Oracle Financial Services Analytical Applications

Data Model Extension Guidelines

Release 8.0.7.0.0

March 2019

ORACLE®
Financial Services

ORACLE®

Oracle Financial Services Analytical Applications Data Model Extension Guidelines

Document Control

Version Number	Revision Date	Change Log
2.0	March 2019	Added the chapter Oracle SQL Data Modeler

Table of Contents

1 Preface

This Preface provides supporting information for the Oracle Financial Services Data Foundation Data Model Extension Guidelines and includes the following topics:

- [Purpose of This Document](#)
- [Audience](#)
- [Related Documents](#)

1.1 Purpose of this Document

The purpose of this document is to detail the extension process of the OFSAA data model. The objective is to ensure that implementation teams extend the OFSAA application as per guidelines, conformance, and standards established for content and design established.

1.2 Audience

This document is intended to be used by Oracle Financial Services Analytical Applications Data Foundation System Administrator.

1.3 Related Documents

This section identifies additional documents related to OFSDF Application 8.0.8.0.0. You can access Oracle documentation online from the Documentation Library ([OHC](#)).

- Oracle Financial Services Data Foundation Application Pack User Guide ([OHC](#)).
- Oracle Financial Services Data Foundation Application Pack Installation Guide ([OHC](#)).

2 Overview of Data Model

The OFSAA data model is organized into distinct areas based on usage and function performed by each area of the model.

2.1 Overview of OFSAA Data Model

The OFSAA data model is organized into distinct areas based on usage and function performed by each area of the model.

The data model conforms to the data architecture processes for each function and the modelling methodology employed in each of these functional areas representing the usage pattern.

3   Structure of OFSAA Data Model

This section details the structure of OFSAA Data Model.

3.1   Common Staging Area

The common staging area is the single source of data for all of OFSAA. Primary purpose of staging is data acquisition and integration. The common staging area is a consolidation of entities fact, dimensions and others. Completely de-normalized in structure and bears no relationship to one another in the physical model. Logical relationships exist and are established and validated through the staging area processes such as the Value code Mapping, Data Quality and GL Reconciliation processes.

3.2   Application Specific Processing Area

Application specific processing area is a collection of application specific entities used for incoming data, Application maintenance and Application metadata. The modelling methodology employed is based on the processing needs of each Application. The modelling methodology varies from Application to Application. Various degrees of normalization to de-normalization and dimensional modelling structures are employed. The modelling in the processing area is governed by the needs of specific processing engines. Processing engines have evolved at different points of time and have different data needs structurally.

3.3   Common Results Area

The result area is a rich combination of source data, value added metrics from each of the analytical applications and supports the derivation of higher order value added metrics across conformed dimensions spanning multiple functional domains regulatory, economic and more.

The common results area is a conformed set of facts and dimensions that combine results from various applications and create an integrated result environment for analytics across applications. Integrated reporting at transaction, account and customer granularities are managed through an integrated common result modelling approach.

4 Extension Process

The Data model life cycle process involves understanding of data requirements, data architecture and data flow requirements of the applications that are being implemented.

The governance process is broadly described as follows:

- Data Requirements

  Identification of the additional data requirements for an implementation of the application is determined through customization/configuration documents. Use-cases have to be established for different areas in the OFSAA model.

- Data Flow

  Data flow needs to be established for these additional data requirements.

- Model Design

  Model extensions to be designed based on the guidelines provided for each area. Extensions should also be raised as Enhancement Requests to respective applications through the standard support process. Enhancement Requests will be analyzed by the Product Engineering team and design approaches will be provided where applicable. Review the data model customization review template attached in the appendix.

- Model Build

  Model extensions to be built based on the guidelines provided for each area.

  The supported tools are:

  - Erwin
  - Oracle SQL Data Modeler

  For information about extension through Oracle SQL Data Modeler, see Oracle SQL Data Modeler.

5   Design Guidelines for Staging Area

The following are the design guidelines for staging area:

- Subject Area: Staging area is designed to hold data related to multiple applications. Entities are typically shared across subject areas.

- Entities: Typically, Staging area will have wide tables to accommodate data requirements related to multiple OFSA applications in a single row.

- Relationships: Staging area model should not contain any relationships amongst entities. Referential integrity checks are performed as part of Data Quality checks.

- No redundancy of information: Information required for multiple applications need to be captured only once, in a way suitable to all the applications.

- Multiple periods: Staging is designed to capture and store data for multiple periods. This will be useful to support operational requirements such-as re-run of processes for earlier periods.

- No update: Staging data, as received from source systems should not be modified. In case of exceptions that require data updates, following are the options:

    - Data needs to be corrected in source systems and re-loaded into staging.

    - Data quality framework should provide the ability to identify data quality issues and correct the data with audit trail.

- Alpha-numeric Codes:

    - Master data should be designed to hold alpha-numeric codes.

    - Stage tables corresponding to the dimension column should also be explicitly designed as alpha-numeric.

- Static Information: Data created and maintained within the application such as setup, static master information and so on should not be represented in staging.

- Impact Analysis: Staging area data model is shared across multiple OFSA applications. Hence, when a property of any entity or attribute is modified, impact analysis needs to be performed with application owners before performing the change. Metadata browser can be used for performing the impact analysis.

- Data Origin: Each staging entity needs to include a column for storing the source of data. This information is useful to understand the lineage of data.

- Download Specification: Staging area is designed to receive data from source systems. Download specification for each application can be generated from staging area model. User-defined properties for table and column are referenced in the generation of download specs. Mention UDP values for any attributes added as part of the extension process.

- Naming Standards: Staging data model should follow naming standards as detailed in the Appendix.

- Customization related User Defined Properties: Column level UDP's named 'Custom' (NO/YES), 'Customization Reason' (Pending ER, Specific to Customer, Modified data element) and 'Type of Change' (Length, Datatype, Logical Name, Description, Addition) have to be specified for attributes added/modified as part of the extension process.

6   Design Guidelines for Processing Area

Data model for the processing area is designed as per the requirements for the applications. The following are the guidelines for processing area:

- Design: Processing area for each application is designed for optimal performance of the calculation engines respective to the applications.

- Performance: Data will be moved from staging into processing area. Typically, processing involves multiple SELECT and UPDATE SQL statements and hence data model should be designed to enable high performance SELECT and UPDATE operations on the table.

- Relationships: Processing area data model should enforce non-identifying and identifying relationships amongst entities as applicable.

- Naming Standards: Processing data model should follow naming standards as detailed in the Appendix.

- Diagram Representation: Processing area entities should be grouped logically and represented in the erwin data model. In erwin, default display view should have primary key designators.

- Limited History: Processing is designed to support storage of limited history. This historical data is useful for verification of process outputs for earlier dates, if required.

7   Design Guidelines for Reporting Area

The purpose of the Reporting data model is to perform the following:

- Address application specific reporting requirements

- Address integrated reporting across applications

- Drill through and drill-across in reports.

The following are the guidelines for reporting area:

1. Application-specific Reporting: Application specific reporting area can have specific fact tables to store measures related to that application. However, dimensions that are referred in any of the application specific reporting need to be conformed across all the applications.

2. Integration of Results: Integration of results is achieved through common structures for account and customer as described as follows:

   ▪ Common Account Summary

      — Consolidates measures at the granularity of the account that are output from different applications.

      — Any account level reporting should be driven out of integrated Account Summary set of entities. Account attributes captured from source system and those which do not vary between runs needs to be part of the Common Account Summary entity. Any application specific attributes and measures that are computed by application needs to be part of application specific account summary entities. Such a data model design enables integrated reporting of account information.

   ▪ Common Customer Summary

      — Consolidates measures at the granularity of the customer that are output from different applications.

      — Any customer level reporting should be driven out of integrated Customer Summary set of entities. Customer attributes captured from source system and those, which do not vary between runs, need to be part of the Common Customer Summary entity. Any application specific attributes and measures that are computed by application needs to be part of application specific customer summary entities. Such a data model design enables integrated reporting of customer information.

