

Oracle Data Integrator 11g

Hands-on Workshop

Student's Lab Book

PTS: Oracle Data Integrator 11g Workshop



ORACLE®
Platform Technology Solutions

Expert Series Enablement

Authors

Jignesh Mehta
Shirley Lum

Main Contributors

Naresh Nemani
Leon Liem
Jignesh Mehta
Gabriel Nistor
Farzin Barazandeh
Andrei Deneasa
Aniket Raut

Reviewers

Denis Gray
Alex Kotopoulos
Julien Testut
Danilo Camacho
Karthic Balasubramani
Richard Jacobs
Henry Xue

Workshop Image

Jignesh Mehta

Oracle Corporation
World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065
USA

Worldwide inquiries:
Phone: +1 650 506 7000
Fax: +1 650 506 7200

www.oracle.com

Oracle is the information company

Oracle is a registered trademark of Oracle Corporation.
Various product and service names referenced herein may be trademarks of Oracle Corporation. All other product and service names mentioned may be trademarks of their respective owners.

Copyright © 2010 Oracle Corporation

All rights reserved.

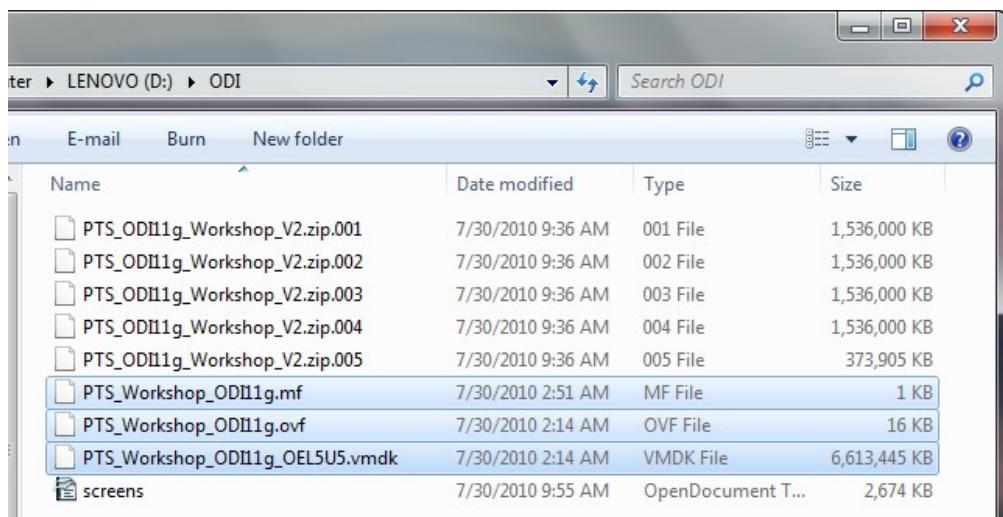
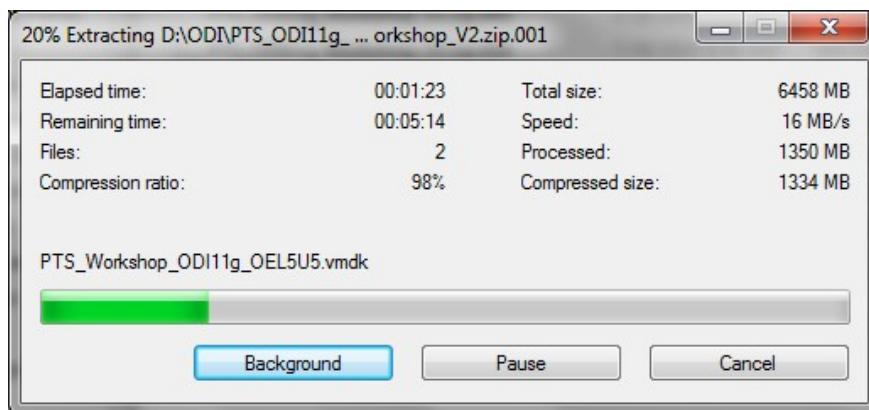
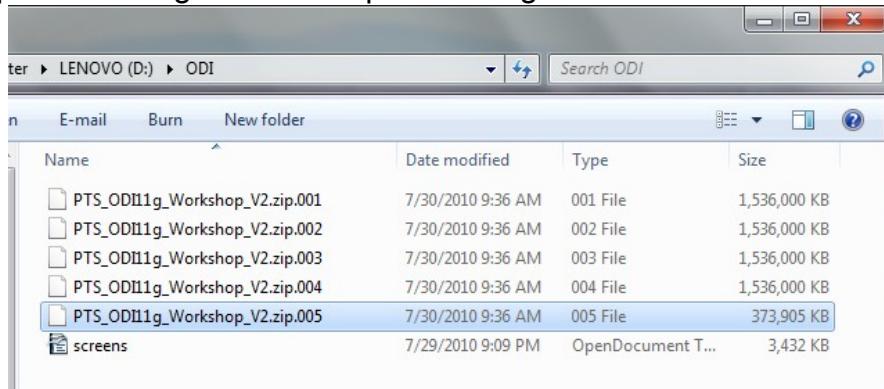
ORACLE®

Table of Contents

Lab 1: PTS ODI 11g Workshop Image Setup.....	4
Lab 2: Workshop Image Overview.....	13
Lab 3: Building First Interface Interface – Simple Transformation.....	19
Lab-4.1: Creating Master and Work Repository.....	31
Lab 4-2: Creating and Managing Topology.....	50
Lab 4-3: Creating Agents.....	67
Lab 5: Creating Models.....	78
Lab 6: Creating Project and Folders.....	86
Lab 7: Customizing Knowledge Modules.....	92
Lab 8: Variables, Functions, Procedures and Packages.....	103
Lab 9: Workflows and Scenarios.....	115
Lab 10: Advanced Interface – Complex Transformation.....	125
Lab 11: Advanced Interface – Using Data Sets with ODI Interface.....	147
Lab 12: Advanced Interface – Temporary Interface with Lookup.....	156
Lab 13: Change Data Capture with ODI.....	194
Lab 14: Version, Solution and Migration.....	201
Lab 15: Data Quality and Profiling.....	210
Lab 16: ODI and GoldenGate.....	232
Lab 17: Web Services and Extending ODI with SDK.....	254

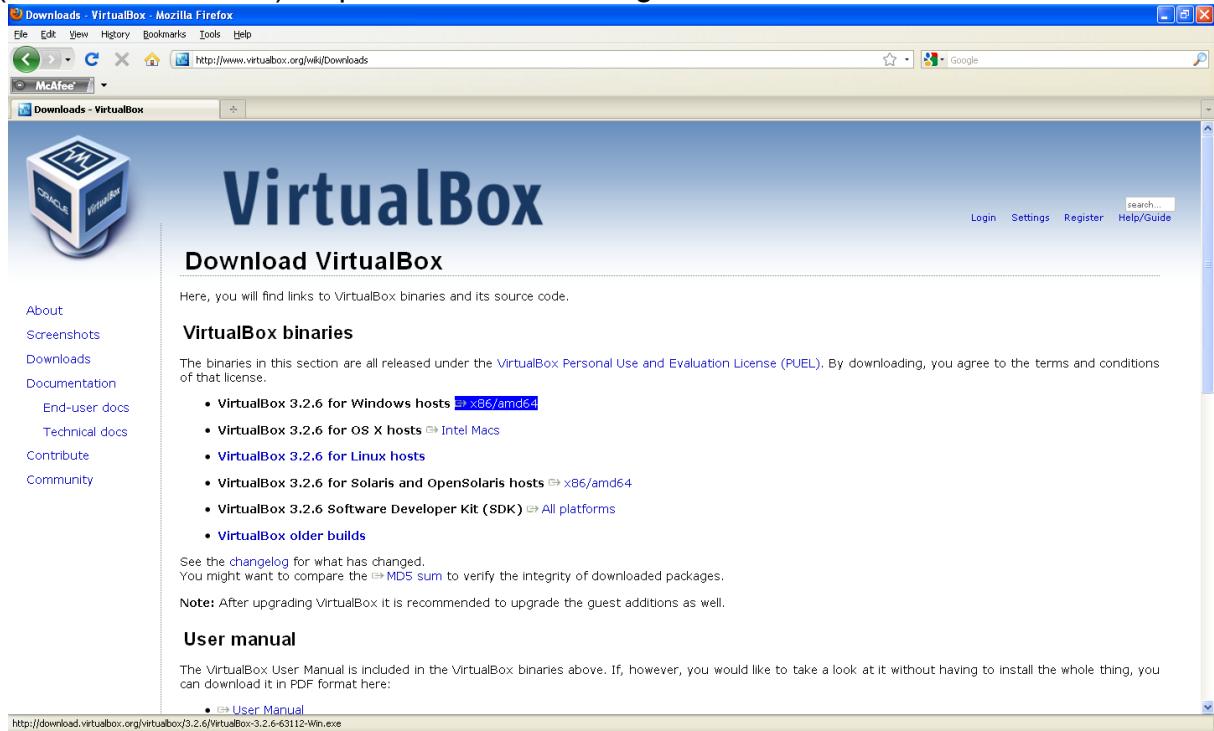
Lab 1: PTS ODI 11g Workshop Image Setup

1. Download five zip files and **unzip the first zip file only** which will create all required VM files and folder containing documents. Use of 7Zip utility is preferred. Right click on zip file ending with 001 and click on extract all.

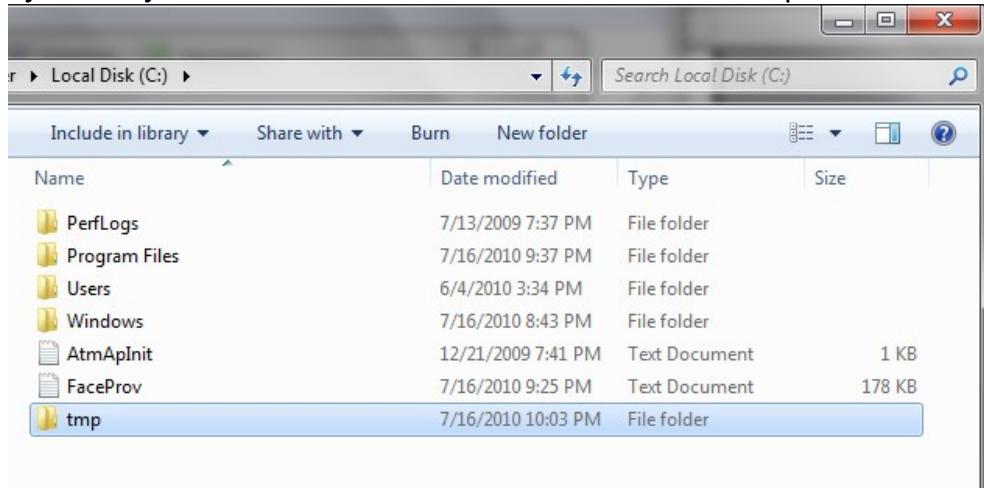


2. Download and install, if not already, Oracle VirtualBox for Windows hosts

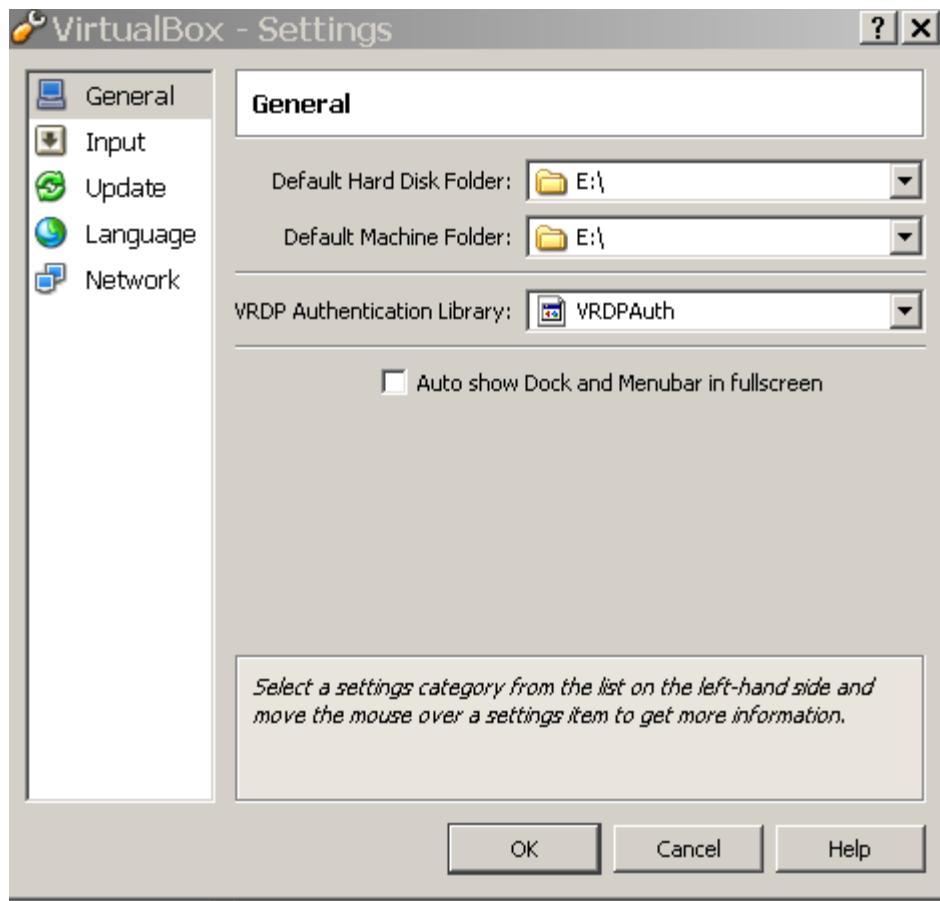
(3.2.8 latest version): <http://www.virtualbox.org/wiki/Downloads>



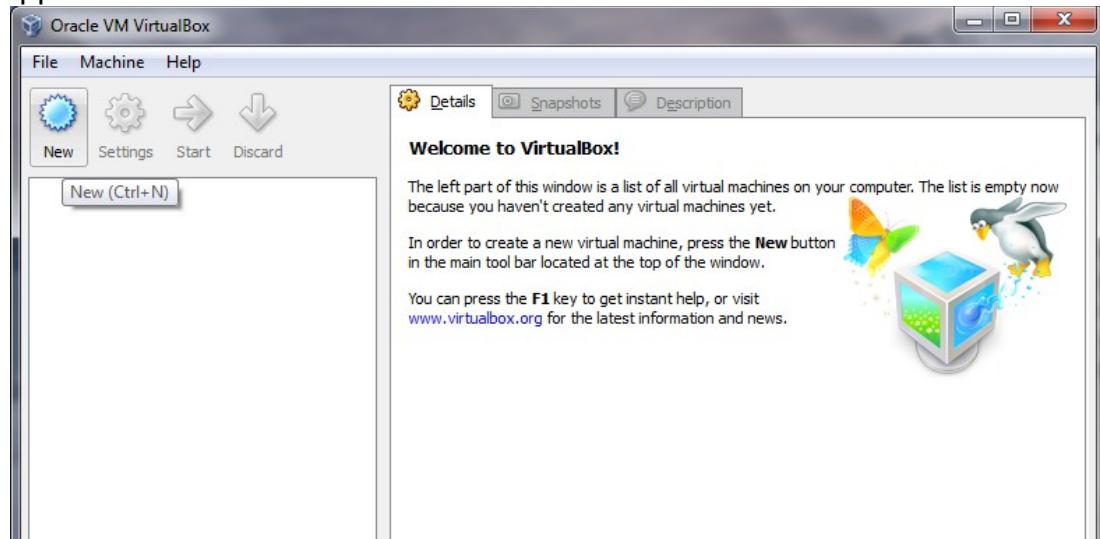
3. Make sure you have C:\tmp dir which is shared dir with Linux guest system if you want to share document across. This is optional.

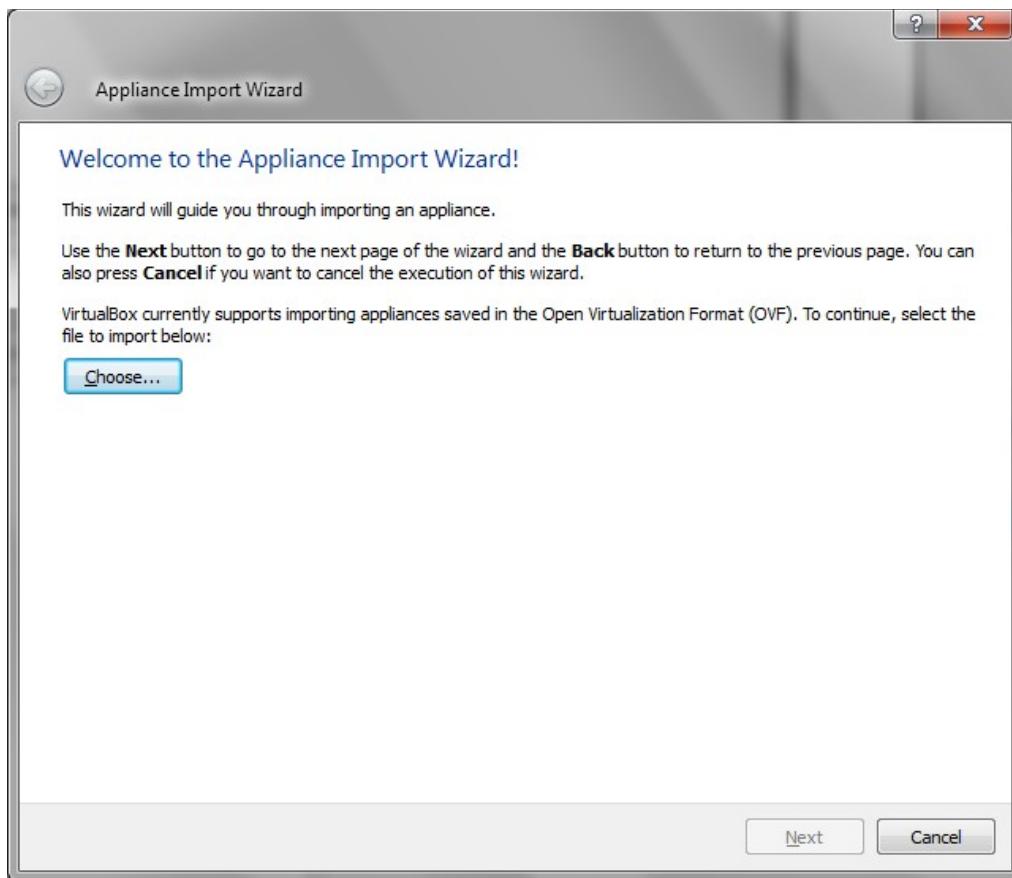


4. Open VirtualBox from Program Files. OPTIONAL: If you want to change the default properties for your virtual machine, click on Preferences from File menu and set the desired default folder locations.

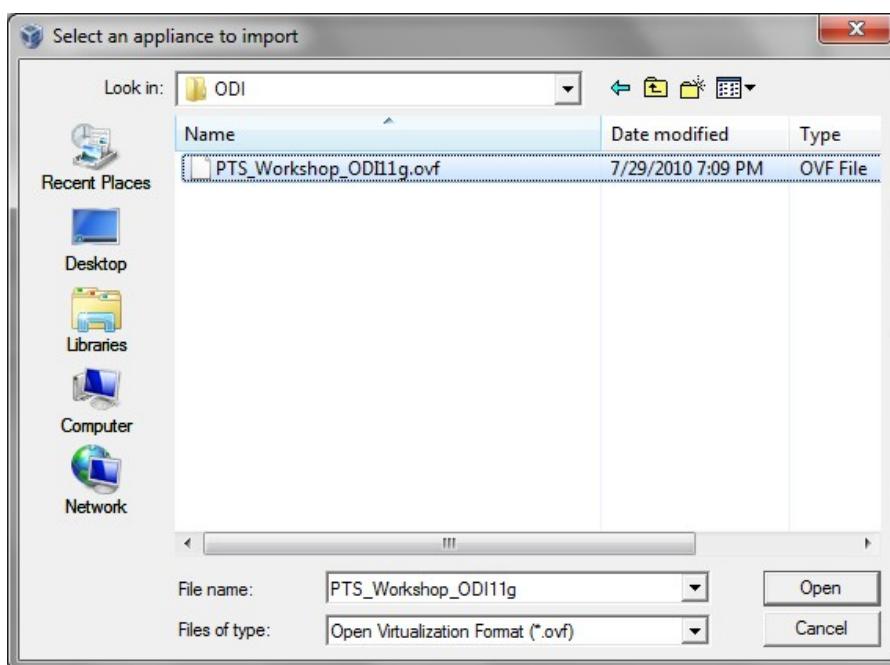


5. Click on Import Appliance Menu from File menu list which will bring Import Appliance wizard: Click Choose button

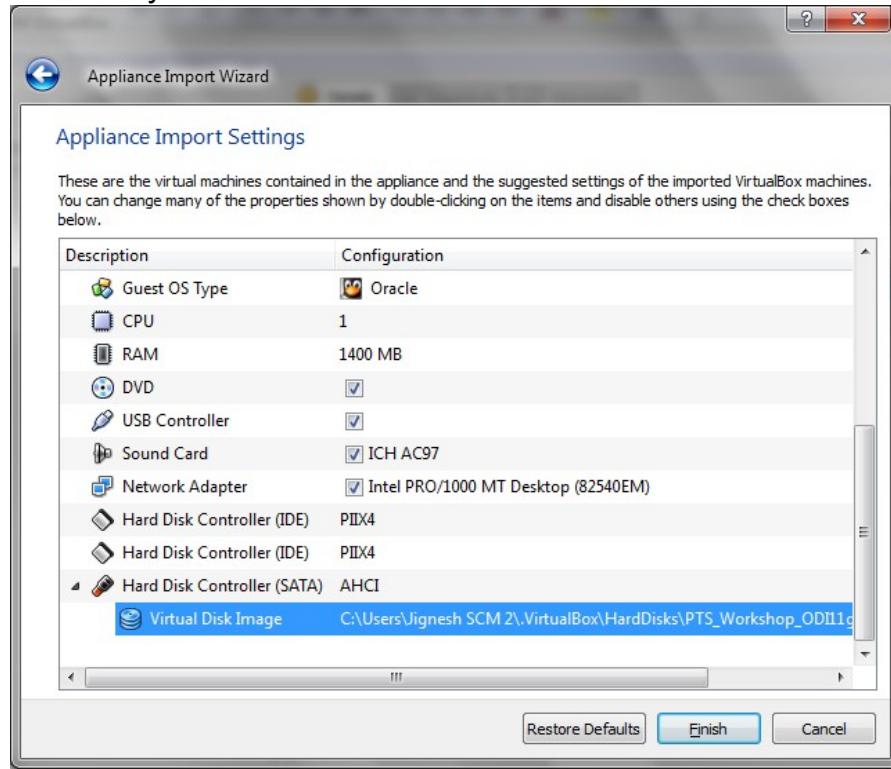




6. Select PTS_Workshop_ODI11g.ovf file from unzipped folder:



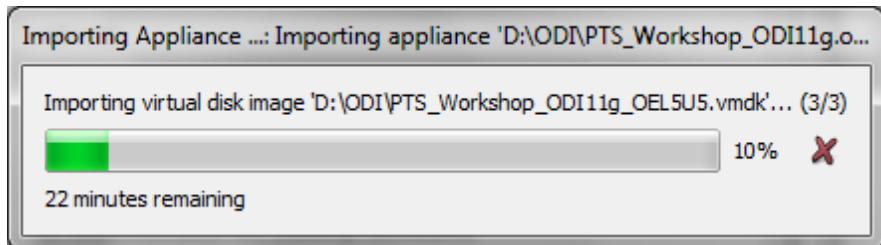
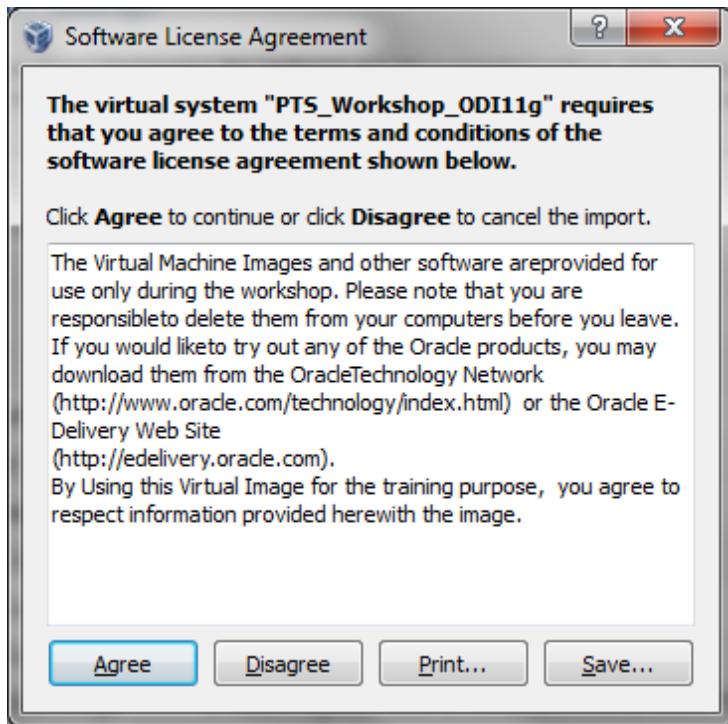
7. Import will display Image parameters, scroll down and select different directory for Image file that will be created, if you want to store it on different drive/directory than the default one:



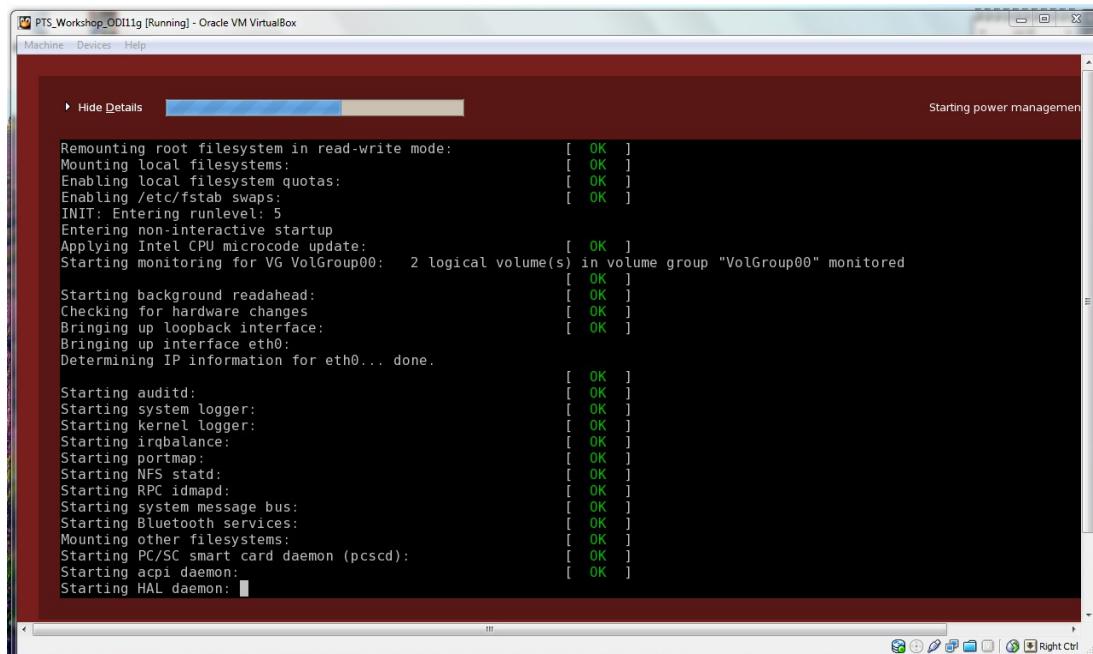
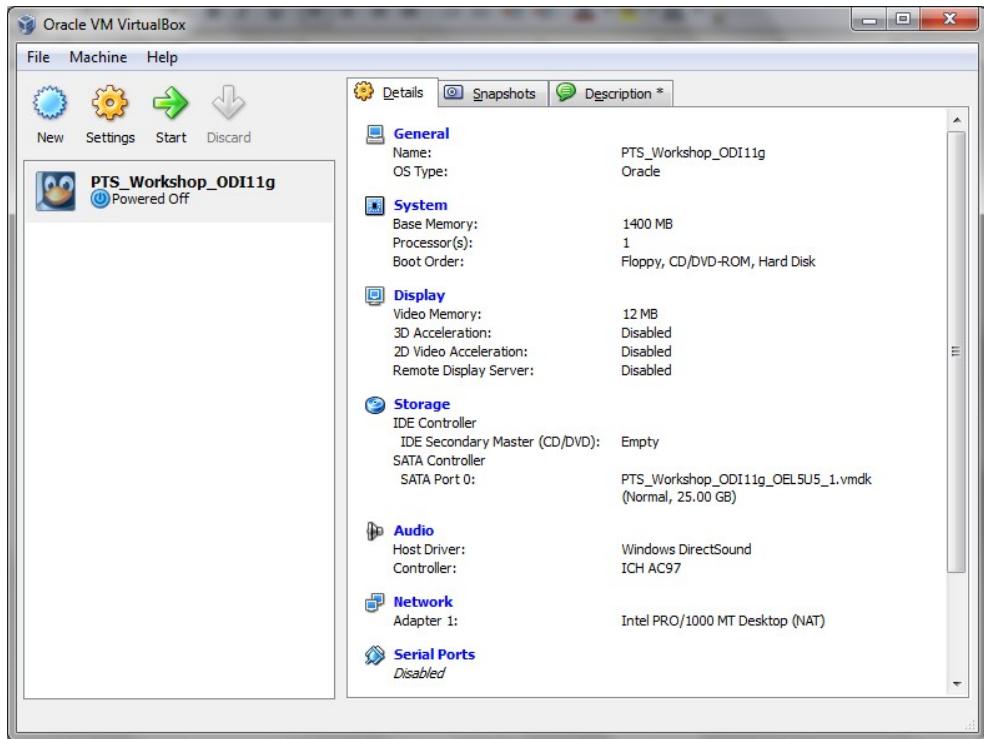
Note: Selecting a different directory for Image file from the Appliance Import Settings screen may not take effect. Alternatively, you can change the directory from File > Preferences as described in step 4 above.

Click Finish.

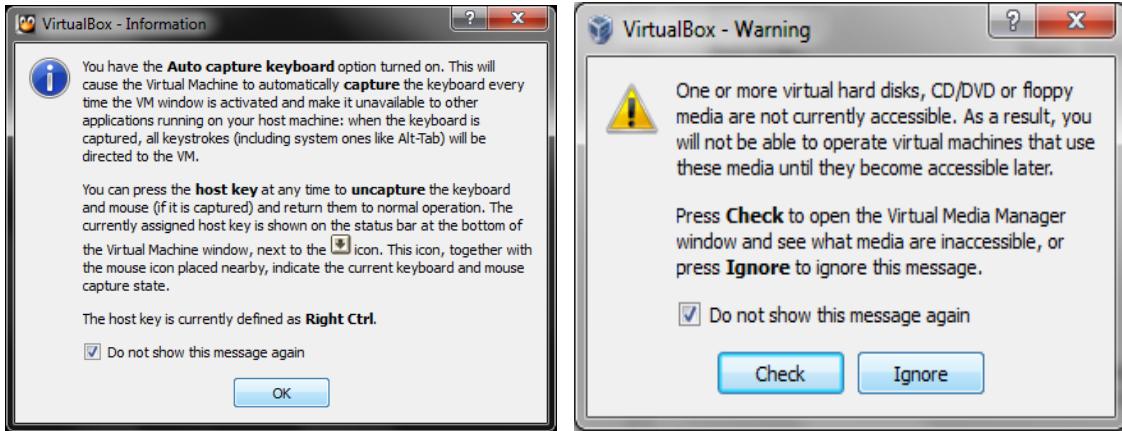
An agreement window will pop up to explain the purpose of VM image. Please click on Agree button to continue, which will create VM image disk on directory you specified on parameter page.



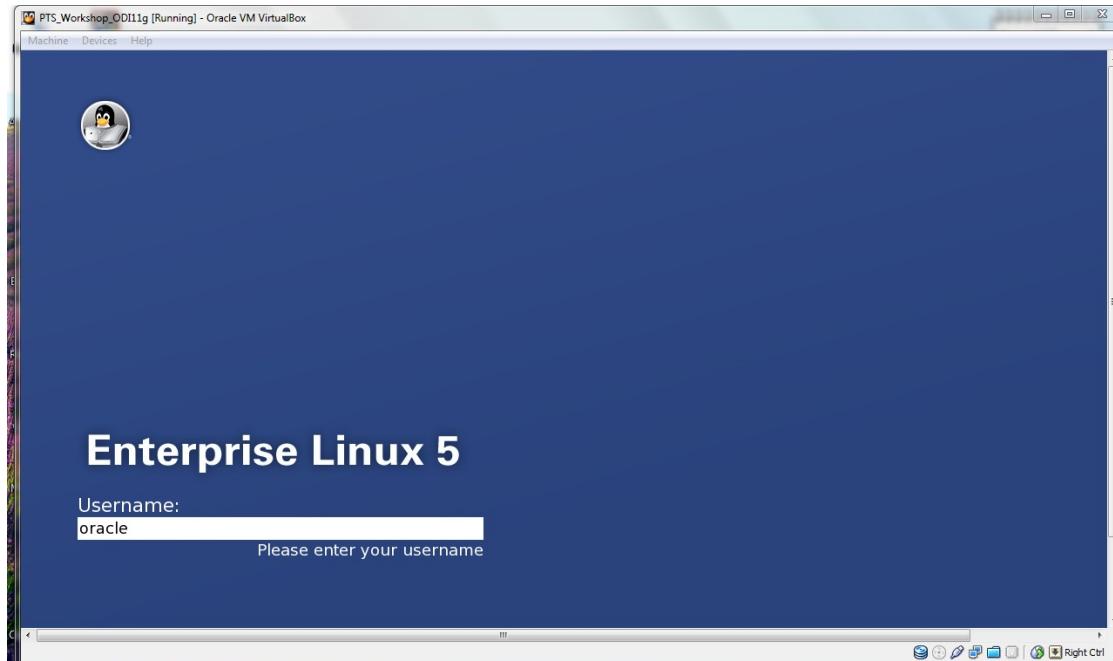
8. Once disk is created, you will be presented with configured VM image within VirtualBox window. Click on Green Start arrow to launch the image and to work within Linux guest operating environment.



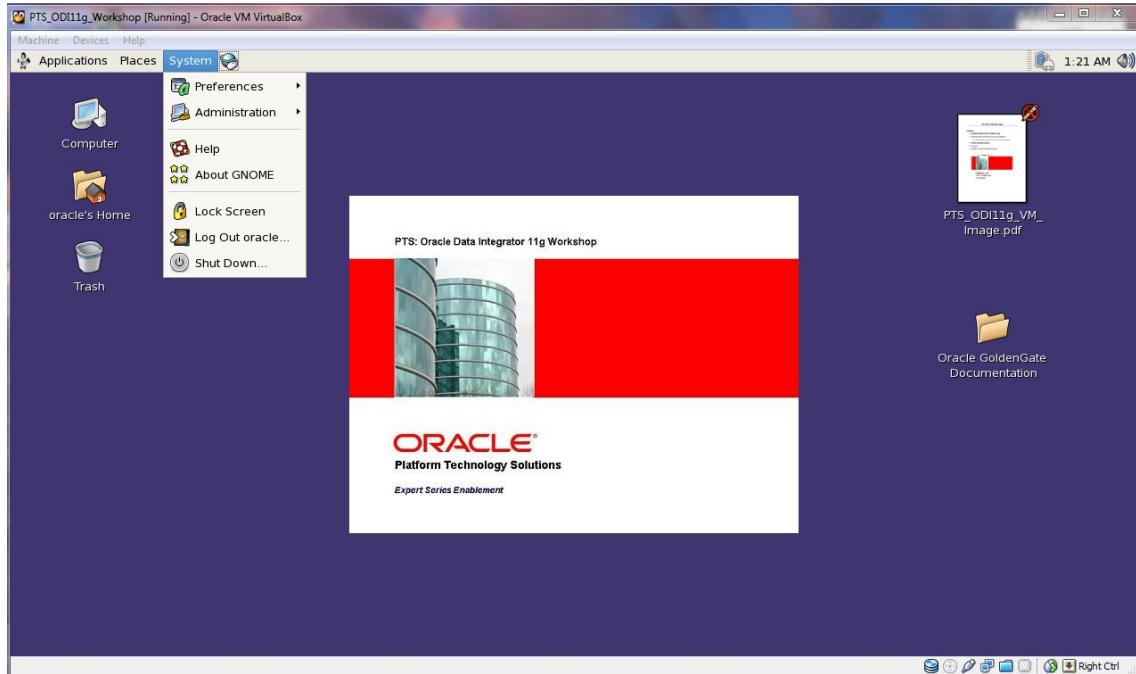
9. If popup come asking confirmation on Keyboard/Mouse capture, Click on check box and OK as image starts.



10. Once image starts, enter 'oracle' as user name and password



11. To open terminal window, just right click anywhere on the desktop and select Open Terminal menu. When you wish to shutdown VM image, remember to shutdown the image by going through system menu and selecting shutdown.



Lab 2: Workshop Image Overview

2.1 Lab Introduction

Lab Introduction: Workshop labs will help Data Integration Developers, Warehouse Developer, DBAs and Architects gain hands on experience with some of the best practices for implementing E-LT techniques to improve performance and reduce data integration costs using Oracle Data Integrator Enterprise Edition (ODI-EE). Labs are designed to help gain understanding of ODI features and techniques by allowing participants to walk through the steps that are needed to load and transform various sources into a relational table target. This will also include how to deploy and manage interfaces using ODI-EE and ODI environment.

Scenario: Our Company is in the process of building a Sales Data Warehouse to support BI system for reporting on our customer and sales data. The physical schema for our target star schema has been finalized and we have been assigned to build the interfaces and packages in ODI to load this star schema.

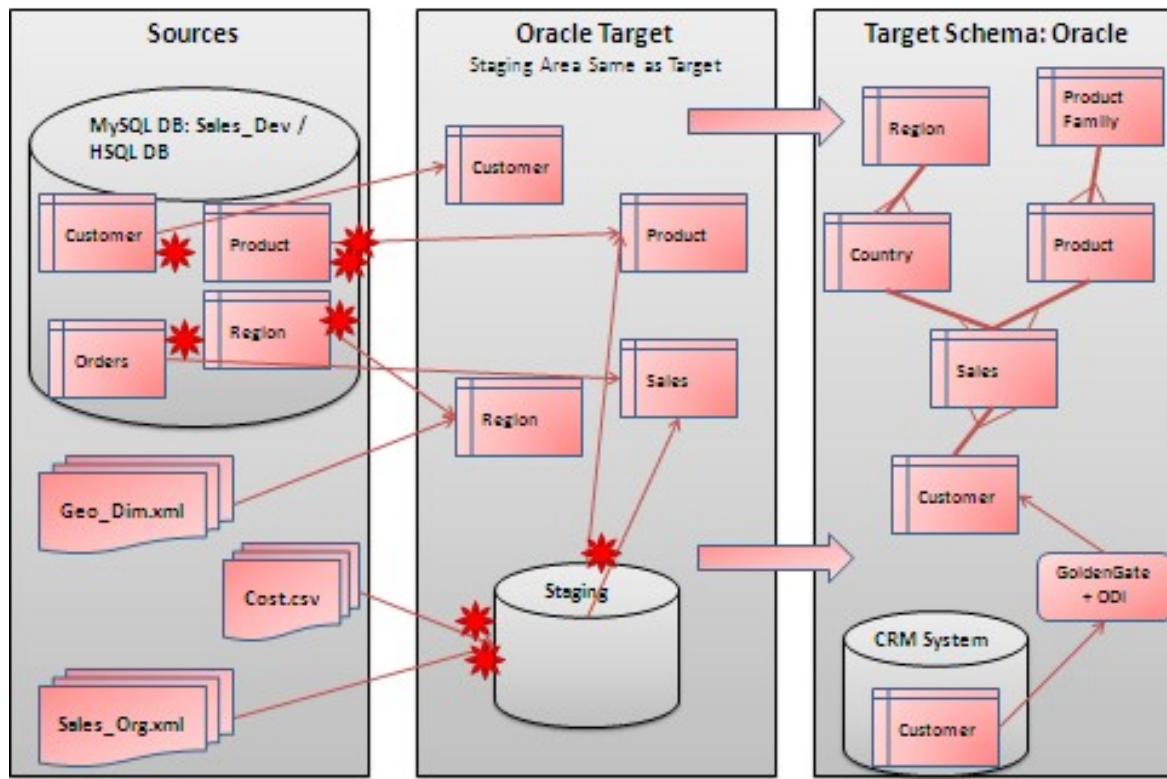
Our supporting data is in various heterogeneous systems, located across the enterprise. Sales, customer and country data will be imported from our Sales system located on MySQL Database Server. Our product management system contains our product and costing data and this information can be provided as comma delimited extracts. Our sales and marketing system is an internet based system; we will be provided an XML extract of our promotional data and sales org data from this system. The following lessons will walk us through various steps that are needed to create the interfaces and packages need to load the reporting star schema. With help of different labs, we will examine various aspects of interface design, deployment, execution and monitoring ODI projects. Each lab may provide more contexts to the scenario with respect to specific lab activities.

Prerequisites

Before you begin this workshop, you should

- Have a general understanding of RDBMS concepts.
- Have a general understanding of ETL concepts.

2.2 Architecture Overview



Sources:

Data Sources	Type
Customer	MySQL DB - Sales Dev / HSQL DB
Orders	MySQL DB - Sales Dev /HSQL DB
Product	MySQL DB - Sales Dev / HSQL DB
Region	MySQL DB - Sales Dev / HSQL DB
Sales Org	XML File
Cost	CSV File
Geo DIM	XML File

Targets:

Target DB	Type
Customer	Oracle DB
Product	Oracle DB
Sales	Oracle DB
Region	Oracle DB
CRM Customer Source/Target	Oracle DB

2.3 Workshop Image Overview

Once you start PTS ODI Workshop VirtualBox image, log into Enterprise Linux with username / password as **oracle / oracle**.

