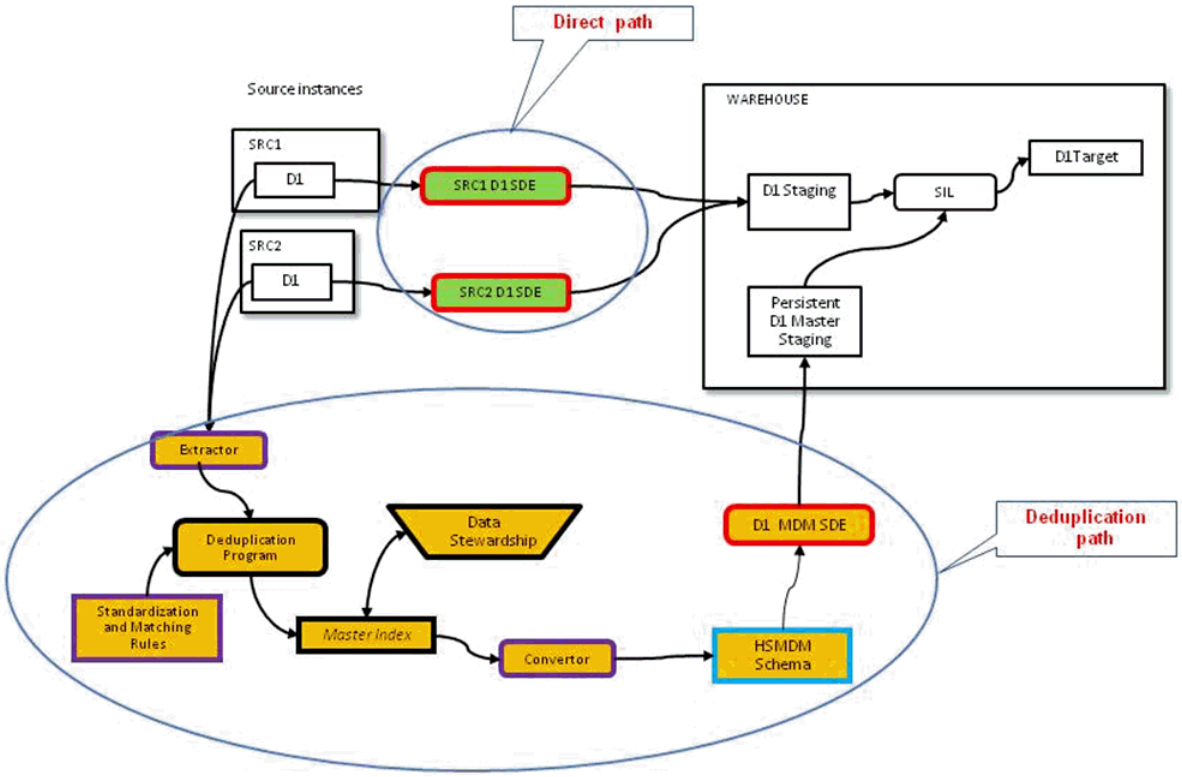
To arrive at the final number of queries, you will have to take the sum of the two rows.

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 4-1 illustrates how one dimension is loaded when using CDA's multi-source integration capability.

*Figure 4–1 CDA 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:** CDA retains all the records in the duplicate set, but marks them as merged. CDA 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. The system then writes the results to the Master Index for the dimension. The Data Steward makes decisions about potential matches not automatically resolved by rules. Until this

happens, such records are treated as singletons. Any changes made by the Data Steward are applied to the Master Index.

2. An SDE mapping that knows how to read the dimension from the dimension Master Index does so, writing records to a persistent staging table.
3. 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).
4. On the Direct Path, the SIL writes the contributor records to the dimension target table.
5. 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 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.

### 4.1.1 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.

### 4.1.2 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 CDA 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.

### 4.1.3 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 CDA's multi-source integration, which allows you to produce the same effects.

### 4.1.4 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 programatically 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 CDA 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 CDA 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 4.3, "Processes for Using Oracle Healthcare Master Person Index Deduplication Projects" on page 4-14.

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 4-1 lists the dimensions for which CDA provides multi-source integration support.