- Data Modeling Technique: Reporting data model is structured as typical star schemas' corresponding to analytical reporting requirements.

- Aggregate Entities: Depending on performance requirements for each application, information can be reported out of aggregate entities. However, a drill through to the base entity from the aggregate entity is mandatory.

- Reporting and Local Currency: Include additional attributes in the fact tables to store reporting and local currency equivalent of base measures. These attributes need to be computed by looking into the exchange rates.

- Dimensional Conformance: Same dimensional information should not be represented in different forms within reporting area.

- Slowly Changing Dimensions: Dimensions that need to track changes in attributes need to be handled through slowly changing dimensions (SCD) process. Such dimension entities need to have a standard set of attributes required for SCD process.

- Measures: Same fact information should not be represented in multiple fact tables.

- Intermediate Results: Reporting data model should hold both final and intermediate computed outputs of each application.

- Performance: Data is going to be loaded into reporting area from processing. Since reporting will involve multiple SELECT queries, data model should be designed to enable high performance SELECT operations on the table. Also, since reporting model provides integrated information across applications, entities will be wide compared to entities modelled in the processing area.

- Relationships: Reporting area data model should enforce non-identifying and identifying relationships between fact and dimension entities as applicable.

- Naming Standards Reporting data model should follow naming standards as detailed in the Appendix.

- Subject Area: Subject areas should be created for a logical set of reports. Ideally, each subject area should have a single fact and multiple dimension entities.

- Diagram Representation: Within the subject area, fact entities should be represented in the centre surrounded by dimension entities. In erwin, default display view should have primary key designators.

- Full History: Reporting area model is designed to support storage of full history.

8 Configuring Entity UDPs for Application Data Interface of Data Integration Hub

In OFSAA DIH Application, the uploaded data model is visible as Application Data Interface (ADI). These ADI names are given through erwin data model entity UDP for each table. For any new table addition which is added as part of customization should have valid value in following entity UDPs to get the proper ADI names in DIH.

- OFSAA Data Interface Name: This Entity UDP expects the logical name of the table or logical name for a group of tables which will be displayed as an ADI in DIH. This UDP is mandatory for all Stage and Results tables to get a valid ADI in DIH which corresponds to each table in model.

- OFSAA Data Interface Subtype: This Entity UDP expects the logical name of a table where the table is part of a group of tables where we have a group name in OFSAA Data Interface Name UDP. This UDP is optional and applicable only if you have a one OFSAA Data Interface Name against a group of tables.

For Example:

- STG_CASA, STG_LOAN_CONTRACTS, STG_TD_CONTRACTS, FCT_COMMON_ACCOUNT_SUMMARY are defined as group of tables and the UDPs for the tables are given in the following table:

TABLE_NAME	OFSAA Data Interface Name	OFSAA Data Interface Subtype
STG_CASA	Customer Account	CASA Contracts
STG_LOAN_CONTRACTS	Customer Account	Loan Contracts
STG_TD_CONTRACTS	Customer Account	Term Deposits
FCT_COMMON_ACCOUNT_SUMMARY	Customer Account	Common Account Summary

- If you have Stage and Fact tables for the same purpose, you need to define it as same name given for OFSAA Data Interface Name, OFSAA Data Interface Subtype.

TABLE_NAME	OFSAA Data Interface Name	OFSAA Data Interface Subtype
STG_ACCT_FEATURE_MAP	Account Feature Map	Account Feature Map
FCT_ACCOUNT_FEATURE_MAP	Account Feature Map	Account Feature Map

- If you have independent tables like only one either stage or dim or fact, then Subtype can be left blank.

TABLE_NAME	OFSAA Data Interface Name	OFSAA Data Interface Subtype
STG_VEHICLE_MASTER	Vehicle Master	
FCT_TXN_CHANNEL	Transaction Channel	
DIM_ACCOUNT	Account Dimension	

9   Sample Extensions to Model

The following table summarizes some of the common types of extensions to model:

Extension type	Staging	Processing/Reporting
Add new dimension	• Create entity named STG_<DIMENSION NAME>_MASTER. • Primary key for the master entity should be the alpha-numeric dimension key identifier and MIS Date. • Include attributes for the dimension as columns in the master table.	• Create a set of 4 tables if the dimension/hierarchy needs to be maintained within OFSAA – DIM_<DIMENSION NAME>_B – stores members; DIM_<DIMENSION NAME>_TL – stores translated members; DIM_<DIMENSION NAME>_ATTR - stores values of attributes of dimension members; DIM_<DIMENSION NAME>_HIER - stores hierarchy. • Create the dimension table if the dimension needs to be used in reporting. Design the structure of the dimension table depending on the requirement to have level based or parent-child hierarchies. Dimension table should have support for slowly changing dimension. • Configure data movement processes to load dimension tables in processing and reporting.
Add new Attributes to dimension	Enhance staging master table to include this attribute.	• In case the dimension is maintained within OFSAA, configure this attribute for the dimension. Model need not be changed. • In case the dimension is used for reporting and attribute is required for reporting, then include this attribute in the dimension table as a column. • Modify data movement processes to load this additional attribute into dimension tables in processing and reporting.
Modify data type/data length for any attributes of the dimension	Check if this data type/data length change does not impact any applications.	• Modify the data type/data length in the corresponding dimension table in reporting. • Some of the dimensional attributes might be de-normalized in fact entities for performance reasons. Datatype/Data length needs to be • modified in all such cases.

Extension type	Staging	Processing/Reporting

Add new entities for receiving hierarchy data from any MDM system	Create a set of four entities named STG_<DIMENSION NAME>_B – holds the dimension members. STG_<DIMENSION NAME>_TL – holds the translated dimension members. STG_<DIMENSION NAME>_ATTR – holds the values of attributes of dimension. STG_<DIMENSION NAME>_HIER - holds the hierarchy	• Create a set of 4 tables if the dimension/hierarchy needs to be maintained within
		  ▪ OFSAA – DIM_<DIMENSION NAME>_Bstores members
		  ▪ DIM_<DIMENSION NAME>_TL stores translated members
		  ▪ DIM_<DIMENSION NAME>_ATTR stores values of attributes of dimension members
		  ▪ DIM_<DIMENSION NAME>_HIER stores hierarchy.
		• Create the dimension table if the dimension needs to be used in reporting. Design the structure of the dimension table depending on the requirement to have level based or parent-child hierarchies. Dimension table should have support for slowly changing dimension.
		• Configure data movement processes to load dimension tables in processing and reporting.

Add attributes to product processor	• Make sure that attributes that are being added do not exist in any form within the entity. • Extend product processor to hold these attributes.	• Extend processing fact tables to hold this information. Modify T2Ts to move this information from staging. • Extend common account summary entity to hold this information if the attribute is required for reporting. Modify T2Ts to move this information from staging to reporting.
Add attributes to any fact entity	• Make sure that attributes that are being added do not exist in any form within the entity. • Extend the entity with this additional attribute.	• Extend the processing/reporting entity to include this attribute. • Modify the data movement processes to move the information from staging.

Extension type	Staging	Processing/Reporting
Make changes to primary key of the fact entity	• Make sure that the impact of changing the granularity is carried for all the OFSA applications that are being implemented. • Extend the fact entity in staging to include attribute as part of the PK. • Modify the data movement processes to move data as per processing/reporting model.	• Make sure that the impact of changing the granularity is carried for all the OFSA applications that are being implemented. • Any primary key change in processing/reporting will impact OFSA applications. Hence, raise service requests in product support with the use-cases.
Add a new entity	• Make sure that the information to be captured in the new entity does not exist elsewhere in staging model. • Modify the data movement processes to move data as per processing/reporting model.	• In cases where data captured in the new entity in staging needs to be moved to an existing processing/reporting entity, modify the data movement processes. • In cases where there is a need to represent fact information in a separate entity, extend processing/reporting model to create this entity. Build data movement processes to move data from staging.