2.3.1 Software Components and Access Information

MySQL: Database: sales_dev - Source Schema. Username /pwd: pts / oracle

HSQL DB: username/pwd: sa / blank

To start HSQL, on terminal window, type > **startHSQL**

Oracle DB: Database: XE login info

Sales Data Warehouse: cust_dw_dev / cust_dw_dev - Target (TRG_*)
 cust_dw_prod / cust_dw_prod - Target Schema
 cust_dw_dev / cust_dw_dev - Dup Source (SRC_*)
 rep_trainingm / rep_trainingm - ODI Master Rep
 rep_trainingw / rep_trainingw - ODI Work Rep

ODI Repository username/password: SUPERVISOR / SUPERVISOR

Weblogic: login: weblogic / oracle123

Agent: J2EE Agent (OracleDIAgent) and Standalone Agent (localagent)

GoldenGate username / pwd: system / oracle

2.3.2 Components Access

MySQL and Oracle Database starts as a service when Image starts.

- A. Oracle Database Console: Oracle DB on VM Bookmark on Browser
- B. ODI Studio
 - a. Open Terminal
 - b. > **odistudio** # to launch ODI Studio
 - c. Login to PTS ODI11g Master or Work Rep
login: SUPERVISOR / SUPERVISOR
 - d. # open new terminal to start stand alone agent
 - e. > **startLocalAgent** # to start local agent deployed as stand alone agent
- C. SQL Developer
 - a. Open Terminal
 - b. > **sqldev** # to launch SQL Developer Tool
 - c. There are preconfigured connection to Oracle DB and MySQL DB to browse the tables and data and/or to clean up data for lab.
- D. Goldengate user / pwd: ggs_owner / ggs_owner
 - a. **ggsci** # log into Goldengate to test connection
 - b. ggsci> **DBLOGIN** USERID system, **PASSWORD** oracle
 - c. ggsci> **exit;** # exit from Goldengate connection to database

E. Weblogic Server

A. Start Node Manager

- a. Open Terminal # Do not close this window, minimize it
- b. > **startNodeManager** # to start node manager for weblogic

B. Start Weblogic Server – admin domain

- a. Open Terminal
- b. > **startWLS** # to start weblogic

C. Start ODI Domain – ptsodi_domain

- a. Open Terminal
- b. > **startODIServer** # to start odi doamin. When asked, please enter username / pwd: weblogic / oracle 123

D. Middleware EM, WLS and ODI Console

- a. <http://pts.us.oracle.com:7001/em>
Login: weblogic / oracle123
- b. <http://pts.us.oracle.com:7001/console>
Login: weblogic / oracle123
- c. <http://pts.us.oracle.com:8001/odiconsole>
Login: SUPERVISOR / SUPERVISOR

E. Stop Weblogic Services

- a. Open Terminal
- b. > **stopODIServer** # to stop odi doamin, enter user/pwd
- c. > **stopWLS** # to stop admin doamin
- d. Close terminal window running node manager

2.3.3 Directory Path and Shortcuts

1. > oraclehome # dir to oracle home where database is installed
2. > odihome # dir to dir where ODI Developer is installed
3. > wlshome # dir where weblogic is installed
4. > odidomain # dir to weblogic domain for odi
5. > hostshare # dir to shared files/folder with host OS

Note: On Windows host make sure you have C:\tmp dir, which is default shared directory with linux environment for our image

2.3.4 Other Resources

1. ODI Documentation:

<http://www.oracle.com/technetwork/middleware/data-integrator/overview/index.html>

<http://www.oracle.com/technetwork/middleware/data-integrator/documentation/index.html>

<http://www.oracle.com/technetwork/middleware/data-integrator/community/index.html>

2. Database Documentation:

<http://www.oracle.com/pls/db112/homepage>

3. Weblogic Documentaiton:

http://download.oracle.com/docs/cd/E14571_01/index.htm

Lab 3: Building First Interface – Simple Transformation

Introduction

An integration interface consists of a set of rules that define the loading of a datastore or a temporary target structure from one or more source datastores.

An integration interface is made up of and defined by the following components:

- **Target Datastore** - The target datastore is the element that will be loaded by the interface.
- **Datasets** - One target is loaded with data coming from several datasets. Set-based operators (Union, Intersect, etc.) are used to merge the different datasets into the target datastore.
- **Diagram of Source Datastores** - A diagram of sources is made of source datastores – possibly filtered – related using joins. The source diagram also includes lookups to fetch additional information for loading the target.
- **Mapping** - A mapping defines the transformations performed on one or several source columns to load one target column. These transformations are implemented in the form of SQL expressions.
- **Staging Area** - The staging area is a logical schema into which some of the transformations (joins, filters and mappings) take place. It is by default the same schema as the target's logical schema. It is possible to locate the staging area on a different location (including one of the sources).
- **Flow** - The flow describes how the data flows between the sources, the staging area if it is different from the target, and the target as well as where joins and filters take place. The flow also includes the loading and integration methods (Loading and Integration Knowledge Modules - LKM, IKM).
- **Control** - An interface implements two points of control. Flow control checks the flow of data before it is integrated into the target. Post-Integration control performs a static check on the target table at the end of the interface (Check Knowledge Module - CKM).

The Interface Editor provides a single environment for designing integration interfaces. The Interface Editor enables you to create and edit integration interfaces.

Prerequisites

Login to ODI Studio by following instructions provided in Lab 2.3.2 Components Access. Connect to Work Repository **PTS Training Work Repository** from ODI Studio with user **SUPERVISOR** and password **SUPERVISOR**.

Process Overview

The following step sequence is usually performed when creating an interface, and can be used as a guideline to design your first interfaces:

1. Create a New Interface
2. Define the Target Datastore
3. Define the Datasets
4. Define the Source Datastores and Lookups
5. Define the Mappings
6. Define the Interface Flow
7. Set up Flow Control and Post-Integration Control
8. Execute the Integration Interface

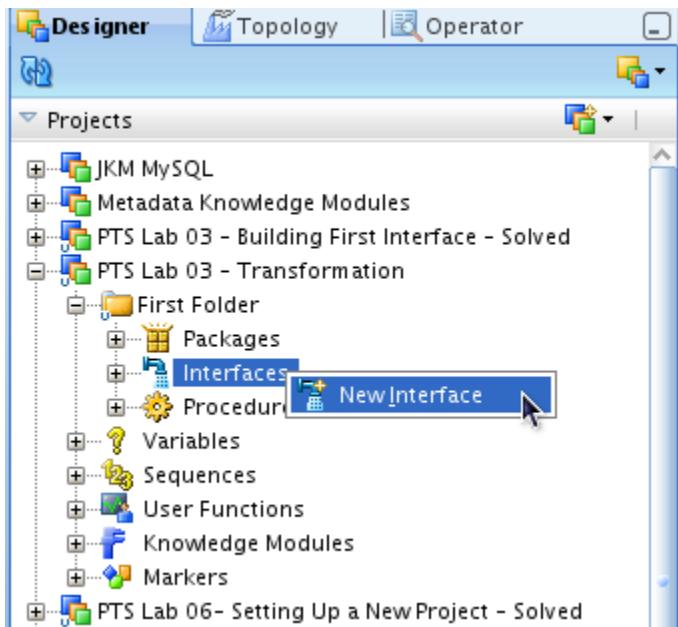
Scenario

We will first start with what ODI developers do most: design data transformations and data flows. In our first set of exercises, we will use an environment where all administration tasks have been done for us: connectivity to the databases, import of the metadata, so that we can focus immediately on what makes “a day in the life of an ODI developer.” Later in this training, we will get back to the definition of the connectivity and the import of the metadata.

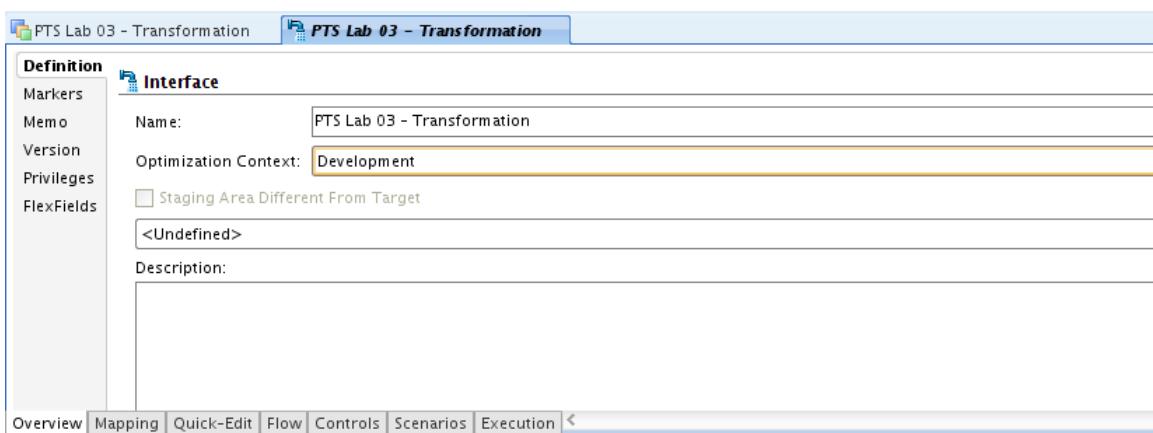
Instructions

- 1) In the project **PTS Lab 03 – Building First Interface**, expand First Folder, right-click on Interfaces, and create a new interface called “PTS Lab 03 – Transformation” loading the TRG_CUSTOMER datastore in the Oracle Sales Warehouse model with the content of the src_customer table from the MySQL Sales Data Source model. You must map the columns of the same name without any transformation. Ensure that you take only distinct records.

- a) In Designer, click the Projects tab, and expand the “PTS Lab 03 – Building First Interface” project. Expand the First Folder. Select the Interfaces node, right-click and select the New Interface option.



- b) In the Interface window, enter PTS Lab 03 - Transformation as the name of the interface. Ensure that the Optimization Context is Development. Click the Mapping tab.



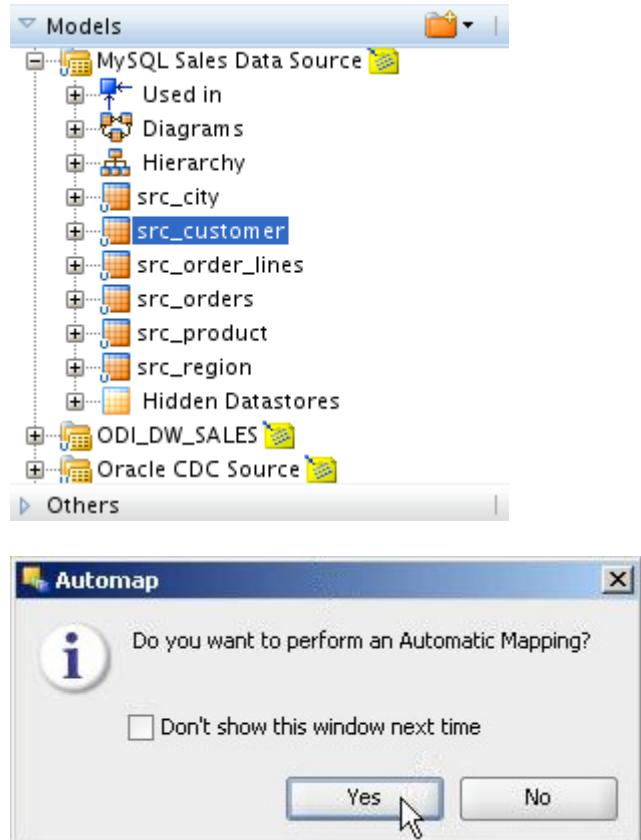
- c) In the Designer Navigator to the left, click the Models tab. In the tree view, expand the Oracle Sales Warehouse model. Drag the TRG_CUSTOMER datastore from the tree view to the **Target Datastore** zone (see screenshot below: the panel to the *right* of the panel with text stating “Target Datastores.... as sources for this dataset”). The datastore appears in this zone.

The screenshot shows the Oracle Data Integrator Designer interface. On the left, the **Models** tree view displays the Oracle Sales Warehouse model with various datastores: TRG_CITY, TRG_COUNTRY, TRG_CUSTOMER, TRG_INVENTORY, TRG_PRODUCT, TRG_PROD_FAMILY, TRG_REGION, and TRG_SALES. The **Target Datastore - TRG_CUSTOMER** panel on the right lists columns with their corresponding names and mappings:

Position	Indicators	Name	Mapping
1		*CUST_ID	
2		DEAR	
3		CUST_NAME	
4		ADDRESS	
5		*CITY_ID	
6		PHONE	
7		AGE	
8		AGE_RANGE	
9		SALES_PERS	
10		CRE_DATE	
11		UPD_DATE	

A tooltip in the center of the screen reads: "Drag datastores from the Designer Navigator Models tree view here to use them as sources for this dataset".

- d) Expand the MySQL Sales Data Source model and drag the src_customer datastore from the model tree to the **Sources** zone of your diagram (the panel to the **left** of the target panel). An Automap dialog box appears. Click Yes. Now the system automatically maps fields in the source and target datastores.

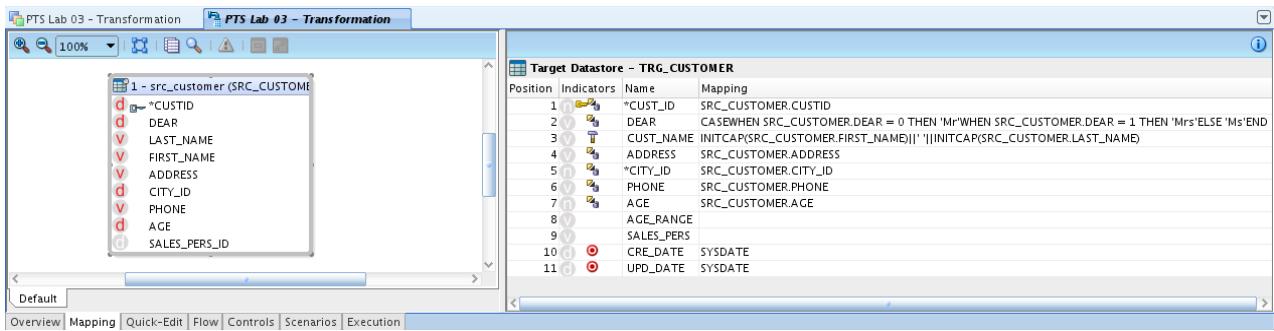


e) Use the following table to map and verify all columns mapping:

MySQL server column	Oracle Column	Transformation	SQL Syntax
Dear	Dear	Change 0 into Mr Change 1 into Mrs Change 2 into Ms	Case when <expression> then <value> when <expression> then <value> else <value> End
First_Name, Last_Name	CUST_NAME	Concatenate the first name and the last name. Capitalize the first letter for both. Put a space between first and last name	Use a + sign on SQL server, or a on Oracle. Use InitCap on Oracle to initialize the first letter.
Address	Address	None	
Phone	Phone	None	
Age	Age	None	
	Age_Range	Not mapped	
	Sales_Pers	Not Mapped	
	Cre-Date	System date on the target machine. Do not modify this value if there is an update	Sysdate on Oracle
	Upd_Date	System date on the target machine	Sysdate on Oracle
Custid	Cust_id	None (Beware: this field is not automatically mapped!)	
City_id	City_id	None	

To perform a mapping:

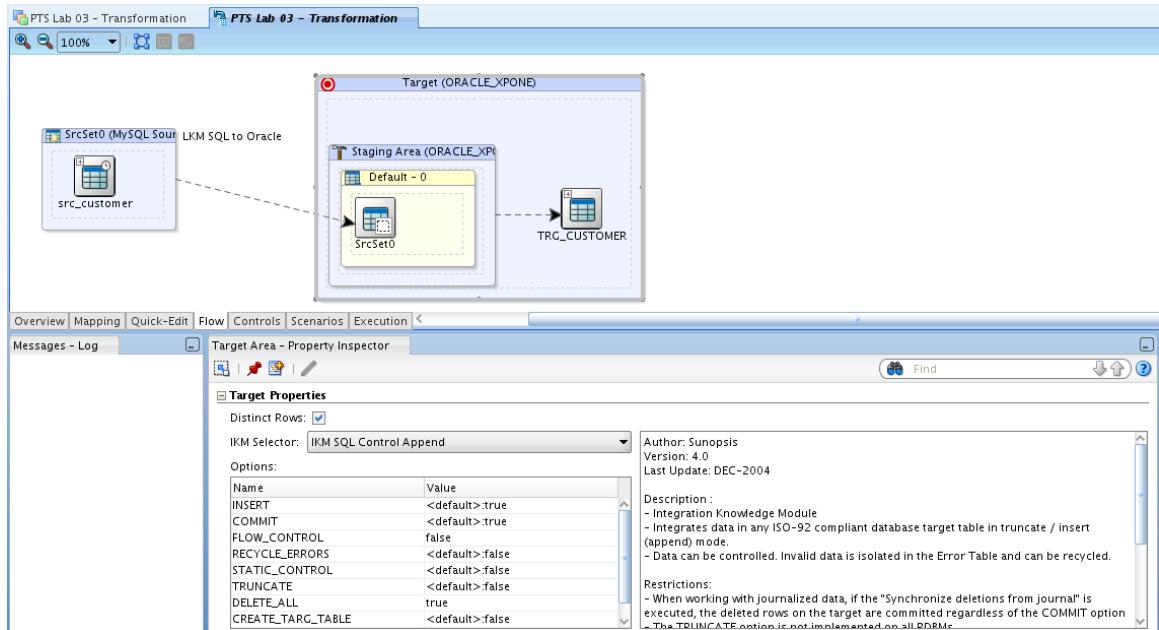
- Clicking on the target column will open the mapping area at the bottom of your screen
- Drag and drop the columns you need for your transformations in that mapping area (**not on the target column**, how tempting it may be). You can build your transformation logic in this field.
- If you scroll down a little bit in the mapping area, you can choose on which system the mapping will be executed (source, staging area, target). Except for literals, mappings are always on the source or the staging area. The staging area is on the target system by default.
- Write the mapping expression using the appropriate SQL for the database you have chosen. You can use the expression editor  to get some help with the appropriate syntax.



The screenshot shows the Oracle Data Integrator Transformation interface. The title bar says "PTS Lab 03 - Transformation". The left pane displays a source table "1 - src_customer (SRC_CUSTOMER)" with columns: CUST_ID, DEAR, LAST_NAME, FIRST_NAME, ADDRESS, CITY_ID, PHONE, AGE, and SALES_PERS_ID. The right pane shows the "Target Datastore - TRG_CUSTOMER" mapping table:

Position	Indicators	Name	Mapping
1		"CUST_ID"	SRC_CUSTOMER.CUSTID
2		DEAR	CASEWHEN SRC_CUSTOMER.DEAR = 0 THEN 'Mr' WHEN SRC_CUSTOMER.DEAR = 1 THEN 'Mrs' ELSE 'Ms' END
3		CUST_NAME	INITCAP(SRC_CUSTOMER.FIRST_NAME) ' ' INITCAP(SRC_CUSTOMER.LAST_NAME)
4		ADDRESS	SRC_CUSTOMER.ADDRESS
5		"CITY_ID"	SRC_CUSTOMER.CITY_ID
6		PHONE	SRC_CUSTOMER.PHONE
7		AGE	SRC_CUSTOMER.AGE
8		AGE_RANGE	
9		SALES_PERS	
10	(red circle)	CRE_DATE	SYSDATE
11	(red circle)	UPD_DATE	SYSDATE

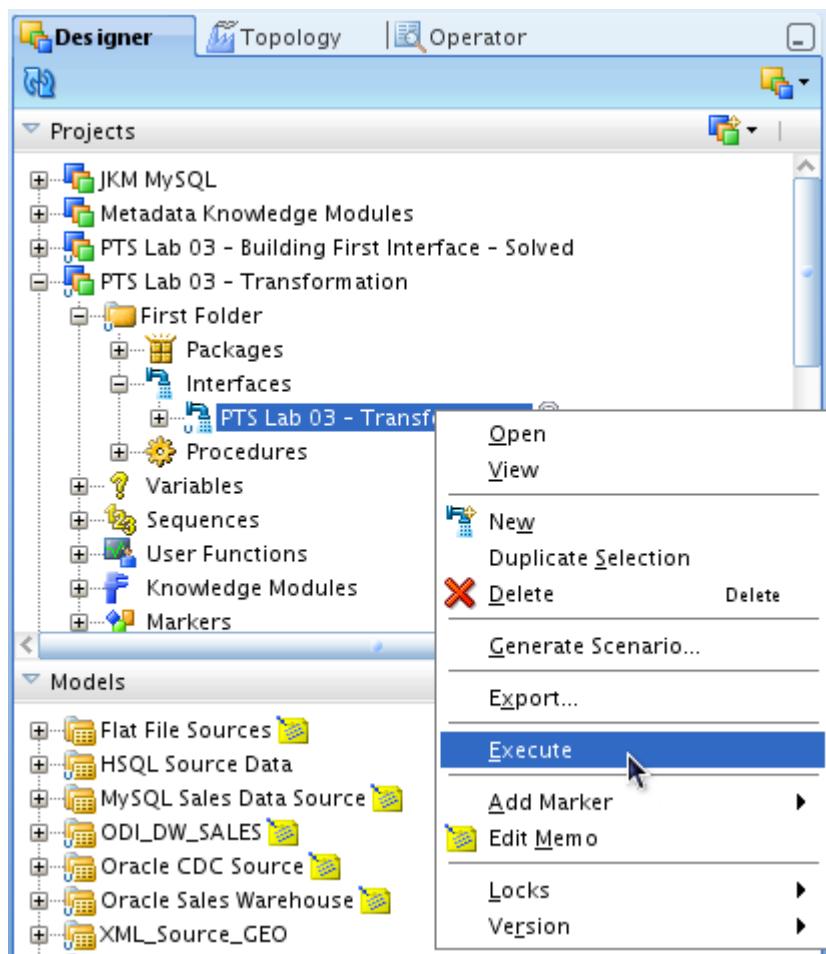
- f) Click the Flow tab. After the flow diagram initializes, you may wish to use the zoom out button to see all objects in the diagram. Click the box for Target + Staging Area labeled Target (ORACLE_ORCL_LOCAL), and then in the Target Properties panel, select the Distinct Rows check box. Set FLOW_CONTROL to false and DELETE_ALL to true. Click  button to minimize Messages log. Your screen looks as shown below.



- g) Click the Save button to save your interface. If the Locking Object window appears, select “Don’t show this window next time,” and click Yes.



- 2) Run this interface, and check the content of the TRG_CUSTOMER table.
- Expand the Interfaces node, right-click the newly created interface PTS Lab 03 – Simple Transformation, and then select Execute.



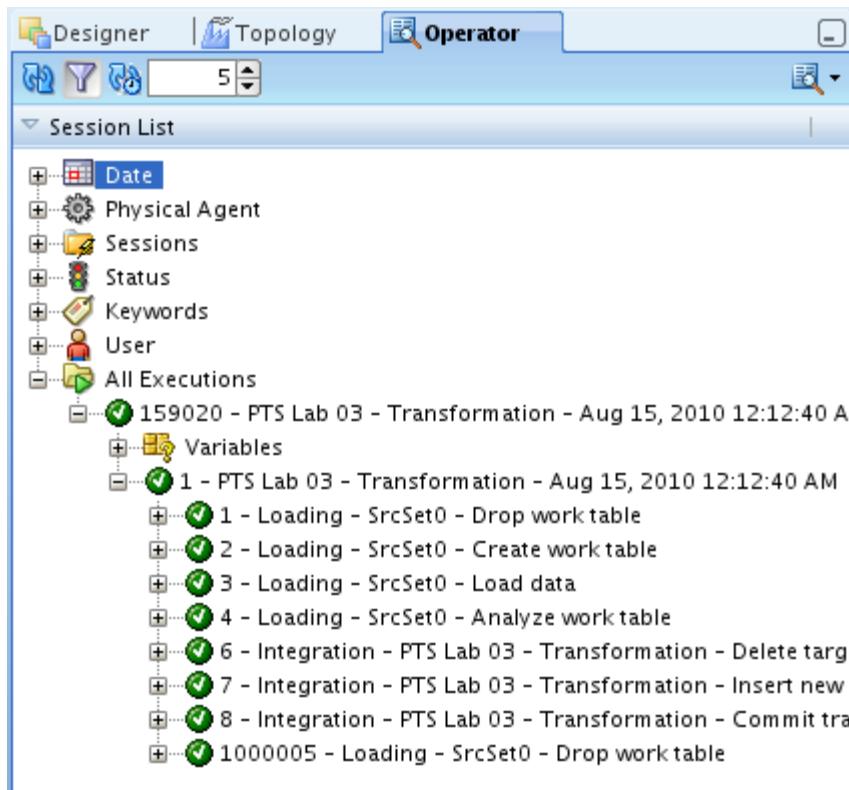
- b) Click OK in the Execution window that appears, and then click OK when the Session Started message appears.



- c) Click the Operator tab to open the ODI Operator. The Operator window appears.

Note: In Operator, you may need to click the Refresh button to view the new session.

- d) Expand the All Executions node. The session called "PTS Lab 01 - Transformation" should appear as complete. Expand this session in the tree view as shown below:



- e) In your interface window, click the Mapping tab. Select the TRG_CUSTOMER target datastore (click the name of the datastore), right-click and select Data.

Position	Indicators	Name	Mapping
1		*CUST_ID	SRC_CUSTOMER.CUSTID
2		DEAR	CASEWHEN SRC_CUSTOMER.DEAR = 0 THEN
3		CUST_NAME	INITCAP(SRC_CUSTOMER.FIRST_NAME ' '
4		ADDRESS	SRC_CUSTOMER.ADDRESS
5		*CITY_ID	SRC_CUSTOMER.CITY_ID
6		PHONE	SRC_CUSTOMER.PHONE
7		AGE	SRC_CUSTOMER.AGE
8		AGE_RANGE	
9		SALES_PERS	
10		CRE_DATE	SYSDATE
11		UPD_DATE	SYSDATE

- f) A window appears with the loaded data.

The screenshot shows a 'Data Editor' window with a title bar. Below the title bar is a toolbar with various icons for file operations like Open, Save, and Print. The main area contains a grid table with columns labeled CUST_ID, DEAR, CUST_NAME, ADDRESS, CITY_ID, and PHONE. The first row is highlighted in blue. Below the grid is a section titled 'Executed Query:' containing the SQL command: 'select * from CUST_DW_DEV.TRG_CUSTOMER'. At the bottom left, it says 'Record 1 of 50'. On the right side, there is a vertical scroll bar and a 'Close' button.

	CUST_ID	DEAR	CUST_NAME	ADDRESS	CITY_ID	PHONE
1	105	Mr	Harold Abel	58 Country Dr	13	(527) 987-8403
2	109	Mrs	Loralee Teall	55 Pittman Rd	43	144761860
3	112	Mr	Emilio Coward	63 Worrell Lo	7	(548) 884-7652
4	115	Ms	Louetta McClean	29 Schmidt Ln	6	(132) 399-3443
5	119	Ms	Wanetta Kuzminski	43 Levee Rd	41	755499541
6	122	Ms	Marybelle Ravencraft	20 Kyle Bend Ln	73	863246440
7	128	Mrs	Nichelle Ahart	87 Springfield Av	4	(333) 446-7238

Executed Query:
select * from CUST_DW_DEV.TRG_CUSTOMER

Record 1 of 50

Close

- g) Verify your data and close this window. Click Save button and then close your interface window. If the Unlocking Object window appears, select “Don’t show this window next time” check box and click Yes. Close “PTS Lab 03 – Building First Interface” tab.



Lab 4-1: Creating Master and Work Repository

Introduction

The central component of the architecture is the Oracle Data Integrator Repository. It stores configuration information about the IT infrastructure, metadata of all applications, projects, scenarios, and the execution logs. Many instances of the repository can coexist in the IT infrastructure. The architecture of the repository is designed to allow several separated environments that exchange metadata and scenarios (for example: Development, Test, Maintenance and Production environments). The repository also acts as a version control system where objects are archived and assigned a version number. The Oracle Data Integrator Repository can be installed on an OLTP relational database.

The Oracle Data Integrator Repository is composed of a master repository and several Work Repositories. Several Work repositories can be designated with several Master repositories, if necessary. However, a Work repository can be linked with only one Master repository for version management.

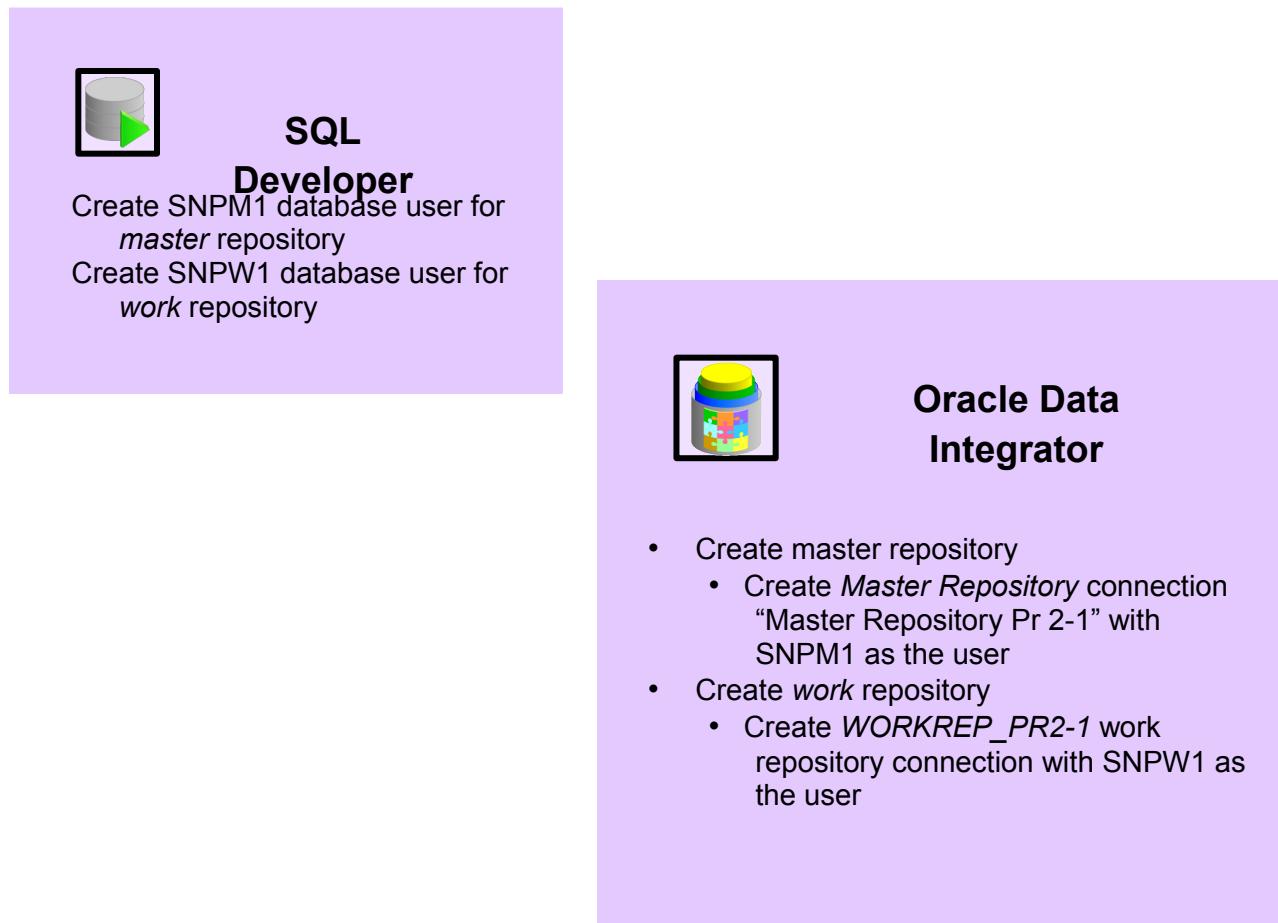
- **Master Repository:** It is the data structure containing information about the topology of the company's IT resources, security, and version management of projects and data models. This repository is stored on a relational database accessible in client/server mode from the different ODI modules. In general, you need only one Master repository.
- **Work Repository:** It is the data structure containing information about data models, projects, and their use. This repository is stored on a relational database accessible in client/server mode from the different ODI modules.

When the Work Repository contains only the execution information (typically for production purposes), it is then called an Execution Repository.

Process Overview

The first steps to setting up Oracle Data Integrator are to create the Master repository, connect to the Master repository, create the Work repository, and then connect to the Work repository.

In this practice, students create and connect to the ODI Master Repository and the ODI Work Repository. Students also create multiple Work repositories that will be used in subsequent practices.



Scenario

You work as a database administrator for FN Enterprise. In FN Enterprise, you are responsible for performing database management and integration tasks on various resources within the organization. In particular, you are responsible for data loading, transformation, and validation. To set your ODI environment you need to set up the security with your OD and create Master and Work repositories.

Instructions

- 1) The RDBMS used in this practice is Oracle DB. To connect to your RDBMS, Create the RDBMS schema/user (Oracle XE) for the Master repository. Perform the following steps:

Step	Screen/Page Description	Choices or Values
a.	Linux	Connect to RDBMS by selecting Applications >Accessories > Terminal> Type sql
b.	SQL Prompt	The RDBMS user/schema can be created by executing the following SQL command: create user snpm1 identified by welcome1 default tablespace users temporary tablespace temp; Press Enter button
c.	SQL Prompt	Grant connect privileges to the newly created user by executing the following SQL command: grant connect, resource to snpm1; Press Enter button

a)

```

File Edit View Terminal Tabs Help
[oracle@pts ~]$ sql

SQL*Plus: Release 11.2.0.1.0 Production on Thu Jul 22 20:06:23 2010

Copyright (c) 1982, 2009, Oracle. All rights reserved.

Connected to:
Oracle Database 11g Enterprise Edition Release 11.2.0.1.0 - Production
With the Partitioning, Oracle Label Security, OLAP, Data Mining,
Oracle Database Vault and Real Application Testing options

```

b)

```

SQL> create user snpml identified by welcome1
default tablespace users temporary tablespace temp;
2
User created.

```

c)

```

SQL> grant connect, resource to snpml;

Grant succeeded.

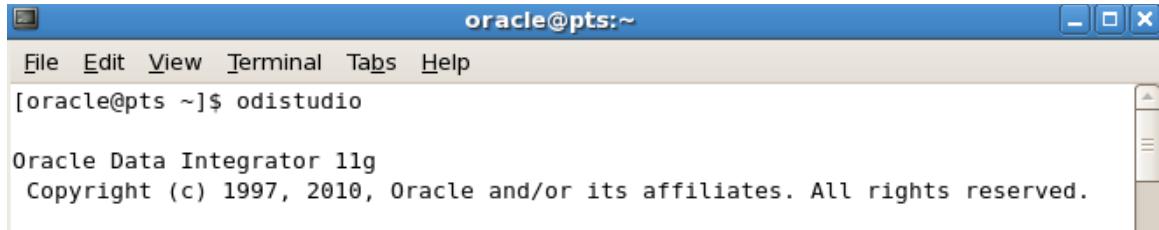
```

2) Create the ODI Master repository:

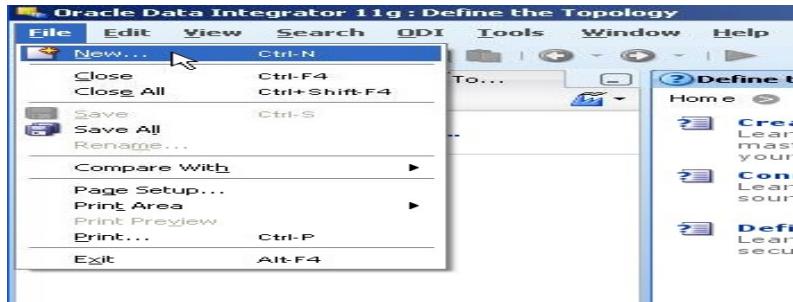
Step	Screen/Page Description	Choices or Values
a.	Linux	Start ODI Studio by selecting Applications > Accessories > Terminal > Type odistudio
b.	Oracle Data Integrator	<p>Open the New Gallery by choosing File > New.</p> <p>In the New Gallery, in the Categories tree, select ODI.</p> <p>Select from the Items list the Master Repository Creation Wizard. Click OK.</p> <p>The Master Repository Creation Wizard appears.</p>
c.	Master Repository Creation Wizard	In the Master Repository Creation Wizard, select Oracle JDBC Driver and click OK. Edit the JDBC

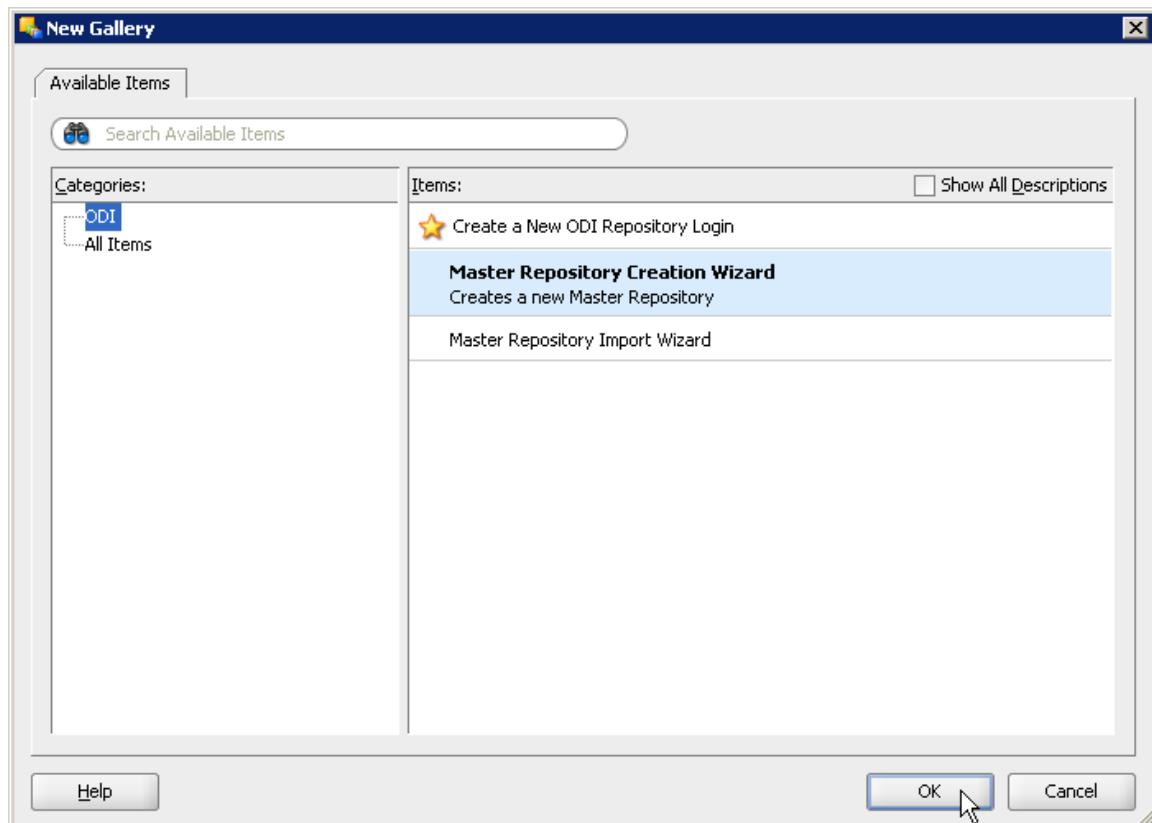
Step	Screen/Page Description	Choices or Values
		<p>URL to read: jdbc:oracle:thin:@pts.us.oracle.com:1521:XE</p> <p>Enter the User as snpm1 and the Password as welcome1. Click the Test Connection button and verify successful connection. Click OK. Click Next on the Master Repository Creation Wizard screen.</p>
d.	Master Repository Creation Wizard	<p>In the Authentication window, enter Supervisor Password as welcome1. Enter welcome1 again to confirm the password. Click Next.</p>
e.	Password Storage	<p>In the Password Storage window, select Internal password Storage, and then click Finish. When Master Repository is successfully created, you will see the Oracle Data Integrator Information message. Click OK. The ODI Master repository is now created.</p>
f.	Message	<p>Verify that ODI Master Repository created successfully, and then click OK.</p>

a)