***Table 4–1 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 |

*Table 4–1 (Cont.) Warehouse Dimensions Supported by Multi-Source Integration*

| Warehouse Table |
|---|
| W_RXI_STUDY_SITE_D |
| W_RXI_STUDY_SUBJECT_D |
| W_RXI_VALDTN_PROCEDURE_D |
| W_USER_D |

## 4.1.5 Layering and Options

CDA'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.

### 4.1.5.1 Oracle Health Sciences Clinical Data Analytics and Oracle Healthcare Master Person Index

CDA has been designed to work with Oracle Healthcare Master Person Index (OHMPI) as its deduplication program. CDA provides all the files required to integrate OHMPI into its Deduplication Path. You can, however, use another deduplication program to serve this role. An outline of the tasks necessary to enable this is in section 2.5.2.5. The remainder of this document, other than Section 2.5.2.5, describes how CDA works with OHMPI as its deduplication program.

## 4.1.6 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 CDA 2.1, 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.

## 4.1.7 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. CDA provides a DAC Execution 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, 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 CDA to run the Deduplication Path for a dimension, then CDA will perform the following tasks whenever a job is executed for an incremental load.

1. CDA will gather information about any new duplicates from the Master Index for the dimension.
2. CDA runs the incremental load along the Direct Path.
3. CDA 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, CDA 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 CDA 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, CDA learns of the new assumed match, and makes the appropriate changes to the warehouse table.

## 4.1.8 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.

### 4.1.8.1 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.

### 4.1.8.2 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:

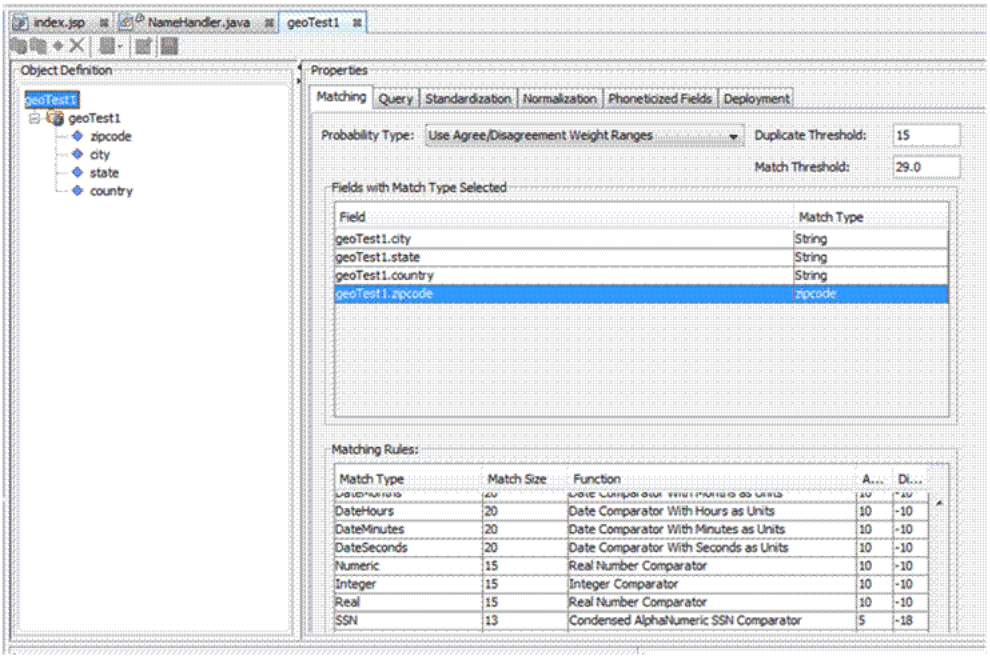
- Duplicate Threshold - a record must have a sum of weights greater than this to be considered a potential match to a Profile in the Master Index.
- Match Threshold - a record must have a sum of weights greater than this value to be considered an assumed match to a Profile in the Master Index.

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 4–2 shows the configuration screen for a project.