10   Oracle SQL Data Modeler

Deploy the patch **29467329** from https://support.oracle.com/ as per the steps detailed in the readme file of the patch. Follow the below mentioned steps to configure and extend the model. For more information, see OFSAA Datamodel extensions through SQL Data Modeler (Doc ID 2521335.1) at My Oracle Support.

NOTE	Before proceeding with the model customization procedure, take a backup of the DB.xml file, and config and atomic schemas.

10.1   Deploying SQL Data Modeler Configuration Files

Follow these steps to deploy SQL Data Modeler configuration files:

1. Copy the downloaded file `defaultdomains.xml` available at `<>/OFS_BFND/SQL_Modeler_Conf_Files/types` to `$fic_home /ficapp/common/FICServer/conf/SQLDM`.

2. Rename the file defaultdomains.xml (from previous step) to OFSAA_DOMAINS_MASTER.xml.

3. Repeat the steps 1 and 2 each time in case of changes, if any, to the file defaultdomains.xml.
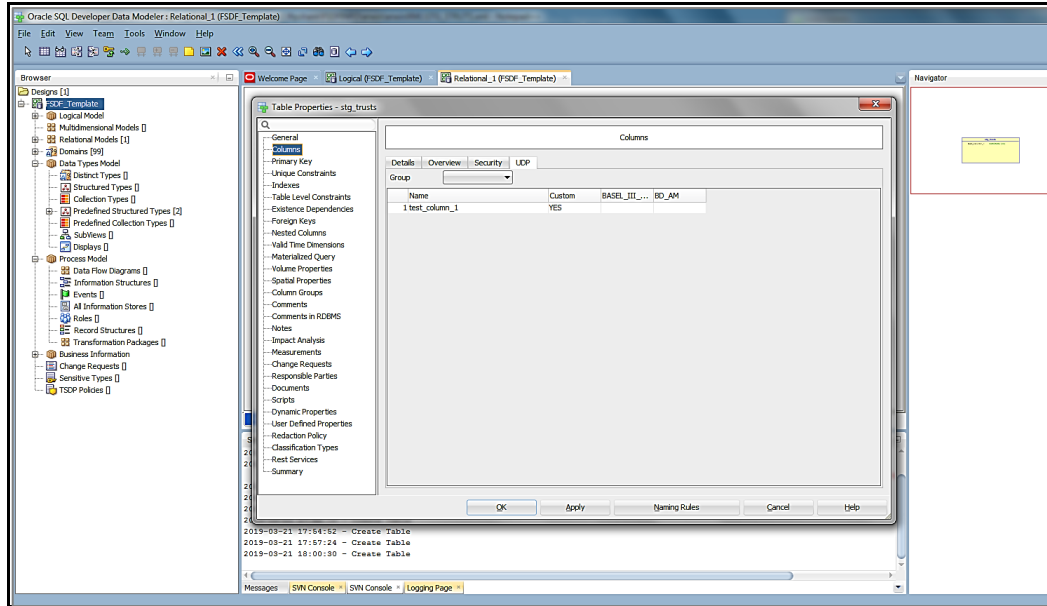
4. Perform model changes as required.

10.2   Generating Customized Model
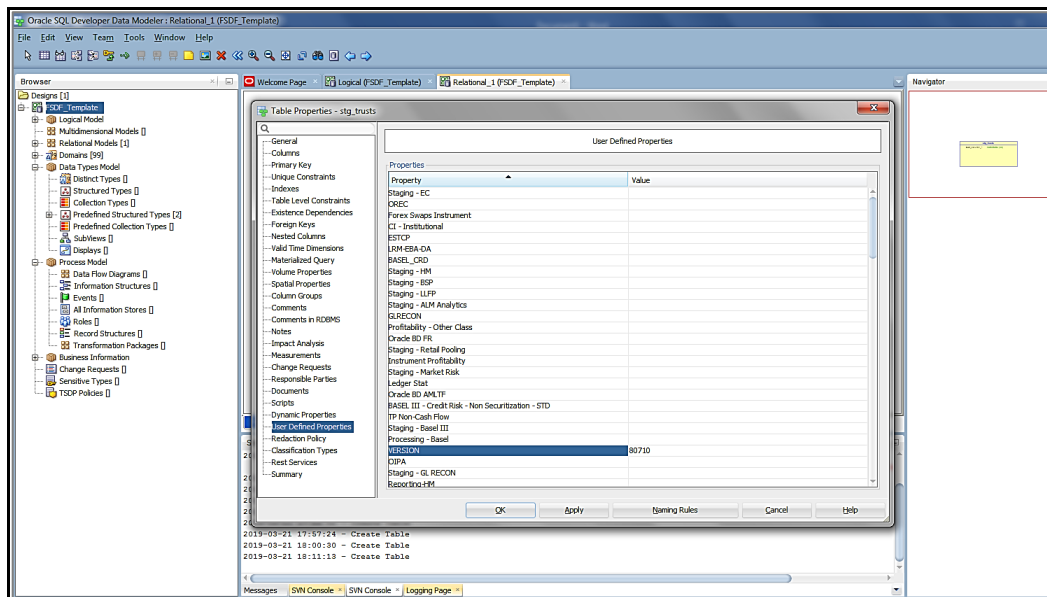
These are the generic tasks to generate customized model:

*   Adding Custom Column
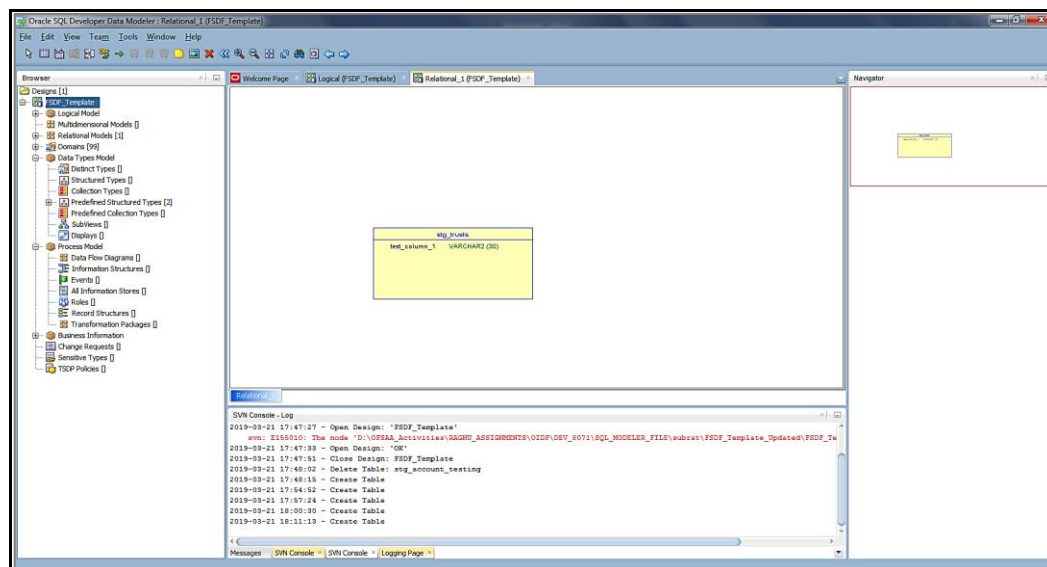
*   Setting User defined property (UDP) for Column to YES.

- Setting Version UDP value to current version.