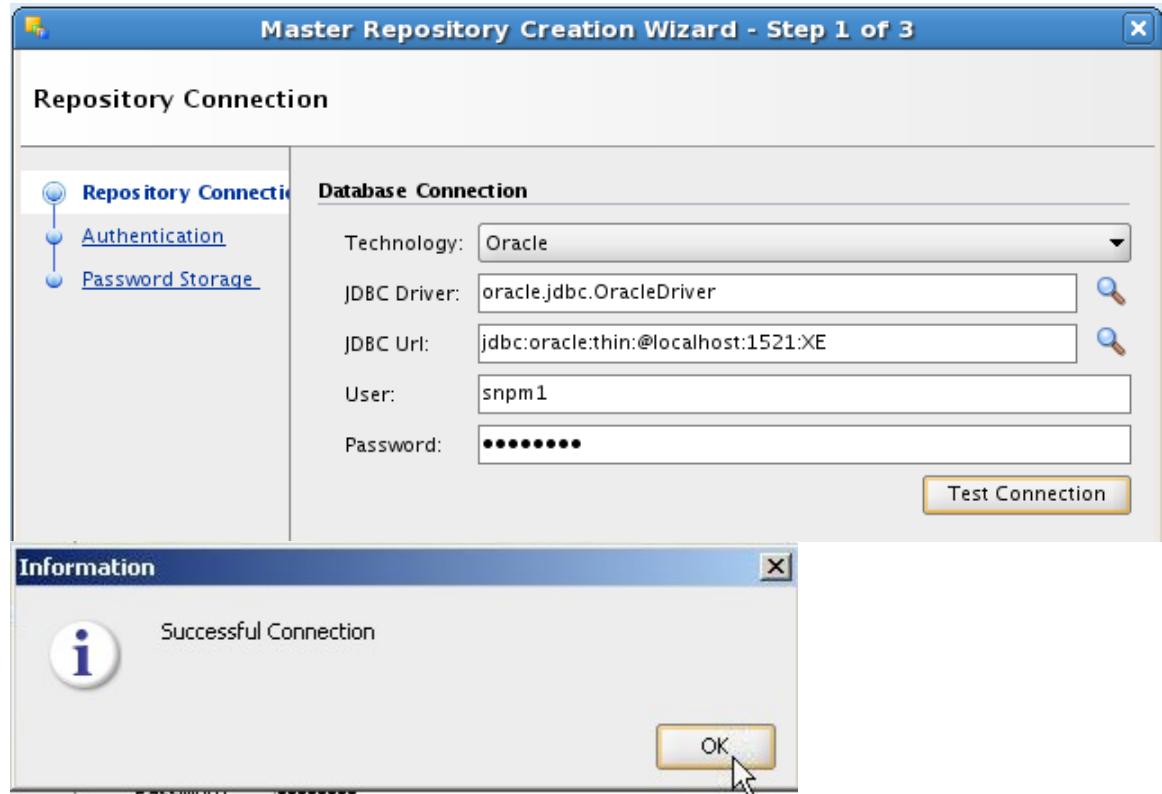


b)





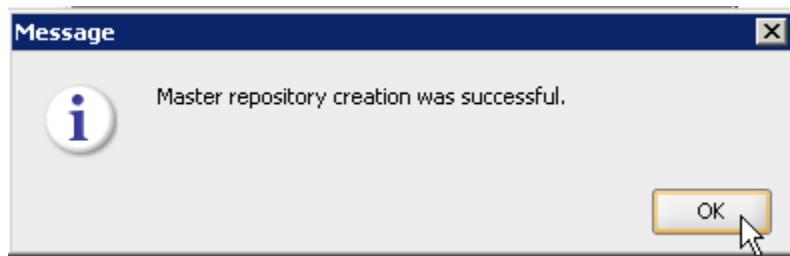
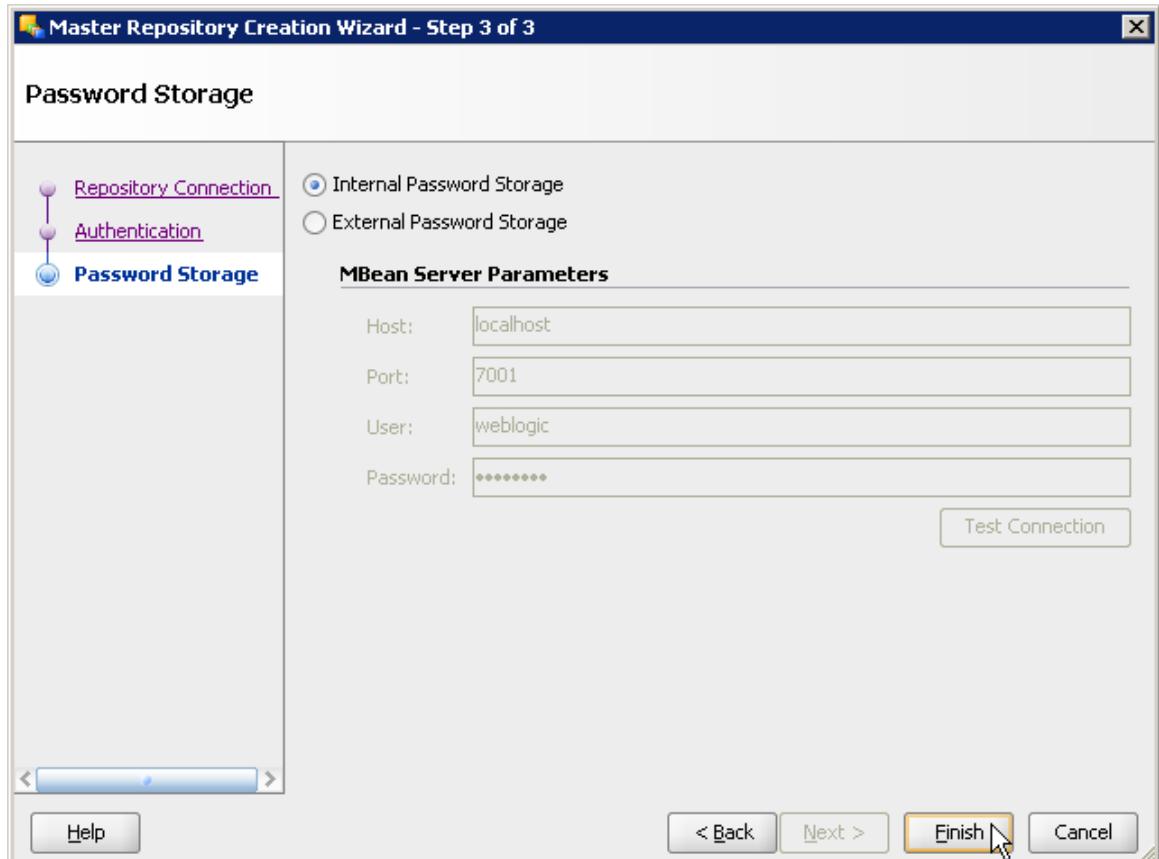
c)



d)



e)



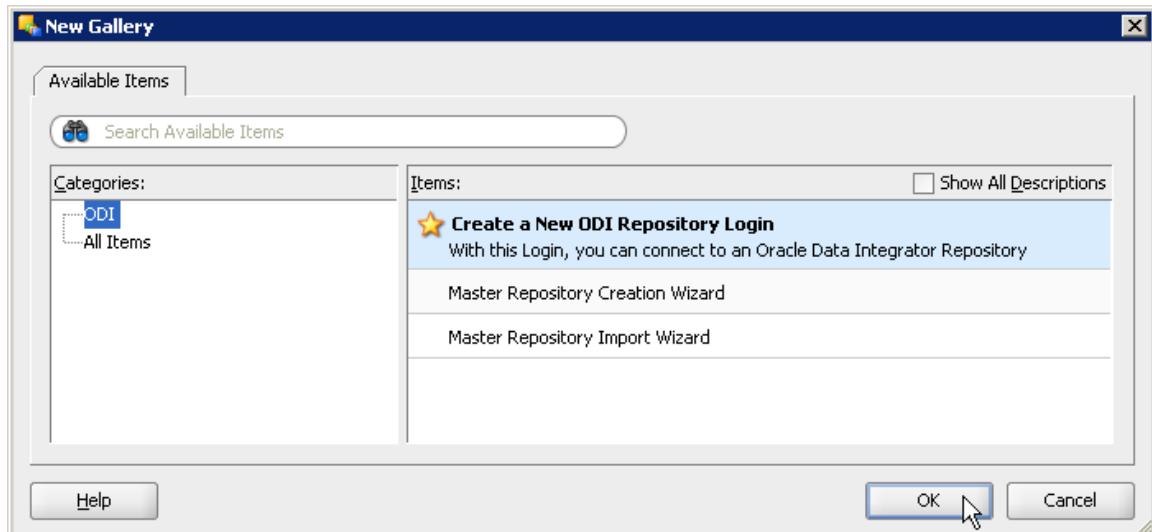
3) Connect to the ODI Master repository:

Step	Screen/Page Description	Choices or Values
a.	Oracle Data Integrator	Open the New Gallery by choosing File > New. In the New Gallery, in the Categories tree, select ODI. From the Items list select Create a New ODI Repository login.

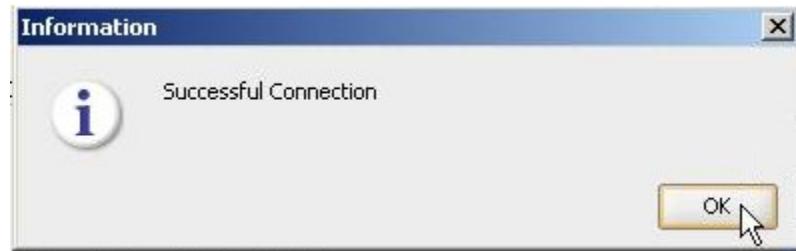
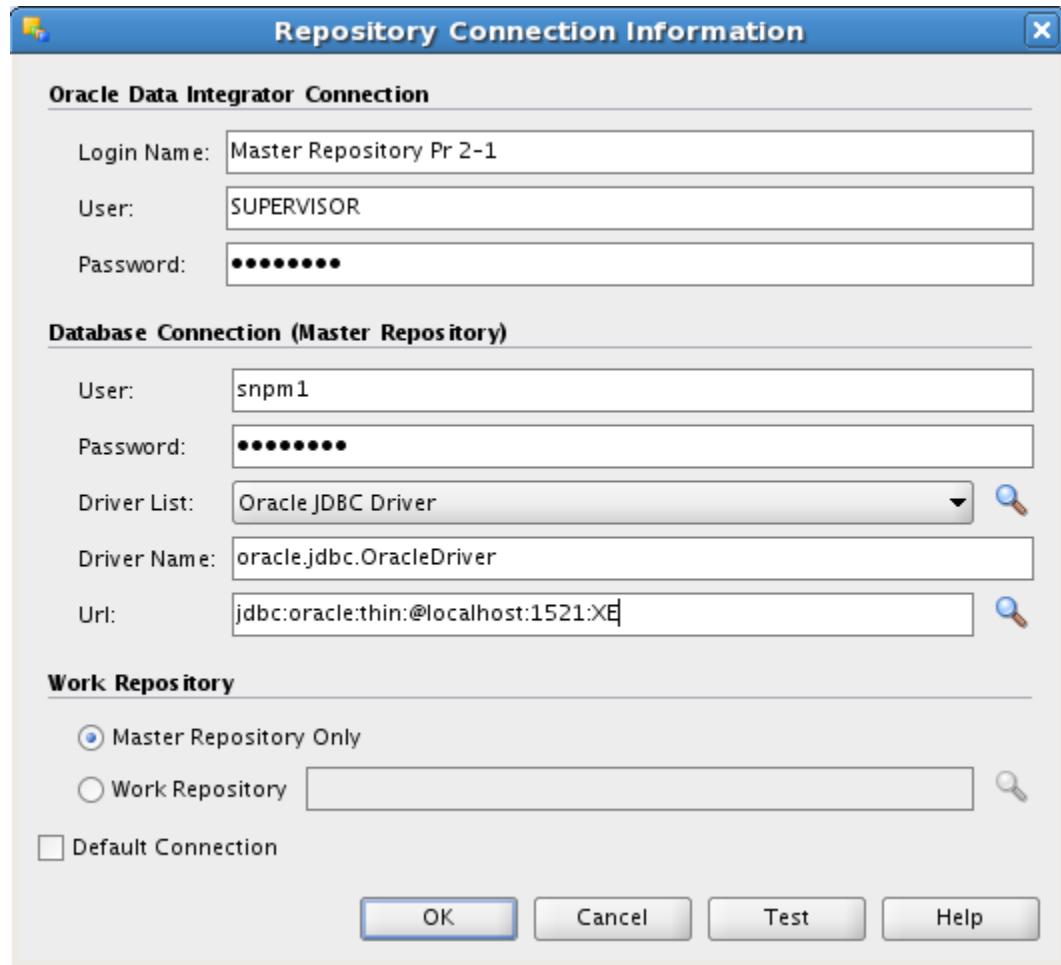
Step	Screen/Page Description	Choices or Values																								
b.	Repository Connections	<p>Configure Repository Connections with the parameters from the tables provided below. To enter the JDBC URL, click the button next to JDBC URL field and select <code>jdbc:oracle:thin:@<host>:<port>:<sid></code> as shown in the screenshot, then edit the URL. Select Master Repository only button. Click Test button. Verify successful connection and click OK. Click OK to save the connection.</p> <table border="1" data-bbox="600 614 1225 1036"> <tr> <td colspan="2" data-bbox="600 614 1225 677">Oracle Data Integrator Connection</td> </tr> <tr> <td data-bbox="600 688 861 751">Parameter</td><td data-bbox="861 688 1225 751">Value</td></tr> <tr> <td data-bbox="600 762 861 868">Login Name</td><td data-bbox="861 762 1225 868">Master Repository Pr 2-1</td></tr> <tr> <td data-bbox="600 878 861 941">User</td><td data-bbox="861 878 1225 941">SUPERVISOR</td></tr> <tr> <td data-bbox="600 952 861 1015">Password</td><td data-bbox="861 952 1225 1015">welcome1</td></tr> </table> <table border="1" data-bbox="600 1110 1426 1681"> <tr> <td colspan="2" data-bbox="600 1110 1426 1174">Database Connection (Master Repository)</td> </tr> <tr> <td data-bbox="600 1184 861 1248">Parameter</td><td data-bbox="861 1184 1426 1248">Value</td></tr> <tr> <td data-bbox="600 1258 861 1322">User</td><td data-bbox="861 1258 1426 1322">snpm1</td></tr> <tr> <td data-bbox="600 1332 861 1396">Password</td><td data-bbox="861 1332 1426 1396">welcome1</td></tr> <tr> <td data-bbox="600 1406 861 1469">Driver List</td><td data-bbox="861 1406 1426 1469">Oracle JDBC Driver</td></tr> <tr> <td data-bbox="600 1480 861 1586">Driver Name</td><td data-bbox="861 1480 1426 1586">oracle.jdbc.OracleDriver</td></tr> <tr> <td data-bbox="600 1596 861 1660">Url</td><td data-bbox="861 1596 1426 1660"><code>jdbc:oracle:thin:@pts.us.oracle.com:1521:XE</code></td></tr> </table> <p>Note: ODI user name (SUPERVISOR) is case-sensitive.</p>	Oracle Data Integrator Connection		Parameter	Value	Login Name	Master Repository Pr 2-1	User	SUPERVISOR	Password	welcome1	Database Connection (Master Repository)		Parameter	Value	User	snpm1	Password	welcome1	Driver List	Oracle JDBC Driver	Driver Name	oracle.jdbc.OracleDriver	Url	<code>jdbc:oracle:thin:@pts.us.oracle.com:1521:XE</code>
Oracle Data Integrator Connection																										
Parameter	Value																									
Login Name	Master Repository Pr 2-1																									
User	SUPERVISOR																									
Password	welcome1																									
Database Connection (Master Repository)																										
Parameter	Value																									
User	snpm1																									
Password	welcome1																									
Driver List	Oracle JDBC Driver																									
Driver Name	oracle.jdbc.OracleDriver																									
Url	<code>jdbc:oracle:thin:@pts.us.oracle.com:1521:XE</code>																									
c.	Oracle data	Click Connect to Repository. Select the newly created																								

Step	Screen/Page Description	Choices or Values
	Integrator	repository connection Master Repository Pr 2-1 from the drop-down list. Click OK. The ODI Topology Manager starts. You are now successfully logged in to the ODI Topology Manager.
d.	Import Service Description	Click Topology tab. Click the Repositories tab in the left panel of the Topology Manager. Verify that your newly created Master repository is in the Repositories window.

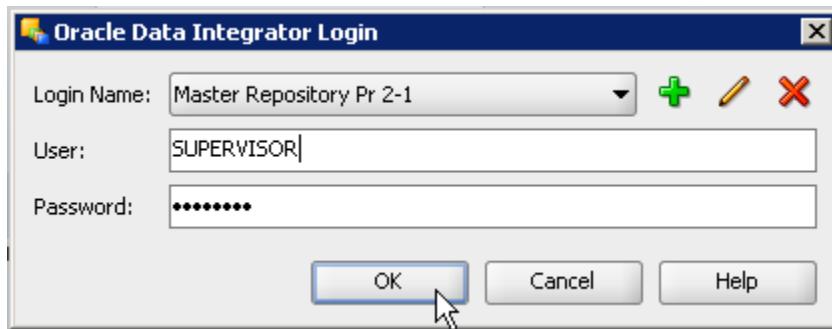
a)



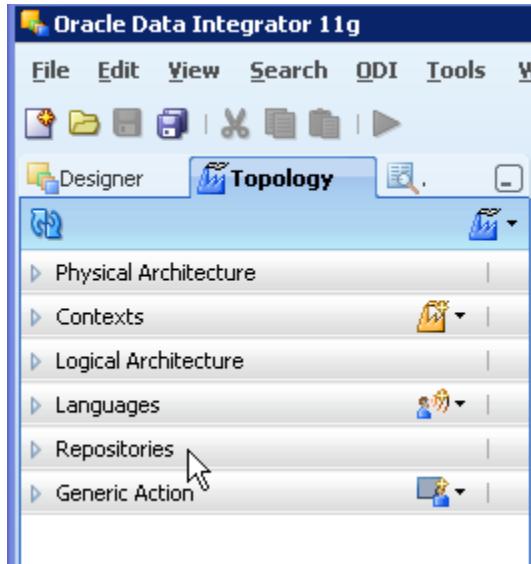
b)

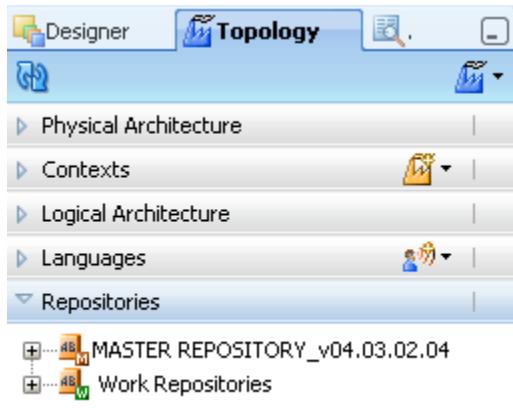


c)



d)





- 4) Create the RDBMS schema/user for the Work repository.

Step	Screen/Page Description	Choices or Values
a.	Oracle SQL Prompt	If not opened, open the SQL Prompt. Create the schema by executing the following SQL command: create user snpw1 identified by welcome1 default tablespace users temporary tablespace temp; Press Enter
b.	Oracle SQL Prompt	Grant connect privileges to the newly created user by executing the following SQL command: grant connect, resource to snpw1;
c.	Oracle SQL Prompt	Run the following command to verify that user snpw1 was successfully created and shown among other users in the list: select * from all_users;

a)

```
SQL> create user.snpw1 identified by welcome1  
default tablespace users temporary tablespace temp;  
2  
User created.
```

b)

```
SQL> grant connect, resource to.snpw1;  
Grant succeeded.
```

c)

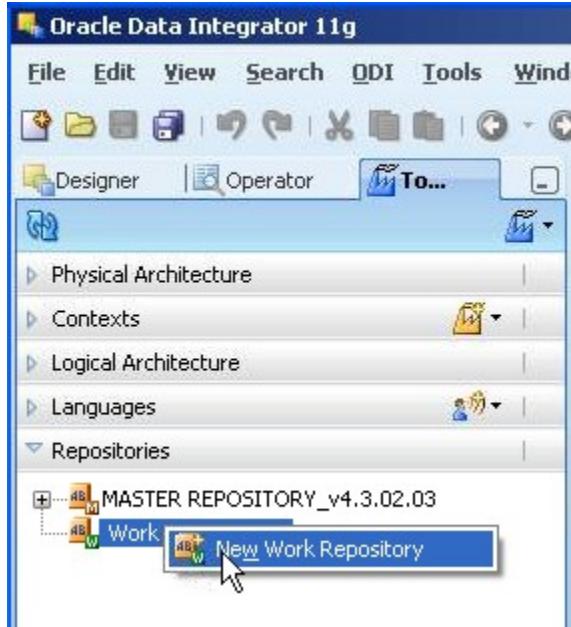
```
SQL> select * from all_users;  
  
USERNAME          USER_ID CREATED  
-----  
SNPW1            93 22-JUL-10  
SNPM1            92 22-JUL-10
```

5) Create the ODI Work repository:

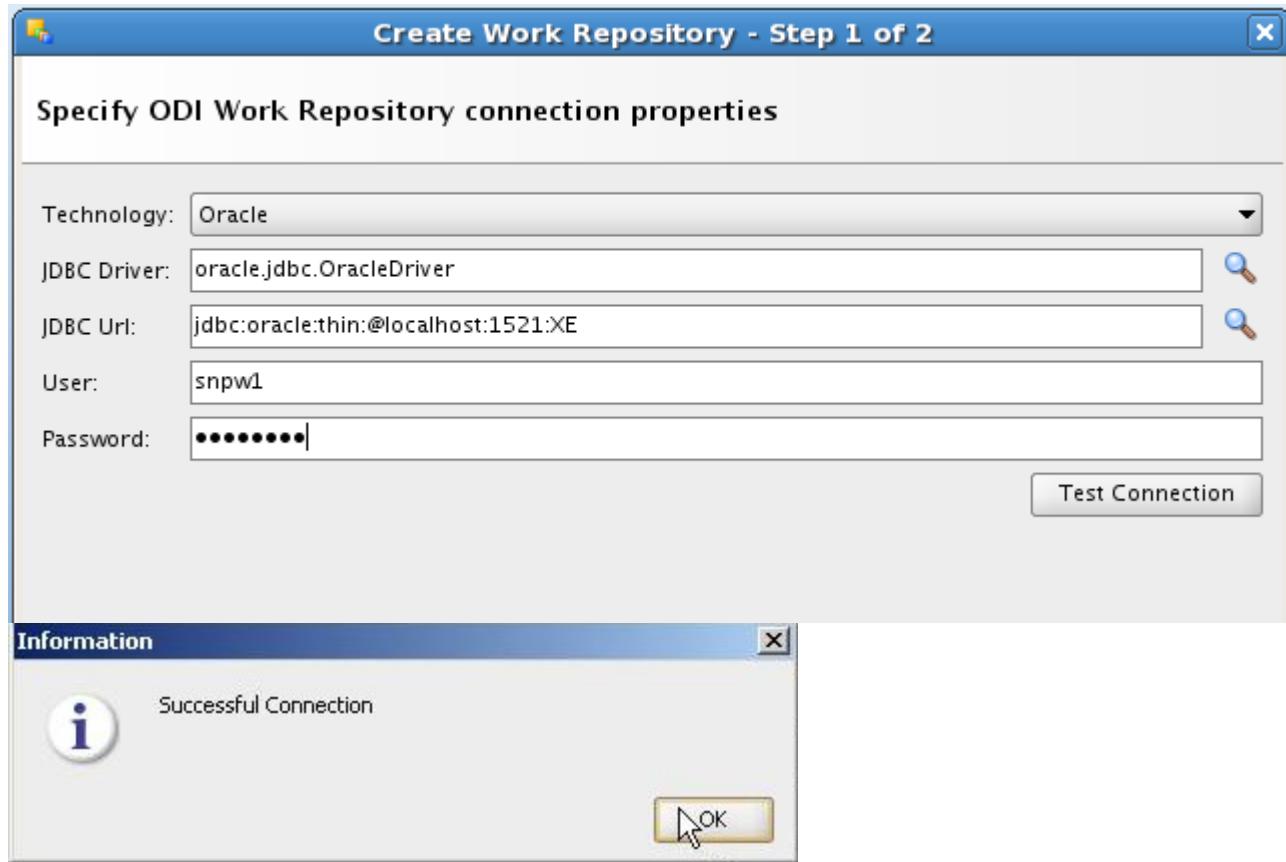
Step	Screen/Page Description	Choices or Values												
a.	Oracle Data Integrator	In ODI, click the Topology Navigator tab and then click to the Repositories panel. Right-click the Work Repositories node and select New Work Repository. The Create Work Repository Wizard opens.												
b.	Specify ODI Work Repository Connection Properties	In the screen that follows, enter the parameters shown in the following table. Click the Test button. Verify successful connection and click OK. Click Next. <table border="1"><thead><tr><th>Parameter</th><th>Value</th></tr></thead><tbody><tr><td>Technology</td><td>Oracle</td></tr><tr><td>Driver Name</td><td>oracle.jdbc.OracleDriver</td></tr><tr><td>JDBC Url</td><td>jdbc:oracle:thin:@pts.us.oracle.com:1521:XE</td></tr><tr><td>User</td><td>snpw1</td></tr><tr><td>Password</td><td>welcome1</td></tr></tbody></table>	Parameter	Value	Technology	Oracle	Driver Name	oracle.jdbc.OracleDriver	JDBC Url	jdbc:oracle:thin:@pts.us.oracle.com:1521:XE	User	snpw1	Password	welcome1
Parameter	Value													
Technology	Oracle													
Driver Name	oracle.jdbc.OracleDriver													
JDBC Url	jdbc:oracle:thin:@pts.us.oracle.com:1521:XE													
User	snpw1													
Password	welcome1													
c.	Specify	In the Specify Work Repository properties set the ID to												

Step	Screen/Page Description	Choices or Values
	Work Repository properties	<p>"1". Set the Name to WORKREP_PR2-1. Enter Password as welcome1. For Work Repository Type, leave Development. Click Finish. Verify that the newly created Work repository is now in the work repositories tree view.</p> <p>Note: Development type of repository allows management of design-time objects such as data models and projects (including interfaces, procedures, etc). A development repository includes also the run-time objects (scenarios and sessions). This type of repository is suitable for development environments.</p>
d.	Create Work Repository Login	<p>In the Create Work Repository Login window, click Yes. Enter the Login name: WORKREP_PR2-1 as shown on the screenshot. Click OK.</p>
e.	Oracle Data Integrator	<p>Verify that Work repository WORKREP_PR2-1 created and shown under the Work Repositories node.</p>

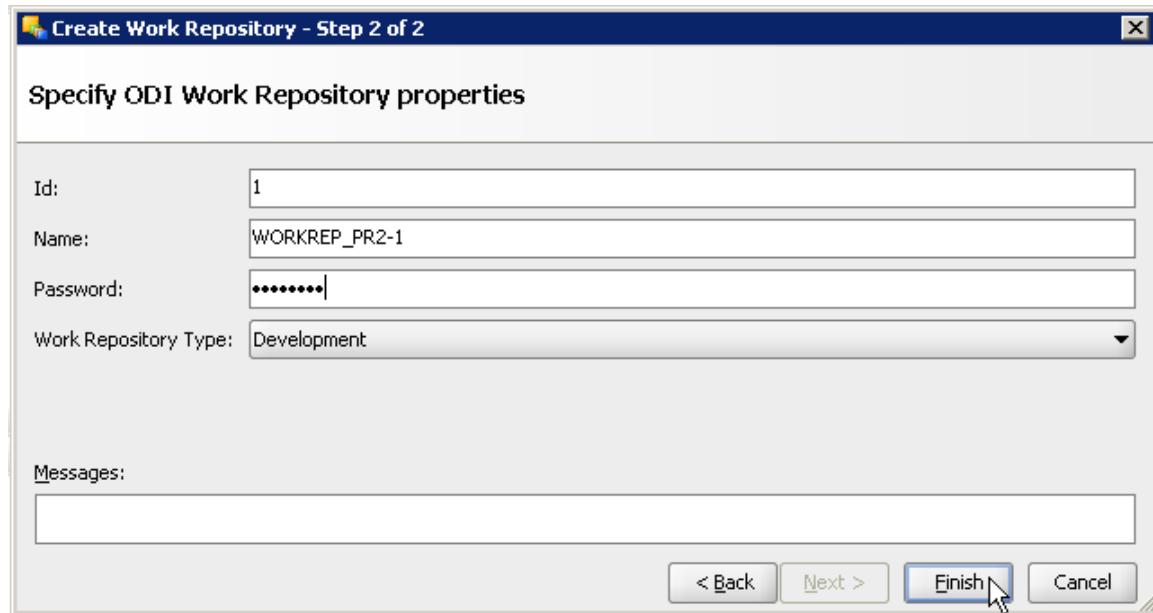
a)



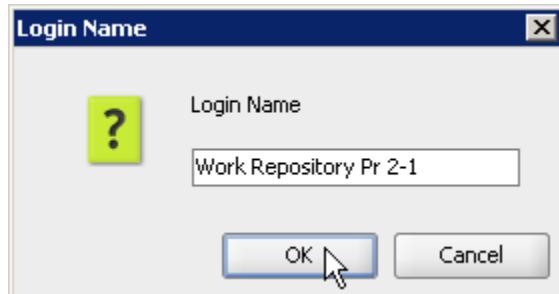
b)



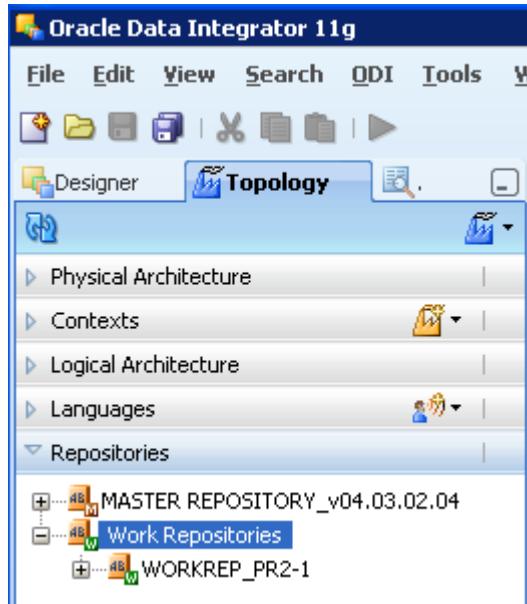
c)



d)



e)

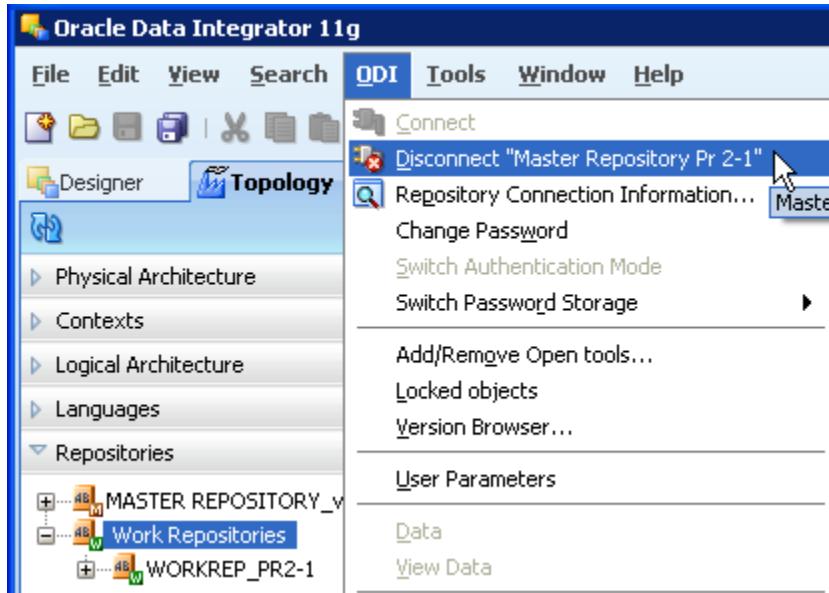


6) Connect to the Work repository:

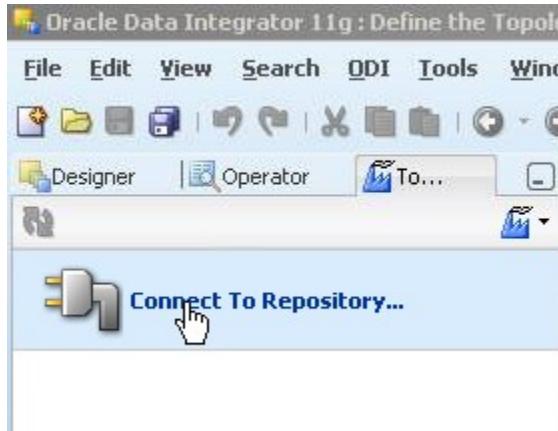
Step	Screen/Page Description	Choices or Values
a.	Oracle Data Integrator	Click ODI menu and select Disconnect "Master Repository Pr 2-1".
b.	Oracle Data	Click "Connect To Repository..."

Step	Screen/Page Description	Choices or Values
	Integrator	
c.	Oracle Data Integrator Login	<p>Select “Work Repository Pr 2-1” from the Login Name drop-down list. Enter Password: welcome1. Click OK. Click the Designer tab. The following ODI Designer screen appears.</p> <p>You have now successfully created and connected to the ODI Work repository.</p>

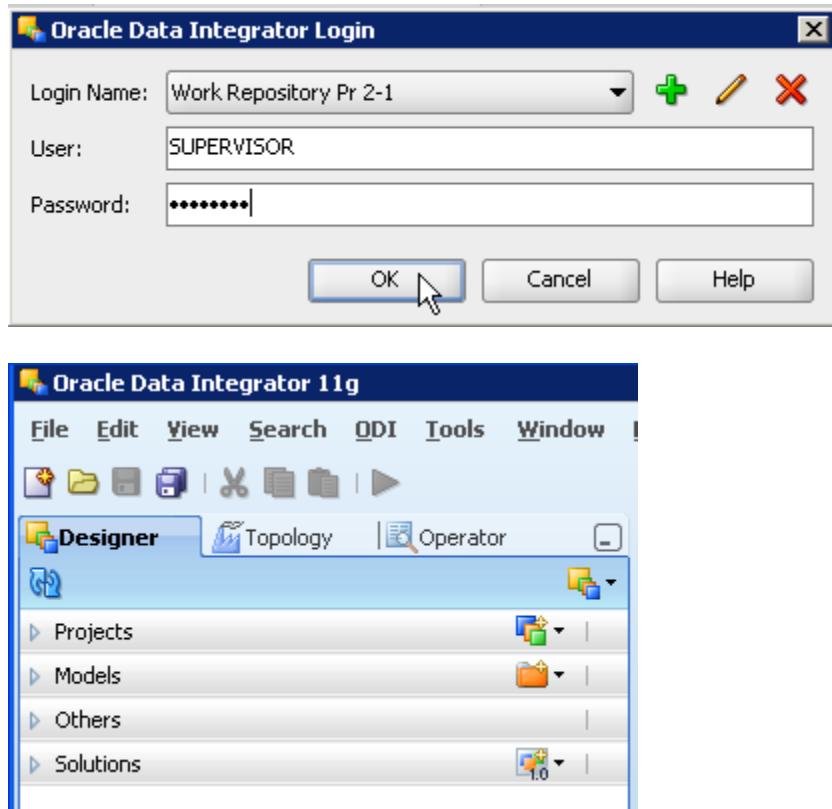
a)



b)



c)



Lab 4-2: Creating and Managing Topology

Introduction

The Oracle Data Integrator Topology is the physical and logical representation of the Oracle Data Integrator architecture and components.

Physical Architecture: The physical architecture defines the different elements of the information system, as well as their characteristics taken into account by Oracle Data Integrator. Each type of database (Oracle, DB2, etc.), file format (XML, Flat File), or application software is represented in Oracle Data Integrator by a technology.

The physical components that store and expose structured data are defined as *data servers*. A data server is always linked to a single technology. A data server stores information according to a specific technical logic which is declared into *physical schemas* attached to this data server.

Contexts: Contexts bring together components of the physical architecture (the real Architecture) of the information system with components of the Oracle Data Integrator logical architecture (the Architecture on which the user works).

For example, contexts may correspond to different execution environments (*Development, Test and Production*) or different execution locations (*Boston Site, New-York Site*, and so forth.) where similar physical resources exist.

Logical Architecture: The logical architecture allows a user to identify as a single Logical Schema a group of similar physical schemas - that is containing datastores that are structurally identical - but located in different physical locations. Logical Schemas, like their physical counterpart, are attached to a technology.

All the components developed in Oracle Data Integrator are designed on top of the logical architecture. For example, a data model is always attached to logical schema, and data flows are defined with this model. By specifying a context at run-time, the model's logical schema resolves to a single physical schema, and the data contained in this schema in the data server can be accessed by the integration processes.

Agents: Oracle Data Integrator run-time Agents orchestrate the execution of jobs. The agent executes jobs on demand and to start the execution of scenarios according to a schedule defined in Oracle Data Integrator.

Languages: Languages defines the languages and language elements available when editing expressions at design-time.

Prerequisites

Completion of **Lab 4-1: Creating Master and Work Repository**.

Process Overview

You use the **Topology Navigator** to define the topology of your information system to ODI so that it can be accessed by other ODI modules. In addition, Topology Navigator allows you to manage the repositories. The Topology Navigator stores this information in a Master repository. This information can be used by all the other modules.

The following steps are a guideline to create the topology. You can always modify the topology after an initial setting:

1. Create the contexts corresponding to your different environments.
2. Create the data servers corresponding to the servers used by Oracle Data Integrator.
3. For each data server, create the physical schemas corresponding to the schemas containing data to be integrated with Oracle Data Integrator.
4. Create logical schemas and associate them with physical schemas in the contexts.
5. Create the physical agents corresponding to the standalone or Java EE agents that are installed in your information systems.
6. Create logical agents and associate them with physical agents in the contexts.

Note: Steps 5 and 6 creating agents are the topic of the next lab.

Scenario

You created Master and Work repositories. To continue setting up your ODI infrastructure, you need to create contexts, a data sever, and physical and logical schemas.

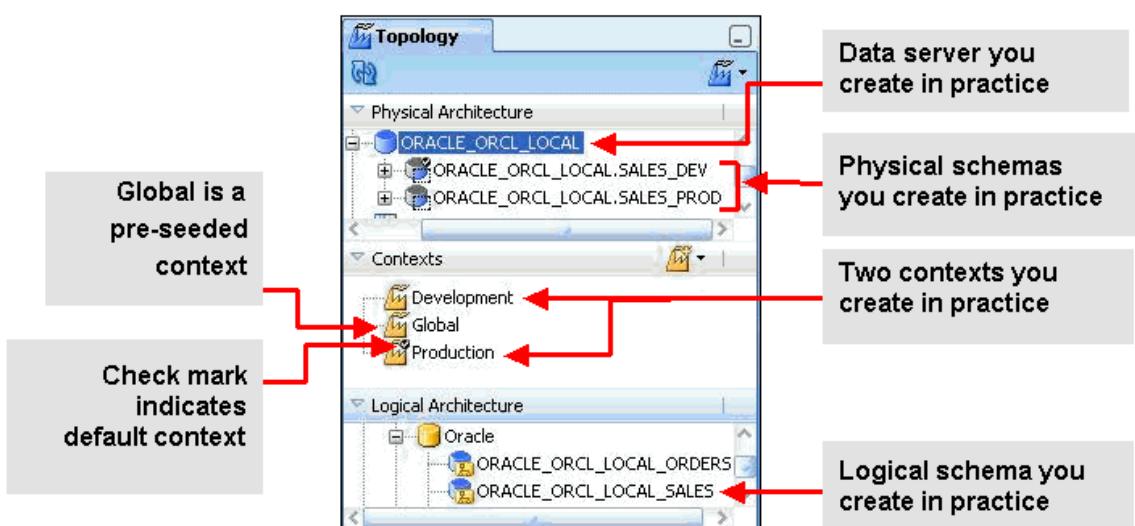
Before you begin working on your ODI projects, you need to describe your ODI infrastructure in the topology. As a starting point of this practice, you will use the Lab 4-2: Creating and Managing Topology

environment provided with ODI installation. The infrastructure includes several servers, and you need to define the following new data server and its attached physical schemas:

Data Server: ORACLE_ORCL_LOCAL

Schemas available in this instance:	
Schema	Description
CUST_DW_DEV	Schema storing the target tables for development purposes
CUST_DW_PROD	Schema storing the production target tables
ODI_TMP	Schema used to store ODI temporary objects

Both the SALES_DEV and SALES_PROD physical schemas (corresponding to CUST_DW_DEV and CUST_DW_PROD data schemas) contain identical table structures and correspond to the same logical schema called ORACLE_ORCL_LOCAL_SALES. The mapping for this logical schema depends on the context.



1. Define Production context.

2. Define Development context (a 3rd context, Global, is pre-seeded).
3. Define ORACLE_ORCL_LOCAL data server.
4. Define ODI physical schemas for data server: SALES_DEV, SALES_PROD.
5. Define ORACLE_ORCL_LOCAL_SALES ODI logical schema.
7. Map logical schema to the two physical schemas, in terms of the three contexts.

MYSQL Data Server: Data Server: MYSQL_PTSDB_LOCAL

<i>Schemas available in this instance:</i>	
DB	Description
SALES_DEV (TRAINING_SRC)	Schema storing the target tables for development purposes
TEST	Schema used to store ODI temporary objects

1. Define Production context.
2. Define Development context (a 3rd context, Global, is pre-seeded).
3. Define MYSQL_PTSDB_LOCAL data server.
4. Define ODI physical schemas for data server: PTSDB.
5. Define MYSQL_PTSDB_LOCAL_SALES logical schema.
7. Map logical schema to the physical schemas, in terms of the three contexts.

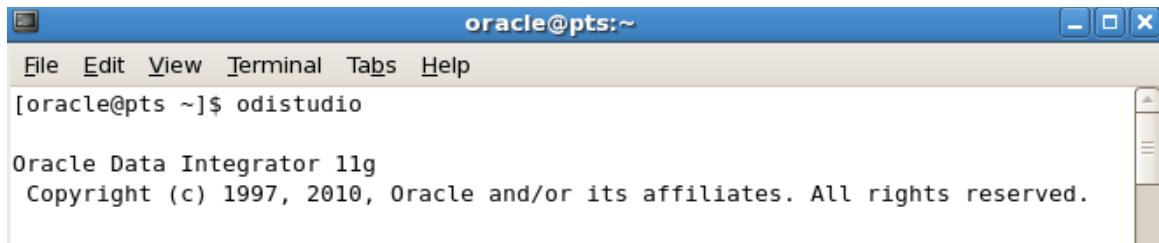
Note: JDBC driver for MYSQL is already installed for this lab. You have to download the latest JDBC driver from mysql website.