***Figure 4–2 Project Configuration Screen***

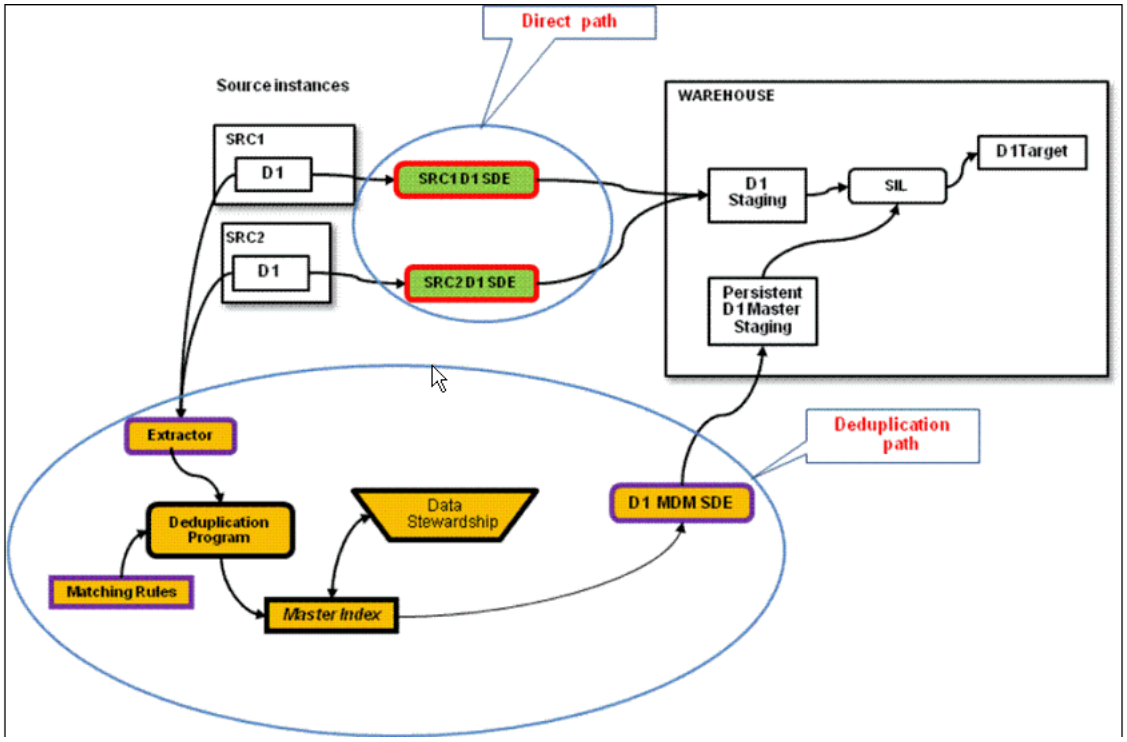


For more information, refer to the *Oracle Healthcare Master Person Index Match Engine Reference*.

## 4.1.9 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 4–3 Deduplication Path***




### 4.1.9.1 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 4.3.6, "Initial Load Processes" on page 4-16.

### 4.1.9.2 Extractor

The extractor is used during Incremental loads only. This program determines which dimension records are new, have changed, or have 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.

#### 4.1.9.3 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.

#### 4.1.9.4 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.

#### 4.1.9.5 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.

#### 4.1.9.6 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*.

---

#### 4.1.9.7 Dimension MDM SDE

The Dimension MDM SDE is a query that reads from the Master Index, 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.

#### 4.1.9.8 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.

## 4.2 Preliminaries to Using Oracle Healthcare Master Person Index Deduplication Projects

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

### 4.2.1 Installation Results

The process of installing or upgrading CDA 2.1 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 CDA installation. Others are installed as part of the CDA 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 file 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

The following item is not dimension specific:

- A DAC execution plan to do extractions for the Extractor component of the Deduplication path.

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

### 4.2.2 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 of the 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 4–2 OHMPI Folders***

| Folder | Description |
|---|---|
| cleanser | Work in this directory to do cleansing of the extracted, profiled data prior to bulk load |

*Table 4–2 (Cont.) OHMPI Folders*

| Folder | Description |
|---|---|
| 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.<br>*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. |
| 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. |

## 4.3 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.

### 4.3.1 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*.

### 4.3.2 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 4.1.8.2, "Matching Rules using Project Configuration" on page 4-10.