11 Data Model Merge Guidelines

This section details the instructions to merge two data models. To merge the two data models:

1. Open the erwin model.

2. Navigate to the Actions menu and click the Complete Compare option.

3. The Complete Compare dialog opens, which consists of seven input forms.

4. To access any of the seven input forms, in the left panel of the Complete Compare dialog, click the required form title. The seven input forms are:

   - Overview

   - Left Model

   - Right Model

   - Left Object Selection

   - Right Object Selection

   - Type Selection

   - Advanced Options

5. Overview: This form provides a brief description of the Complete Compare function

6. Left Model: Using the Left Model input form, you can select data model for the left side of the comparison in the Complete Compare dialog.
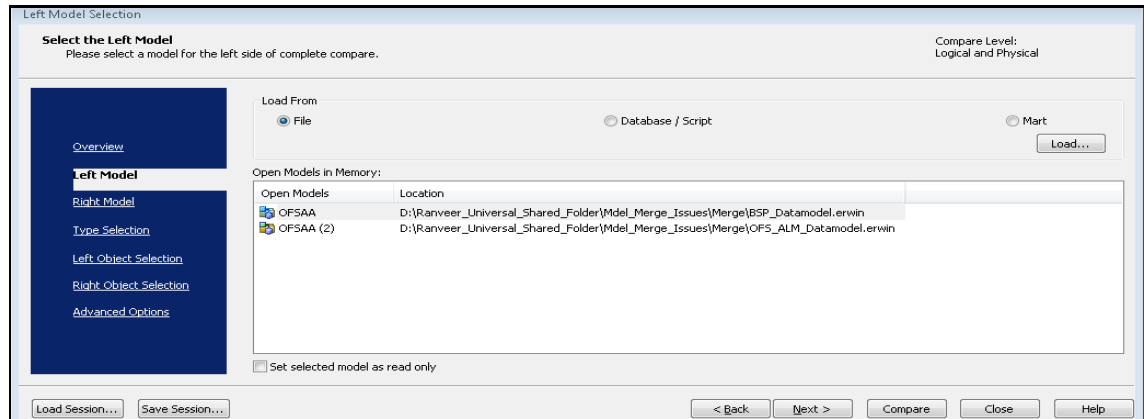
   To select the required model for the left side of comparison, click Load. The file Open dialog appears. Search for the required data model file, select it and click Open. The selected data model loads into the Complete Compare dialog. You can see the selected data model in the Open Models in Memory list box.

   Alternatively, you can locate the required data model file on your computer and double-click to open it. The selected data model opens by default in the erwin model. Alternatively, if available, you can select required data model in the Open Models in Memory list box.

   If the default selection of the data model for the left side of the comparison is not correct, follow step 4 to select the required data model. Else, click Next to go to the Right Model input form.

   > **NOTE**   When you launch the Complete Compare function with one data model, that data model loads into the Right Model form of the Complete Compare dialog. When you open the second data model for comparison, existing data model in the Right Model form moves to the Left Model form, and the second data model loads into the Right Model form.

7. Right Model: Using the Right Model input form, you can select the data model for the right side of the comparison in the Complete Compare dialog. The Right Model input form is identical to the Left Model input form.

    To select the required model for the right side of comparison, click Load. The file Open dialog appears. Search for the required data model file, select it and click Open. The selected data model loads into the Complete Compare dialog. You can see the selected data model in the Open Models in Memory list box. Click Next to go to the Type Selection input form.

NOTE	When you launch the Complete Compare function with one data model, that data model loads into the Right Model form of the Complete Compare dialog. When you open the second data model for comparison, existing data model in the Right Model form moves to the Left Model form, and the second data model loads into the Right Model form.
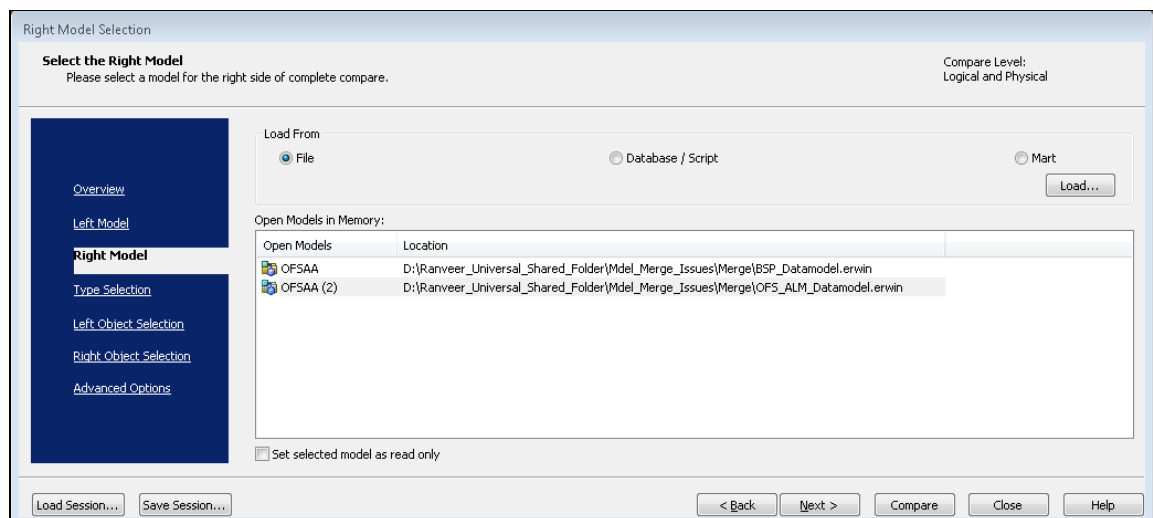
8. Type Selection: This form is critical in the entire Complete Compare process. Using this form, you can customize the layout for the merged output data model based on the existing layout of the input data models. This form restricts the comparison to specific object types and properties of the data models.

a. In the Type Selection input form, expand Model. In the list box, select the required options using checkbox.

NOTE	• If the Type Resolution dialog does not appear, go to the next form. Therefore, click Next. • When you select Type Selection form, it may prompt the display of the Type Resolution dialog to equalize the UDPs across the left and right models. This is GDM-46 error.

b. You can select the following type of multiple options in the Type Selection form:

— Subject Areas

— UDP

— Physical Table and Physical Column

— Logical Entity and Logical Attribute

— Subtype and Supertype

c. Below are the prompted values for Subtype (child table) set and Subtype items (child table elements) selection. To select all values from the Subtype Symbol, select the Subtype Symbol checkbox. Or to select individual Subtype items, expand Subtype Symbol, and select checkboxes against required Subtype items.