Instructions

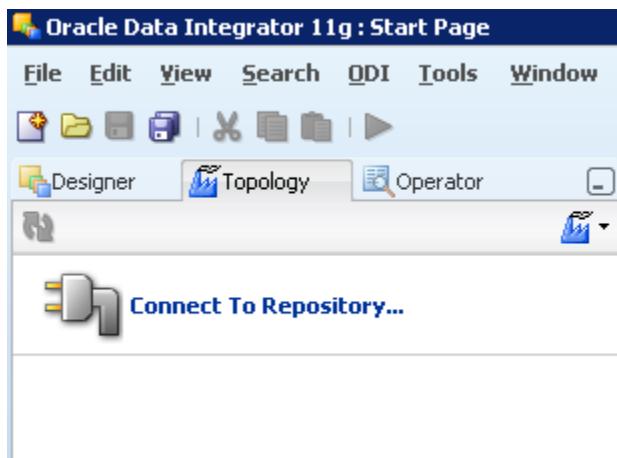
- 1) If not started, start Oracle Data Integrator and open **Master Repository Pr 2-1** connection.

Note: The username and password are case-sensitive.

- a) Start ODI Studio by selecting Applications >Accessories > Terminal> Type **odistudio**

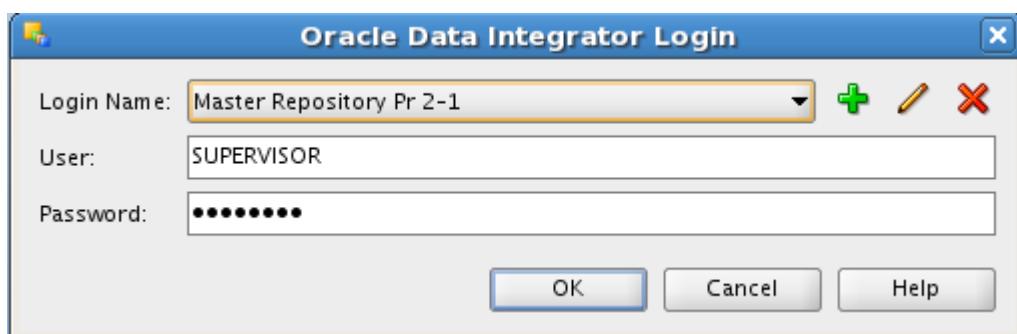


```
oracle@pts:~$ odistudio
[oracle@pts ~]$ Oracle Data Integrator 11g
Copyright (c) 1997, 2010, Oracle and/or its affiliates. All rights reserved.
```



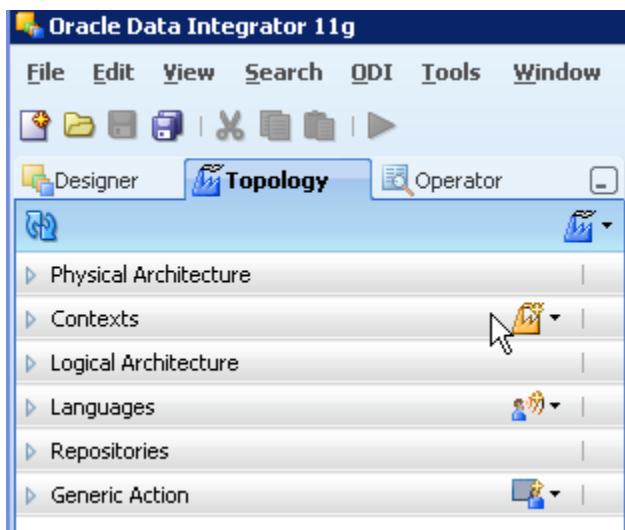
- b) Log in with the following details provided below. Click OK.

- Login name: Master Repository Pr 2-1
- User: SUPERVISOR
- Password: welcome1.



- 2) Create a new context:

- a) Click Topology navigator tab, and then click the Context tab.



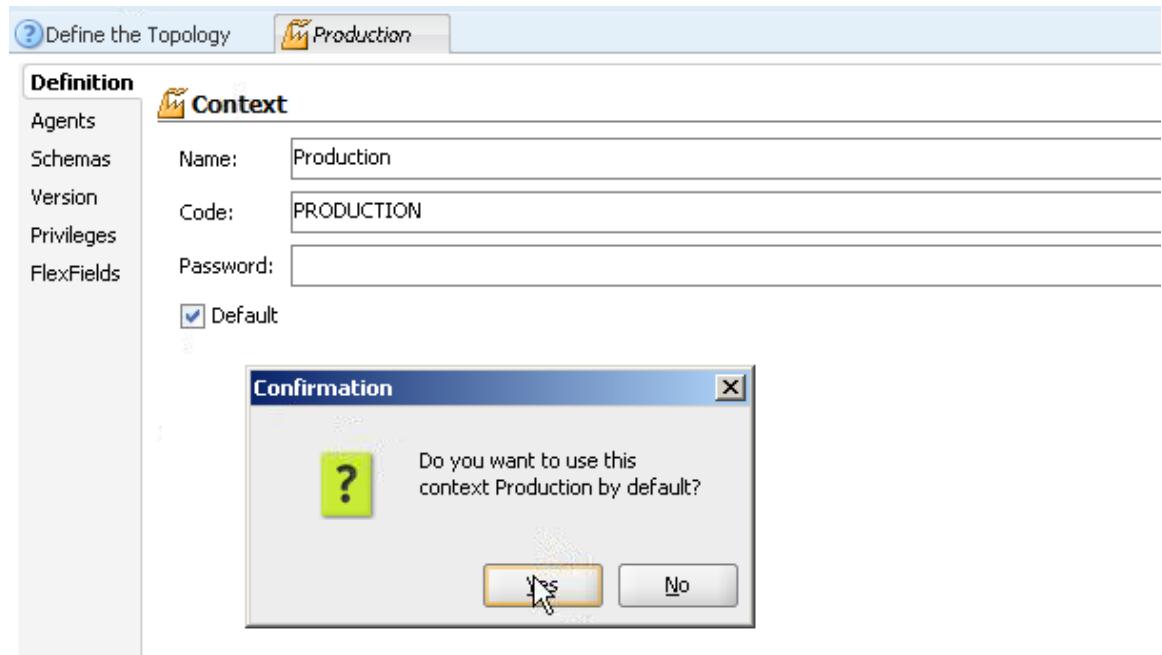
- b) Click the New Context icon on the toolbar, and then select New Context.



- c) Enter your context parameters:

- **Name:** Production
- **Code:** PRODUCTION
- **Password:** Leave this field empty.
- **Default:** Select this check box, and click Yes to confirm in the pop-up window that appears.

The context window should appear as follows:



d) Click Save button

7) Create a new context:

a) Repeat the operations to create another context:

- **Name:** Development
- **Code:** DEVELOPMENT
- **Password:** Leave this field empty.
- **Default:** Leave the check box deselected.



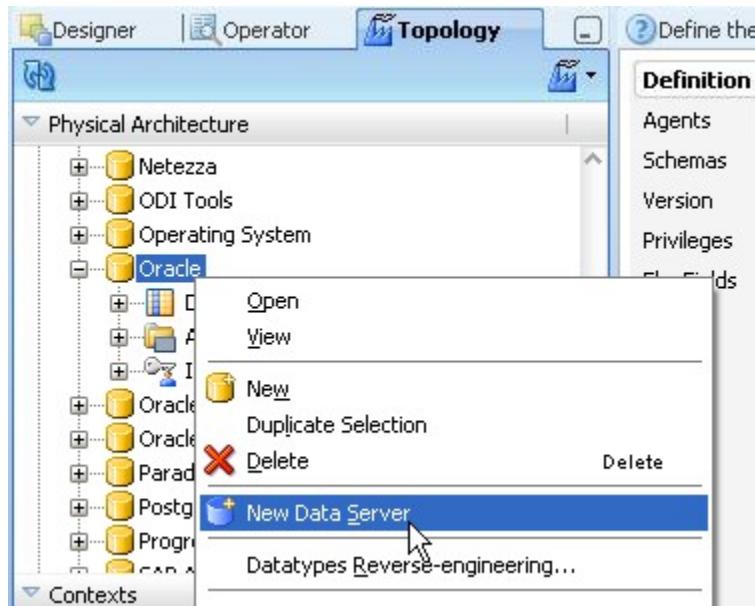
The contexts should appear as follows:



8) Create an ORACLE_ORCL_LOCAL data server:

a) Click the Physical Architecture tab.

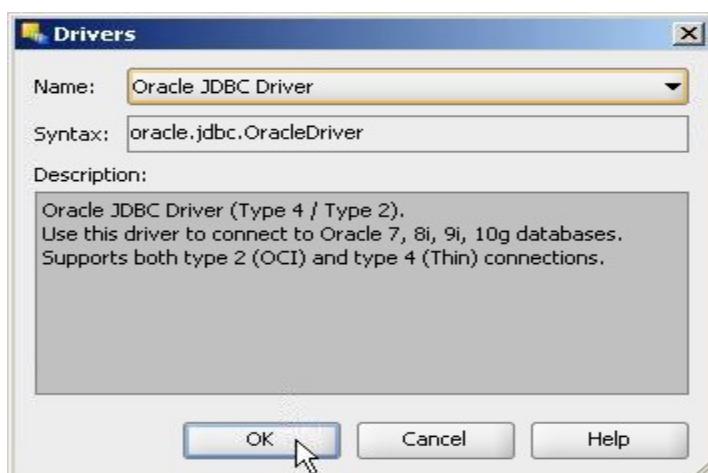
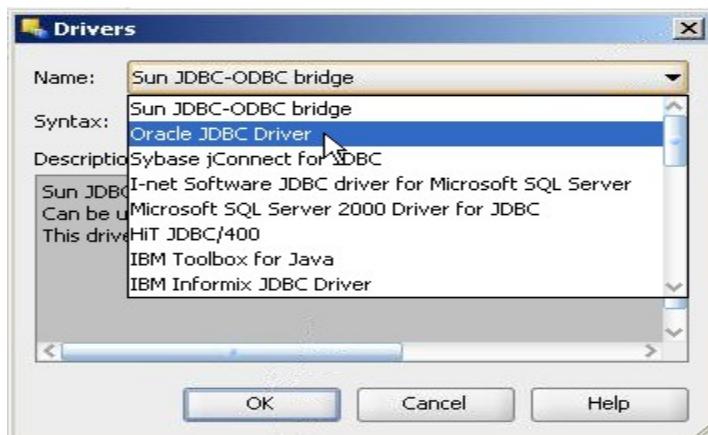
- b) Expand the Technologies node.
 c) Select the Oracle node, and then right-click and select New Data Server.



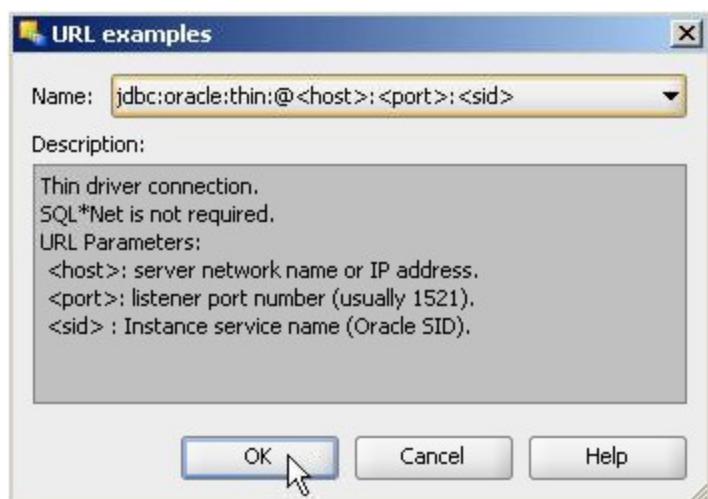
- d) Enter the following information on the Definition tab:
- **Name:** ORACLE_ORCL_LOCAL
 - **Instance Name:** ORCL
 - **User:** ODI
 - **Password:** ODI

Name:	ORACLE_ORCL_LOCAL
Technology:	Oracle
Instance / dblink (Data Server):	ORCL
User:	ODI
Password:	***
<input type="checkbox"/> JNDI Connection	
Array Fetch Size:	30
Batch Update Size:	30

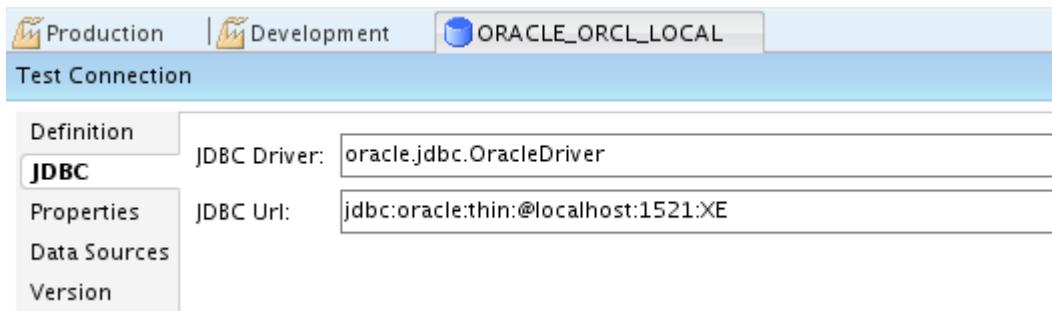
- e) Click the JDBC tab. Click the  button at the right of the JDBC Driver field. In the window that appears, select Oracle JDBC Driver, and then click OK.



- f) Click the  button at the right of the JDBC URL field. In the window that appears, select the first URL, and then click OK.

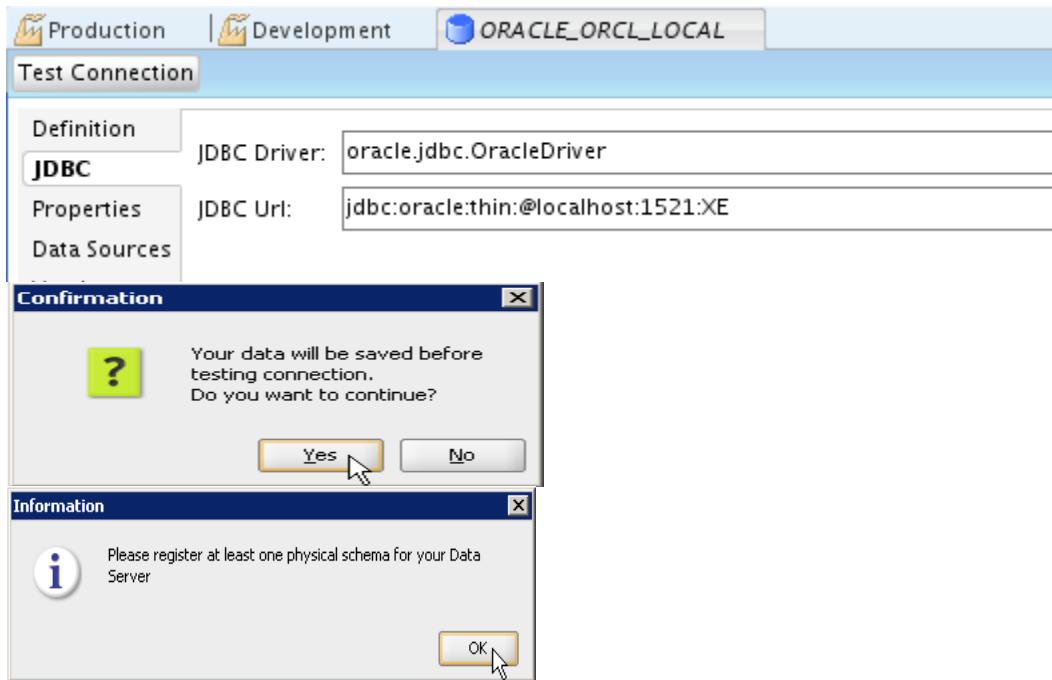


- g) Edit the JDBC URL to have the following:
 URL: `jdbc:oracle:thin:@pts.us.oracle.com:1521:XE`
- h) The JDBC tab should now appear as follows:

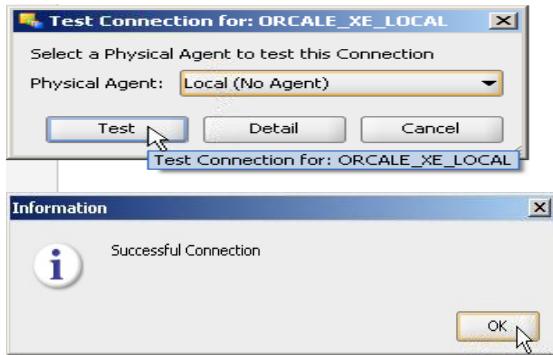


9) Test this data server:

- a) Click Test Connection button. Click Yes to confirm saving your data before testing connection. In the Information window, click OK.



In the dialog box that appears, click the Test button. Click OK.

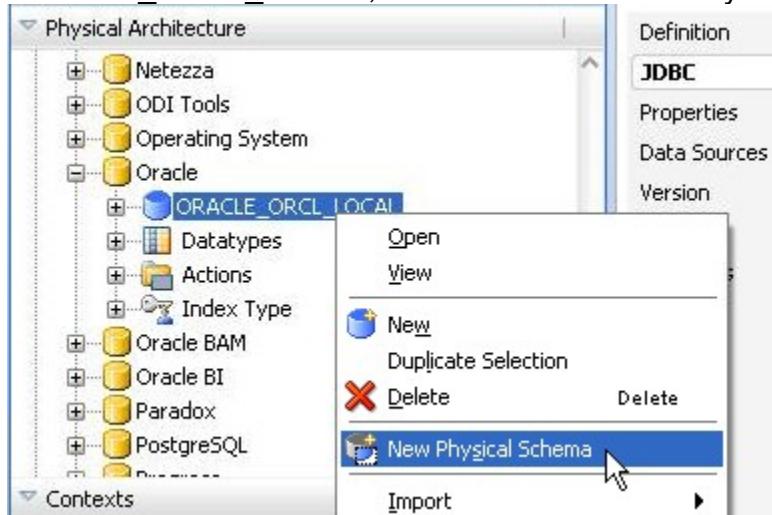


- 10) Create a physical schema for this data server:

ORACLE_ORCL_LOCAL.CUST_DW_DEV

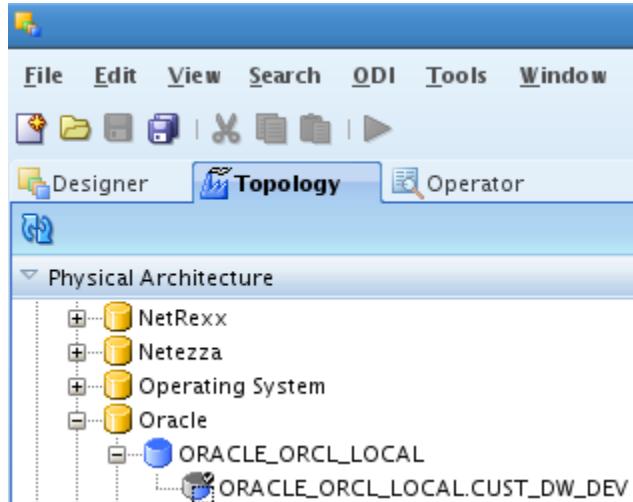
- **Data Schema:** CUST_DW_DEV
- **Work Schema:** ODI_TMP
- **Default check box:** Selected

- a) Expand Oracle node. Right-click the newly created data server ORACLE_ORCL_LOCAL, and then select New Physical Schema.

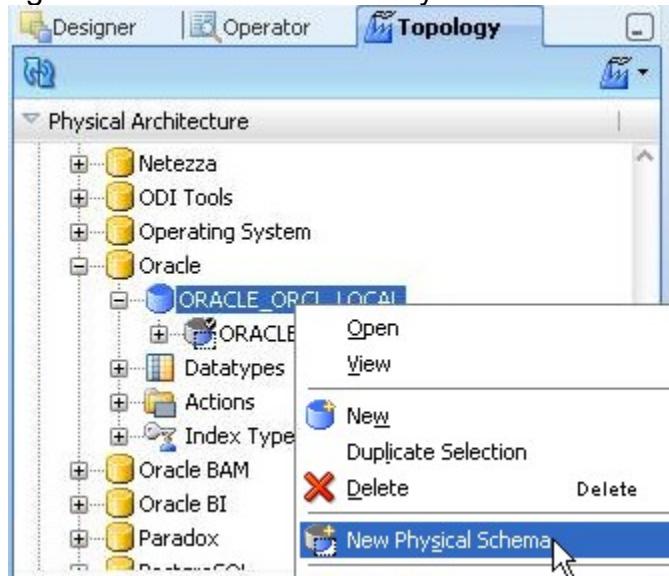


- b) In the new window that appears, select CUST_DW_DEV from the Schema (Schema) drop-down list, and then select ODI_TMP from the Schema (Work Schema) drop-down list. Verify that the Default check box is selected, and leave all other fields unchanged. Click Save button . Click OK in the Information window that appears.

Expand: Oracle > ORACLE_ORCL_LOCAL. The Physical schema ORACLE_ORCL_LOCAL.CUST_DW_DEV appears in the tree view:



- 11) Create a physical schema ORACLE_ORCL_LOCAL.CUST_DW_PROD for the ORACLE_ORCL_LOCAL data server:
- **Data Schema:** CUST_DW_PROD
 - **Work Schema:** ODI_TMP
 - **Default Schema:** Not selected
- a) Select the ORACLE_ORCL_LOCAL data server in the tree view, and then right-click and select New Physical Schema.



- b) In the new window that appears, select CUST_DW_PROD from the Schema (Schema) drop-down list, and then select ODI_TMP from the Schema (Work Schema) drop-down list. Check that the Default check box is *not* selected, and leave all other fields unchanged. Click Save button

In the Information window, click OK.

The screenshot shows the Oracle Data Integrator interface. At the top, there are tabs for Production, Development, ORACLE_ORCL_LOCAL, and ORACLE_ORCL_LOCAL.CUST_DW_PROD. The current view is under the 'Physical Schema [Data Server: ORACLE_ORCL_LOCAL]' tab. On the left, a sidebar has sections for Definition, Context, Version, Privileges, FlexFields, and a 'Default' checkbox. The 'Name:' field is set to ORACLE_ORCL_LOCAL.CUST_DW_PROD. The 'Schema (Schema):' dropdown is set to CUST_DW_PROD, and the 'Schema (Work Schema):' dropdown is set to ODI_TMP. Below this is a 'Physical Architecture' tree view. It starts with Netezza, Operating System, and Oracle. Under Oracle, there is a node for ORACLE_ORCL_LOCAL, which contains two child nodes: ORACLE_ORCL_LOCAL.CUST_DW_DEV and ORACLE_ORCL_LOCAL.CUST_DW_PROD.

- 12) Create a logical schema ORACLE_ORCL_LOCAL_SALES and map this schema in the different contexts.

- **Development Context:** ORACLE_ORCL_LOCAL.CUST_DW_DEV
- **Production Context:** ORACLE_ORCL_LOCAL.CUST_DW_PROD
- **Global Context:** ORACLE_ORCL_LOCAL.CUST_DW_DEV

- a) Select the Logical Architecture tab and expand the Technologies node. Select the Oracle node, and then right-click and select New Logical Schema.

The screenshot shows the Oracle Data Integrator interface with the 'Logical Architecture' tab selected. On the left, a tree view shows various technologies like Netezza, ODI Tools, Operating System, Oracle, Parrot, Postgres, Prog, SAP, and SAS. The Oracle node is expanded. A context menu is open over the Oracle node, with options: Open, View, New, Duplicate Selection, Delete, and New Logical Schema. The 'New Logical Schema' option is highlighted with a blue background and a cursor arrow pointing to it.

Enter the name of the logical schema ORACLE_ORCL_LOCAL_SALES. To map this logical schema to physical schemas in different contexts, select the appropriate physical schema in front of each context, as shown below. Click Save button.

Logical Schema	
Name:	ORACLE_ORCL_LOCAL_SALES
Context	Physical Schemas
Development	ORACLE_ORCL_LOCAL.CUST_DW_DEV
Global	ORACLE_ORCL_LOCAL.CUST_DW_DEV
Production	ORACLE_ORCL_LOCAL.CUST_DW_PROD

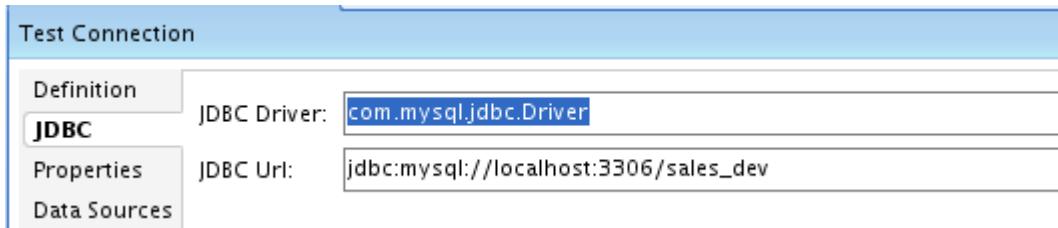
13) Create an MYSQL_PTSDB_LOCAL data server:

- Click the Physical Architecture tab.
- Expand the Technologies node.
- Select the MYSQL node, and then right-click and select New Data Server.
- Enter the following information on the Definition tab:
 - **Name:** MYSQL_PTSDB_LOCAL
 - **User:** pts
 - **Password:** oracle

The screenshot shows the MySQL PTSDB LOCAL Data Server configuration window. The left sidebar has tabs for JDBC, Properties, Data Sources, Version, Privileges, and FlexFields. The main area has two tabs: 'Data Server' and 'Connection'. In 'Data Server', the 'Name' field is 'MYSQL_PTSDB_LOCAL', 'Technology' is 'MySQL', and 'User' is 'pts'. In 'Connection', 'Password' is masked. At the bottom, 'Array Fetch Size' is 30 and 'Batch Update Size' is 30.

- Click the JDBC tab.
- Enter the JDBC Driver and URL to have the following:
DRIVER: com.mysql.jdbc.Driver
URL: jdbc:mysql://localhost:3306/sales_dev

g) The JDBC tab should now appear as follows:



14) Test this data server:

- Click Test Connection button. Click Yes to confirm saving your data before testing connection. In the Information window, click OK.

In the dialog box that appears, click the Test button. Click OK.



15) Create a physical schema (Database) for this data server:

MYSQL PTSDB LOCAL ptsdb

- **Database (Catalog):** sales_dev
- **Database (Work Catalog):** test
- **Default check box:** Selected

- Expand MySQL node. Right-click the newly created data server MYSQL PTSDB LOCAL, and then select New Physical Schema.
- In the new window that appears, select sales_dev from the Database (Schema) drop-down list, and then select test from the Database (Work Schema) drop-down list. Verify that the Default check box is selected, and leave all other fields unchanged. Click Save button . Click OK in the Information window that appears.

Definition

Physical Schema [Data Server: MYSQL PTSDB LOCAL]	
Context	Name: <input type="text" value="MYSQL PTSDB LOCAL.sales_dev"/>
Version	Database (Catalog): <input type="text" value="sales_dev"/>
Privileges	Database (Work Catalog): <input type="text" value="test"/>
FlexFields	<input checked="" type="checkbox"/> Default

- 16) Create a logical schema MYSQL PTSDB LOCAL SALES and map this schema in the different contexts.

- **Development Context:** MYSQL PTSDB LOCAL. sales_dev
- **Production Context:** MYSQL PTSDB LOCAL. sales_dev
- **Global Context:** MYSQL PTSDB LOCAL. sales_dev

- a) Select the Logical Architecture tab and expand the Technologies node. Select the MySQL node, and then right-click and select New Logical Schema.
- b) Enter the name of the logical schema MYSQL PTSDB LOCAL SALES. To map this logical schema to physical schemas in different contexts, select the appropriate physical schema in front of each context, as shown below. Click Save button.

Definition

Logical Schema	
Privileges	Name: <input type="text" value="MYSQL PTSDB LOCAL SALES"/>
FlexFields	
Context	Physical Schemas
Development	MYSQL PTSDB LOCAL sales_dev
Global	MYSQL PTSDB LOCAL sales_dev
Production	MYSQL PTSDB LOCAL sales_dev

Lab 4-3: Creating Agents

Introduction

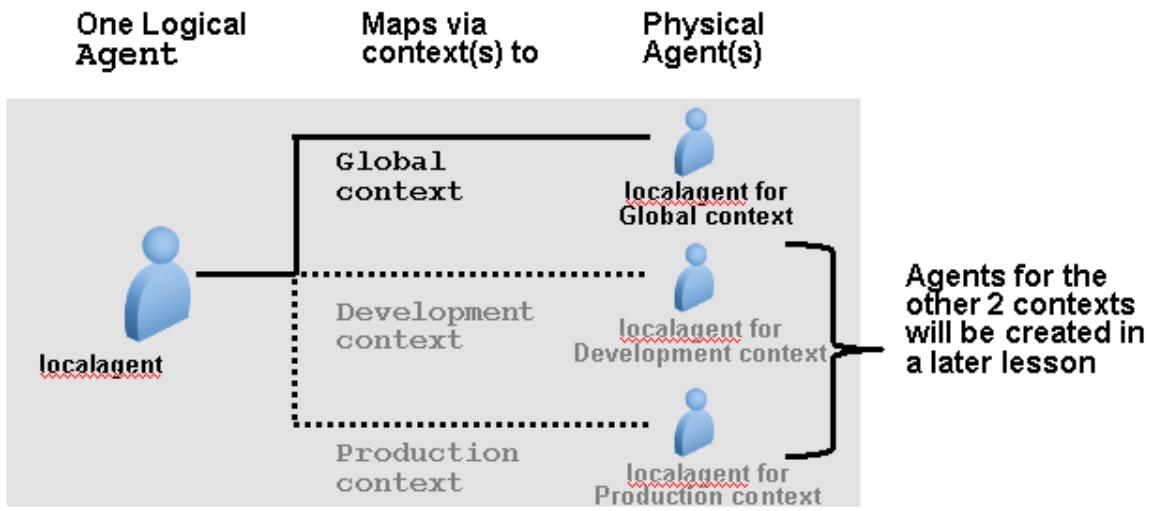
Oracle Data Integrator run-time Agents orchestrate the execution of jobs. These agents are Java components. The run-time agent functions as a *listener* and a *scheduler* agent. The agent executes jobs on demand (model reverses, packages, scenarios, interfaces, and so forth), for example when the job is manually launched from a user interface or from a command line. The agent is also used to start the execution of scenarios according to a schedule defined in Oracle Data Integrator.

There are two types of Oracle Data Integrator agents. A **standalone agent** runs in a separate Java Virtual Machine (JVM) process. It connects to the work repository and to the source and target data servers via JDBC. A **Java EE agent** is deployed as a web application in a Java EE application server. The Java EE agent can benefit from all the features of the application server.

A physical agent corresponds to a single standalone agent or a Java EE agent. Similarly to schemas, physical agents having an identical role in different environments can be grouped under the same logical agent. A logical agent is related to physical agents through contexts. When starting an execution, you indicate the logical agent and the context. Oracle Data Integrator will translate this information into a single physical agent that will receive the execution request.

Process Overview

A common task that is performed using ODI is to set up and install ODI Agent as a service. After the ODI scenarios are created, they can be scheduled and orchestrated using an ODI Agent, which is a lightweight Java process that orchestrates the execution of ODI scenarios. In this practice, you learn how to set up and install an ODI Agent, which will then be used in subsequent practices for orchestration of the execution of ODI objects.



1. Run encode password to generate an encrypted password string.
2. Edit odiparams.bat, inserting the encrypted password.
3. Execute agent.bat, to create/install/start an agent named localagent.
4. In ODI, define a physical agent named localagent.
5. In ODI, define a logical agent named localagent, mapping it to the physical agent named localagent, for the Global context.

Scenario

In the previous labs, you set up your topology by creating Master and Work repositories, contexts, data servers, and physical and logical schemas. Now you complete setting up your ODI environment by installing an ODI Agent as a background service.

Instructions

- To set up an ODI Agent, perform the following steps:

Step	Screen/Page Description	Choices or Values
a.	Linux Shell and text editor	<p>Open new terminal window and go to Lab4 directory by typing command: > cd /home/oracle/PTS_Labs/Lab4</p> <p>To open param files in a text editor, type following command: > gedit Lab4_odiparms.sh</p> <p>Leave this editor open; you will run a batch file from a Command window and return to this text editor to copy the results into the Lab4_odiparms.sh file.</p> <p><u>Explanation:</u> You need to edit the Lab4_odiparms.sh file to set the repository connection information for the ODI Agent to use. The password information is stored as an encrypted string. You will generate the encrypted string in a Command window.</p>
b.	Command Prompt	<p>Leave the text editor open, and open the terminal (Applications >Accessories > Terminal> Type odihome) and change the directory to “cd bin”</p> <p>Type pwd the following directory path should be shown /app/Oracle/Middleware/Oracle_ODI/oracledi/agent/bin</p>
c.	Command Prompt	<p>To generate the encrypted password string, you will be using the agent command: encode <password>. Because your password is the word “welcome1”, enter and execute the following command in the Command window:</p> <p>./encode.sh welcome1</p> <p>Keep the Command window open, as you will be returning to it after editing the Lab4_odiparms.sh file.</p>
d.	Command Prompt and text editor	<p>Copy the generated encrypted password from the Command window (highlight the password with your mouse, from the Command window’s pull-down menu, select Edit > Copy) and insert it into the Lab4_odiparms.sh file as the value for the ODI_MASTER_ENCODED_PASS parameter. Verify and, if necessary, edit settings for <i>other</i> parameters from the</p>

Step	Screen/Page Description	Choices or Values																					
		<p>table below, save the labodiparams.sh file, and then close it. The agent's connectivity is now set up.</p> <table border="1" data-bbox="612 445 1486 1248"> <thead> <tr> <th data-bbox="612 445 1013 508">Parameter</th><th data-bbox="1013 445 1486 508">Value</th></tr> </thead> <tbody> <tr> <td data-bbox="612 508 1013 572">ODI_MASTER_DRIVER</td><td data-bbox="1013 508 1486 572">oracle.jdbc.driver.OracleDriver</td></tr> <tr> <td data-bbox="612 572 1013 656">ODI_MASTER_URL</td><td data-bbox="1013 572 1486 656">jdbc:oracle:thin:@pts.us.oracle.com:1521:XE</td></tr> <tr> <td data-bbox="612 656 1013 720">ODI_MASTER_USER</td><td data-bbox="1013 656 1486 720">snpm1</td></tr> <tr> <td data-bbox="612 720 1013 825">ODI_MASTER_ENCODED_PASS</td><td data-bbox="1013 720 1486 825"><i>Insert your encoded password.</i></td></tr> <tr> <td data-bbox="612 825 1013 889">ODI_SECU_WORK REP</td><td data-bbox="1013 825 1486 889">WORKREP_PR2-1</td></tr> <tr> <td data-bbox="612 889 1013 994">ODI_SUPERVISOR_ENCODED_PASS</td><td data-bbox="1013 889 1486 994"><i>Insert your encoded password.</i></td></tr> <tr> <td data-bbox="612 994 1013 1058">ODI_USER</td><td data-bbox="1013 994 1486 1058"><i>Leave the default value.</i></td></tr> <tr> <td data-bbox="612 1058 1013 1121">ODI_ENCODED_PASS</td><td data-bbox="1013 1058 1486 1121"><i>Leave the default value.</i></td></tr> <tr> <td data-bbox="612 1121 1013 1184">ODI_JAVA_HOME</td><td data-bbox="1013 1121 1486 1184"><i>Leave the default value</i></td></tr> </tbody> </table> <p>Note</p> <ul style="list-style-type: none"> ▪ snpm1 is RDBMS schema/user (Oracle XE) for the Master repository. It was pre-created for this and subsequent practices. ▪ Work Repository name is WORKREP_PR2-1. ▪ Because each time when you encode the password, it gets different values, your encoded password will differ from the one provided in the screenshot. ▪ Do <i>not</i> change the default value of the ODI_USER and ODI_ENCODED_PASS parameters. Those parameters were pre-coded during ODI installation. 	Parameter	Value	ODI_MASTER_DRIVER	oracle.jdbc.driver.OracleDriver	ODI_MASTER_URL	jdbc:oracle:thin:@pts.us.oracle.com:1521:XE	ODI_MASTER_USER	snpm1	ODI_MASTER_ENCODED_PASS	<i>Insert your encoded password.</i>	ODI_SECU_WORK REP	WORKREP_PR2-1	ODI_SUPERVISOR_ENCODED_PASS	<i>Insert your encoded password.</i>	ODI_USER	<i>Leave the default value.</i>	ODI_ENCODED_PASS	<i>Leave the default value.</i>	ODI_JAVA_HOME	<i>Leave the default value</i>	
Parameter	Value																						
ODI_MASTER_DRIVER	oracle.jdbc.driver.OracleDriver																						
ODI_MASTER_URL	jdbc:oracle:thin:@pts.us.oracle.com:1521:XE																						
ODI_MASTER_USER	snpm1																						
ODI_MASTER_ENCODED_PASS	<i>Insert your encoded password.</i>																						
ODI_SECU_WORK REP	WORKREP_PR2-1																						
ODI_SUPERVISOR_ENCODED_PASS	<i>Insert your encoded password.</i>																						
ODI_USER	<i>Leave the default value.</i>																						
ODI_ENCODED_PASS	<i>Leave the default value.</i>																						
ODI_JAVA_HOME	<i>Leave the default value</i>																						

- a) Open Lab4_odiparams.sh file and keep it open to update:

```

oracle@pts:~/PTS_Labs$ gedit Lab4_odiparams.sh
File Edit View Search Tools Documents Help
New Open Save Print... Undo Redo Cut Copy Paste Find Replace
*Lab4_odiparams.sh x
# FILENAME
# Lab4_odiparams.sh
#
# DESCRIPTION
# Configuration script. This script contains the parameters for starting
# Oracle Data Integrator modules for PTS Workshop Lab4. This file demonstrate
# configuration for odi parameters and use of it to create/launch agent.
#
# VARIABLES DESCRIPTION
#
# The following variables should be configured in order to run operations
# requiring a repository connection (agent, agentstop, startscen, agentscheduler, etc.)
#
# ODI_MASTER_DRIVER
# JDBC driver used to connect the Master Repository.
# ODI_MASTER_URL
# JDBC URL used to connect the Master Repository.
# ODI_MASTER_USER
# Database account used to connect the Master Repository.
# ODI_MASTER_ENCODED_PASS
# Database account password. The password must be encoded with the
# "encode.[sh|bat] <password>" command.
#
# The following variables should be configured. They are user credentials
# used only by the agent startup program.
#
# ODI_SUPERVISOR
# OracleDI supervisor user name.
# ODI_SUPERVISOR_ENCODED_PASS

```

- b) Encrypt the password from different terminal window and from within \$ODI_HOME/bin directory

```

oracle@pts:/app/Oracle/Middleware/Oracle_ODI/oracledi/agent/b
File Edit View Terminal Tabs Help
[oracle@pts agent]$ cd bin
[oracle@pts bin]$ pwd
/app/Oracle/Middleware/Oracle_ODI/oracledi/agent/bin

```

```

[oracle@pts bin]$ ./encode.sh welcomel
aYypxcLQIix4qVRZ,dSuyp
[oracle@pts bin]$ 

```

- c) Update param files with encrypted password and other parameters as specified for the step

```

*labodiparams.sh (/app/Oracle/Middleware/Oracle_O
File Edit View Search Tools Documents Help
New Open Save Print... Undo Redo Cut Copy Paste Find Replace
*labodiparams.sh x
# Repository Connection Information
#
ODI_MASTER_DRIVER=oracle.jdbc.OracleDriver
ODI_MASTER_URL="jdbc:oracle:thin:@//pts.us.oracle.com:1521/XE"
ODI_MASTER_USER=snpm1
ODI_MASTER_ENCODED_PASS=h2yarFYQkw4apmVu2jNSnP4f
#ODI_MASTER_ENCODED_PASS=ggfXH9R4p5.05N2t6EW2kQPwjj

#
# User credentials for agent startup program
#
ODI_SUPERVISOR=SUPERVISOR
#ODI_SUPERVISOR_ENCODED_PASS=dLyaICtBftvZGCx2TyvDp
ODI_SUPERVISOR_ENCODED_PASS=aYpxcLQIix4qVRZ,dSuyp

#
# User credentials for ODI tools
#
ODI_USER=SUPERVISOR
ODI_ENCODED_PASS=$ODI_SUPERVISOR_ENCODED_PASS

#
# Work Repository Name
#
ODI_SECU_WORK REP=WORKREP_PR2-1

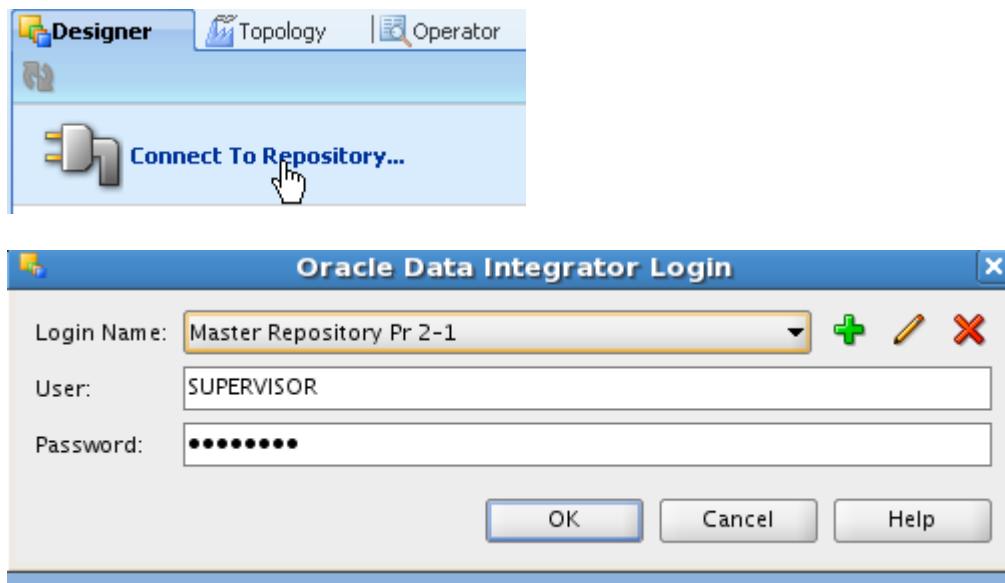
```

- 2) After the ODI Agent is installed, you need to set up ODI Agent with ODI Topology Navigator. To set up ODI Agent with Topology Navigator, perform the following steps:

Step	Screen/Page Description	Choices or Values
a.	ODI Login	If not started, start ODI Studio by selecting Applications > Accessories > Terminal > Type odistudio . Click ODI menu and select “Connect Master Repository Pr 2-1”, and then click “Connect To Repository”. Click OK.
b.	Topology navigator	Click Topology Navigator tab. In Topology Navigator, select the Physical Architecture panel. Right-click the

Step	Screen/Page Description	Choices or Values
		Agents node. Select New Agent.
c.	Agent: localagent	<p>Fill in the following fields:</p> <p>Name: localagent</p> <p>Host: Network name or IP address of the machine the agent has been launched on. Verify that this parameter is set to localhost.</p> <p>Port: Listening port used by the agent. By default, this port is 20910. Leave the default value.</p> <p>Web Application Context: Name of the web application corresponding to the J2EE agent deployed on an application server. For standalone agents, this field should be set to oraclediagent.</p> <p>Set Maximum number of sessions supported by this agent to 250. Click Save button.</p>
d.	Topology Navigator Local Agent: New	Now you have to insert a logical agent in Logical Architecture, and map it to the newly created Physical agent. Click the Logical Architecture tab. Right-click Agents and select New Logical Agent. On the screen that follows, set the Name to " localagent ." Set Physical Agents to localagent . From file menu click Save, and then close the Logical Agent tab.