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 only occur for real duplicates, and avoid false assumed matches. Do this by adjusting 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 CDA 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.

### 4.3.3 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.

### 4.3.4 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.

### 4.3.5 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.

### 4.3.6 Initial Load Processes

#### 4.3.6.1 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 DAC execution plan, CDA - Complete Initial De Dup, for extracting the flat files for cleansing.

#### 4.3.6.2 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 refer to *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. Use the flat file generated as part of DAC execution plan.
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.

#### 4.3.6.3 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 4.5.1, "Rules" on page 4-21.

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 CDA, Oracle recommends that you do not use the cleansing process to modify dimension data. Instead, you should modify the data in its the source database. Refer to Rule one in Section 4.5.1, "Rules" on page 4-21.

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*.

#### 4.3.6.4 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\.

   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:

   a. Run Bulk Match.
   b. 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.

3. If the outcomes are not what you want:
   - Adjust the match rules. For more information, refer to Section 4.1.8.2, "Matching Rules using Project Configuration" on page 4-10.
   - 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*.

#### 4.3.6.5 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*.

### 4.3.7 Incremental Load Processes

#### 4.3.7.1 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

## 4.4 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.

### 4.4.1 Merged Fact Records Consequent to LOV Dimension Merge

One of the dimensions that can be deduplicated starting in CDA 2.1 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 will be found to both point

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, CDA 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.

## 4.4.2 General Case: Discovered Duplicate Fact Records

Driving Dimensions

### 4.4.2.1 Impact of Dimension Deduplication on Fact Tables

This section applies to the following facts:

- 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)

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, CDA 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 4–3, " Study-Site Dimension Table After Deduplication" displays the results of deduplication of the Study-Site dimension. Data sources 1 and 2, both 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 4–3 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 4–4, " 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 4–4 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 |

CDA 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 the two records in the fact table that both point 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.

## 4.5 Rules and Recommendations

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

### 4.5.1 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.

### 4.5.2 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 programatically 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.

## 4.6 Oracle Health Sciences Clinical Development Analytics' Match Rules

This section provides information about the matching rules that are shipped with each CDA 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.

### 4.6.1 Policies for Creating Shipped Match Rules

This section describes how CDA 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. CDA'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 4–5, " Preferred Source for Each Deduplicated Dimension" on page 4-23 for the preferred sources defined in the Projects shipped by CDA.
4. If two or more records match, CDA'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 4–5, " Preferred Source for Each Deduplicated Dimension" on page 4-23
   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 4–5 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 |
| 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 |

*Table 4–5 (Cont.) Preferred Source for Each Deduplicated Dimension*

| Warehouse Table | Preferred Source |
|---|---|
| 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 |

## 4.6.2 Configurations

Table describes the configuration of the OHMPI Projects shipped with CDA. 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, CDA 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 4–6 Project Weights and Thresholds*

| Dimension Name | Project Name | Duplicate Threshold | Match Threshold | Match Attribute | Match Type | ICustomized 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 |

***Table 4–6 (Cont.) Project Weights and Thresholds***

| Dimension Name | Project Name | Duplicate Threshold | Match Threshold | Match Attribute | Match Type | ICustomized or Built-in | Comparator Function | Disagree Weight | Agree Weight | Null Field |
|---|---|---|---|---|---|---|---|---|---|---|
| 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 | HouseNumber | 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 |
| 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 |

***Table 4–6 (Cont.) Project Weights and Thresholds***

| Dimension Name | Project Name | Duplicate Threshold | Match Threshold | Match Attribute | Match Type | ICustomized or Built-in | Comparator Function | Disagree Weight | Agree Weight | Null Field |
|---|---|---|---|---|---|---|---|---|---|---|
| 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 | OCDACityString | Customized | Condensed String Comparator | 0 | 25 | Zero Weight |
| Geography | OCDA_Geography | 25 | 85 | COUNTRY | OCDACountryString | Customized | Condensed String Comparator | -6 | 10 | Zero Weight |
| Geography | OCDA_Geography | 25 | 85 | STATE_PROV | OCDAStateString | Customized | Condensed String Comparator | 0 | 20 | Zero Weight |
| Geography | OCDA_Geography | 25 | 85 | ZIPCODE | OCDAZipString | 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 |