| NOTE	When the EAC-3033 error occurs, upload the merged model using the slice model and ignore the error.
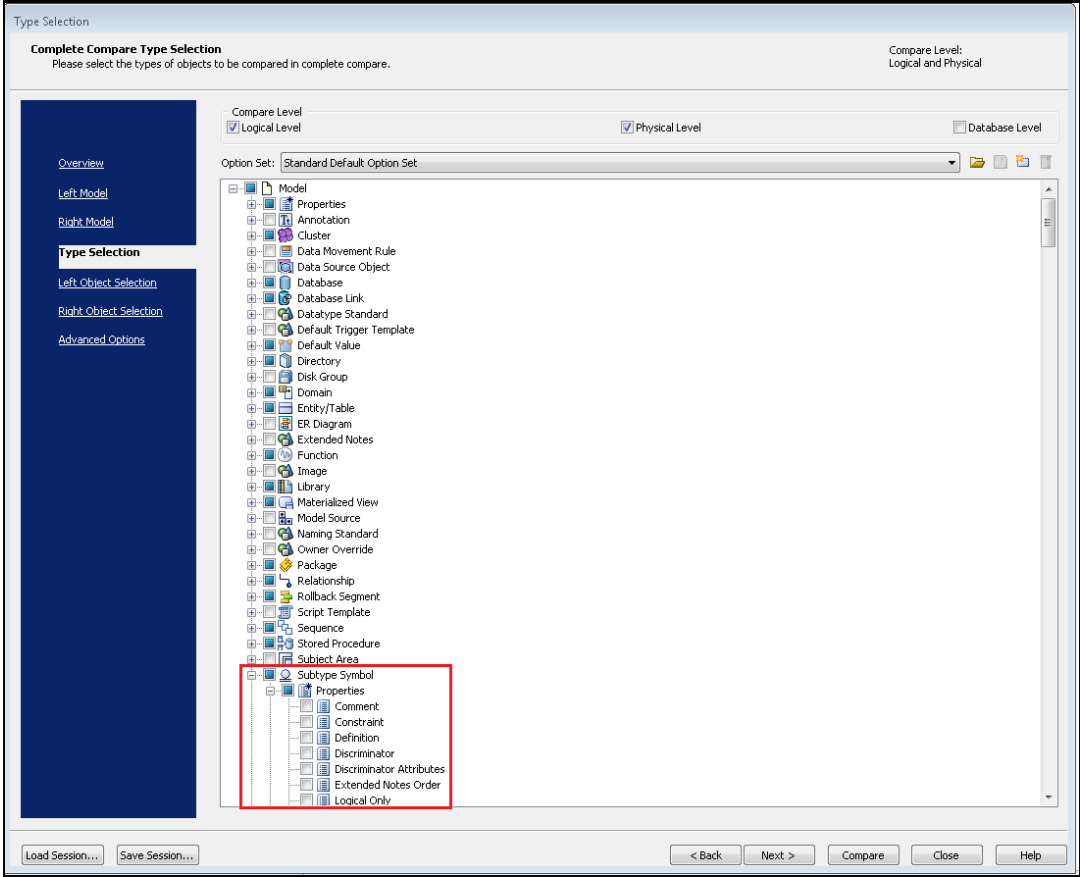
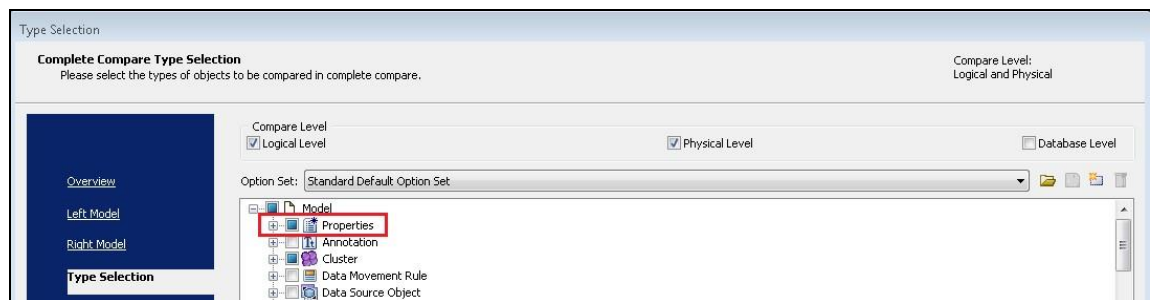**d.** Then click Next to go to the Left Object Selection form.

9. Left Object Selection: Using the Left Object Selection input form, you can select which objects (table properties) can be included from the Left Model form in the comparison process.

As shown in the image below, to choose the required Object Sets in the Choose Objects Using Sets list box, select the checkboxes against the Object Set names. The selected Object Sets appear in the Selected Objects list box on the right side.

NOTE	You can select individual item in each Object set. For example, expand Subject Areas object set, then select the individual checkboxes against required Subject Area item.
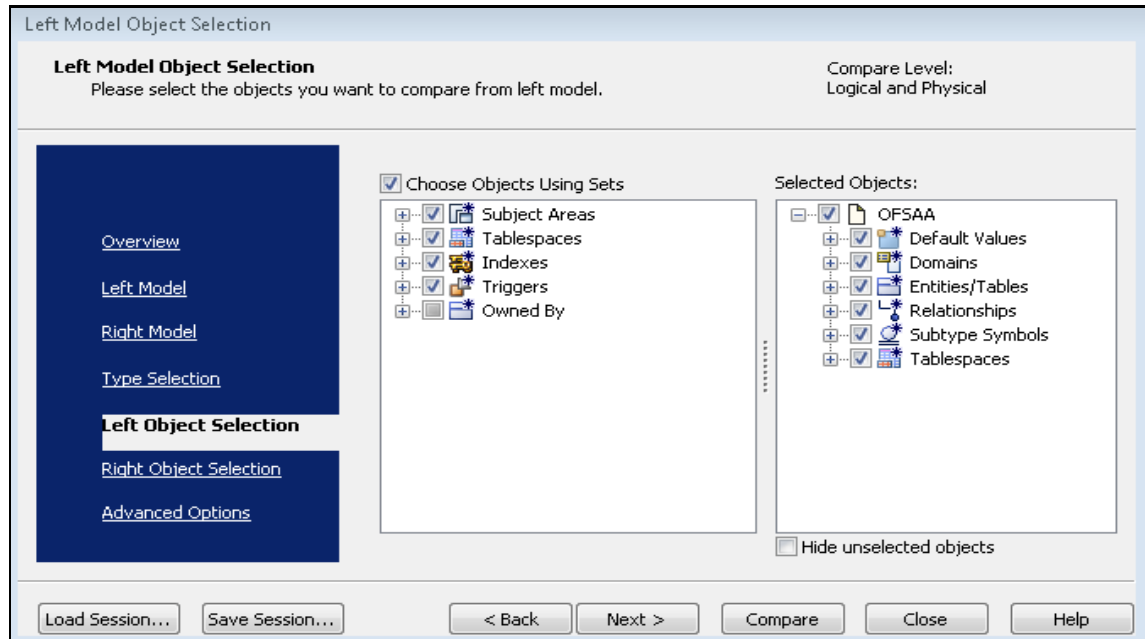
To include tables in the comparison process, you must select table Properties in the Type Selection form. Then the tables will appear in the Left Object Selection input form.


10. Right Object Selection: Using the Right Object Selection input form, you can select which objects (table properties) can be included from the Right Model form in the comparison process.

    To choose the required Object Sets from the Choose Objects Using Sets list box, select the checkboxes against the Object Set names (you can select Object Sets like Subject Areas, Tablespaces, Indexes, Triggers). The selected Object Sets appear in the Selected Objects list box on the right side. Click Next to go to the Advanced Options form.