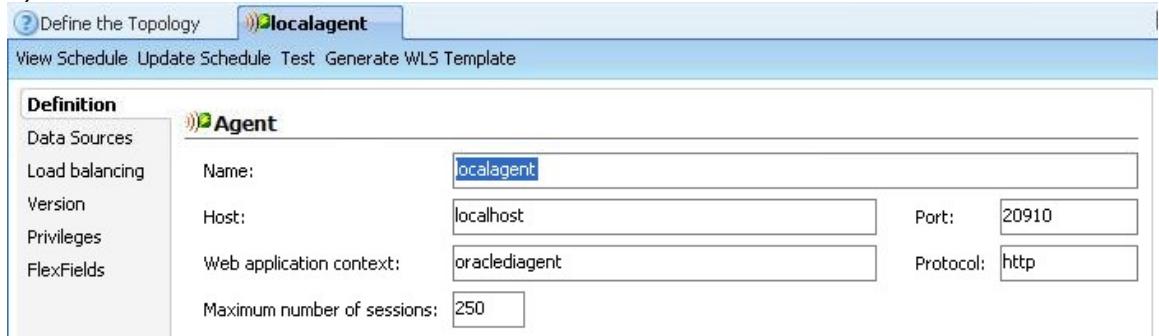
a)



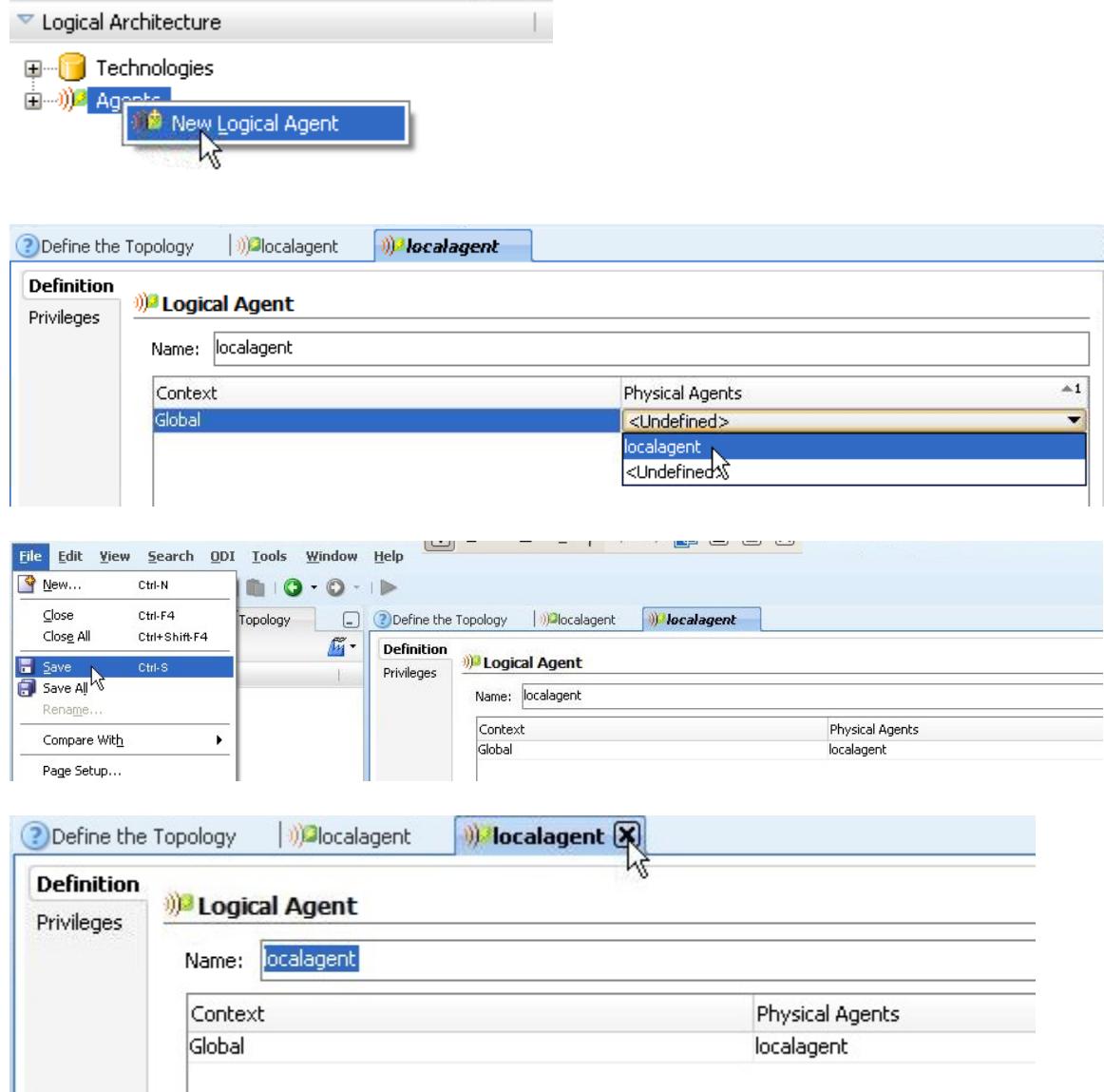
b)



c)



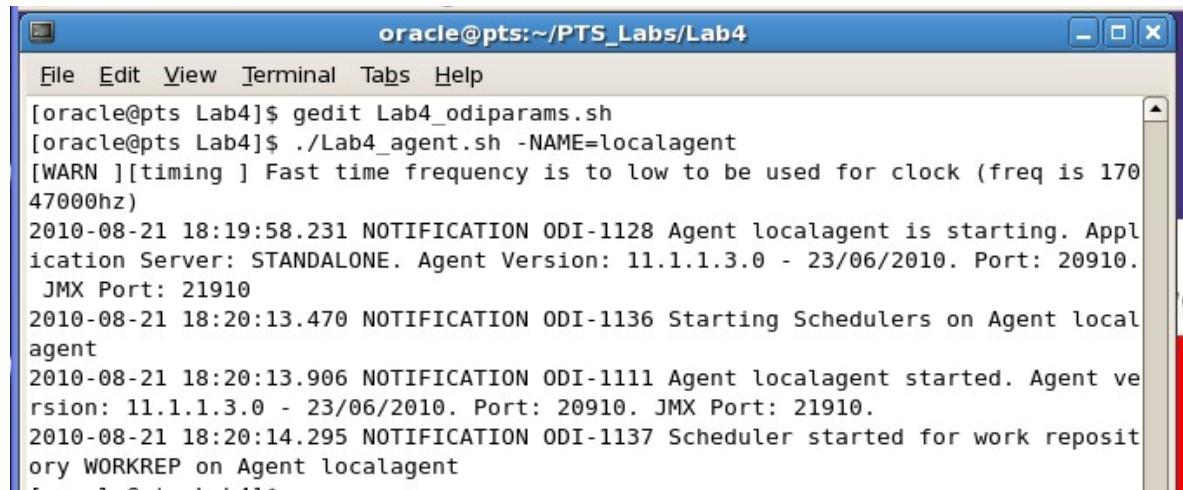
d)



- 3) Now that ODI Agent is set up, it can be executed directly from the command line.

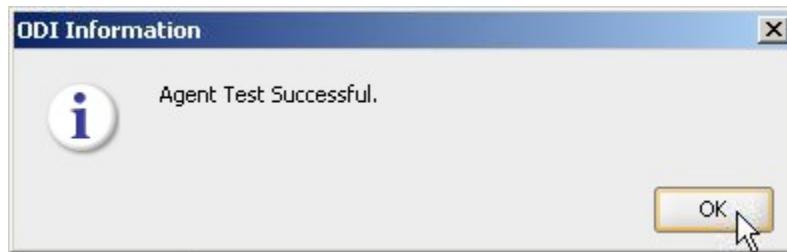
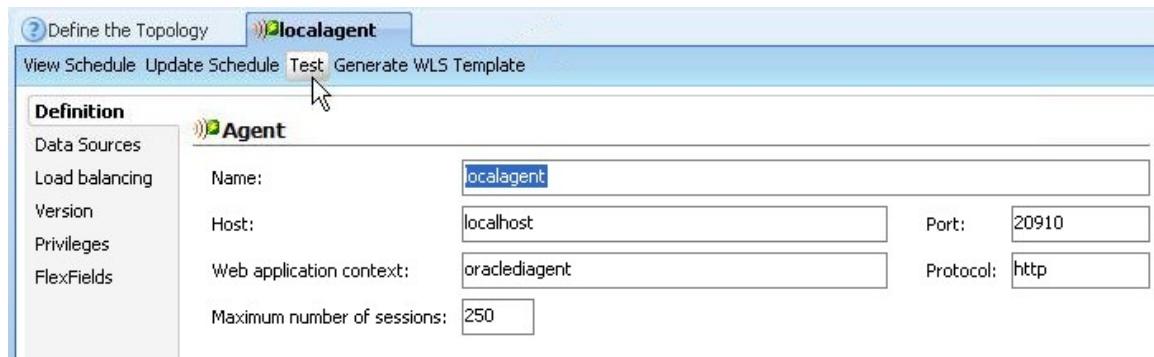
Step	Screen/Page Description	Choices or Values
a.	Command Prompt	Switch back to the terminal shell you left open at the Lab4 directory /home/oracle/PTS_Labs/Lab4 . Execute the Lab4_agent.sh file using the following command: ./Lab4_agent.sh -NAME=localagent
b.	Services	The agent is now starting. Verify that ODI Agent is successfully started. Minimize the window Command Prompt – Lab4_agent.sh-NAME=localagent .
c.	Oracle Data Integrator	In ODI, click Test icon to verify connection of the agent localagent . @ Close localagent tab.

a & b)



```
oracle@pts:~/PTS_Labs/Lab4
File Edit View Terminal Tabs Help
[oracle@pts Lab4]$ gedit Lab4_odiparams.sh
[oracle@pts Lab4]$ ./Lab4_agent.sh -NAME=localagent
[WARN ][timing ] Fast time frequency is to low to be used for clock (freq is 170
47000hz)
2010-08-21 18:19:58.231 NOTIFICATION ODI-1128 Agent localagent is starting. Appl
ication Server: STANDALONE. Agent Version: 11.1.1.3.0 - 23/06/2010. Port: 20910.
JMX Port: 21910
2010-08-21 18:20:13.470 NOTIFICATION ODI-1136 Starting Schedulers on Agent local
agent
2010-08-21 18:20:13.906 NOTIFICATION ODI-1111 Agent localagent started. Agent ve
rsion: 11.1.1.3.0 - 23/06/2010. Port: 20910. JMX Port: 21910.
2010-08-21 18:20:14.295 NOTIFICATION ODI-1137 Scheduler started for work reposi
tory WORKREP on Agent localagent
```

c)



Lab 5: Creating Models

Introduction

A Model is the description of a set of datastores. It corresponds to a group of tabular data structures stored in a data server. A model is based on a Logical Schema defined in the topology. In a given Context, this Logical Schema is mapped to a Physical Schema. The Data Schema of this Physical Schema contains physical data structure: tables, files, JMS messages, elements from an XML file, that are represented as datastores.

Models can be organized into model folders and the datastores of a model can be organized into sub-models.

A new model is created with no datastores. Reverse-engineering is the process that populates the model in Oracle Data Integrator by retrieving metadata from the data server containing the data structures. There are two different types of reverse-engineering:

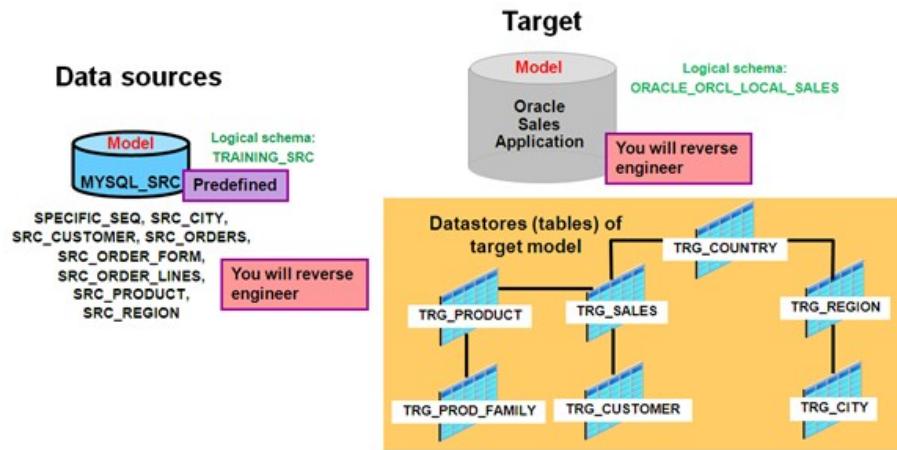
- **Standard reverse-engineering** uses standard JDBC driver features to retrieve the metadata.
- **Customized reverse-engineering** uses a technology-specific Reverse Knowledge Module (RKM) to retrieve the metadata, using a method specific to the given technology. This method usually retrieves more information than the Standard reverse-engineering method.

Prerequisites

Completion **Lab 4-2: Creating and Managing Topology**.

Process Overview

In the previous practice, you configured the schemas containing the application data stored in the Oracle database. You now create the models corresponding to this data and reverse engineer the schemas' data structures.



Steps to create a model by reverse engineering:

1. Create an empty model.
2. Set up the reverse engineering strategy.
 - Standard versus customized reverse engineering
3. Run the reverse engineering process.
 - Use the selective reverse option (standard reverse engineering).
4. Flesh out the model.
 - Add data stores, columns, constraints, and so on.

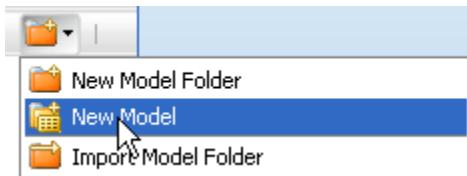
Scenario

Now that you have your ODI environment set up, you can create new ODI Models and reverse engineer all tables in the models.

Instructions

- 1) Connect to Work Repository **WORKREP_PR4** from ODI Studio with user **SUPERVISOR** and password **welcome1**.
- 2) Create a model for the Oracle schema.
 - a) In Designer, select the Models tab.

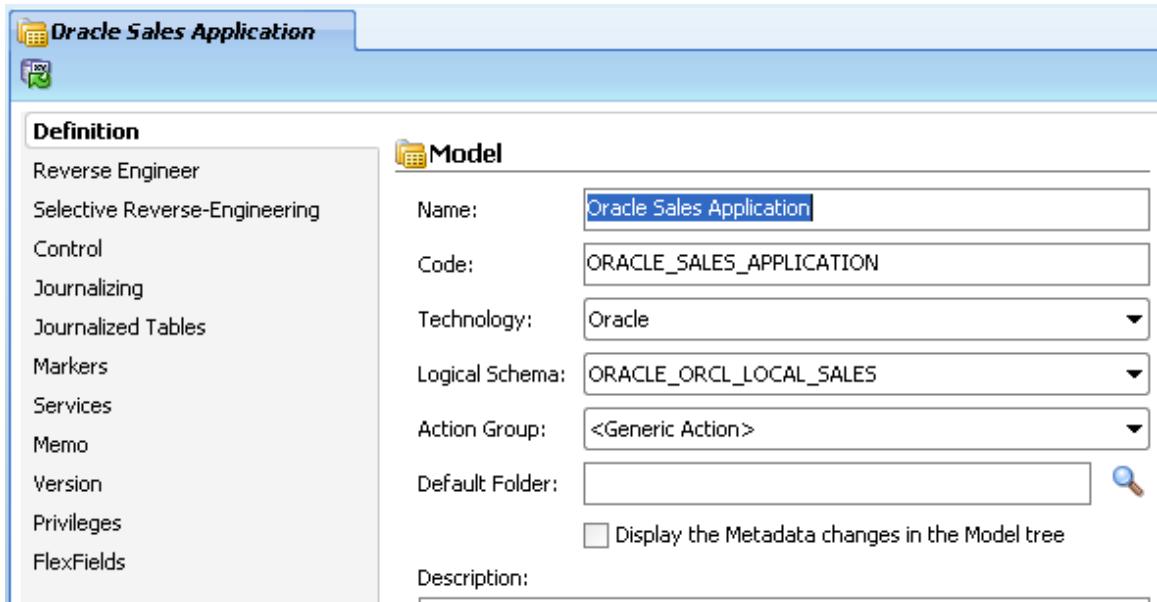
- b) Click the New Model  icon and then select New Model.



- c) Specify (enter or select) the following parameters on the Definition tab:

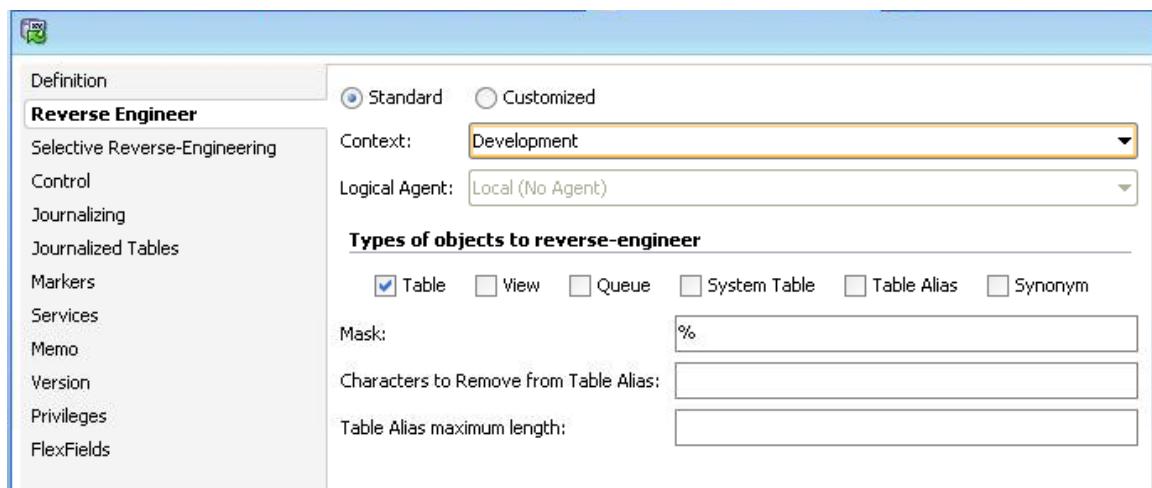
- o **Name:** Oracle Sales Application
- o **Code:** ORACLE_SALES_APPLICATION
- o **Technology:** Oracle
- o **Logical Schema:** ORACLE_ORCL_LOCAL_SALES

The Definition tab should appear as below:



A screenshot of a software application window titled 'Oracle Sales Application'. The window has a tabs bar with 'Definition' and 'Model' tabs. The 'Definition' tab is active and contains a list of options: Reverse Engineer, Selective Reverse-Engineering, Control, Journalizing, Journalized Tables, Markers, Services, Memo, Version, Privileges, and FlexFields. The 'Model' tab is also visible and contains the following configuration fields:
Name: Oracle Sales Application
Code: ORACLE_SALES_APPLICATION
Technology: Oracle
Logical Schema: ORACLE_ORCL_LOCAL_SALES
Action Group: <Generic Action>
Default Folder: (empty)
 Display the Metadata changes in the Model tree
Description: (empty)

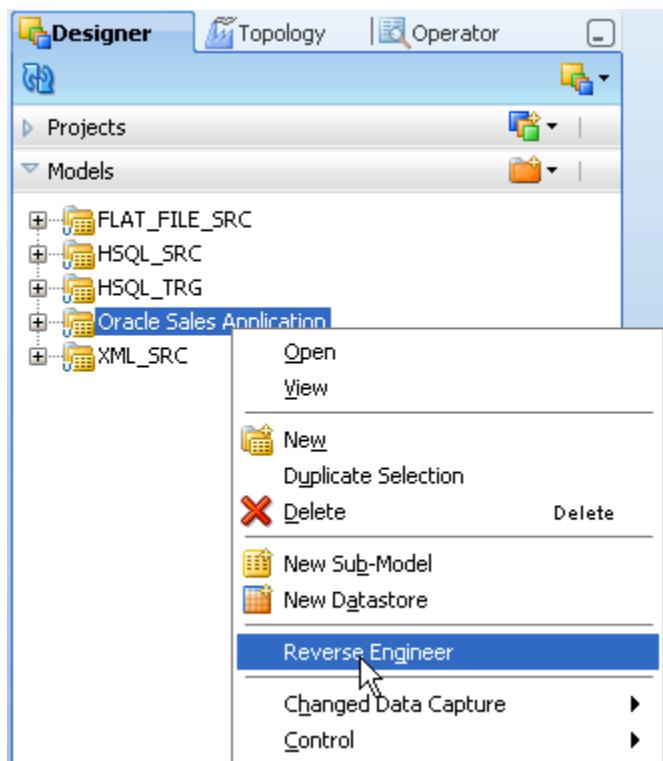
- d) Click the Reverse Engineer tab, and then select Development from the Context drop-down list. Click Save button.



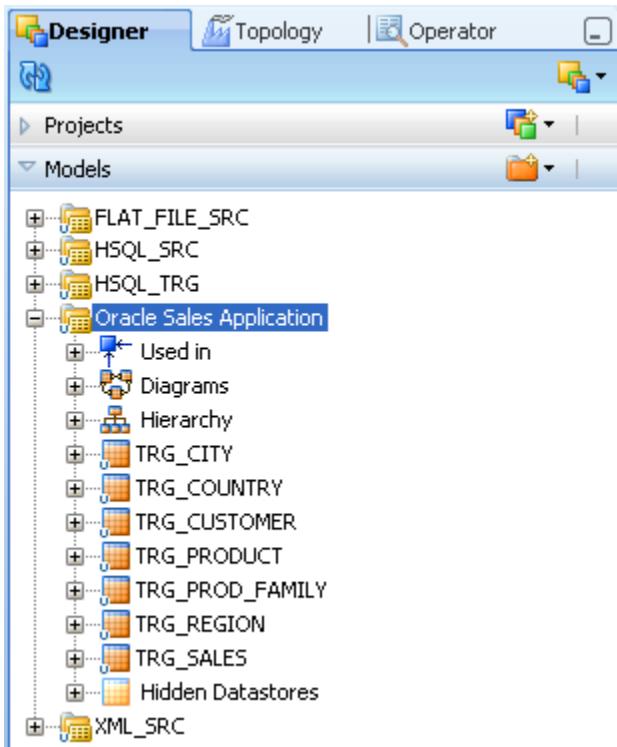
- 3) Reverse engineer all the tables in this model.

- a) Right-click Oracle Sales Application model and select the Reverse Engineer option. If the Confirmation window appears, click Yes.

Note: The progress of the reverse engineering process is shown on the status bar.



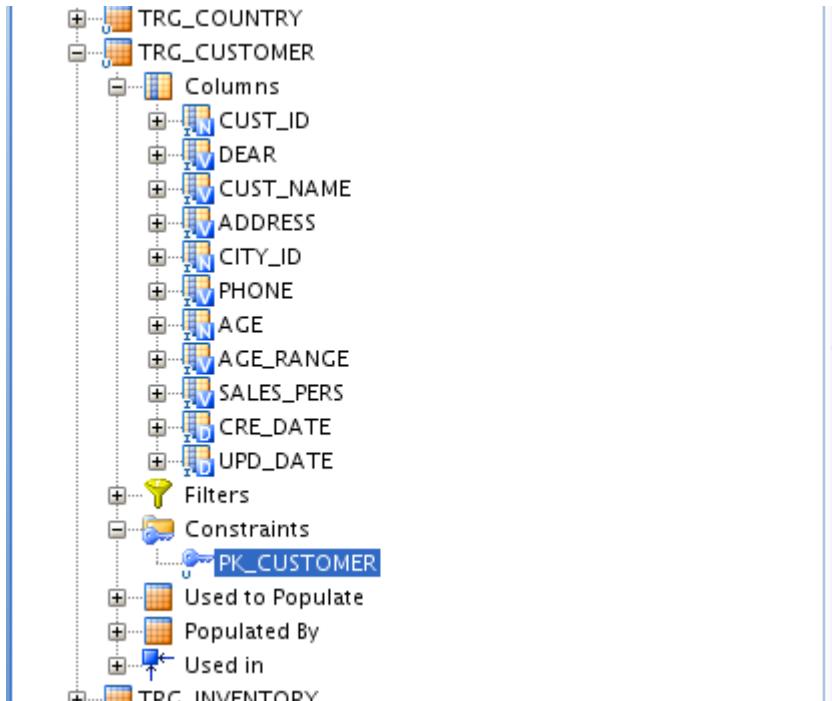
- b) Verify the model reverse engineered successfully. In the tree view, expand the Oracle Sales Application model. The *datastores* of the model appear.



- 4) Check that the columns and constraints that were reverse engineered for the TRG_CUSTOMER table correspond to its data definition language (DDL) given below:

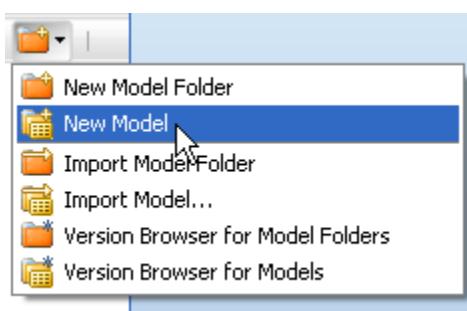
```
create table TRG_CUSTOMER (
    CUST_ID      NUMERIC(10) not null,
    DEAR         VARCHAR(4),
    CUST_NAME    VARCHAR(50),
    ADDRESS      VARCHAR(100),
    CITY_ID      NUMERIC(10) not null,
    PHONE        VARCHAR(50),
    AGE          NUMERIC(3),
    AGE_RANGE    VARCHAR(50),
    SALES_PERS   VARCHAR(50),
    CRE_DATE     DATE,
    UPD_DATE     DATE,
    constraint PK_TRG_CUSTOMER primary key (CUST_ID));
```

- a) Expand the TRG_CUSTOMER *datastore*, and then expand the Columns and Constraints nodes. The list of columns and constraints that were reverse engineered for this table appear in the tree view. Close the tabs.



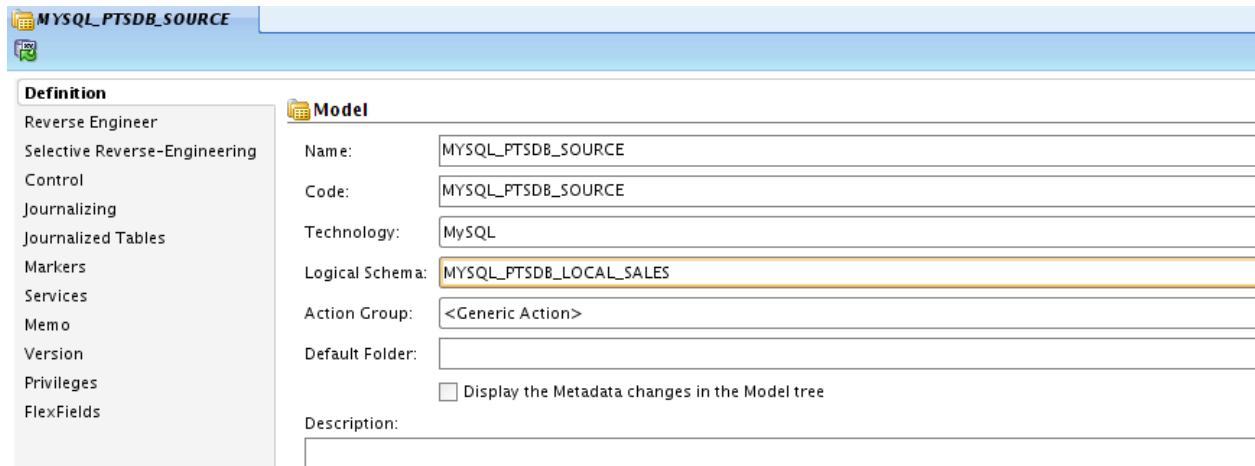
5) Create a model for MYSQL Schema.

- a) Click the New Model  icon and then select New Model.

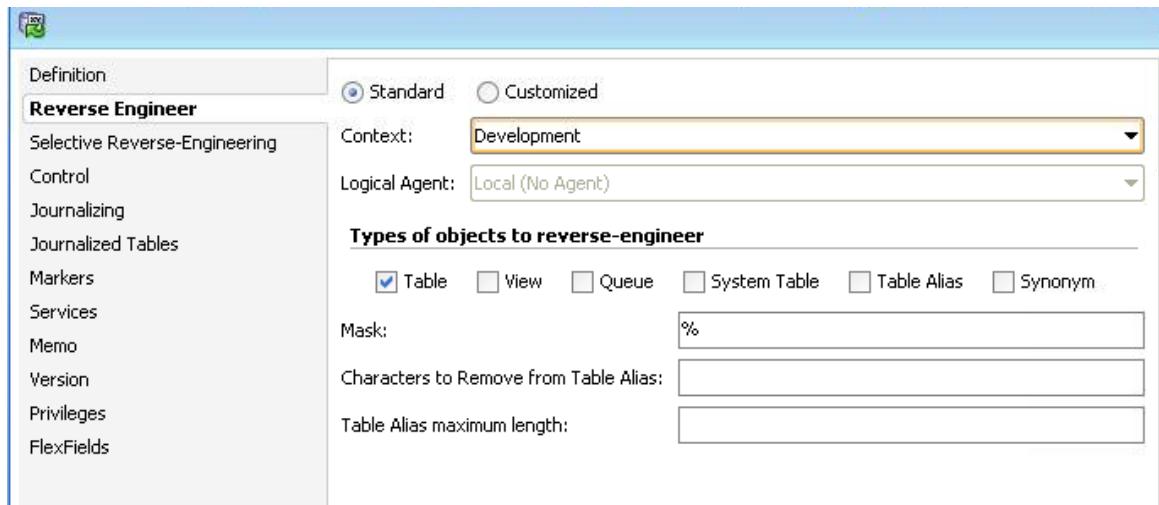


- b) Specify (enter or select) the following parameters on the Definition tab:
- o **Name:** MYSQL PTSDB SOURCE
 - o **Code:** MYSQL PTSDB SOURCE
 - o **Technology:** MySQL
 - o **Logical Schema:** MYSQL PTSDB LOCAL SALES

- c) The Definition tab should appear as below.



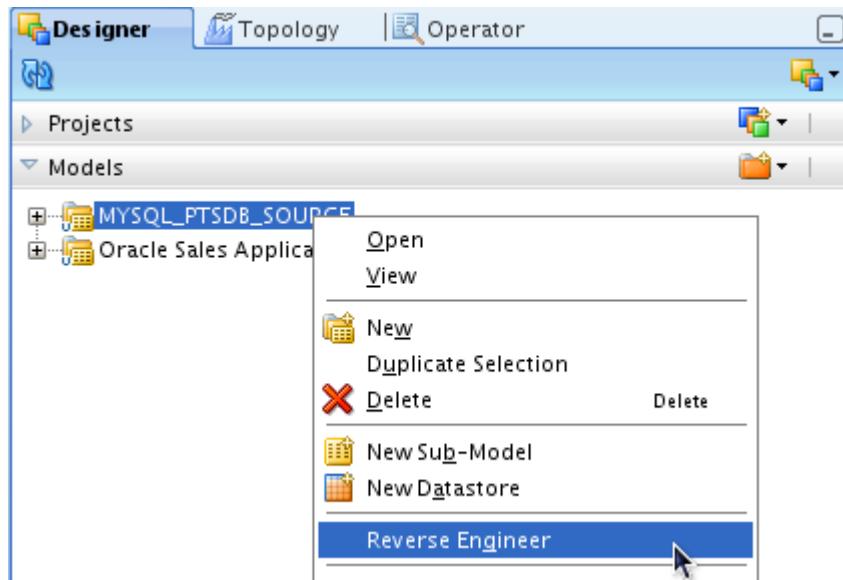
- d) Click the Reverse Engineer tab. Select Development from the Context drop-down list. Click the Save button.



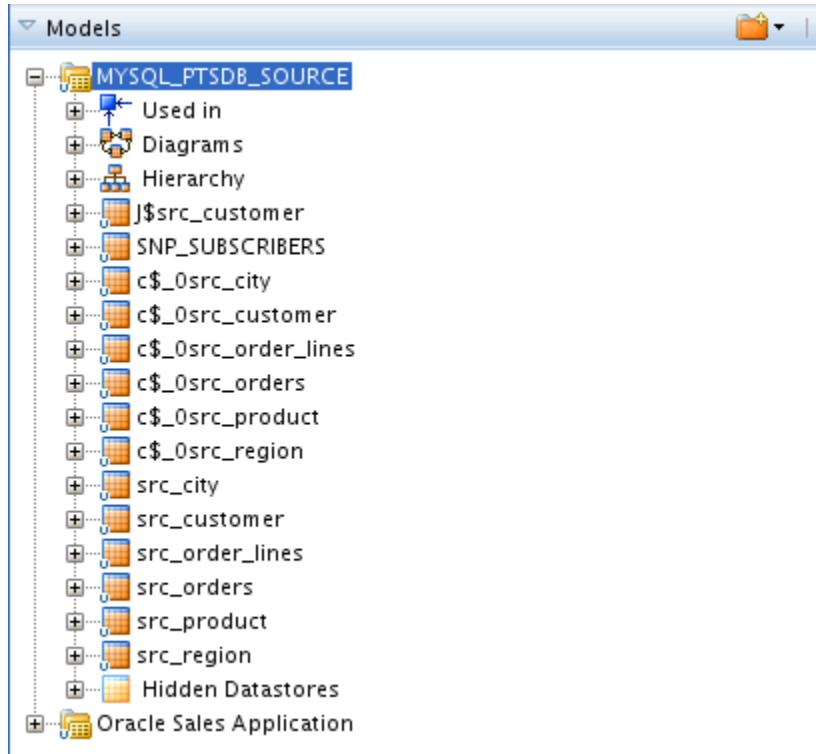
- 6) Reverse engineer all the tables in this model.

- a) In the Models tree view, right-click MYSQLDB PTSDB_SOURCE model and select the Reverse Engineer option.

Note: The progress of the reverse engineering process is shown on the status bar.



- b) Verify the model reverse engineered successfully. In the tree view, expand the Geographic Information model. The *datastores* of the model appear.



Lab 6: Creating Project and Folders

Introduction

An integration project is composed of several components:

- **Folders** – components that help organize the work into a project:
 - **Packages** – workflow, made up of a sequence of steps organized into an execution diagram
 - **Interfaces** – reusable dataflow comprised of a set of declarative rules that describe the loading of a datastore or a temporary target structure from one or more source datastores
 - **Procedures** – reusable component that groups a sequence of operations
- **Variables, sequences, user functions**
- **Knowledge Modules** – code template related to a given technology that provides a specific function (loading data, reverse-engineering, journalizing)
- **Markers** - tags attached to any ODI object to help organize a project

Process Overview

Guidelines in creating a new project:

- Set up a new project
- Use folders to organize your work
- Import the right knowledge modules
- Import and export objects
- Use markers to manage the project

It is recommended to maintain a good organization of the project by using folders. Folders simplify finding objects developed in the project and facilitate the maintenance tasks.

Scenario

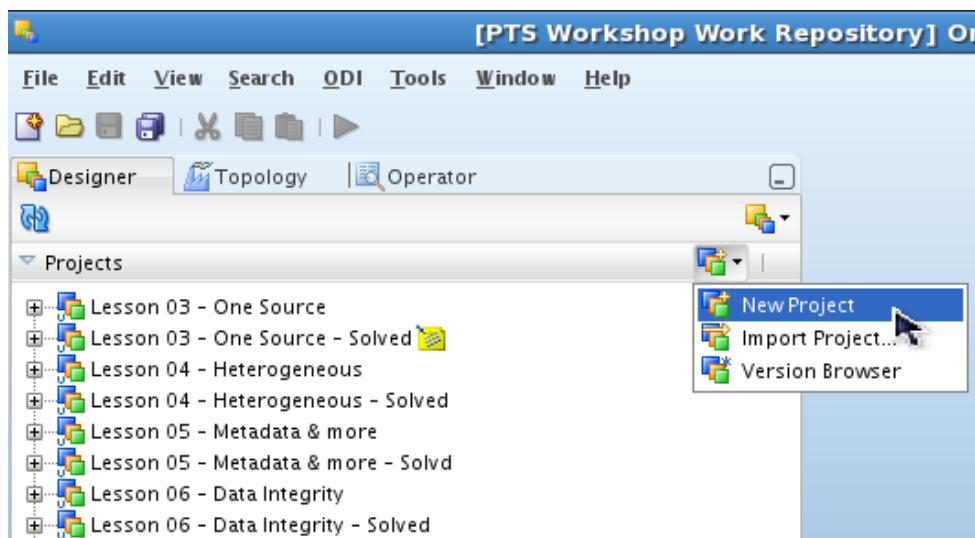
You are responsible for data loading, transformation, and validation. To start an integration project, you need to create a new ODI project and import Knowledge Modules that will be used for development of your integration project.

Instructions

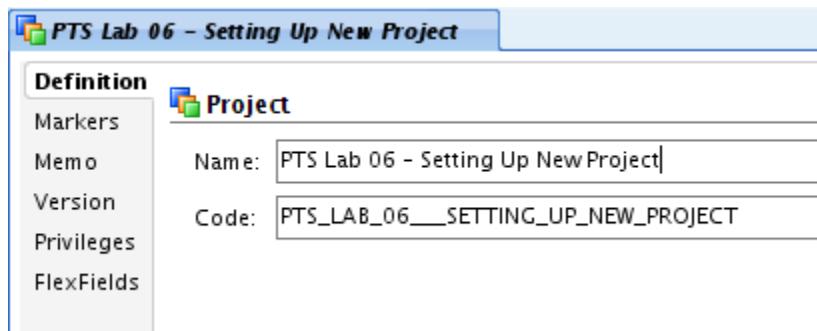
- 1) If necessary, connect to Work Repository user **PTS Training Work Repository** from ODI Studio with user **SUPERVISOR** and password **SUPERVISOR**. Create New Project.

Create a project **PTS Lab 06 – Setting Up New Project** with a folder called **HandsOn**.

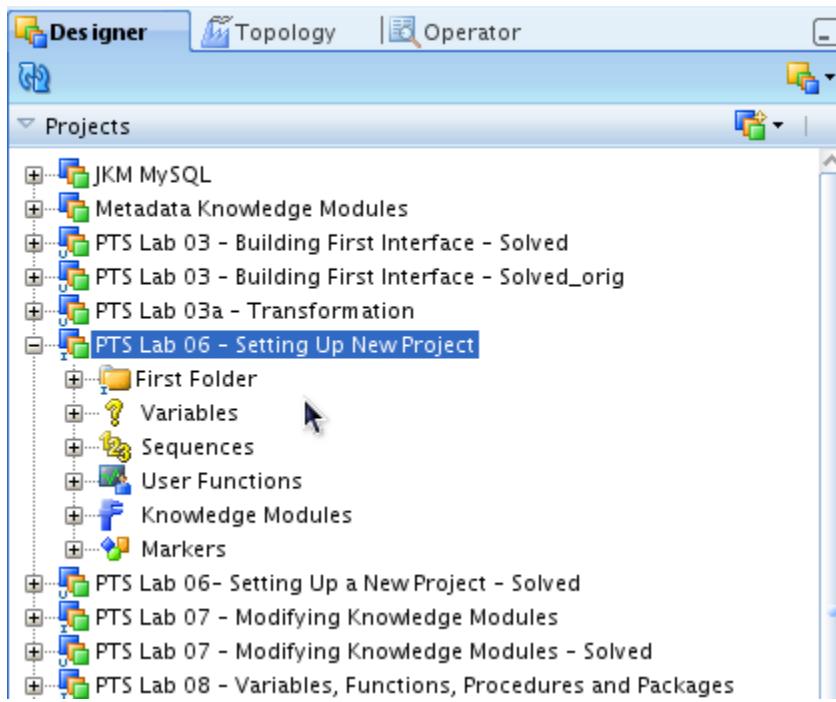
- a) Switch to the Designer Navigator. Click the **Designer** tab. Select **Projects** tab, click the **New Project** icon, and then select **New Project**.