***Table 4–6 (Cont.) Project Weights and Thresholds***

| Dimension Name | Project Name | Duplicate Threshold | Match Threshold | Match Attribute | Match Type | ICustomized or Built-in | Comparator Function | Disagree Weight | Agree Weight | Null Field |
|---|---|---|---|---|---|---|---|---|---|---|
| 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 | HouseNumber | 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 |
| 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 |

*Table 4–6 (Cont.) Project Weights and Thresholds*

| Dimension Name | Project Name | Duplicate Threshold | Match Threshold | Match Attribute | Match Type | ICustomized or Built-in | Comparator Function | Disagree Weight | Agree Weight | Null Field |
|---|---|---|---|---|---|---|---|---|---|---|
| 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 |

## 4.7 User-supplied Deduplication System

If you want to use a deduplication program other than OHMPI, the program must plug into the CDA deduplication path at two points: it must read from the source databases for the dimension, and the deduplication program's Master Index must be read into 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
- a modified MDM SDE that can read the Master Index

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.

## 4.8 Extending the Warehouse

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

***Table 4–7 Extension Tasks by Load Path***

| Task | Path |
|---|---|
| Add M to the D1 Staging table | Direct |
| Add M to the D1 warehouse table | Direct |
| Add M to the D1 Persistent staging table | Deduplication |
| Add M to the OHMPI Project definition for dimension D1 | Deduplication |
| Add M to the Master Index for dimension D1 | Deduplication |
| Add M to the MDM SDE for the dimension | Deduplication |
| Add M to D1 SDE for each transactional application you source from | Direct |
| Add M to D1 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.

### 4.8.1 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.

## 4.9 Informatica Mappings used in Multi-Source Integration

Table 4–8 shows the Informatica 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 4–8 Informatica 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_MDM_<Dimension>_Dim | Dedup | Dedup 4 | Load Persistent Staging | Master Index | Persistent Staging table | Both | Populate Persistent Staging from Master Index |
| SIL_CDA_PS_<Dimension>_Dim | Dedup | Dedup 5a | 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 5b | 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:



## Deleting Control Table Entries

The Control Table (W_CONTROL_S) contains a record for each ETL execution of a specific target table. The record stores a start and end timestamp. That is, source records were extracted only if their creation or modification timestamp was between the start and end timestamps specified in the Control Table record for a given ETL mapping execution.

There are occasions when it is desirable to be able to delete selected entries from the Control Table. This is useful when you want to:

- Delete all the entries from the control table for full load.
- Delete selected entries from the control table based on the different input options. This will help when new columns are added and data needs to be reloaded into a selected dimension or fact.

Accordingly, the Control Table population program has been enhanced to support deletion of  entries from control table, while supporting its original function of populating the control table.

1. Navigate to **OCDA_domain** > **OCDA_SOURCES_APP_AREA** > **OCDA_CONTROL_TABLE_WA**.
2. Click **OCDA_CONTROL_TABLE_POPULATE_PRG**.
3. Submit the program with the following parameters:

   **Submission Details**

   Submission Type: Immediate

   Submission Mode: Incremental

   Force Execution: Yes

   **Submission Parameters**

Following are input parameters for the control table populate program that are used to delete rows:

a. **Delete_mode** parameter has a list of values used for mode of deletion. Below are the values in the list and description of operation performed.

   **None -** Performs a normal operation on the control table, that is, populates the table.

   **All** - Deletes all the entries in the control table.

   **ETL_RUN_ID -** This requires a list of comma seperated ETL_RUN_ID in the input_values field. When submitted, it will delete entries corresponding to input_values.

   **MASTER_JOB_ID -** This requires a list of comma seperated Master_Job_Id in the input_values field. When submitted, it will delete entries corresponding to input_values.

   **Program_Name** - This requires a list of comma seperated Program_Names in the input_values field. When submitted, it will delete entries corresponding to input_values.

b. **Input_values** parameter takes a comma-separated list of values that need to be deleted.

## 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 (i) nulls will display as blank cells, and (ii) the 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 (i) nulls will display as zeros, (ii)

the 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.