| NOTE	• You can select individual item in each Object set. For example, if you are performing Subject Areas-wise Complete Compare, expand Subject Areas object set, then select the individual checkboxes against required Subject Area item. In the image below, based on apps requirement, we have selected all the Subject Areas.
	• If there are very less number of tables to merge with a very huge model, then you can create a Dummy Subject Area in a small model and move the tables into Dummy Subject Area. To Complete Compare under the Left Object Selection form or the Right Object Selection form, select the Dummy Subject Area and perform Complete Compare.
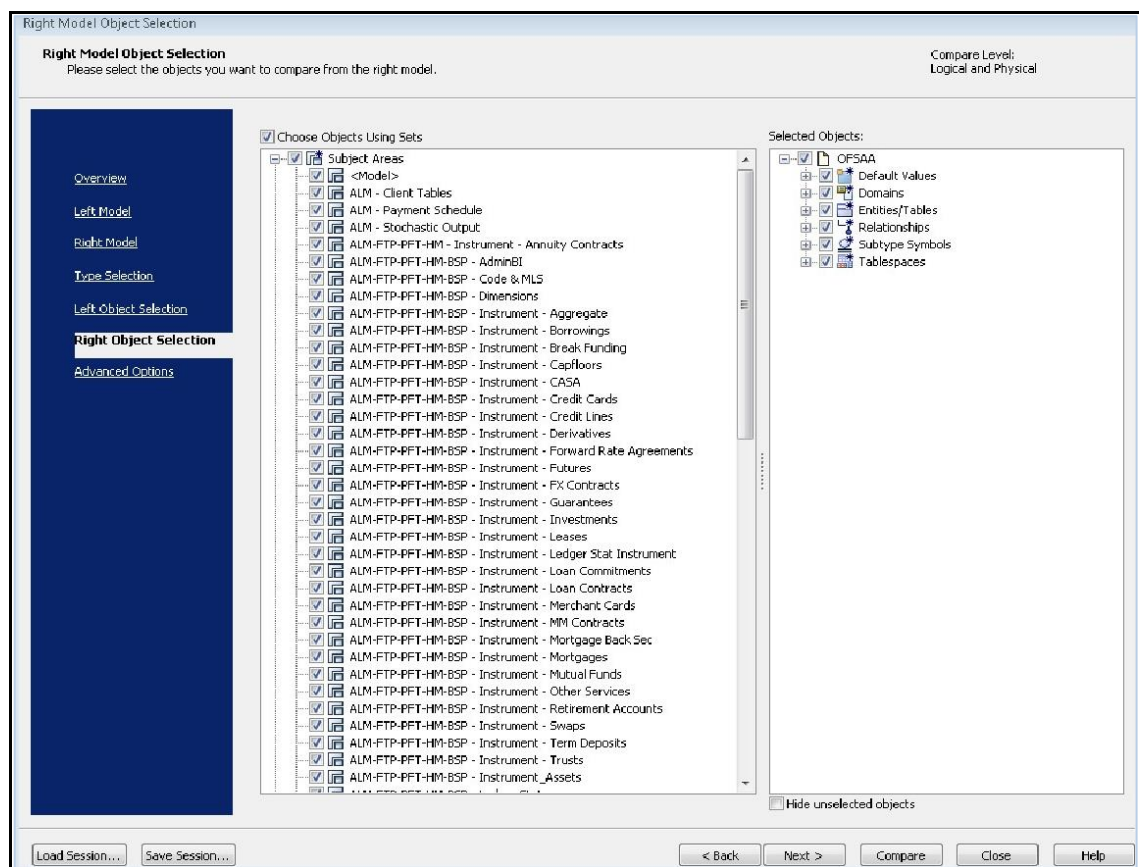
11. Advanced Options: This section consists of the functional Options. In the image below, the selected options are mandatory. As per your requirement, select further options.
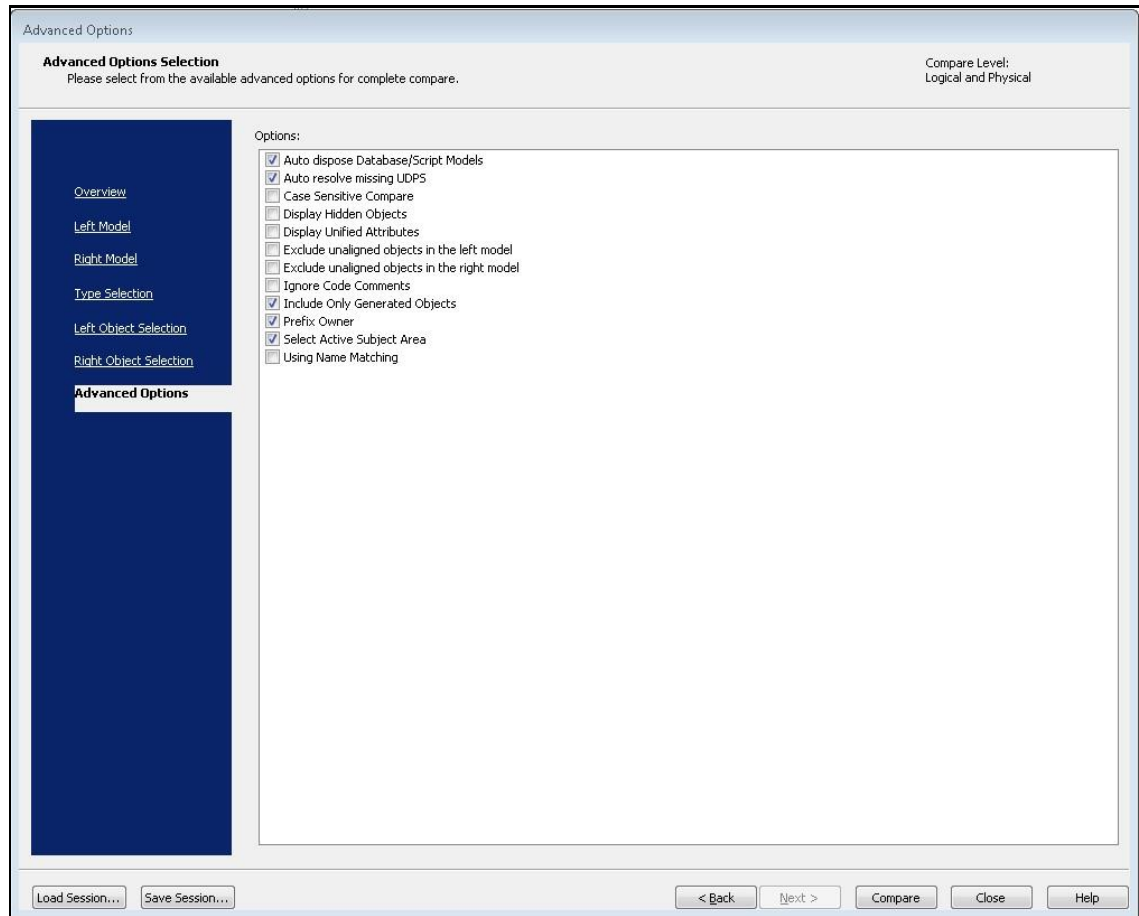
12. To initiate the comparison process, click Next and then click Compare. The Resolve Differences window opens.

13. In the Resolve Differences window, there are the three methods to merge data models:

   ▪ In the first method, you can merge two data models at the data model level. To merge a data model (Left Model) with another data model (Right Model), expand Model, then click arrow symbol. Or, to merge Right Model with Left Model, click arrow symbol.
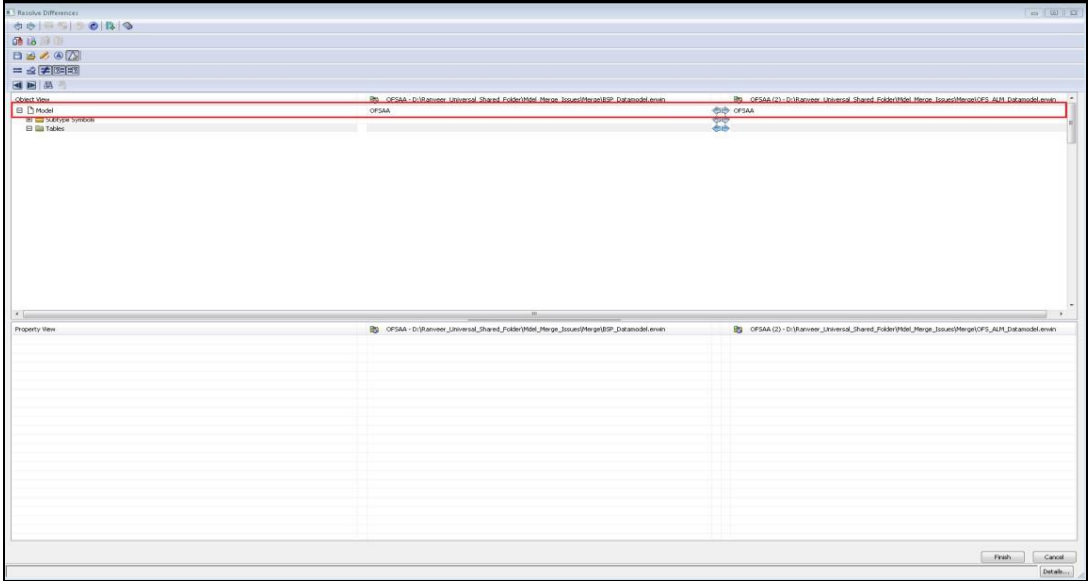
- In the second method, you can merge two data models at the table level. To merge two data models at the table level, expand Model, expand Tables, select ![icon] from the menu bar. Then to merge the Right Model with the Left Model at table level, click the required table and click the ![arrow] arrow symbol, and repeat the same for each table. Or, to merge the Left Model with the Right Model at the table level, click the required table and click the ![arrow] arrow symbol, and repeat the same for each table.
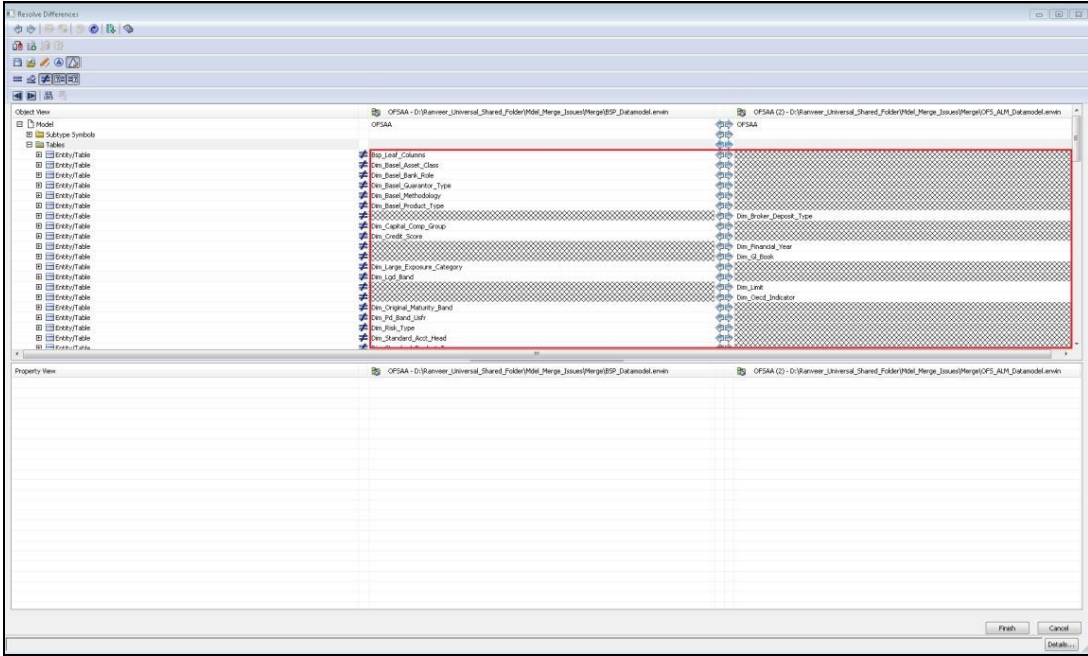