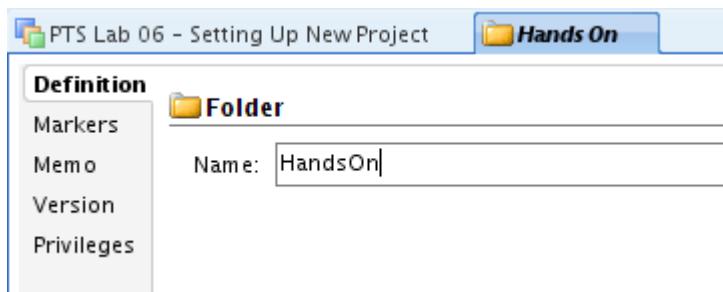
Enter the name of the project: **PTS Lab 06 – Setting Up New Project**.
The code is automatically generated. Note: If the maximum of 35 characters is reached, this will result in a blank code. In such cases, you may need to manually enter a value for the code.

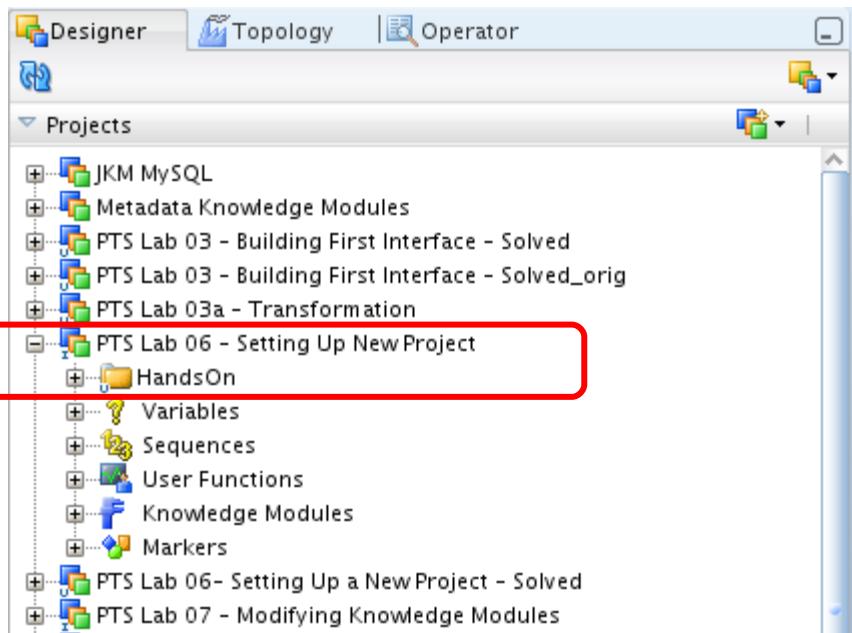


- b) Click the **Save** button to create the project. The project appears in the tree view. Expand the **PTS Lab 06 – Setting Up New Project** project node.



- c) ODI creates a folder called **FirstFolder**. Double-click this folder. In the editing window that appears, edit the folder Name: **HandsOn**. Click **Save** button.



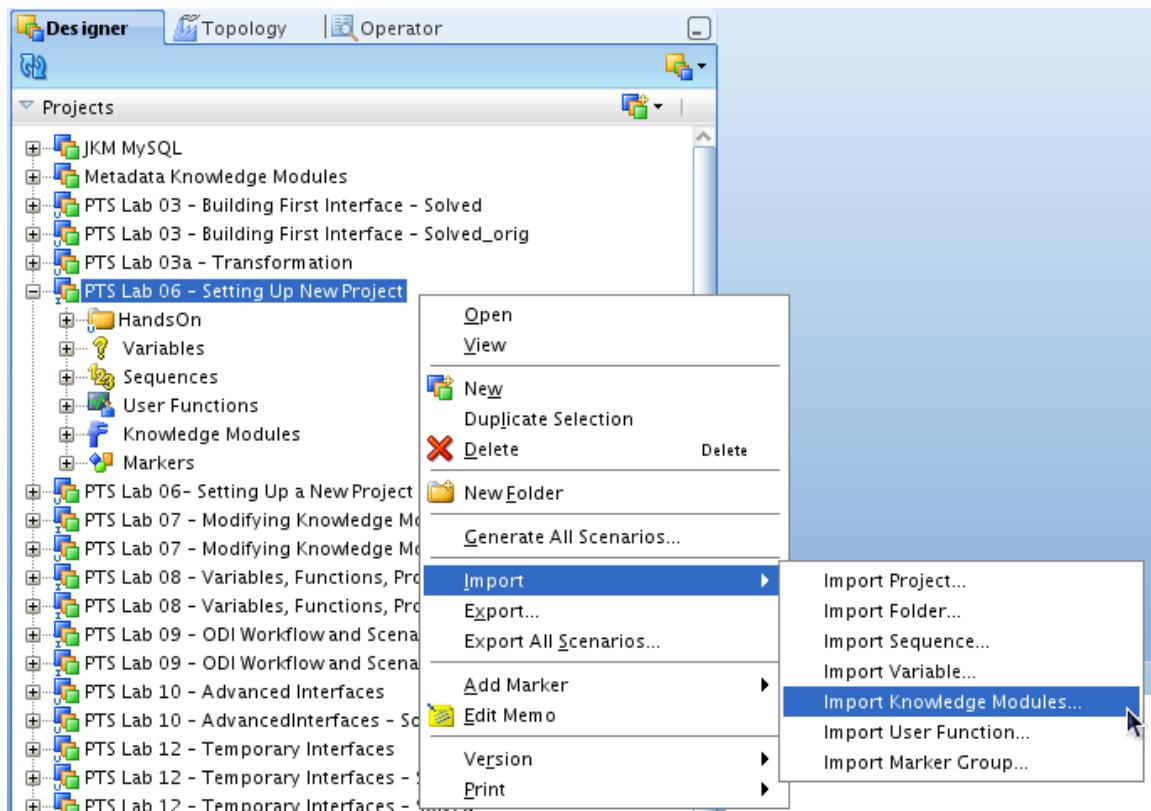


2) Import Knowledge Modules

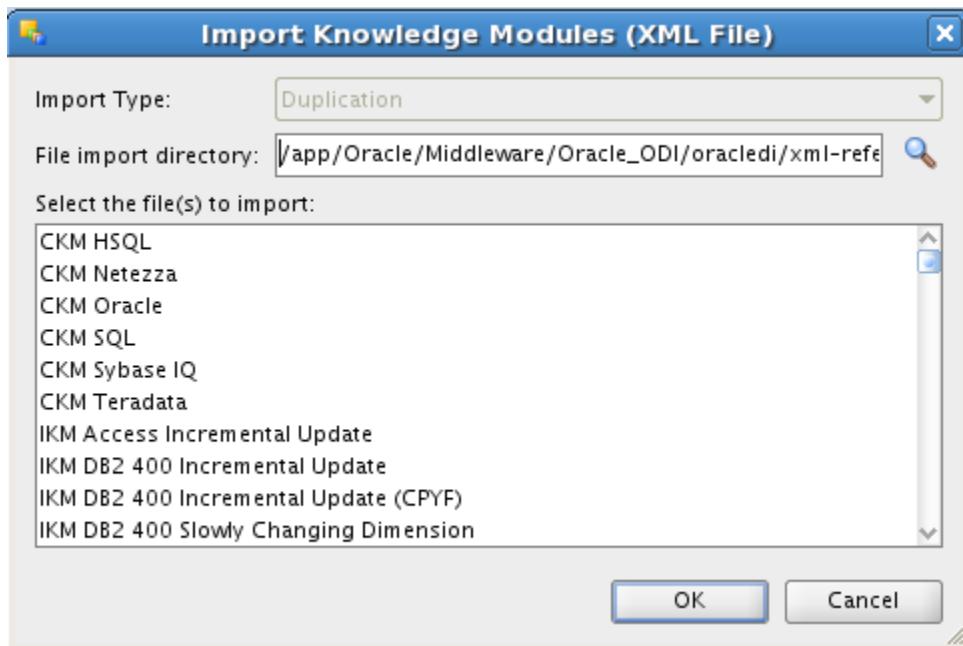
Import the knowledge modules required for working with the following technologies:

- Sources:
 - MySQL
 - File
- Targets
 - Oracle

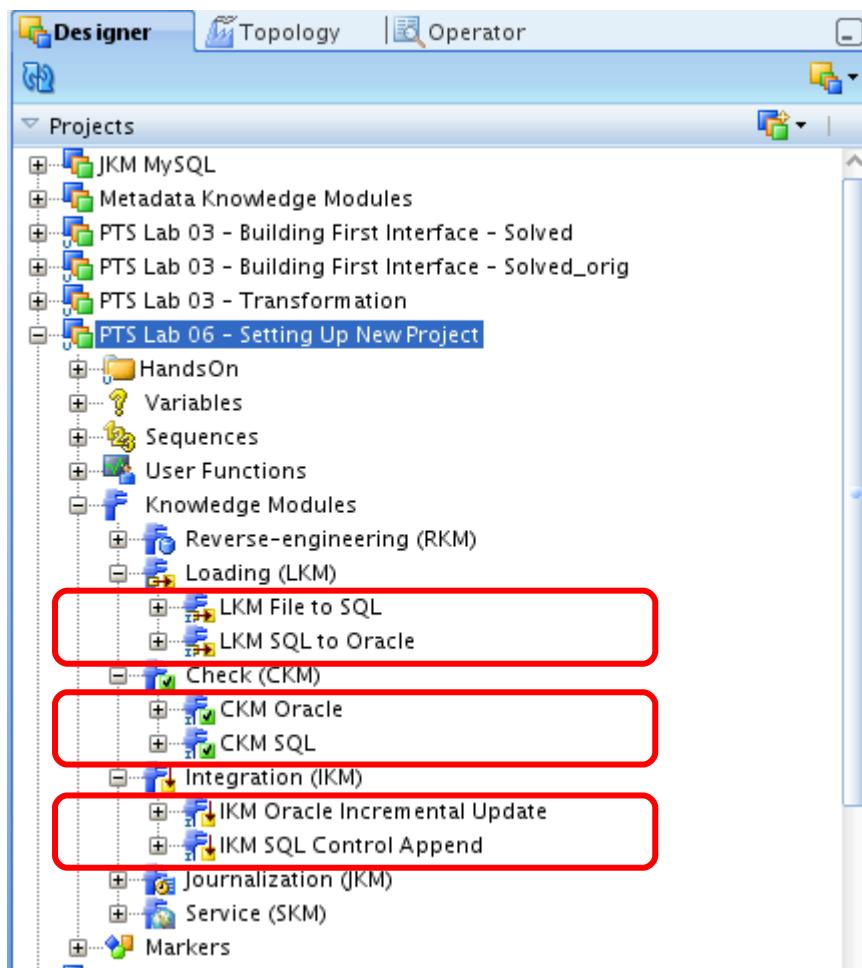
- a) Select the **PTS Lab 06 – Setting Up New Project** project node. Right-click and then select **Import > Import Knowledge Modules**.



- b) The Import Knowledge Modules window appears.
If no Knowledge Modules appear, type or navigate to the following directory: /app/Oracle/Middleware/Oracel_ODI/oracledi/xml-reference



- c) Select the following knowledge modules. Use the Ctrl key for multiple selections. Click **OK**. In the **Import Report**, click **Close**.
- CKM SQL
 - CKM Oracle
 - IKM Oracle Incremental Update
 - IKM SQL Control Append
 - LKM File to SQL
 - LKM SQL to Oracle
- d) Check the imported knowledge modules by expanding corresponding nodes **PTS Lab 06 – Setting Up New Project > Knowledge Modules** as shown below:



Lab 7: Customizing Knowledge Modules

Introduction

Knowledge Modules (KMs) are components of Oracle Data Integrator's Open Connector technology. KMs contain the knowledge required by Oracle Data Integrator to perform a specific set of tasks against a specific technology or set of technologies.

Oracle Data Integrator uses six different types of Knowledge Modules:

- **RKM (Reverse Knowledge Modules)** are used to perform a customized reverse-engineering of data models for a specific technology. These KMs are used in data models.
- **LKM (Loading Knowledge Modules)** are used to extract data from source systems (files, middleware, database, etc.). These KMs are used in interfaces.
- **JKM (Journalizing Knowledge Modules)** are used to create a journal of data modifications (insert, update and delete) of the source databases to keep track of the changes. These KMs are used in data models and used for Changed Data Capture.
- **IKM (Integration Knowledge Modules)** are used to integrate (load) data to the target tables. These KMs are used in interfaces.
- **CKM (Check Knowledge Modules)** are used to check that constraints on the sources and targets are not violated. These KMs are used in data model's static check and interfaces flow checks.
- **SKM (Service Knowledge Modules)** are used to generate the code required for creating data services. These KMs are used in data models.

Process Overview

Guidelines for developing your own Knowledge Modules:

- Very few KMs are ever created. They usually are extensions or modifications of existing KMs.
- Duplicate existing steps and modify them. This prevents typos in the syntax of the odiRef methods.

- All interfaces using the KM inherit the new behavior. Remember to make a copy of the KM if you do not want to alter existing interfaces. Modify the copy, not the original.

Scenario

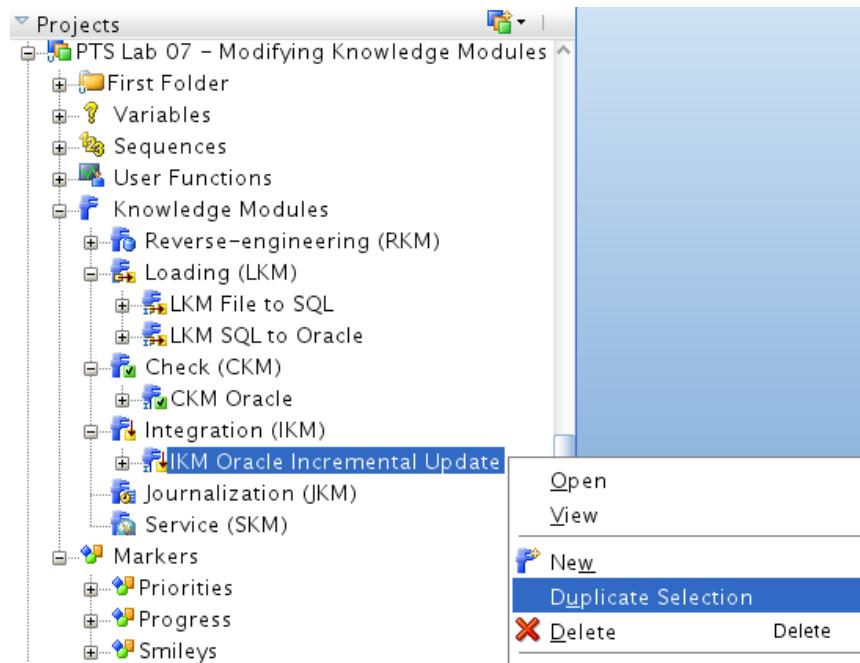
We will now modify an existing Knowledge Module to see how efficient this modification can be. We will modify a Knowledge Module that is already in use and propagate our changes to the interfaces using this Knowledge Module.

We will add auditing capability to our interface, by modifying the KM. This is typical of what would be done by a customer who wants to add SOX-type capabilities to an existing deployment.

Lab 7.1: Modify Knowledge Module

Instructions

- 1) In the project ***PTS Lab 07 - Modifying Knowledge Modules***, right-click on the IKM Oracle Incremental Update Knowledge Module, and select Duplicate Selection.



2) Open the copy and rename it to *IKM Oracle IU – Audit*.

The objective here is to add two steps that will:

- create an audit table (and only generate a warning if the table already exists). Name the table after the target table, simply add _H at the end of the table name. (for instance, *CUSTOMER_H*)
- in the audit table, insert three columns, the primary key of the record being processed by the interface, a time stamp, and an indicator that will tell us what operation was done on the record ('I' = Insert, 'D' = Delete, 'U' = Update).

3) In the Details tab, add a step

The screenshot shows the 'IKM Oracle IU - Audit' application window. The left sidebar has tabs for 'Definition' (selected), 'Details' (selected), 'Options', 'Markers', 'Memo', 'Version', 'Privileges', and 'FlexFields'. The 'Lines' tab is also visible. The main area is titled 'Details' and contains a table with the following data:

Or...	Command	Context	Logica...	Transac...	Commit	Ignore E...	Log Level	Line Cou...
10	Validate KM options					<input type="checkbox"/>	5	
20	Check RDBMS version					<input type="checkbox"/>	5	
30	Create target table					<input checked="" type="checkbox"/>	5	
40	Drop flow table					<input checked="" type="checkbox"/>	5	
50	Create flow table I\$					<input checked="" type="checkbox"/>	5	
60	Delete target table					<input type="checkbox"/>	3	Delete
70	Truncate target table					<input type="checkbox"/>	3	
80	Analyze target table					<input checked="" type="checkbox"/>	5	
90	Lock journalized table					<input type="checkbox"/>	4	
100	Create Temp. Indexes o...					<input type="checkbox"/>	5	
110	Insert flow into I\$ table					<input type="checkbox"/>	4	
120	Recycle previous errors					<input checked="" type="checkbox"/>	4	
130	Create Index on flow ta...					<input checked="" type="checkbox"/>	5	
140	Analyze integration table					<input checked="" type="checkbox"/>	5	
150	Synchronize deletions f...					<input type="checkbox"/>	3	Delete
160	Remove deleted rows fr...					<input type="checkbox"/>	4	
170	Flow control					<input type="checkbox"/>	4	
180	Flag rows for update					<input type="checkbox"/>	4	
190	Flag useless rows					<input type="checkbox"/>	4	
200	Update existing rows			Transac...	No Co...	<input type="checkbox"/>	3	Update
210	Insert new rows			Transac...	No Co...	<input type="checkbox"/>	3	Insert
220	Commit transaction			Transac...	Commit	<input type="checkbox"/>	3	
230	Cleanup journalized ta...					<input type="checkbox"/>	4	
240	Analyze target table					<input checked="" type="checkbox"/>	5	
250	Post-integration control					<input type="checkbox"/>	4	
260	Drop flow table					<input checked="" type="checkbox"/>	5	
270	Drop Temp. Indexes on...					<input checked="" type="checkbox"/>	5	
280	sub-select inline view					<input type="checkbox"/>	3	

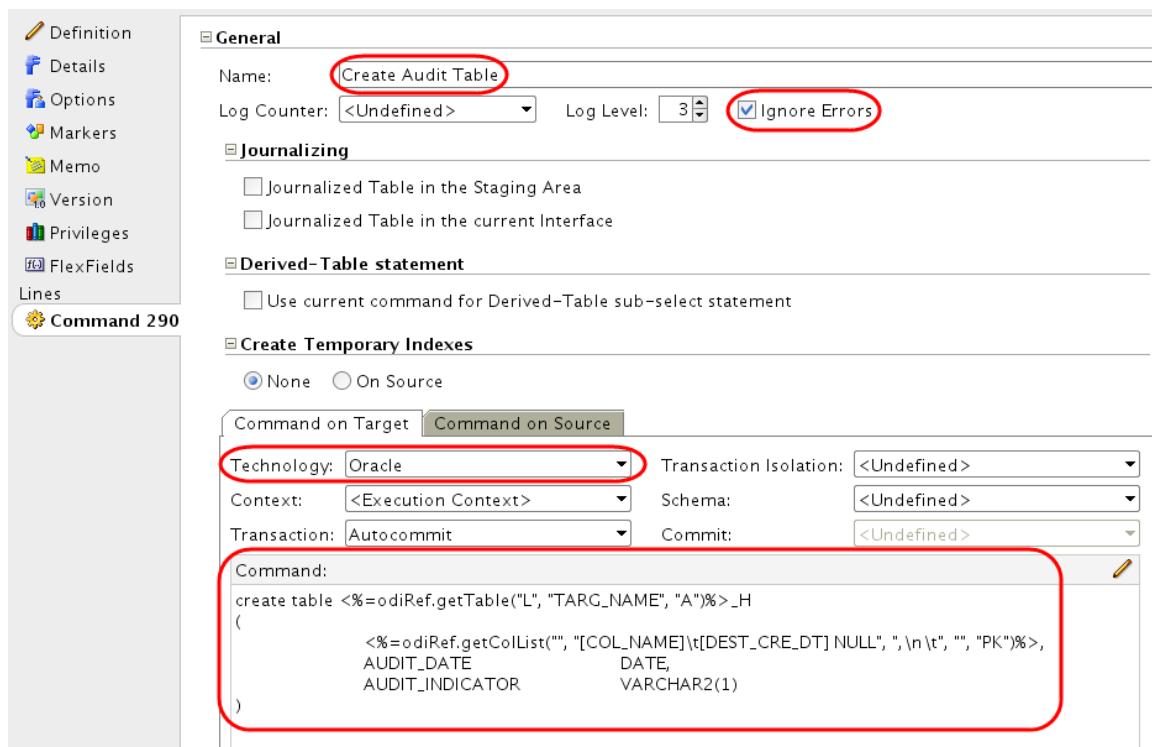
4) In the new Command window, name this command *Create Audit Table*. Select the *Ignore Errors* check box. Set the Technology on Target to *Oracle*. Enter the following command to create the audit table. Verify that check boxes in *Journalizing* sections are not selected. If necessary deselect them.

Note: To reduce typing, you can copy the code from a similar step and modify as needed.

```

create table <%=odiRef.getTable("L", "TARG_NAME", "A")%>_H
(
    <%=odiRef.getColList("", "[COL_NAME]\t[DEST_CRE_DT]NULL",
    ",\n\t", "", "PK")%>,
        AUDIT_DATE          DATE,
        AUDIT_INDICATOR     VARCHAR2(1)
)

```



- 5) Click **Details** tab. Scroll down and select the *Create Audit Table* command at the bottom. Click the icon several times to move the command up and place it right after the *Insert new rows* step. Verify that the *Ignore Errors* checkbox is selected.

Order	Command	Context	Logical ...	Transacti...	Commit	Ignore Err...	Log Level	Log Coun...
10	Validate KM options					<input type="checkbox"/>	5	
20	Check RDBMS version					<input type="checkbox"/>	5	
30	Create target table					<input checked="" type="checkbox"/>	5	
40	Drop flow table					<input checked="" type="checkbox"/>	5	
50	Create flow table I\$					<input checked="" type="checkbox"/>	5	
60	Delete target table					<input type="checkbox"/>	3	Delete
70	Truncate target table					<input type="checkbox"/>	3	
80	Analyze target table					<input checked="" type="checkbox"/>	5	
90	Lock journalized table					<input type="checkbox"/>	4	
100	Create Temp. Indexes o...					<input type="checkbox"/>	5	
110	Insert flow into I\$ table					<input type="checkbox"/>	4	
120	Recycle previous errors					<input checked="" type="checkbox"/>	4	
130	Create Index on flow table					<input checked="" type="checkbox"/>	5	
140	Analyze integration table					<input checked="" type="checkbox"/>	5	
150	Synchronize deletions fr...					<input type="checkbox"/>	3	Delete
160	Remove deleted rows fro...					<input type="checkbox"/>	4	
170	Flow control					<input type="checkbox"/>	4	
180	Flag rows for update					<input type="checkbox"/>	4	
190	Flag useless rows					<input type="checkbox"/>	4	
200	Update existing rows			Transacti...	No Com...	<input type="checkbox"/>	3	Update
210	Insert new rows			Transacti...	No Com...	<input type="checkbox"/>	3	Insert
220	Create Audit Table					<input checked="" type="checkbox"/>	3	
230	Commit transaction			Transacti...	Commit	<input type="checkbox"/>	3	
240	Cleanup journalized table					<input type="checkbox"/>	4	
250	Analyze target table					<input checked="" type="checkbox"/>	5	
260	Post-integration control					<input type="checkbox"/>	4	
270	Drop flow table					<input checked="" type="checkbox"/>	5	
280	Drop Temp. Indexes on ...					<input checked="" type="checkbox"/>	5	
290	sub-select inline view					<input type="checkbox"/>	3	

- 6) Similarly, create a step that inserts the records into the audit table, call it *Insert Audit Records*. Select the *Ignore Errors* check box. Set the Technology on Target to *Oracle*. Enter the following command to insert the records.

```
Insert into <%=odiRef.getTable("L", "TARG_NAME", "A")%>_H
(
<%=odiRef.getColList("", "[COL_NAME]", ",\n\t", "", "PK")%>,
AUDIT_DATE,
AUDIT_INDICATOR
)
select <%=odiRef.getColList("", "[COL_NAME]", ",\n\t", "", "PK")%>,
sysdate,
IND_UPDATE
from <%=odiRef.getTable("L","INT_NAME","W")%>
```

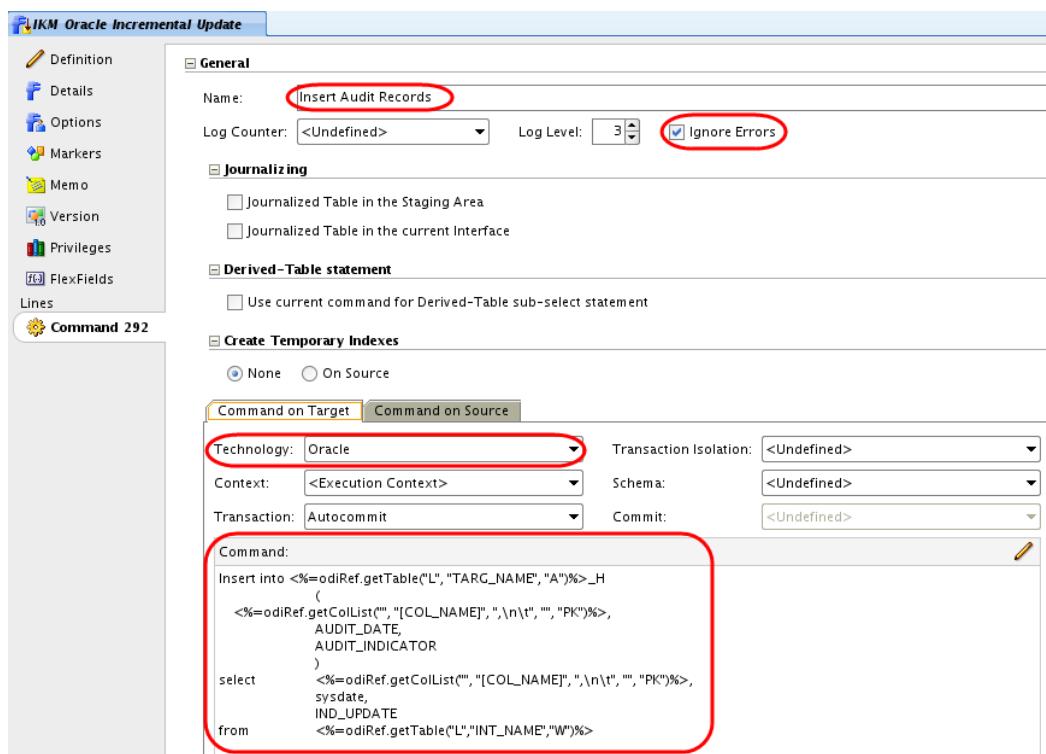
Note: These substitution methods use the following parameters:

GetTable:

- “L”: Local naming convention. For example, in Oracle that would be schema.table (versus “R” for remote: schema.table@server).
- “A”: Automatic. It enables ODI to determine which physical schema to use (the Data schema [“D”] or the Staging schema [“W”])

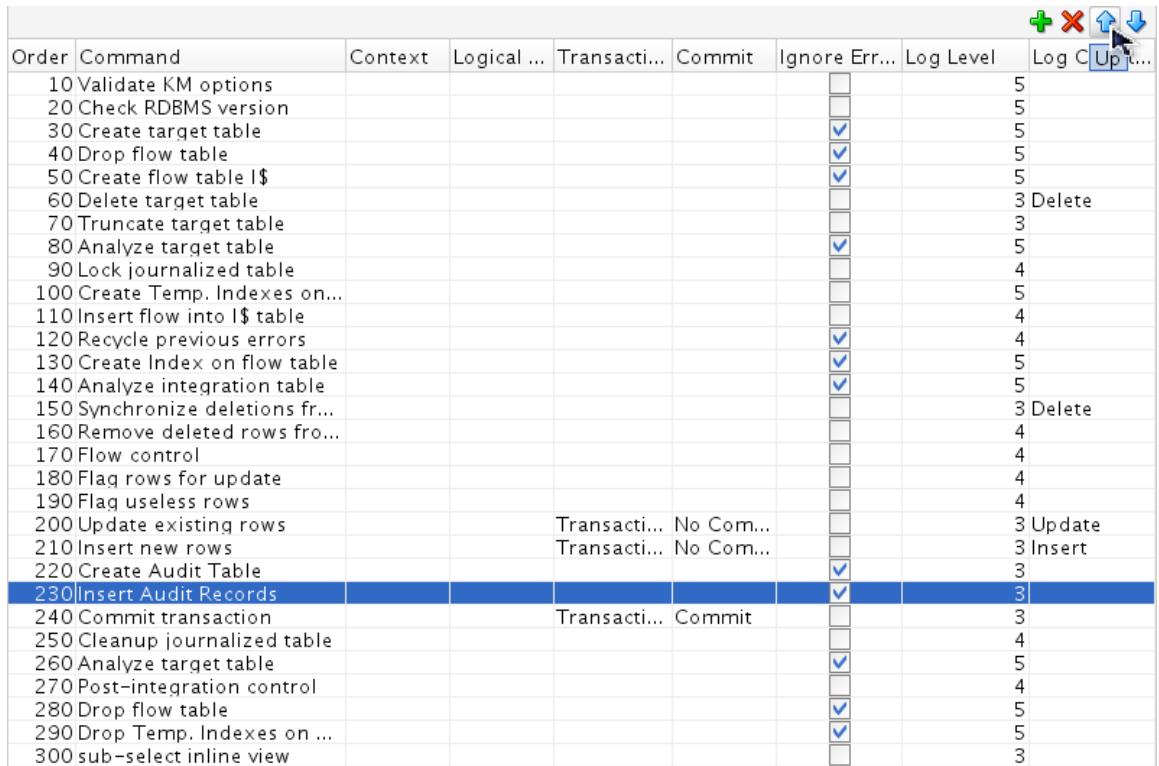
getColList:

- Notice the “PK” parameter. If it is used, only the columns that are part of the primary key are included.



7) Click *Details* tab. Scroll down and select the *Insert Audit Records* command.

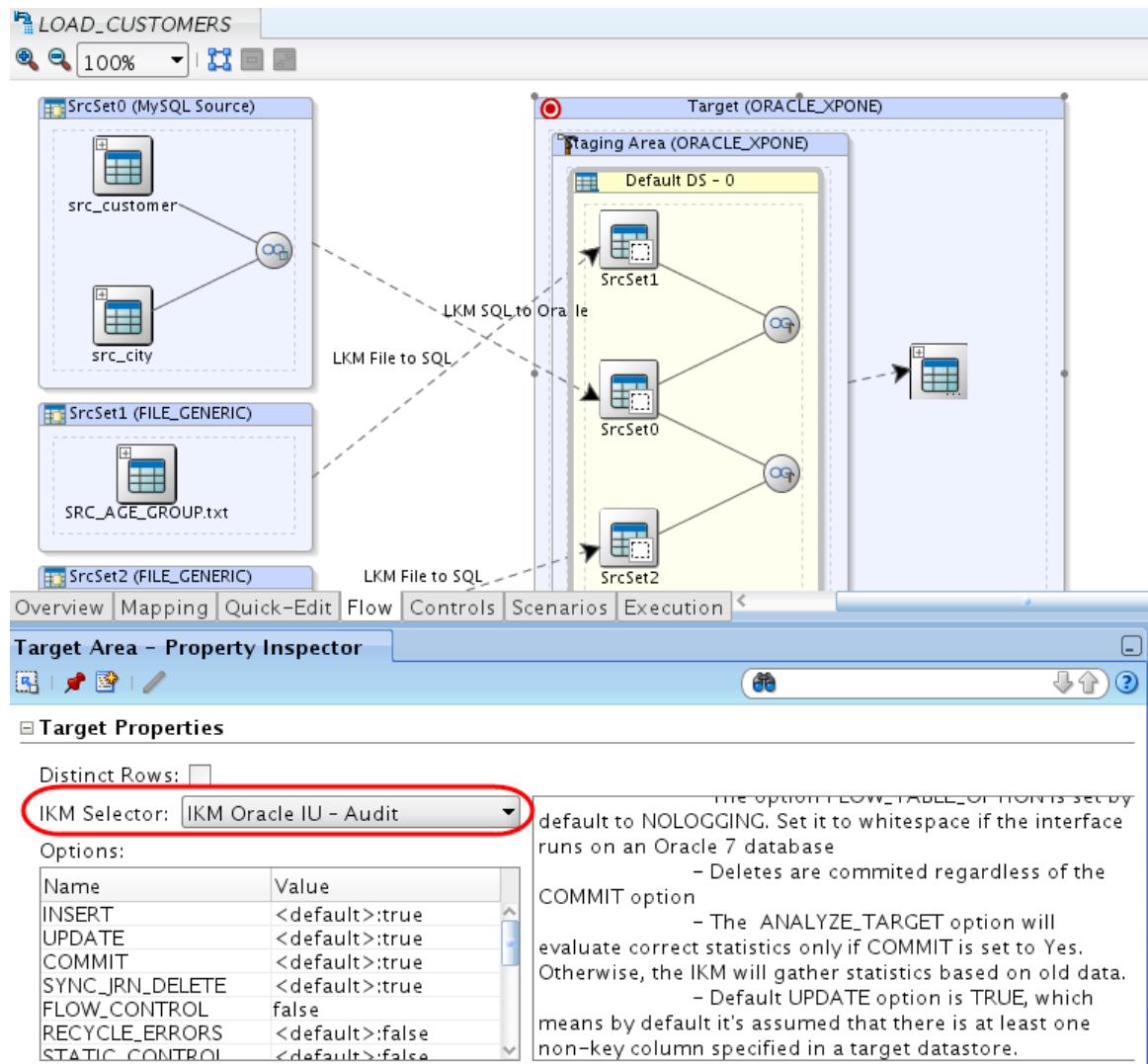
Click the icon several times to move the command up and place it right after the *Create Audit Table* step, as shown below. Click *Save*, and then close the tab.



The screenshot shows a table with columns: Order, Command, Context, Logical ..., Transacti..., Commit, Ignore Err..., Log Level, and Log C... Up.... The table lists various database operations from 10 to 300. The 'Log C... Up...' column contains numerical values (e.g., 5, 3, 4) and some text entries like 'Delete'. A blue selection bar highlights row 230, 'Insert Audit Records'. The toolbar at the top right includes icons for add (+), delete (x), and up/down arrows.

Order	Command	Context	Logical ...	Transacti...	Commit	Ignore Err...	Log Level	Log C... Up...
10	Validate KM options					<input type="checkbox"/>	5	
20	Check RDBMS version					<input checked="" type="checkbox"/>	5	
30	Create target table					<input checked="" type="checkbox"/>	5	
40	Drop flow table					<input checked="" type="checkbox"/>	5	
50	Create flow table I\$					<input checked="" type="checkbox"/>	5	
60	Delete target table					<input checked="" type="checkbox"/>	3 Delete	
70	Truncate target table					<input type="checkbox"/>	3	
80	Analyze target table					<input checked="" type="checkbox"/>	5	
90	Lock journalized table					<input type="checkbox"/>	4	
100	Create Temp. Indexes on...					<input type="checkbox"/>	5	
110	Insert flow into I\$ table					<input type="checkbox"/>	4	
120	Recycle previous errors					<input checked="" type="checkbox"/>	4	
130	Create Index on flow table					<input checked="" type="checkbox"/>	5	
140	Analyze integration table					<input checked="" type="checkbox"/>	5	
150	Synchronize deletions fr...					<input type="checkbox"/>	3 Delete	
160	Remove deleted rows fro...					<input type="checkbox"/>	4	
170	Flow control					<input type="checkbox"/>	4	
180	Flag rows for update					<input type="checkbox"/>	4	
190	Flag useless rows					<input type="checkbox"/>	4	
200	Update existing rows			Transacti...	No Com...	<input type="checkbox"/>	3 Update	
210	Insert new rows			Transacti...	No Com...	<input type="checkbox"/>	3 Insert	
220	Create Audit Table					<input checked="" type="checkbox"/>	3	
230	Insert Audit Records					<input checked="" type="checkbox"/>	3	
240	Commit transaction			Transacti...	Commit	<input type="checkbox"/>	3	
250	Cleanup journalized table					<input type="checkbox"/>	4	
260	Analyze target table					<input checked="" type="checkbox"/>	5	
270	Post-integration control					<input type="checkbox"/>	4	
280	Drop flow table					<input checked="" type="checkbox"/>	5	
290	Drop Temp. Indexes on ...					<input checked="" type="checkbox"/>	5	
300	sub-select inline view					<input type="checkbox"/>	3	

- 8) Verify that your new knowledge module *IKM Oracle UI - Audit* appears in the Knowledge Modules tree.
- 9) Modify the *LOAD_CUSTOMERS* interface to be executed with your newly created knowledge module. Change IKM entry to use *IKM Oracle UI-Audit* in the Flow tab. Make sure that in the IKM Selection section, *FLOW_CONTROL* and *STATIC_CONTROL* options are set to false. Save and execute the Interface.



10) In Operator, verify that the audit table was created and populated.

The screenshot shows the SAP Data Services Operator interface. The top navigation bar includes tabs for 'Designer', 'Topology', and 'Operator', with 'Operator' being the active tab. Below the tabs is a toolbar with icons for search, filter, and refresh, followed by a dropdown menu set to '5'. The main area is titled 'Session List' and displays a hierarchical log of database operations. The log entries are numbered from 1 to 38, with some steps circled in red at the bottom. The entries include various database actions such as loading data, creating tables, and inserting audit records. The log ends with step 38, which is circled in red, indicating the creation of the audit table.

- 1 - Copy of Customer Lesson11 - Jul 22, 2010 12:05:51 PM
- + 1 - Loading - SrcSet2 - Drop work table
- + 2 - Loading - SrcSet2 - Create work table
- + 3 - Loading - SrcSet2 - Load data
- + 5 - Loading - SrcSet0 - Drop work table
- + 6 - Loading - SrcSet0 - Create work table
- + 7 - Loading - SrcSet0 - Load data
- + 9 - Loading - SrcSet1 - Drop work table
- + 10 - Loading - SrcSet1 - Create work table
- + 11 - Loading - SrcSet1 - Load data
- + 12 - Loading - SrcSet1 - Analyze work table
- + 14 - Integration - Copy of Customer Lesson11 - Drop flowtable
- + 15 - Integration - Copy of Customer Lesson11 - Create flowtable !\$
- + 16 - Integration - Copy of Customer Lesson11 - Delete target table
- + 17 - Integration - Copy of Customer Lesson11 - Insert flow into !\$ table
- + 18 - Integration - Copy of Customer Lesson11 - Analyze integration table
- + 19 - Integration - Copy of Customer Lesson11 - Remove deleted rows from flowtable
- + 20 - Control - CUSTOMER - create check table
- + 21 - Control - CUSTOMER - delete previous check sum
- + 22 - Control - CUSTOMER - create error table
- + 23 - Control - CUSTOMER - delete previous errors
- + 24 - Control - CUSTOMER - insert PK errors
- + 25 - Control - CUSTOMER - insert FK errors
- + 26 - Control - CUSTOMER - insert CK errors
- + 27 - Control - CUSTOMER - insert Not Null errors
- + 28 - Control - CUSTOMER - insert Not Null errors
- + 29 - Control - CUSTOMER - delete errors from controlled table
- + 30 - Control - CUSTOMER - insert check sum into check table
- + 31 - Integration - Copy of Customer Lesson11 - Create Unique Index on flowtable
- + 32 - Integration - Copy of Customer Lesson11 - Flag rows for update
- + 33 - Integration - Copy of Customer Lesson11 - Update existing rows
- + 34 - Integration - Copy of Customer Lesson11 - Insert new rows
- + 35 - Integration - Copy of Customer Lesson11 - Create Audit Table
- + 36 - Integration - Copy of Customer Lesson11 - Insert Audit Records
- + 37 - Integration - Copy of Customer Lesson11 - Commit transaction
- + 38 - Integration - Copy of Customer Lesson11 - Drop flowtable
- 1000004 - Loading - SrcSet2 - Drop worktable

Lab 7.2: Add an Option to your KM

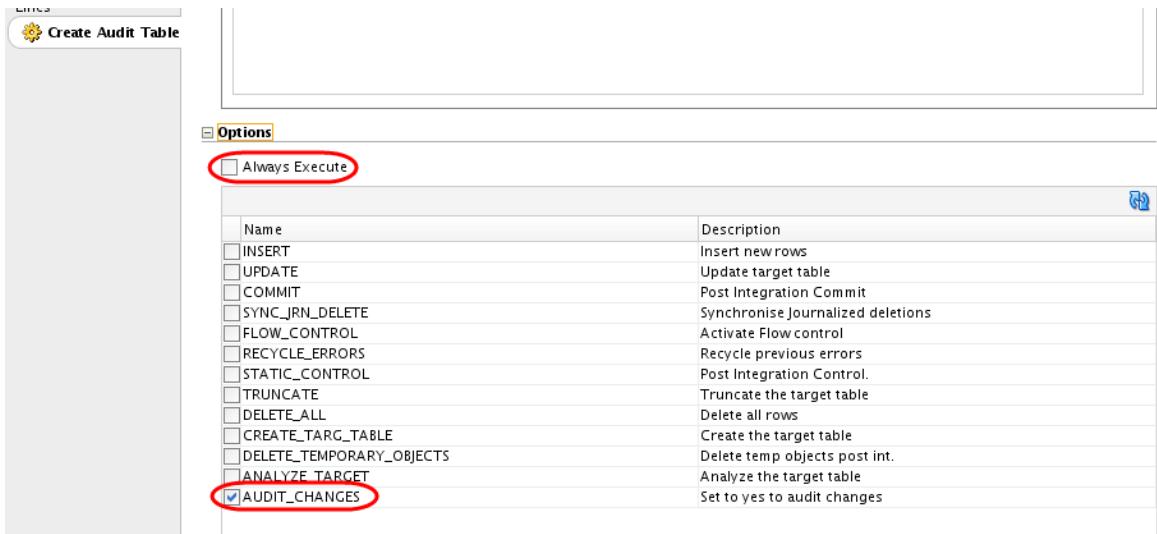
Instructions

To make your KM more user friendly, you can add an option that will let the end-user choose when he/she want to generate audits:

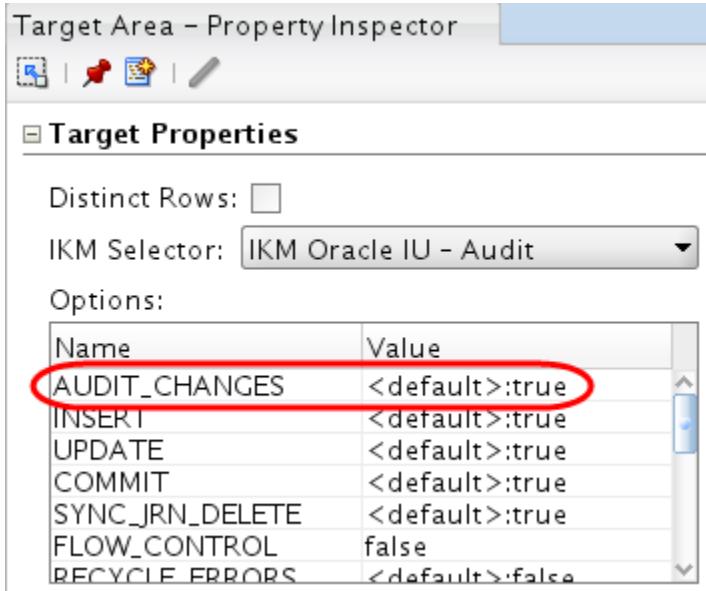
- 1) Add an option:
 - a) In the Projects tab, right-click *IKM Oracle UI – Audit*
 - b) Select *New Option*
 - c) Name the option AUDIT_CHANGES
 - d) Set the type to *Checkbox*
 - e) Set the default value to *True*
 - f) Save the Option

The screenshot shows a software interface for defining a procedure option. On the left, there's a sidebar with tabs for 'Definition', 'Markers', 'Memo', 'Privileges', and 'Version'. The 'Definition' tab is selected. On the right, there's a main panel titled 'Procedure Option [Knowledge Module: IKM Oracle IU - Audit]'. Inside this panel, there are several input fields: 'Name' (containing 'AUDIT_CHANGES'), 'Type' (containing 'Check Box'), and 'Default Value' (containing 'True'). The 'Name', 'Type', and 'Default Value' fields are each highlighted with a red oval.