# Cancelling Jobs in Oracle Life Sciences Data Hub

Cancelling jobs (Informatica programs) submitted in Oracle LSH does not automatically abort the workflow in Informatica PowerCenter. This needs to be done manually. To abort the workflow:

1. Identify the folder that contains the particular workflow. For more information, refer to Identifying the Folder Containing the Workflow.
2. Abort the workflow in Informatica PowerCenter. For more information, refer to Aborting a Workflow.

### Identifying the Folder Containing the Workflow

Use the particular job's command log file in Oracle LSH (cmdlog.log), to identify the Informatica folder that contains the workflow. Perform the following steps in Oracle LSH to view the log file:

1. In the Job Execution section of My Home, click the Job ID hyperlink of the particular job you want to cancel.

   The Job Execution Details screen is displayed. The Master Job Id field displays the system-generated unique ID of the job that calls this job (parent job).
2. Click **Outputs**, and then click the hyperlink in the View column.
3. Choose to save or open the command log file (cmdlog.log).

**See Also:**

*Oracle Life Sciences Data Hub Developer's Guide, (Monitoring Jobs)*

### Aborting a Workflow

Perform the following steps in Informatica PowerCenter to abort a workflow:

1. Open the Informatica PowerCenter Workflow Monitor.
2. In the Repositories tree, navigate to the particular folder that contains the Informatica job.

For more information on how to identify the folder that contains a particular workflow, refer to Identifying the Folder Containing the Workflow

3. In the Workflow Run pane, select and right-click the workflow, and click **Abort**.

**See Also:**

*Informatica PowerCenter Online Help*

# Errors in Reports

### Error

CDA reports return any of the following errors:

- Assertion failure
- Invalid arithmetic operation on non numeric type

  [nQSError: 22019] Function Median does not support non-numeric types

  [nQSError: 22025] Function TimestampDiff is called with an incompatible type
- State: HY000. Code: 10058. [NQODBC] [SQL_STATE: HY000] [nQSError: 10058] A general error has occurred. [nQSError: 43113]

### Cause

The source of the problem may be the warehouse table W_ETL_RUN_S. This table should contain a row for each execution of the ETL that updates the warehouse. Many reports in CDA depend on the value of the dynamic Repository variable CURRENT_DAY. This Repository variable, in turn, depends on the maximum value of the column LOAD_DT in W_ETL_RUN_S. The value of CURRENT_DAY is updated every 5 minutes by an initialization block in the CDA Repository. If W_ETL_RUN_S does not contain any values for LOAD_DT, then CURRENT_DAY will be null, and the many reports that depend on it will fail with errors such as those shown above.

If the ETL for CDA is working correctly, a row should be added to W_ETL_RUN_S each time an ETL run completes successfully, and these errors will not occur. However, especially during initial setup and load, a partially complete ETL run will load some data, but not set the value of LOAD_DT. Also, test data is loaded directly into a warehouse, and the test data does not supply rows for W_ETL_RUN_S, the conditions will be set for these assertion failures.

### Fix

To fix the above errors, perform the following steps:

1. In the warehouse, check that W_ETL_RUN_S has at least one row, and that the value of LOAD_DT is not null in the rows that are present.
2. If W_ETL_RUN_S already has values for LOAD_DT, it is not the cause of the errors. Skip the remaining steps in this list and look elsewhere for the cause of the errors.
3. If W_ETL_RUN_S has no values for LOAD_DT, correct it, if necessary, by re-executing the ETL in incremental mode.
4. Exit the CDA OBIEE Presentation Server web page.
5. Restart the CDA OBIEE Presentation Server and BI Server , and log in.
6. Wait 5 minutes, to ensure that CURRENT_DAY is updated from the corrected W_ETL_RUN_S.

7. View a dashboad page where the error had occurred. If the error was due to missing LOAD_DT values, the report should now show results or a (No Results) message, if there is no data for the report.

# Glossary

### Case Report Form

A printed, optical, or electronic document designed to record all of the protocol-required information to be reported to the sponsor on each trial subject. The CRF is the way the **Clinical Data** for Patients is collected.

### CDMS

Clinical Data Management System (For example, Oracle Clinical)

### Central Laboratory

A location, under contract to a Clinical Trial sponsor, where samples are sent from multiple sites for analysis.