- In the third method, you can merge two data models at the Subtype and Supertype table level To merge two data models at Subtype and Supertype level:

  During an ongoing Supertype Subtype model complete comparison process, you cannot move the Supertype table directly from the Left Model or the Right Model. You must find all
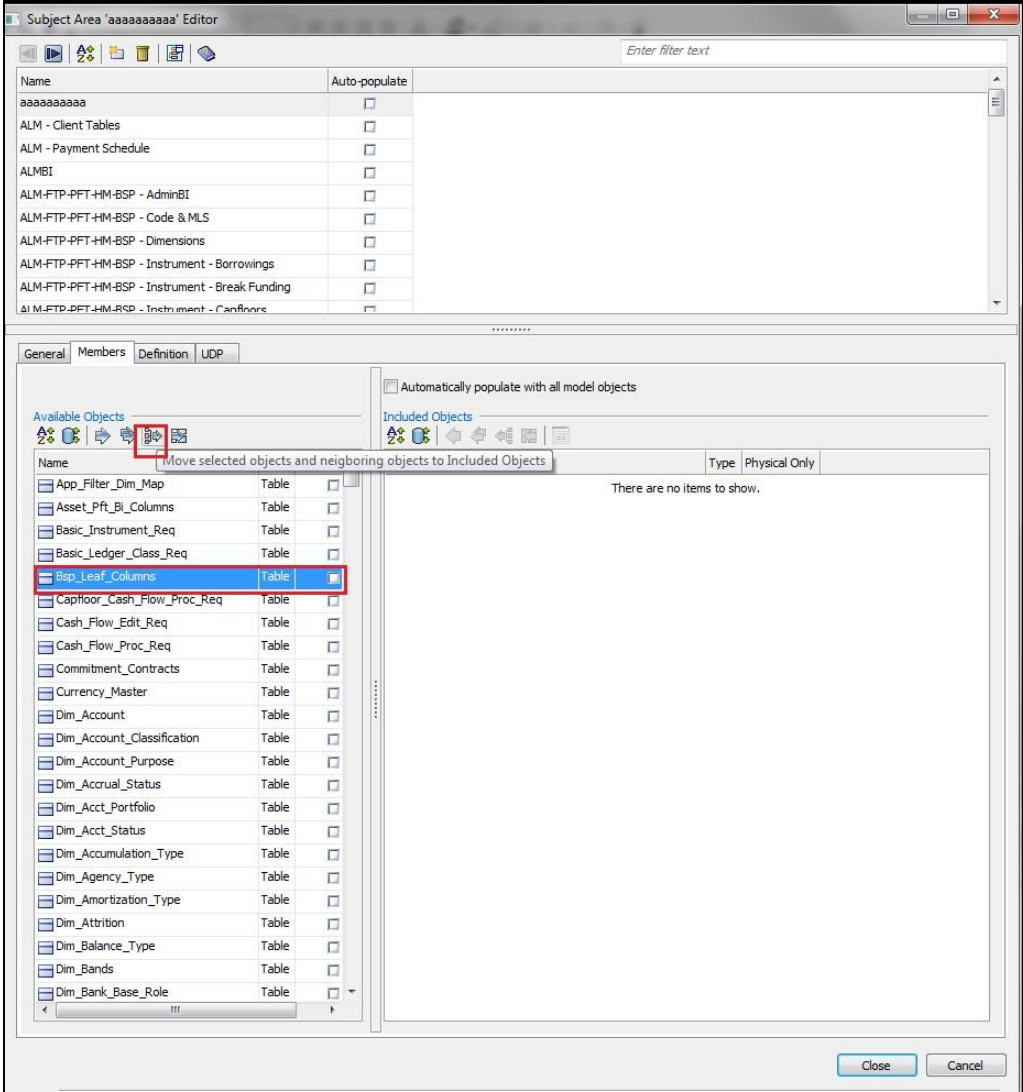
the relationships between the Supertype table (parent table) and Subtype tables (child tables). Then you can move the Subtype table and after that, move the Supertype table.

How to find the Supertype or Subtype table relationship?

i. Open the model where you are unable to move the table from left to right or from right to left. For illustration, we are considering BSP and ALM data models along with Supertype and Subtype.

ii. In Subject Areas, create a Dummy Subject Area with any name. Right-click Dummy Subject Area name, click Properties. The Dummy Subject Area '[name]' Editor window opens.
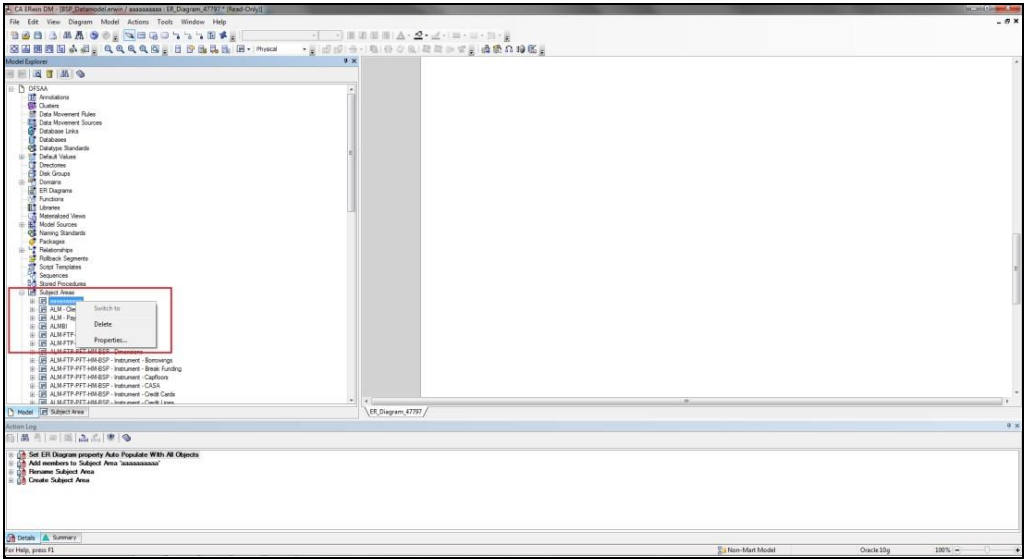
iii. In the Members tab, select the required Table Name and click symbol. This action moves selected object and the objects mapped to it from Available Objects list box to the Included Objects list box. In this example, in the Included Objects list box, BSP_LEAF_COLUMNS is the Supertype (parent table) and the remaining are Subtype (child tables).