- 2) Link the option to your steps. For each one of the steps you have created:
 - a) In the *Details* section of the KM, double-click your step to open it, and then scroll-down to the bottom and click *Options*
 - b) Unselect *Always Execute*
 - c) Select *AUDIT_CHANGES*



- d) Repeat for every step you created
 - e) Save and close the IKM. The execution of these steps can now be set by the end-user.
- 3) In the *Flow* tab of your Interface, click on the “*Target/Staging Area*” box and verify that the *AUDIT_CHANGES* option is set to *True*. Run the Interface, check the behavior in the Operator interface.



- 4) Change the value to *False*, run the interface again, and compare the generated code in the operator interface.

Lab 8: Variables, Functions, Procedures and Packages

Introduction

Packages: The Package is the largest unit of execution in Oracle Data Integrator. A Package is made up of a sequence of steps organized into an execution diagram. Each step can either succeed or fail its execution. Depending on the execution result (success or failure), a step can branch to another step.

Procedures: A Procedure is a set of commands that can be executed by an agent. These commands concern all technologies accessible by Oracle Data Integrator (OS, JDBC, JMS commands, etc).

A Procedure is a reusable component that allows you to group actions that do not fit in the Interface framework. Procedures should be considered only when what you need to do can't be achieved in an interface. In this case, rather than writing an external program or script, you would include the code in Oracle Data Integrator and execute it from your packages. Procedures require you to develop all your code manually, as opposed to interfaces.

Variables: A variable is an object that stores a single value. This value can be a string, a number or a date. The variable value is stored in Oracle Data Integrator. It can be used in several places in your projects, and its value can be updated at run-time.

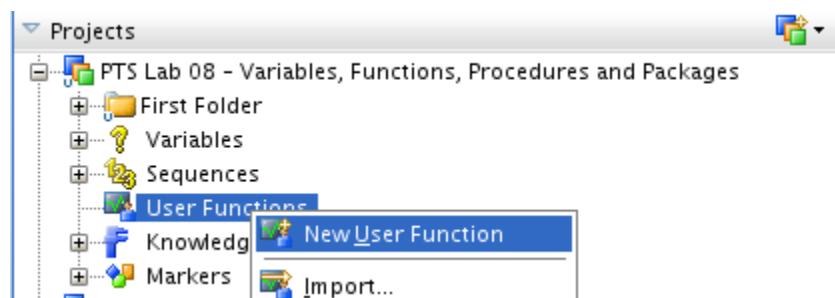
User Functions: User functions are used for defining customized functions that can be used in interfaces or procedures. It is recommended to use them in your projects when the same complex transformation pattern needs to be assigned to different datastores within different interfaces. A function can be created as a global function or in a project.

Lab 8.1: Create a User Function

Instructions

We will first create a User Function to replace the “Case When” operation we have in the *LOAD_CUSTOMERS* interface *Dear* column. Remember to use the dollar sign (\$) when using the parameters.

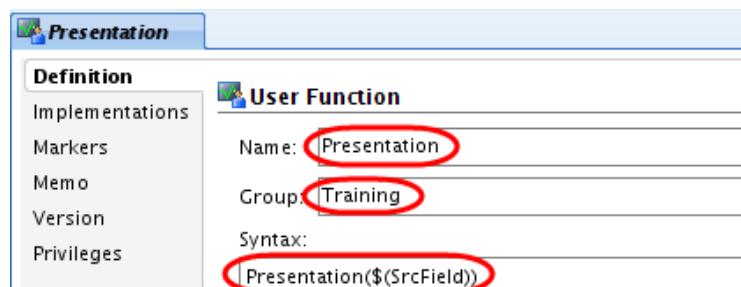
- 1) Under the *PTS Lab 08 - Variables, Functions, Procedures and Packages* project, create a new User Function



- 2) Enter a Name for the Function: *Presentation*

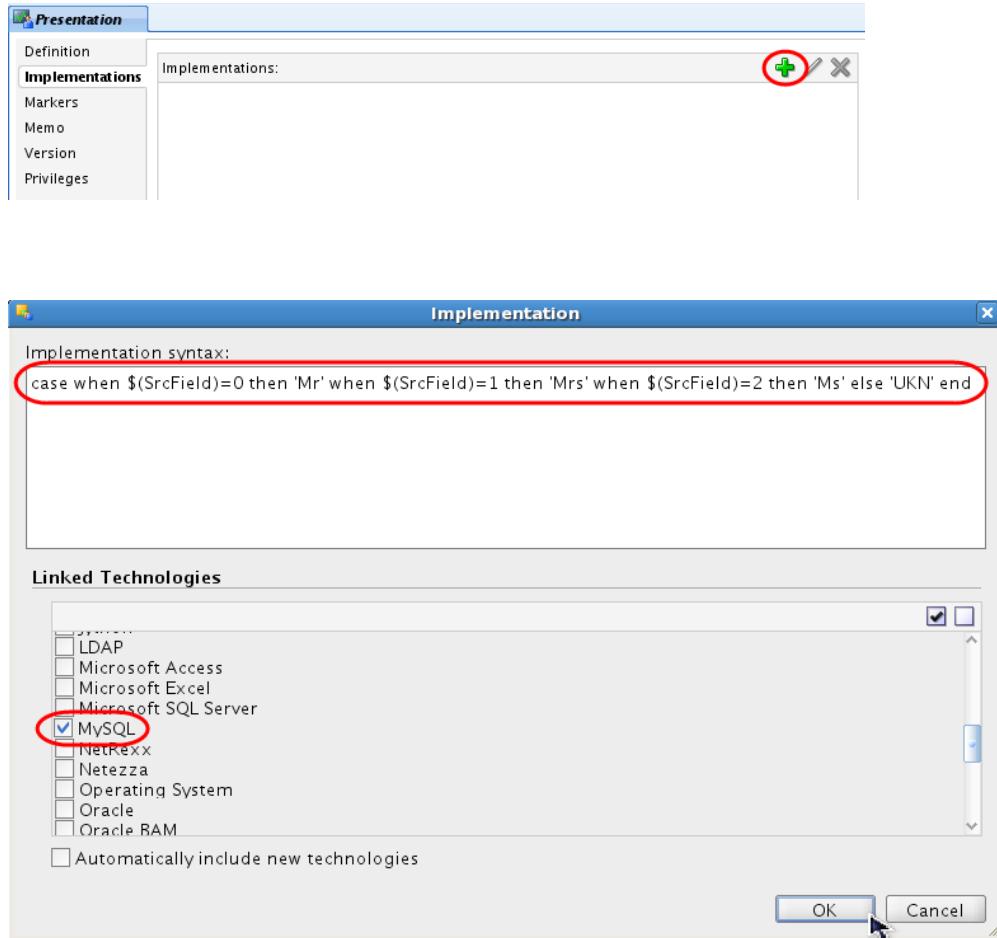
Enter a Group name for the Function: *Training*

Define the syntax for the User Function: *Presentation(\$SrcField)*

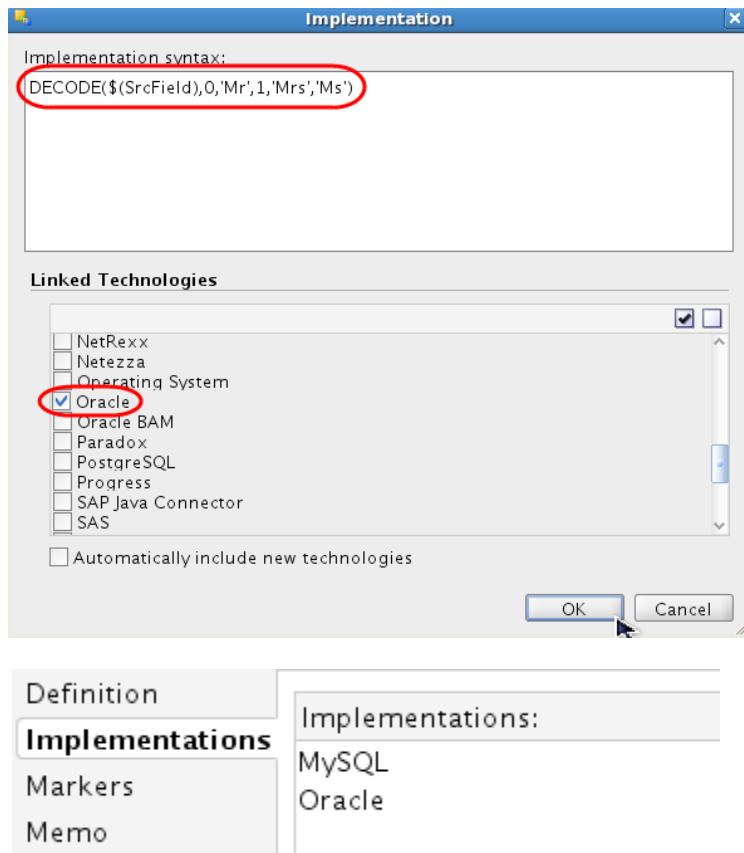


- 3) Add an Implementation and choose the technology for which you are defining the code. Type the following code for MySQL:

```
case when $(SrcField)=0 then 'Mr' when $(SrcField)=1 then 'Mrs' when $  
      (SrcField)=2 then 'Ms' else 'UKN' end
```

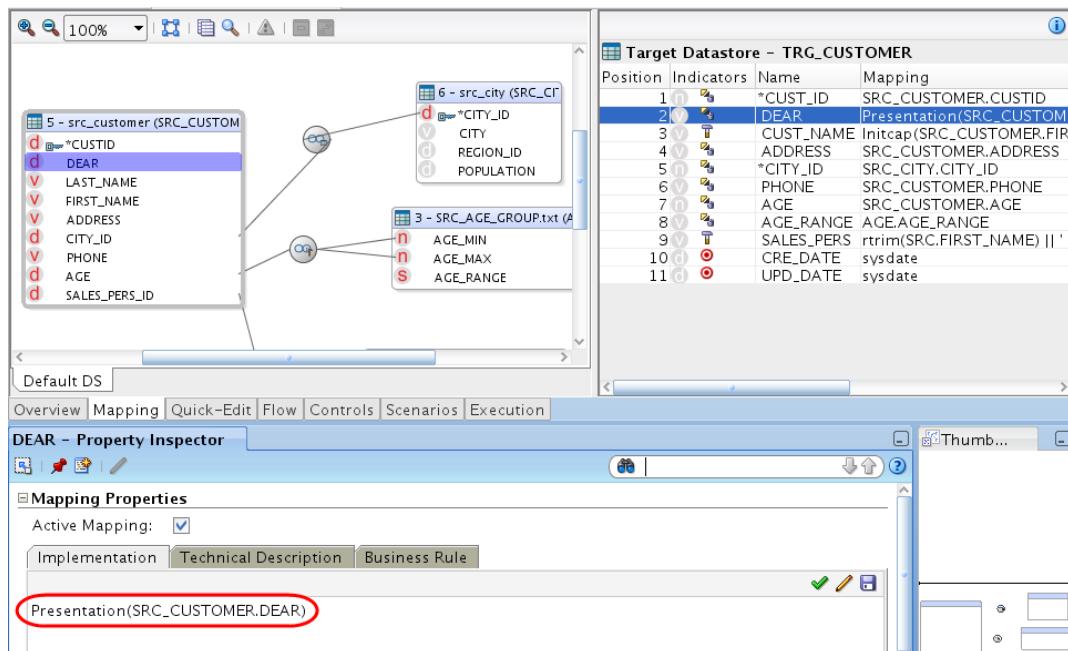


For Oracle, use the following code: DECODE\$(SrcField),0,'Mr',1,'Mrs','Ms')



- 4) Save the Function. In the *LOAD_CUSTOMERS* interface, replace the initial mapping for the *DEAR* column with the user function:

Presentation(SRC_CUSTOMER.DEAR)



- 5) Save and execute the interface. Check the generated code in the operator interface. Notice that the function has been replaced with the implementation code at runtime.

The screenshot shows the Informatica Operator interface with the following details:

- Session List:** Session 127020 - LOAD_CUSTOMERS - Aug 3, 2010 7:03:20 PM
- Source Code:** The generated SQL code includes a CASE statement for the DEAR field. The section of code is highlighted with a red oval.

```

select      SRC_CUSTOMER.CUSTID      as C1_CUST_ID,
           case when SRC_CUSTOMER.DEAR=0 then 'Mr' when
SRC_CUSTOMER.DEAR=1 then 'Mrs' when SRC_CUSTOMER.DEAR=2 then 'Ms' else 'UKN'
           end
           as C2_DEAR,
           SRC_CUSTOMER.ADDRESS      as C3_ADDRESS,
           SRC_CITY.CITY_ID          as C4_CITY_ID,
           SRC_CUSTOMER.PHONE         as C5_PHONE,
           SRC_CUSTOMER.AGE           as C6_AGE,
           SRC_CUSTOMER.LAST_NAME    as C9_LAST_NAME,
           SRC_CUSTOMER.FIRST_NAME   as C8_FIRST_NAME,
           SRC_CUSTOMER.SALES_PERS_ID as C12_SALES_PERS_ID
           sales_dev.src_customer as SRC_CUSTOMER, sales_dev.src_city as
           SRC_CITY
           where
           (1=1)
           And SRC_CITY.CITY_ID=SRC_CUSTOMER.CITY_ID

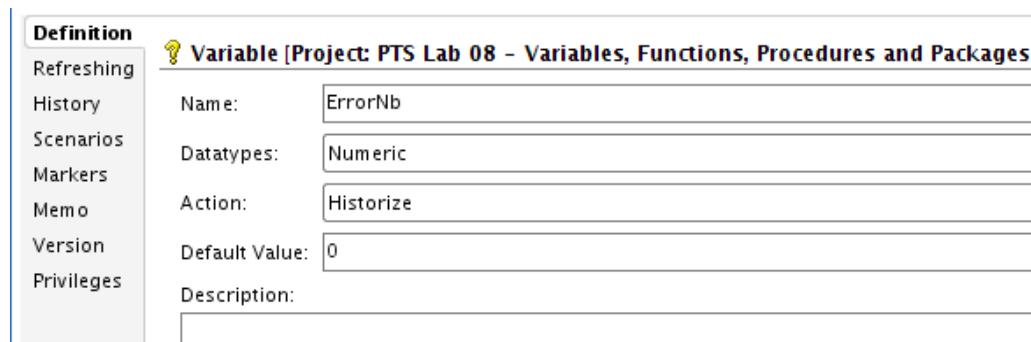
```

Lab 8.2: Variables

Instructions

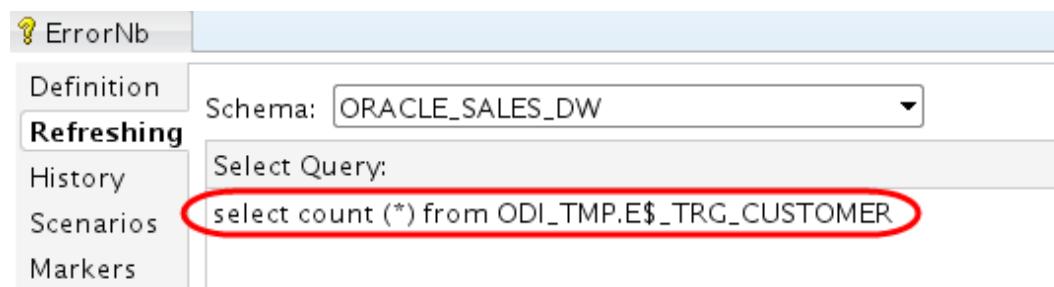
We will now use a variable to count the number of errors in the Customers data load. The objective here is to evaluate if we have identified errors, and to have ODI behave differently if we do.

- 1) Create a new variable. Variables can be local to a project or global for all projects. In this exercise, we will create a local variable. Call the variable *ErrorNb* (Datatype: Numeric, Action: Historize, Default value: 0).



In the *Refreshing* tab of the variable, select the *ORACLE_SALES_DW* schema and type the following query that will return the number of errors for the Customers table:

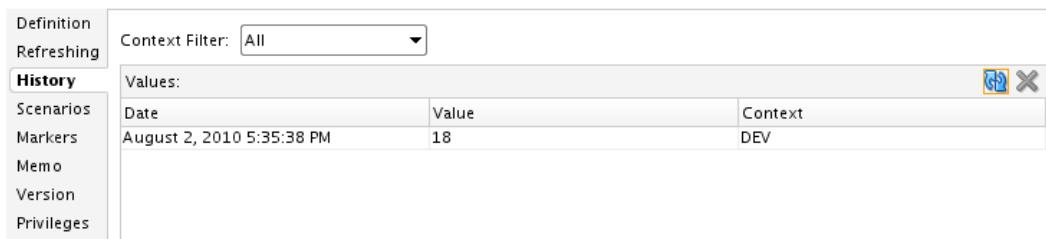
```
select count(*) from ODI_TMP.E$$_TRG_CUSTOMER
```



Note that the logical schema has been selected to point to the appropriate table. A more robust way to write this query is to use ODI substitution APIs to re-build the qualified name of the table:

```
Select count(*) from <  
%=>odiRef.getObjectName("L","E$_TRG_CUSTOMER","W")%>
```

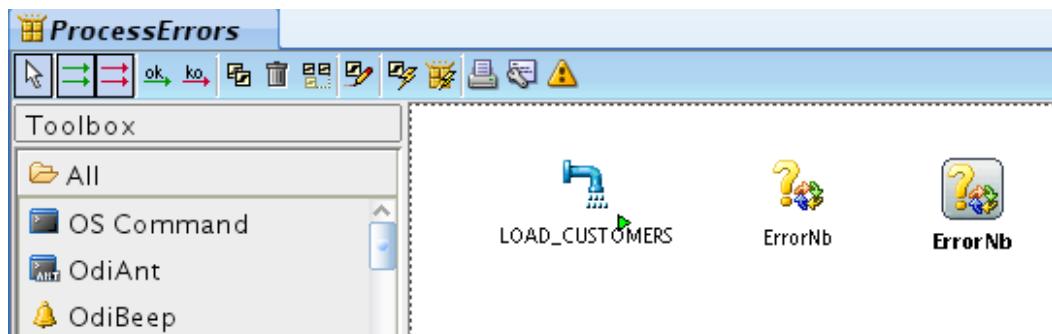
When you click the *Refresh*  button, you can follow up the processing of the queries in the Operator interface. The History tab will keep track of the values taken by the variable:



The screenshot shows the Operator interface with the 'History' tab selected. On the left, there's a sidebar with links: Definition, Refreshing, History (which is bolded), Scenarios, Markers, Memo, Version, and Privileges. The main area has a 'Context Filter' dropdown set to 'All'. Below it is a table titled 'Values:' with one row. The table has columns for 'Date', 'Value', and 'Context'. The data is as follows:

Date	Value	Context
August 2, 2010 5:35:38 PM	18	DEV

- 2) Test the variable in a package. Create a new package called *ProcessErrors*. . In the *Diagram* tab, drag and drop the LOAD_CUSTOMERS interface and the variable twice in the package *ProcessErrors*. For the first variable, set the operation to *Refresh Variable*.



Properties

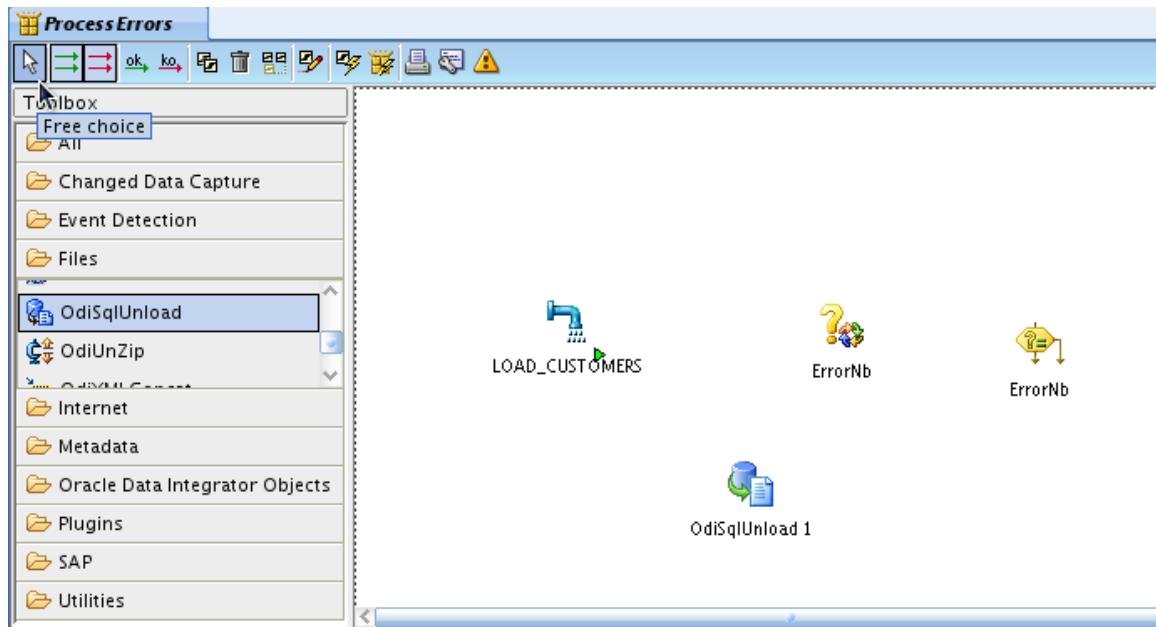
General	Advanced	Memo	Version	Privileges
Step name	Type			
ErrorNb	Refresh Variable			
Linked object	Path			
ErrorNb	[PTS Lab 08 - Variables, Functions, Procedures and Packages]			

- 3) For the second variable, set the operation to *Evaluate Variable*. Your evaluation is whether the value of *ErrorNb* is greater than zero. (Due to the way ODI processes numbers, use the value 0.0 for your comparison).

Properties

General	Advanced	Memo	Version	Privileges
Step name	Type			
ErrorNb	Evaluate Variable			
Linked object	Path			
ErrorNb	[PTS Lab 08 - Variables, Functions, Procedures and Packages]			
Operator	Value			
>	0.0			

- 4) Save these errors in a flat file (to email the contents to an administrator for instance). Use *OdiSQLUnload* to dump the contents of the error table to a csv file on the desktop. Under Files, highlight *OdiSQLUnload* and then click mouse in the diagram pane. Click Free Choice arrow icon in ToolBox to disable selection.



- 5) Fill in the information for *OdiSQLUnload* with the information below (password and user are both *cust_dw_dev*)

Properties

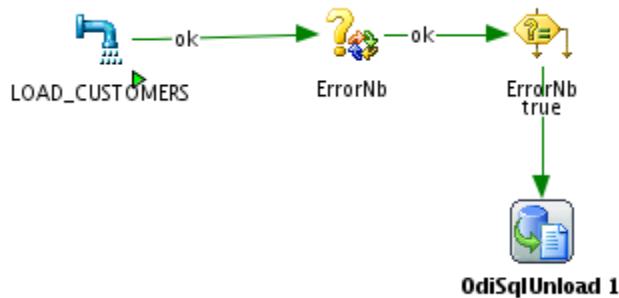
General Command Advanced Memo Version Privileges

OdiSqlUnload

Select a parameter for more information about it.

Step name	OdiSqlUnload1
Parameter	Value
Target File	/home/oracle/Desktop/errors.txt
JDBC Driver	oracle.jdbc.OracleDriver
JDBC URL	jdbc:oracle:thin:@pts.us.oracle.com:1521:XE
User	cust_dw_dev
Password	*****
File Format	Delimited
Field Separator	,
Field Separator (Hexa)	
Record Separator	\r\n
Record Separator (Hexa)	
Date Format	yyyy/MM/dd HH:mm:ss
Charset	ISO8859_1
XML Charset	ISO-8859-1
SQL Query	select * from ODI_TMP.E\$_TRG_CUSTOMER
SQL Query File	
Fetch Size	

- 6) Click the *Next Step on Success* tool icon  on the Package toolbar. Click the *LOAD_CUSTOMERS* step, then press and hold the left mouse button and move the cursor over the first *ErrorNb* variable step. Then, release the mouse button. A green arrow appears between these steps. Repeat to connect the second variable step and the *odiSqlUnload* step.



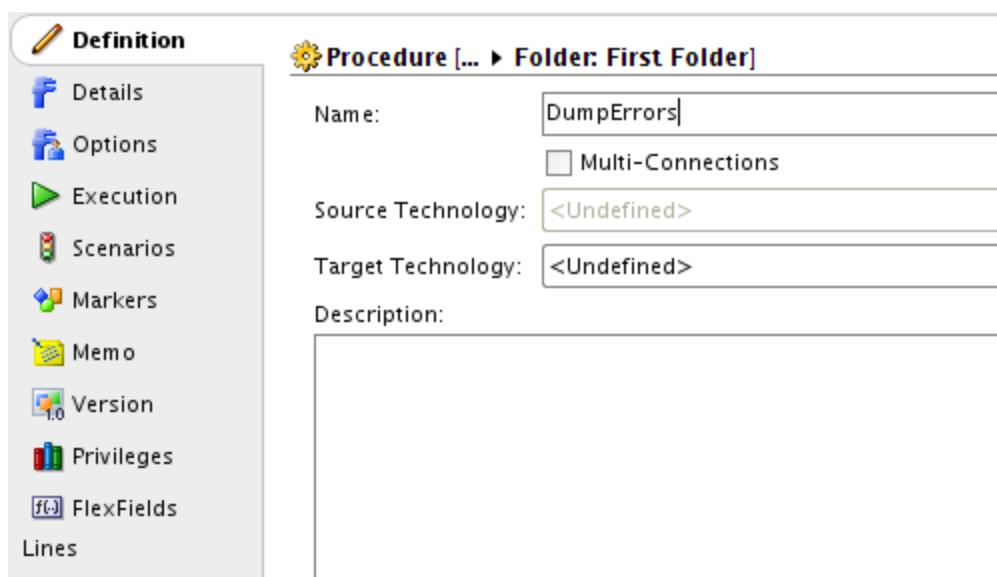
- 7) Click Save to save the package. Execute the package. Check the operator for any errors and view the errors file on the Desktop.

Lab 8.3: Procedures

Instructions

One of the big advantages of Procedures is its reusability. Oftentimes, the same steps occur over and over again in different packages. Instead of recreating those steps in each package, you can create a procedure for the step(s) and use it in multiple packages.

- 1) Create a Procedure call *DumpErrors*.



- 2) Add a step in the *Details* tab. In the *Command on Target* tab, name the step *DumpErrors*, set the Technology to *Sunopsis API*, and type the following command:

```
OdiSqlUnload -FILE=/home/oracle/Desktop/DumpErrors.txt -DRIVER=<  
%=>odiRef.getInfo("SRC_JAVA_DRIVER")%> -URL=<  
%=>odiRef.getInfo("SRC_JAVA_URL")%> -USER=<  
%=>odiRef.getInfo("SRC_USER_NAME")%> -PASS=<  
%=>odiRef.getInfo("SRC_ENCODED_PASS")%>  
  
select * from ODI_TMP.E$_TRG_CUSTOMER
```

Note that there is only one carriage return, just before the *select* key word

General

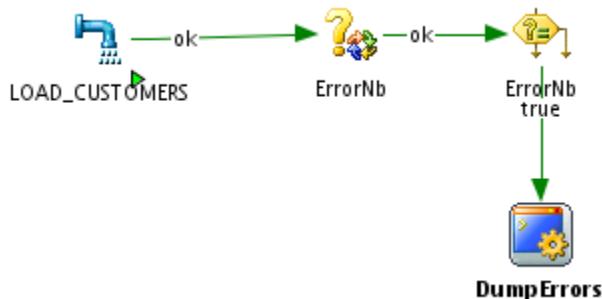
Name:	DumpErrors			
Log Counter:	<Undefined>	Log Level:	3	<input type="checkbox"/> Ignore Errors
<input checked="" type="radio"/> Command on Target <input type="radio"/> Command on Source				
Technology:	Sunopsis API	Transaction Isolation:	<Undefined>	
Context:	<Execution Context>	Schema:	<Undefined>	
Transaction:	Autocommit	Commit:	<Undefined>	
Command: <pre>OdiSqlUnload -FILE=/home/oracle/Desktop/DumpErrors.txt -DRIVER=<%=odiRef.getInfo("SRC_JAVA_DRIVER")%> -URL=<%=odiRef.getInfo("SRC_JAVA_URL")%> -USER=<%=odiRef.getInfo("SRC_USER_NAME")%> -PASS=<%=odiRef.getInfo("SRC_ENCODED_PASS")%> select * from ODI_TMP.E\$_TRG_CUSTOMER</pre>				

In the *Command on Source* tab, set the Technology to *Oracle* and the Schema to *ORACLE_SALES_DW*. Save and close.

General

Name:	DumpErrors			
Log Counter:	<Undefined>	Log Level:	3	<input type="checkbox"/> Ignore Errors
<input checked="" type="radio"/> Command on Target <input type="radio"/> Command on Source				
Technology:	Oracle	Transaction Isolation:	<Undefined>	
Context:	<Execution Context>	Schema:	ORACLE_SALES_DW	
Transaction:	Autocommit	Commit:	<Undefined>	
Command:				

- 3) Replace the *OdiSQLUnload* step in the *ProcessErrors* package with this procedure. Run the package and view the file placed on the Desktop.



Lab 9: Workflows and Scenarios

Introduction

As described in Lab 8, the **package** is the largest unit of execution in Oracle Data Integrator. A package is a workflow, made up of a sequence of steps organized into an execution diagram. Each step can either succeed or fail its execution. Depending on the execution result (success or failure), a step can branch to another step.

Packages assemble and reference other components from a project such as interfaces, procedure or variable.

A **scenario** is designed to put a source component (interface, package, procedure, variable) into production. Remember an interface, procedure, or package can be modified at any time. A scenario is created by generating code from such an object. It thus becomes frozen, but can still be executed on a number of different environments or contexts. Because a scenario is stable and ready to be run immediately, it is the preferred form for releasing objects developed in ODI.

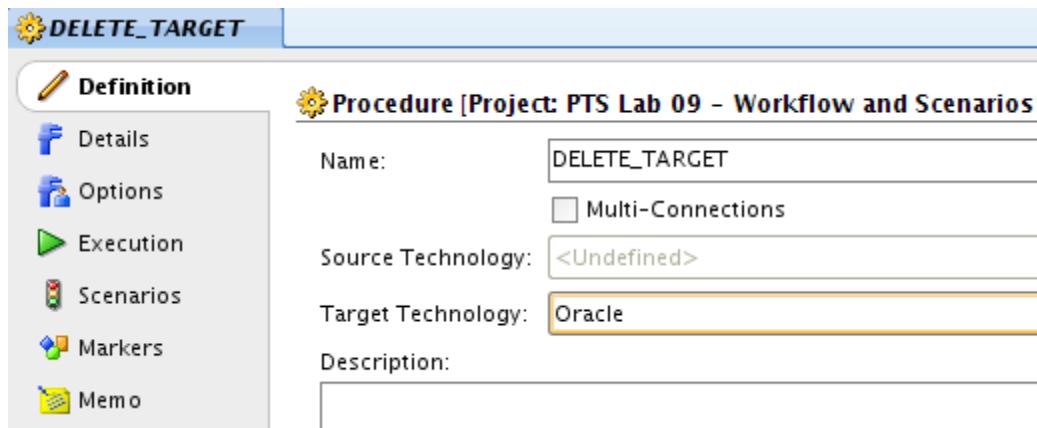
Scenarios can also be versioned. Scenarios can be launched from a command line, from the Oracle Data Integrator Studio and can be scheduled using the built-in scheduler of the run-time agent or an external scheduler.

Lab 9.1: Create a Workflow

Instructions

A common task that is performed using ODI is to create a package that executes a number of objects (Interfaces, Procedures) in the flow.

- 1) Create a procedure *DELETE_TARGET* for deleting the *TRG_CUSTOMER*, *TRG_CITY*, *TRG_REGION*, and *TRG_COUNTRY* tables.
Name: *DELETE_TARGET*
Technology: *Oracle*



2) Add a step in the Details tab and enter the following information:

Name: ***DELETE CUSTOMER***

Schema: ***ORACLE_SALES_DW***

Select *Ignore Errors* checkbox.

Command:

```
DELETE <?=odiRef.getObjectName("L", "TRG_CUSTOMER",
"ORACLE_SALES_DW", "", "D") ?>
```

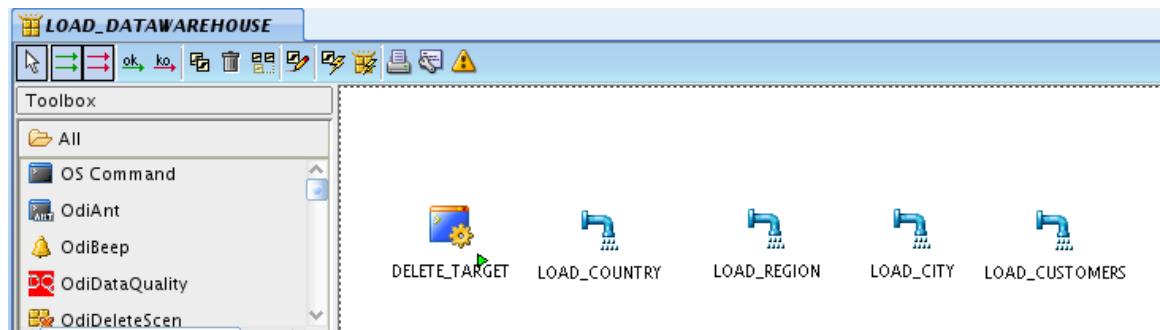
General	
Name:	DELETE CUSTOMER
Log Counter:	<Undefined>
Log Level:	3
<input checked="" type="checkbox"/> <i>Ignore Errors</i>	
<input type="radio"/> Command on Target <input checked="" type="radio"/> <i>Command on Source</i>	
Technology:	Oracle
Context:	<Execution Context>
Transaction:	Autocommit
Schema:	ORACLE_SALES_DW
Commit:	<Undefined>
Command: <i>DELETE <?=odiRef.getObjectName("L", "TRG_CUSTOMER", "ORACLE_SALES_DW", "", "D") ?></i>	

This method returns the table name with run time dependent information, such as the Oracle schema name that may vary depending on the context and topology. Select the details tab.

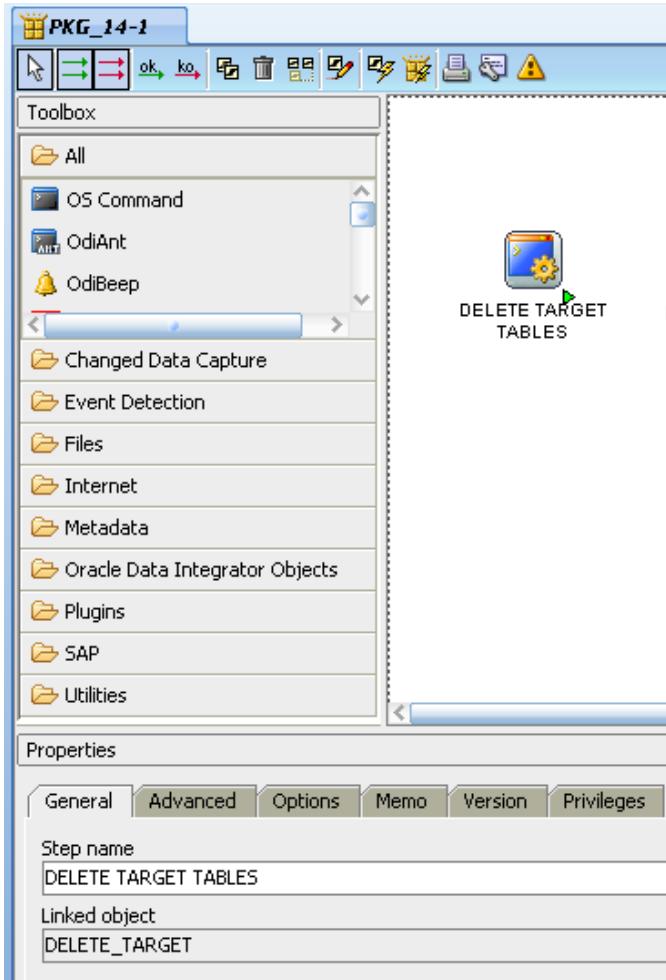
- 3) Repeat the two previous steps to create commands *DELETE CITY*, *DELETE REGION*, and *DELETE COUNTRY* that deletes the *TRG_CITY*, *TRG_REGION* and *TRG_COUNTRY* tables. Ensure that the correct schema has been chosen and that *Ignore Errors* has been selected for each.

Or...	Command	Con...	Logical Schema	Transa...	Com...	Ignore E...	Log Level	Log...
0	DELETE CUST...		ORACLE_SALES_DW			<input checked="" type="checkbox"/>	3	
10	DELETE CITY		ORACLE_SALES_DW			<input checked="" type="checkbox"/>	3	
20	DELETE REGI...		ORACLE_SALES_DW			<input checked="" type="checkbox"/>	3	
30	DELETE COU...		ORACLE_SALES_DW			<input checked="" type="checkbox"/>	3	

- 4) Save and close the Procedure.
- 5) Create a Package called *LOAD_DATAWAREHOUSE*. In the *Diagram* tab, drag the *DELETE_TARGET* procedure from the tree view. Next, drag the *LOAD_COUNTRY*, *LOAD_REGION*, *LOAD_CITY*, and *LOAD_CUSTOMERS* interfaces.



- 6) Edit Step *DELETE_TARGET* to *DELETE TARGET TABLES* as shown in the following screenshot below. In the diagram, click empty space to see the new name.



- 7) Click the *Next Step on Success* tool icon on the Package toolbar. Click the *Delete Target Tables* step. Press and hold the left mouse button and move the cursor over the *LOAD_CUSTOMERS* step. Then, release the mouse button. A green arrow appears between these steps. Repeat until all the objects are connected.



- 8) Save and execute the package. Go to Operator to verify that the package ran successfully.

Lab 9.2: Scenarios

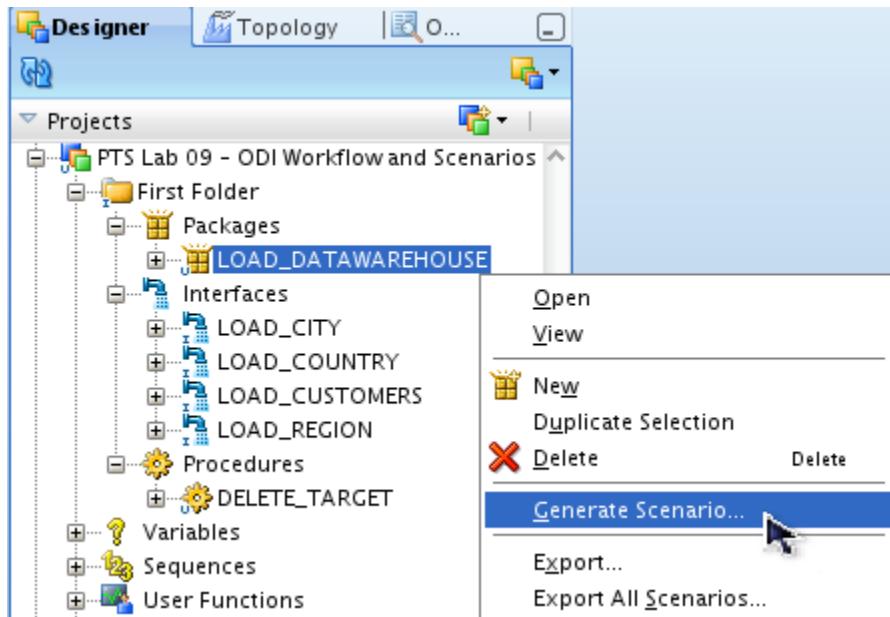
Prerequisites

If you have not already started the local agent deployed as stand alone agent, start it by following instructions provided in Lab 2.3.2 Components Access. Open new terminal window and type in the [command id > startLocalAgent](#).