### Clinical Data

Data pertaining to the medical characteristics or status of a patient or subject.

### Clinical Research Organization

A company or organization that conducts all or part of a clinical trial under contract to a Clinical Trial sponsor.

### Clinical Study

See **Clinical Trial**

### Clinical Trial

Before a pharmaceutical or biotech company can initiate testing on humans, it must conduct extensive pre-clinical or laboratory research. This research typically involves years of experiments on animal and human cells. The compounds are also extensively tested on animals. If this stage of testing is successful, a pharmaceutical company provides this data to the Food and Drug Administration (FDA), requesting approval to begin testing the drug on humans. This is called an Investigational New Drug application (IND). A clinical trial is a carefully designed investigation of the effects of drug, medical treatment, or device on a group of patients (also called Subjects).

### compound

The product being tested or researched within the Clinical Trial.

### CRA

Clinical Research Associate. An employee of the Sponsor, responsible for getting a site prepared to conduct a trial and getting cleaned data back from the site to the Sponsor.

### CRF

See **Case Report Form**

### CRF Book

A set of paper forms or electronic forms that record the results of the set of assessments performed on a subject taking part in a clinical trial.

### CRF Page

A single form within a CRF Book.

### CRO

Clinical Research Organization

### CTMS

Clinical Trial Management System (For example, Oracle's Siebel Clinical)

### discrepancy

Problems found with data reported in the CRF pages by Investigators for specific Patients

### eCRF

A single electronic **CRF**.

### EDC

Electronic Data Capture system (For example, Oracle Remote Data Capture (RDC))

### informed consent

A discussion of all procedures, benefits, risks, and expectations of a clinical trial between clinical investigators and potential patients. The FDA requires all patients to sign an informed consent form before participating in a trial.

### investigator

A person responsible for the conduct of the clinical trial at a trial site. When a Clinical Trial is conducted at a Site by a team the Investigator is the responsible leader of the team and may be called Principal Investigator (PI). Other investigators are called Sub-investigators. Investigators are qualified health care professionals, often are MDs, PhDs or Pham Ds.

### patient

A person who participates in a **Clinical Study** and is the focus of the Clinical Trial's research.

### patient visits

A series of scheduled visits by a Patient to an Investigator based interval specified in the Clinical Trial's Protocol. During the Patient visits the Investigators undertakes the required medical procedures defined in the Clinical Trial Protocol and completes the corresponding CRFs.

### phase

Phase of trial, typically 1,2,3 or 4.

### program

Groups of Clinical Studies or Clinical Trials for the same compound.

### projects

Groups of Studies within a Program (Oracle Clinical Only)

### Protocol

A Protocol is a document that describes the objective(s), design, methodology, statistical considerations, and organization of a trial. It is a plan that states what will be done in the study and why. It outlines how many people will take part in the study, what types of Subjects may take part, what tests they will receive how often, and the treatment plan. The Sponsor of the Clinical Trial typically designs the Protocol.

### Protocol amendment

A written description of a change(s) to or formal clarification of a protocol.

### queries

Each query is a request for information, sent to an Investigator, to resolve a Discrepancy detected in data signed for by that Investigator.

### randomization

The process of assigning trial subjects to treatment or control groups using an element of chance to determine the assignments in order to reduce bias.

### region

A geographic region in which the Clinical Study or Clinical Trial will be carried out.

### Regulatory Authority

An authority such as FDA and EMEA regulating clinical development processes.

### site coordinator

The individual who manages the conduct of the clinical trial. Coordinators are often nurses.

### site visit

A visit or trip by a CRA to a Site for monitoring and support activities.

### sites

Sites are locations where clinical trials are conducted. They are typically a clinic or hospitals where **investigator**s see subjects and perform study procedures, such as medical checks.

### Sponsor

The organization funding the clinical trial. This is typically the Pharmaceutical company whose product is being tested with the clinical trial.

### study document

A required Document to initiate or start a Clinical Trial at a Site (For example, Investigator Resume.)

### study

See **Clinical Trial**

### subject

See **patient**

# Index

## M



## N



## O



## P



## R



## S