iv. The Spanning Neighborhood window opens. In the Ancestors and Descendants sections, in Level, select value 1 to search for one-to-one parent-child table relationship. The list of the Subtype tables mapped to the Supertype table appears. In this example, the list of Subtype tables mapped to the Supertype table BSP_LEAF_COLUMNS appears.

v. To include the required Subtype tables (for this Dummy Subject Area) in the comparison process, select them and click Add. Close the Spanning Neighborhood window.

vi. In the Subject Area '[name]' Editor window, in the Included Objects list box, you can see the Supertype table (in this example, BSP_LEAF_COLUMNS) and its Subtype tables.
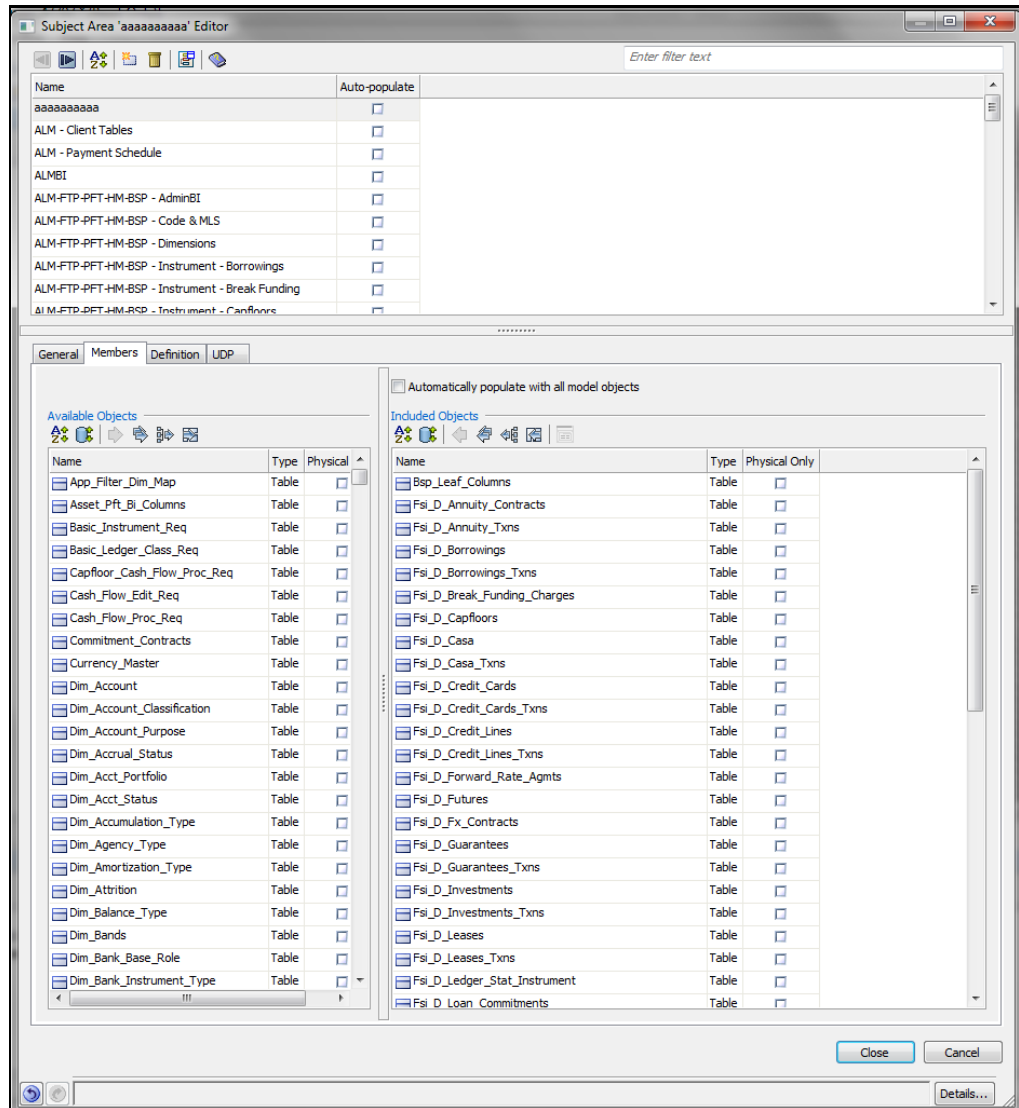
vii. Close the Subject Area '[name]' Editor window. In the data model window, click Save

viii. to save the changes made in the model.

ix. Repeat all the steps for data model merge until the Left Object Selection form (that is, repeat steps from 1 to 8).

x. In the Left Objection Selection form, in the Choose Objects Using Sets list box, select all the Subject Areas along with the newly created Dummy Subject Area. Into this Dummy Subject Area, you moved the Subtype and Supertype tables in previous steps. Click Next.

xi. Repeat steps 10 and 11.

xii. To initiate the comparison process, click Next and then click Compare. The Resolve Differences window opens.

xiii. In the Resolve Differences window, to merge two data models at table level, expand Model, expand Tables, select ⌗ from the menu bar. Select the Supertype table (BSP_LEAF_COLUMN in this example), and move it to left (to merge Right Model with

Left Model, click arrow symbol) or right (to merge Left Model with Right Model, click arrow symbol) side of the comparison as required.

In the considered illustration, BSP_LEAF_COLUMN is the parent table (Supertype) from the source table and its respective child tables (Subtype) are named as FSI_ACCID_HEALTH_INS_CO_CD, FSI_ACCID_HEALTH_INS_CO_MLS,

DIM_ACCOUNT_CLASSIFICATION and so forth. First, we moved the child table names (Subtype table names) from the source model to the target model, and then we moved the Supertype table name BSP_LEAF_COLUMN to the target model.

14. Click Finish to complete the data model merging, and then click Save to save the output-merged model with Merged data model as a part of the output model name.

12   Build Guidelines

Changes identified and approved needs to be included in the model per the naming standards.

- Table addition/modification

    - Data model standards to be followed for each change.

    - Make sure Entity name and description are provided.

    - Make sure values are set for relevant application usage UDP's like ALM, FTP, PFT, Basel II, CREC and so on.

- Column addition/modification

    - Data model standards to be followed for each change.

    - Make sure Attribute name, description are provided.

    - Make sure values are set for relevant application usage UDP's like ALM, FTP, PFT, Basel II, CREC etc.

    - Column level UDPs named 'Custom' (NO/YES), 'Customization Reason' (Pending ER, Specific to Customer, Modified data element) and 'Type of Change' (Length, Datatype, Logical Name, Description, Addition) have to be specified for attributes added/modified as part of the extension process.

> **NOTE**   Make sure that the Model Properties, Model Level UDP – VERSION is not changed while extending the model as per the requirement.

13   Appendix

For more information related to the usage of data models, refer to the following documents:

- [Data Model Naming Standards Guide](#).
- [Data Model Customization Review Template](#).

OFSAAI Support Contact Details

Raise an SR in My Oracle Support (MOS) if you have any queries related to EPM, ERM, and FCCM applications.

Send Us Your Comments

Oracle welcomes your comments and suggestions on the quality and usefulness of this publication. Your input is an important part of the information used for revision.

- Did you find any errors?

- Is the information clearly presented?

- Do you need more information? If so, where?

- Are the examples correct? Do you need more examples?

- What features did you like most about this manual?

If you find any errors or have any other suggestions for improvement, indicate the title and part number of the documentation along with the chapter/section/page number (if available) and contact the Oracle Support.

Before sending us your comments, you might like to ensure that you have the latest version of the document wherein any of your concerns have already been addressed. You can access My Oracle Support site which has all the revised/recently released documents.