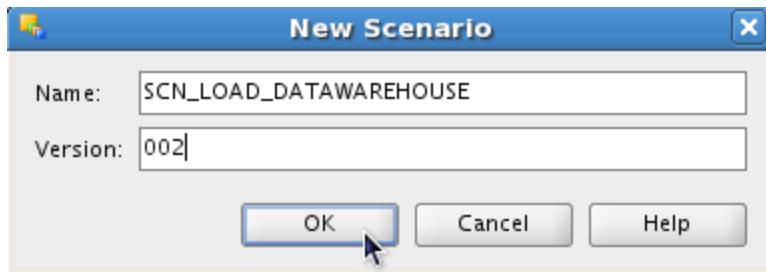
Instructions

We will now create a scenario and schedule it to run at a later time.

- 1) Right-click the just completed package and select *Generate Scenario*.

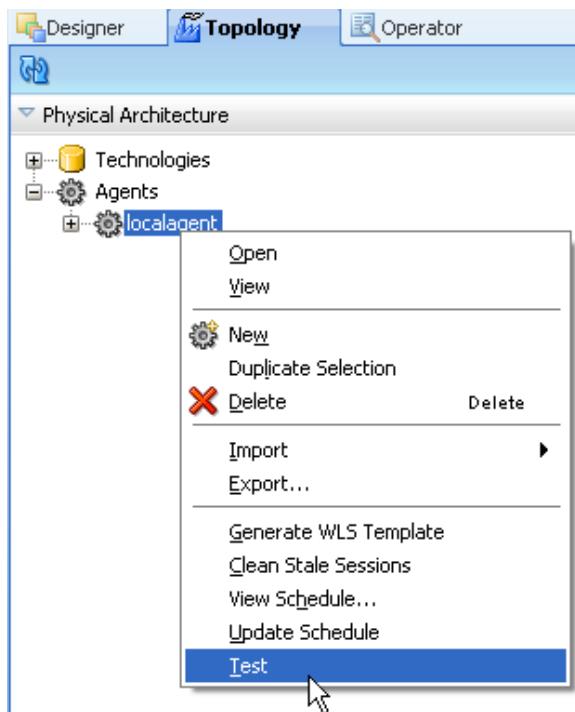


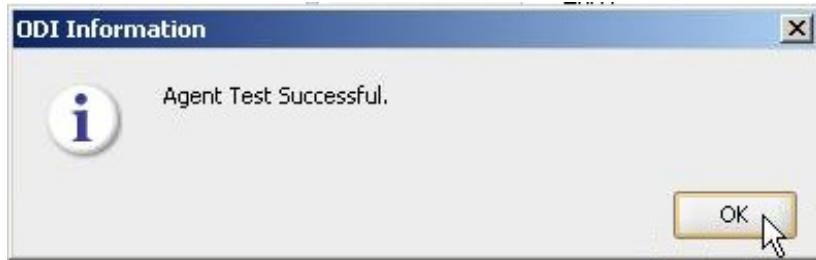
- 2) Name the scenario `SCN_LOAD_DATAWAREHOUSE`. Set the Version to 002. Click OK.



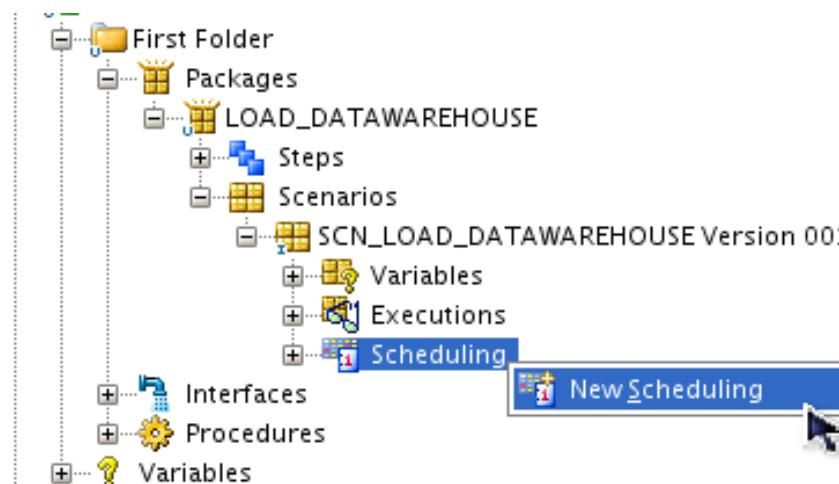
Note: The scenario has now been successfully created. You can now execute the scenario directly, use the scenario within a package, or schedule the scenario within ODI.

- 3) To schedule the scenario, go to the Topology Navigator and verify connection to the ODI agent, as shown below.





- 4) Expand the *SCN_LOAD_DATAWAREHOUSE* scenario. Expand *Packages > LOAD_DATAWAREHOUSE > Scenarios > SCN_LOAD_DATAWAREHOUSE Version 002*. Right-click *Scheduling* and select *New Scheduling*.



- 5) On the screen that follows, select the agent where the scheduled scenario will run: localagent. Set Context as Development and log level to 5. Set Execution to Simple and click the button. Set the execution time to approximately 5 minutes from the current system time as shown in the following screenshot. Click OK and save the scheduled scenario.

Scenario Scheduling: DEV / localagent

Scheduling [Scenario: SCN_LOAD_DATAWAREHOUSE / 001]

Context: Development Agent: localagent

Log Level: 5

Status

Active
 Inactive
 Active for the period:

Starting: Date: Aug 4, 2010 Time: 5:05:13 PM

Ending Date: Aug 4, 2010 Time: 5:05:13 PM

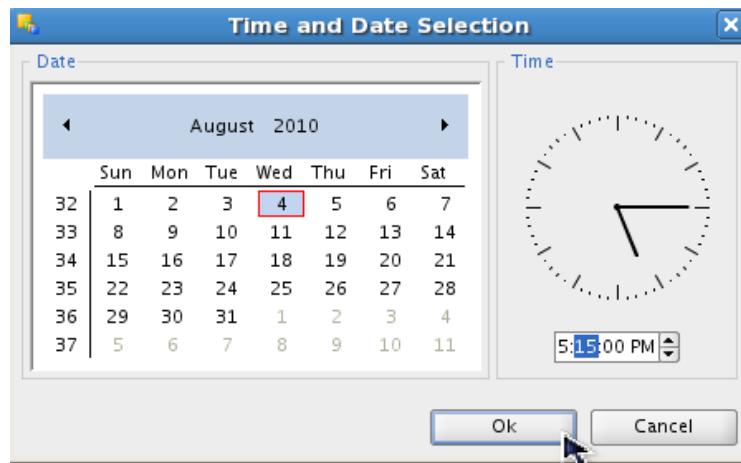
Every day between: from: 5:05:13 PM to: 5:05:13 PM

Except these days of the month

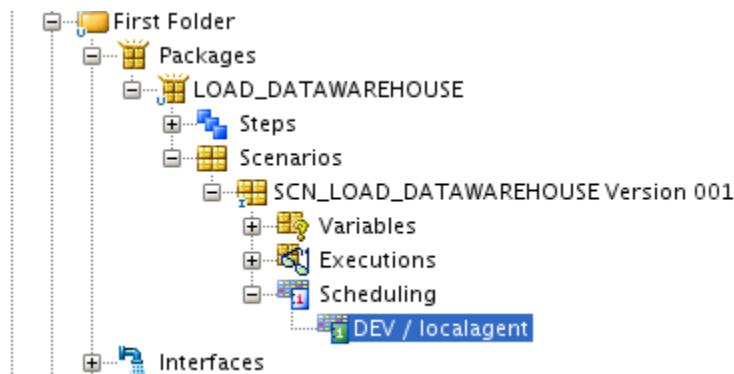
Except these days of the week: Monday Tuesday Wednesday Thursday
 Friday Saturday Sunday

Execution

On startup
 Simple Date: Aug 4, 2010 Time: 5:05:13 PM
 Hourly
 Daily
 Weekly
 Monthly (day of the month)
 Monthly (week day)
 Yearly



- 6) Expand Scheduling and verify that the DEVELOPMENT / localagent entry is now inserted under Scheduling.

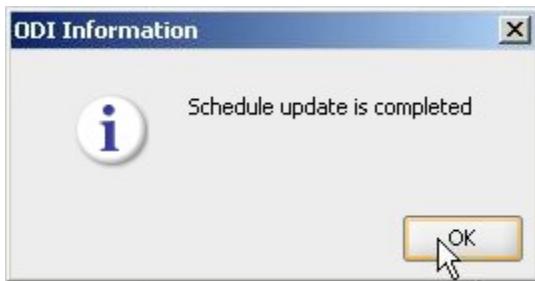


- 7) Go to Topology to review the scheduling of the Agent. In the Physical architecture, expand the Agents node, double-click *localagent*. On the *localagent* screen, click Update Schedule. On the screen that follows, click OK. Click OK again.

The screenshots show the configuration of a local agent in the Oracle Workflow environment:

- Physical Architecture:** Shows the navigation path: Physical Architecture > Agents > localagent.
- Scenario Scheduling:** A dialog titled "Scenario Scheduling: DEVELOPMENT / localagent" for the "localagent" agent. It includes tabs for View Schedule, Update Schedule, Test, and Generate WLS Template. The "Definition" tab is active, showing the agent's configuration:

Name:	localagent
Host:	localhost
Port:	20910
Web application context:	oraclediagent
Protocol:	http
Maximum number of sessions:	250
- Select Repositories:** A dialog titled "Select Repositories" showing "Refresh schedules from the following repositories:" with "WORKREP" listed. It includes a checkbox for "Select All Work Repositories" and buttons for OK, Cancel, and Help. The "OK" button is highlighted.



- 8) Click the View Schedule button. The screen that appears shows you the scheduling information.

The top screenshot shows the "Definition" tab of the "Agent" configuration. The "Name" field is set to "localagent", "Host" to "localhost", "Web application context" to "oraclediagent", and "Maximum number of sessions" to "250". The "View Schedule" button is highlighted with a mouse cursor. The bottom screenshot shows the "Schedule" tab, displaying a table with one row:

Scenario	Start time	Minimum execut...	Average execut...	Maximum executio...	Work Repository
SCN_LOAD_DATAWAREHOUSE	Aug 4, 2010 5:15:00 PM	0h 0m 0s	0h 0m 0s	0h 0m 0s	WORKREP

- 9) Open ODI Operator to verify the execution of the scheduled scenario. In ODI Operator, click the Session List tab. Wait until the scheduled execution time to view the execution results, and then click the Refresh iconExpand: Physical Agent > localagent -2> SCN_LOAD_DATAWAREHOUSE, and view the execution results for the SCN_LOAD_DATAWAREHOUSE package. Note the execution time of this scenario. That is the time that you scheduled with ODI Agent. You have now successfully scheduled a scenario with an agent scheduler.

Lab 10: Advanced Interface – Complex Transformation

Introduction

Recall in Lab 3, an integration interface consists of a set of rules that define the loading of a datastore or a temporary target structure from one or more source datastores.

After the data has been extracted and loaded, the next step is to transform the data according to a set of business rules. The data transformation may include various operations including, but not limited to filtering, sorting, aggregating, joining data, cleaning data, generating calculated data based on existing values, and validating data.

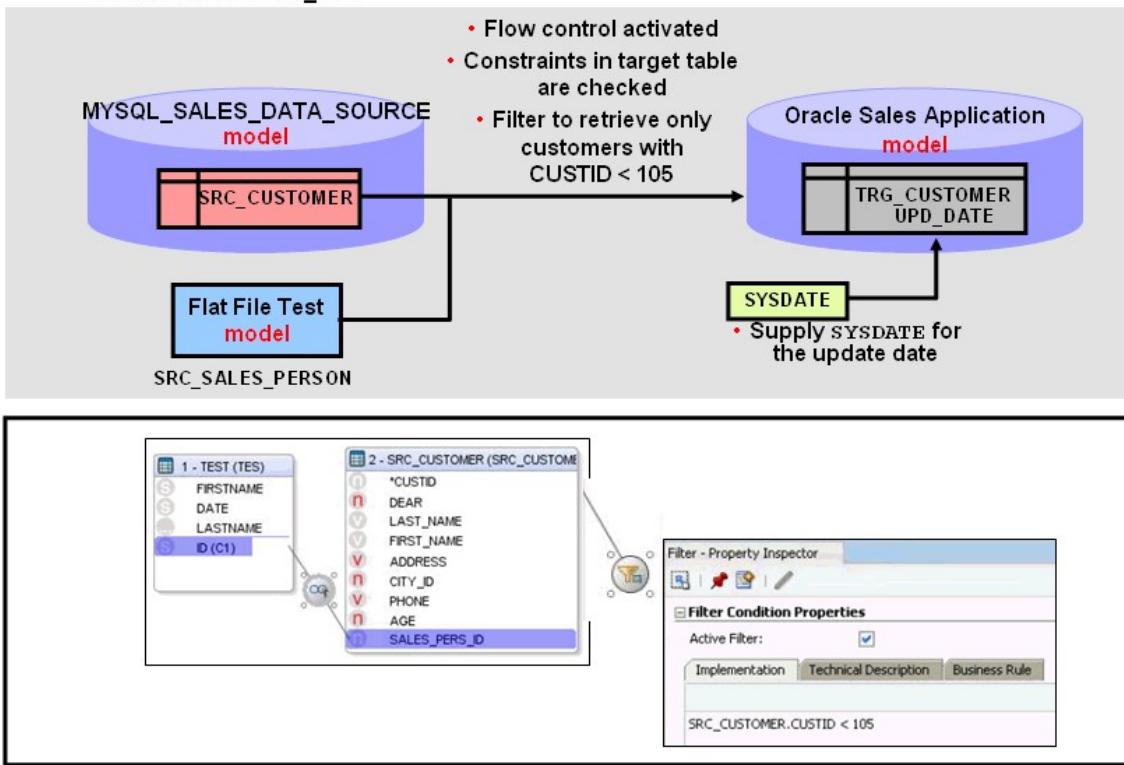
Each business rule is expressed as SQL code that is dependent on the given technology. This SQL code can then make use of other types of ODI objects such as variables and sequences (numeric variables that increment automatically each time they are used), user-defined functions, and substitution methods defined in the ODI substitution API.

Process Overview

In previous labs, you learned how to create a simple ODI Interface.

In this lab, you create a more complex interface with several sources to load the TRG_CUSTOMER datastore in the Oracle Sales Application model with the content of SRC_CUSTOMER table and the SRC_SALES_PERSON files from different models. You apply filtering to retrieve only customers with CUST_ID < 105. In addition, you populate the update date (UPD_DATE) column with System date in the mapping implementation field.

Create interface INT_10-1

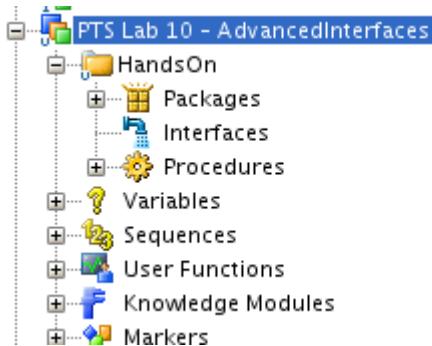


Scenario

You created the interfaces to pass data between models and perform simple ELT transformations. Now you need to create a more complex interface to load data in the target model from different sources. You also need to perform some data transformation, filtering, and provide a date when data was updated.

Instructions

- 1) Create an interface called INT_10-1 loading the TRG_CUSTOMER datastore in the Oracle Sales Application model.
 - a) In ODI Designer, open the PTS Lab 10- Advanced Interfaces, and then the Interfaces node. Right-click and select New Interface.



- b) In the Interface: NEW window, enter INT_10-1, and then click the Mapping tab.

Definition	Name: INT 10-1
Markers	Optimization Context: Development
Memo	<input type="checkbox"/> Staging Area Different From Target
Version	Oracle: ORACLE_SALES_DW
Privileges	
FlexFields	Description:

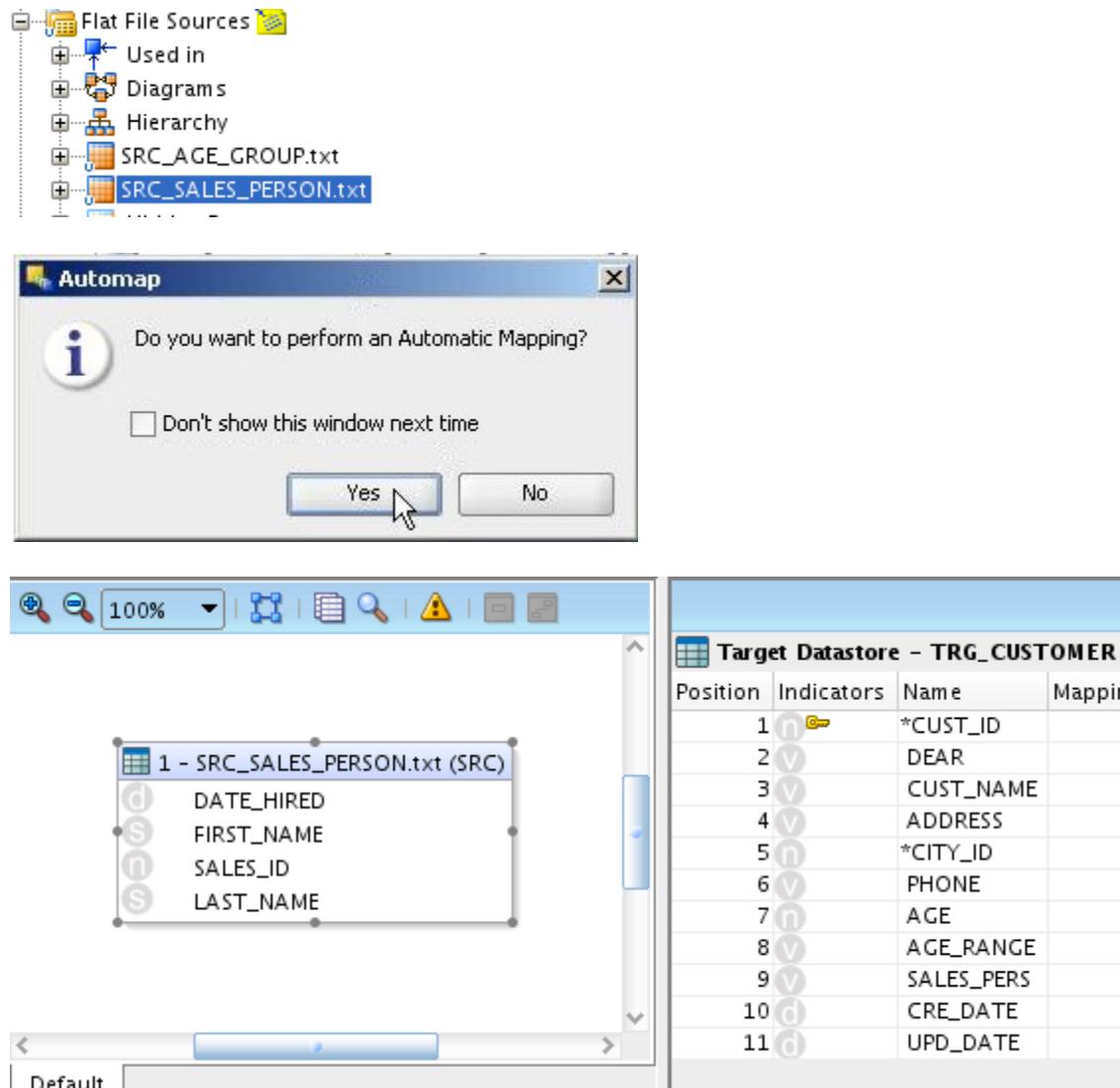
- c) In the Designer Navigator, click the Models tab, and then in the tree view expand the Oracle Sales Warehouse model. Drag the TRG_CUSTOMER datastore from the tree view to the Target Datastore zone. The datastore appears in this zone.

The screenshot shows the Oracle Data Integrator Designer Navigator interface. On the left, the 'Models' tab of the 'Designer Navigator' is selected, displaying a tree view of the 'Oracle Sales Warehouse' model. The 'Used in' node is expanded, showing various datastores like ARCHIVE_CUST, CLEANSED_CUSTOMER, etc. The 'TRG_CUSTOMER' datastore is highlighted with a blue border. On the right, the 'Target Datastore - TRG_CUSTOMER' mapping table is open. The table has columns: Position, Indicators, Name, and Mapping. The data in the table is as follows:

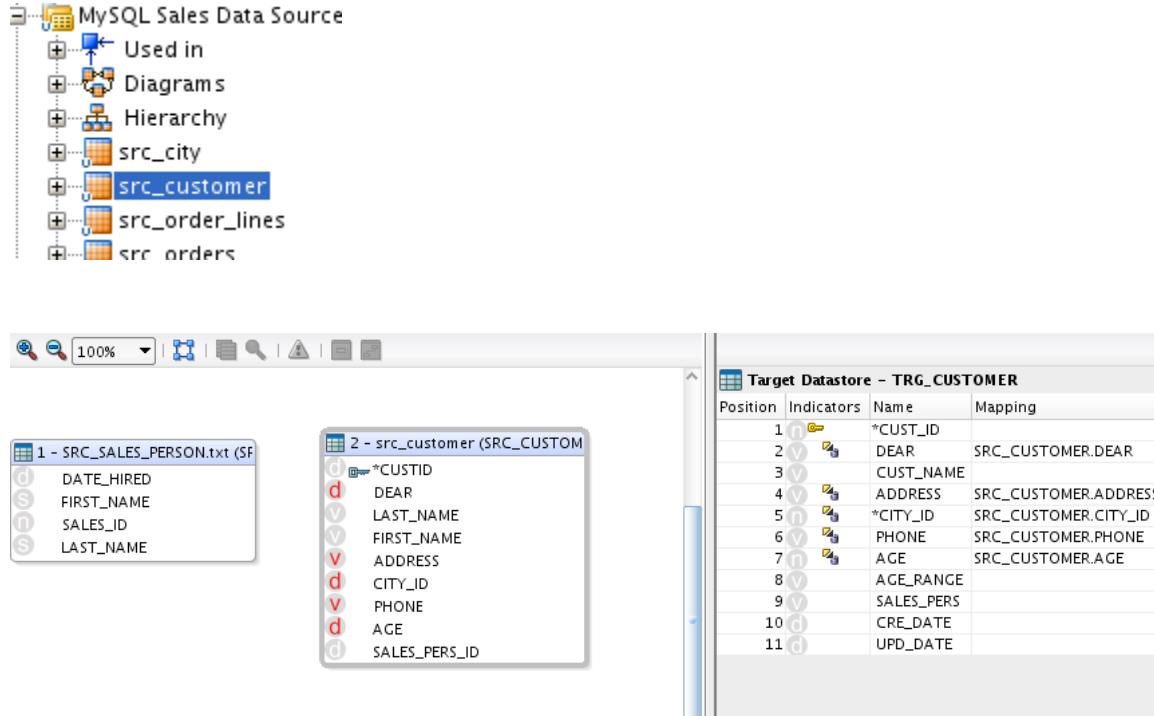
Position	Indicators	Name	Mapping
1	n	*CUST_ID	
2	v	DEAR	
3	v	CUST_NAME	
4	v	ADDRESS	
5	n	*CITY_ID	
6	v	PHONE	
7	n	AGE	
8	v	AGE_RANGE	
9	v	SALES_PERS	
10	d	CRE_DATE	
11	d	UPD_DATE	

A text overlay on the left side of the interface reads: "Drag datastores from the Designer Navigator Models tree view here to use them as sources for this dataset".

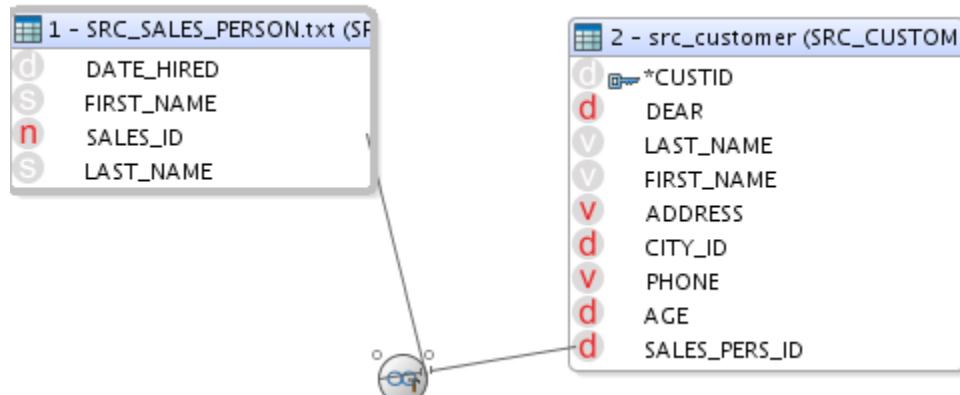
- d) In the Design Navigator Models tab, expand the Flat SRC model. Drag the SRC_SALES_PERSON.txt datastore to the Sources zone of your diagram.



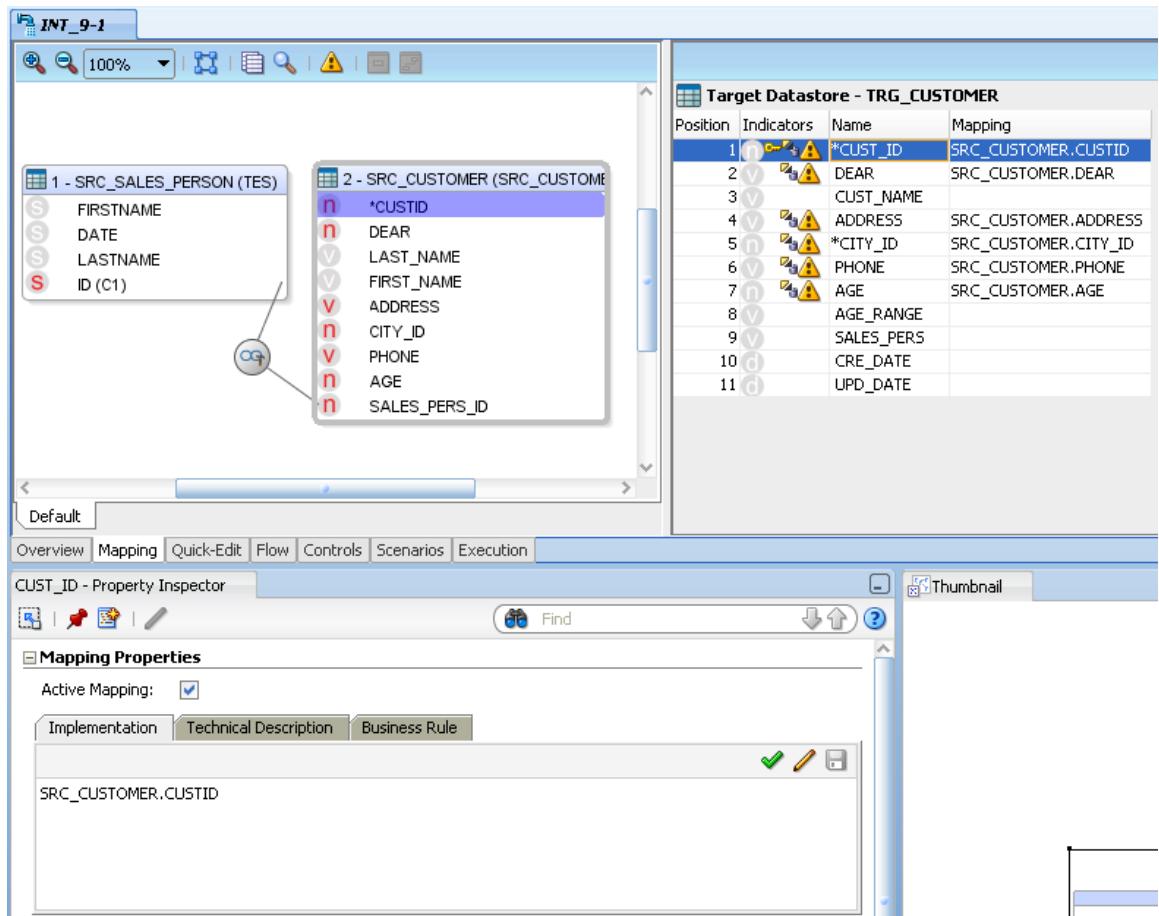
- e) In the Design Navigator Models tab, expand the MYSQL Sales Data Source model and drag the SRC_CUSTOMER datastore to the Sources zone of your diagram. Click Yes to perform Automatic mapping. Click  Arrange button. If necessary, rearrange the datastores to have the diagram as follows:



- f) Drag the SALES_PERS_ID column from the SRC_CUSTOMER source datastore on to the SALES_ID column of the SRC_SALES_PERSON datastore. A join appears between these two sources.



- g) Select and drag the CUSTID column from the SRC_CUSTOMER source datastore to the CUST_ID column in the TRG_CUSTOMER target datastore. Select the CUST_ID column in the Target Datastore zone. The Mapping Properties panel shows the mapping.



Note: Ensure that the Active Mapping check box is selected for the CUST_ID column of the target datastore.

- h) Select the CUST_NAME column in the Target Datastore zone. The Mapping Properties panel changes to show an empty mapping.

The screenshot shows the SAP Data Services interface with the following components:

- Source Zone:** Displays two source tables: "1 - SRC_SALES_PERSON (TES)" and "2 - SRC_CUSTOMER (SRC_CUSTOMER)".
- Target Zone:** Displays the "Target Datastore - TRG_CUSTOMER" table with 11 columns: *CUST_ID, DEAR, CUST_NAME, ADDRESS, *CITY_ID, PHONE, AGE, AGE_RANGE, SALES_PERS, CRE_DATE, and UPD_DATE.
- Mapping Properties Panel:** Shows the "Mapping Properties" tab for the "CUST_NAME" column. It includes sections for "Implementation", "Technical Description", and "Business Rule". The "Implementation" tab is selected, showing an empty mapping expression field.
- Status Bar:** Shows "Execute on: Source" and icons for "Source", "Staging Area", and "Target".

- i) Drag the FIRST_NAME and LAST_NAME columns from the SRC_CUSTOMER source into the Mapping: CUST_NAME Implementation Tab field, and then edit the mapping to have the following mapping expression:

```
INITCAP(SRC_CUSTOMER.FIRST_NAME) || ' ' ||  
INITCAP(SRC_CUSTOMER.LAST_NAME)
```

Note: Ensure that the *Active Mapping* check box is selected for the CUST_NAME column of the target data store.

INT_9-1

The screenshot shows the SAP Data Services interface with the following components:

- Source Datastore - 1 - SRC_SALES_PERSON (TES):**
 - Fields: FIRSTNAME, DATE, LASTNAME, ID (C1).
 - A mapping connection (indicated by a blue line) links the 'LASTNAME' field to the 'LAST_NAME' field in the target table.
- Source Datastore - 2 - SRC_CUSTOMER (SRC_CUSTOMER):**
 - Fields: *CUSTID, DEAR, LAST_NAME, FIRST_NAME, ADDRESS, CITY_ID, PHONE, AGE, SALES_PERS_ID.
- Target Datastore - TRG_CUSTOMER:**

Position	Indicators	Name	Mapping
1	n	*CUST_ID	SRC_CUSTOMER.CUSTID
2	v	DEAR	SRC_CUSTOMER.DEAR
3	v	CUST_NAME	
4	v	ADDRESS	SRC_CUSTOMER.ADDRESS
5	v	*CITY_ID	SRC_CUSTOMER.CITY_ID
6	v	PHONE	SRC_CUSTOMER.PHONE
7	n	AGE	SRC_CUSTOMER.AGE
8	v	AGE_RANGE	
9	v	SALES_PERS	
10	d	CRE_DATE	
11	d	UPD_DATE	
- Mapping Properties - CUST_NAME - Property Inspector:**
 - Active Mapping:
 - Implementation tab selected.
 - Business Rule tab content: `INITCAP(SRC_CUSTOMER.FIRST_NAME) || ' ' || INITCAP(SRC_CUSTOMER.LAST_NAME)`

If the Thumbnail window is open, close it to extend the Mapping panel. Scroll down to select the Staging Area option for the Execute on panel.

The screenshot shows the Dataedo interface with the Mapping panel extended. The left side displays the mapping diagram between two datastores:

- Source Datastore - TEST (TES)** (Left):
 - Fields: FIRSTNAME, DATE, LASTNAME, ID (C1).
 - A mapping arrow points from FIRSTNAME to LAST_NAME in the target datastore.
- Target Datastore - TRG_CUSTOMER** (Right):
 - Fields: *CUSTID, DEAR, LAST_NAME, FIRST_NAME, ADDRESS, CITY_ID, PHONE, AGE, SALES_PERS_ID.
 - Mapping table (11 rows):

Position	Indicators	Name	Mapping
1	PK	*CUST_ID	SRC_CUSTOMER.CUSTID
2	U	DEAR	SRC_CUSTOMER.DEAR
3	T	CUST_NAME	INITCAP(SRC_CUSTOMER...
4	U	ADDRESS	SRC_CUSTOMER.ADDRESS
5	PK	*CITY_ID	SRC_CUSTOMER.CITY_ID
6	U	PHONE	SRC_CUSTOMER.PHONE
7	U	AGE	SRC_CUSTOMER.AGE
8	U	AGE_RANGE	
9	U	SALES_PERS	
10	U	CRE_DATE	
11	U	UPD_DATE	

The bottom section shows the **Target Mapping - Property Inspector** for the CUST_NAME field:

- Implementation:** INITCAP(SRC_CUSTOMER.FIRST_NAME) || ' ' || INITCAP(SRC_CUSTOMER.LAST_NAME)
- Execute on:** <Undefined> (dropdown), Source (radio), **Staging Area** (radio, selected), Target (radio).
- Update:** Insert (checkbox checked), Update (checkbox checked), UD1, UD2, UD3, UD4, UD5, UD6, UD7, UD8, UD9, UD10 (checkboxes unchecked).
- Target Column:**
 - Name:** CUST_NAME (text input).
 - Key:** (checkbox unchecked).
 - Check Not Null (Flow control only):** (checkbox unchecked).
- Datatype:** Length, Scale (text inputs).

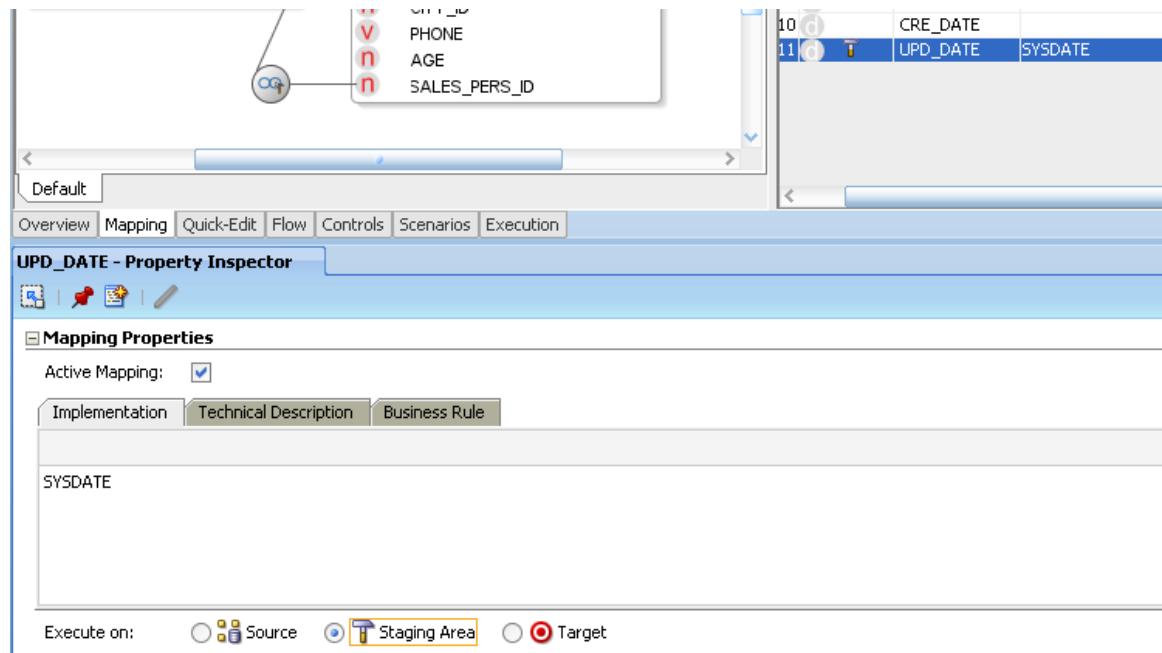
- j) Drag the LAST_NAME column from the SRC_SALES_PERSON source datastore to the SALES_PERS column in the target datastore. Click the SALES_PERS column in the target datastore, and then edit the mapping to have the following mapping expression: UPPER(SRC.LAST_NAME). Select the Staging Area option from the Execute on panel.

Target Datastore - TRG_CUSTOMER			
Position	Indicators	Name	Mapping
1	!	*CUST_ID	SRC_CUSTOMER.CUSTID
2	!	DEAR	SRC_CUSTOMER.DEAR
3	T	CUST_NAME	INITCAP(SRC_CUSTOMER...
4	!	ADDRESS	SRC_CUSTOMER.ADDRESS
5	!	*CITY_ID	SRC_CUSTOMER.CITY_ID
6	!	PHONE	SRC_CUSTOMER.PHONE
7	!	AGE	SRC_CUSTOMER.AGE
8		AGE_RANGE	
9		SALES_PERS	
10		CRE_DATE	
11		UPD_DATE	

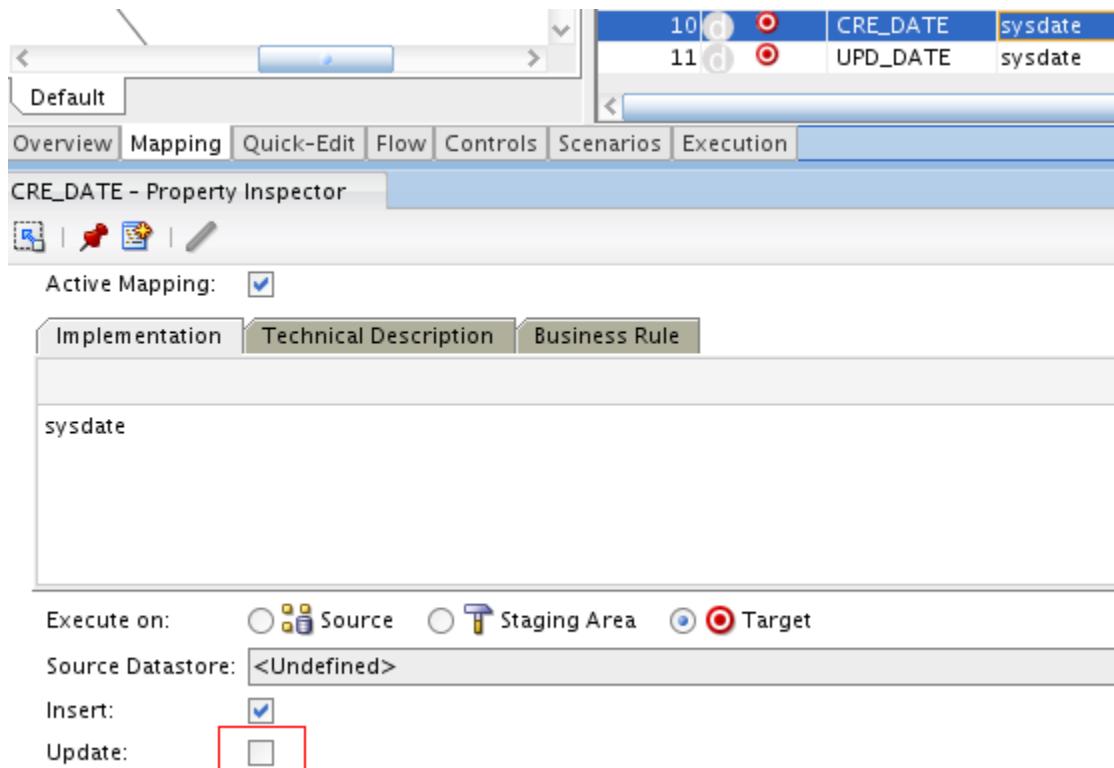
Target Datastore - TRG_CUSTOMER			
Position	Indicators	Name	Mapping
1	!	*CUST_ID	SRC_CUSTOMER.CUSTID
2	!	DEAR	SRC_CUSTOMER.DEAR
3	T	CUST_NAME	INITCAP(SRC_CUSTOMER...
4	!	ADDRESS	SRC_CUSTOMER.ADDRESS
5	!	*CITY_ID	SRC_CUSTOMER.CITY_ID
6	!	PHONE	SRC_CUSTOMER.PHONE
7	!	AGE	SRC_CUSTOMER.AGE
8		AGE_RANGE	
9	T	SALES_PERS	UPPER(SRC.LAST
10		CRE_DATE	
11		UPD_DATE	

- k) Select the UPD_DATE column in the Target Datastore zone and enter SYSDATE in the Mapping implementation field. Select the Staging Area option from the Execute on panel.

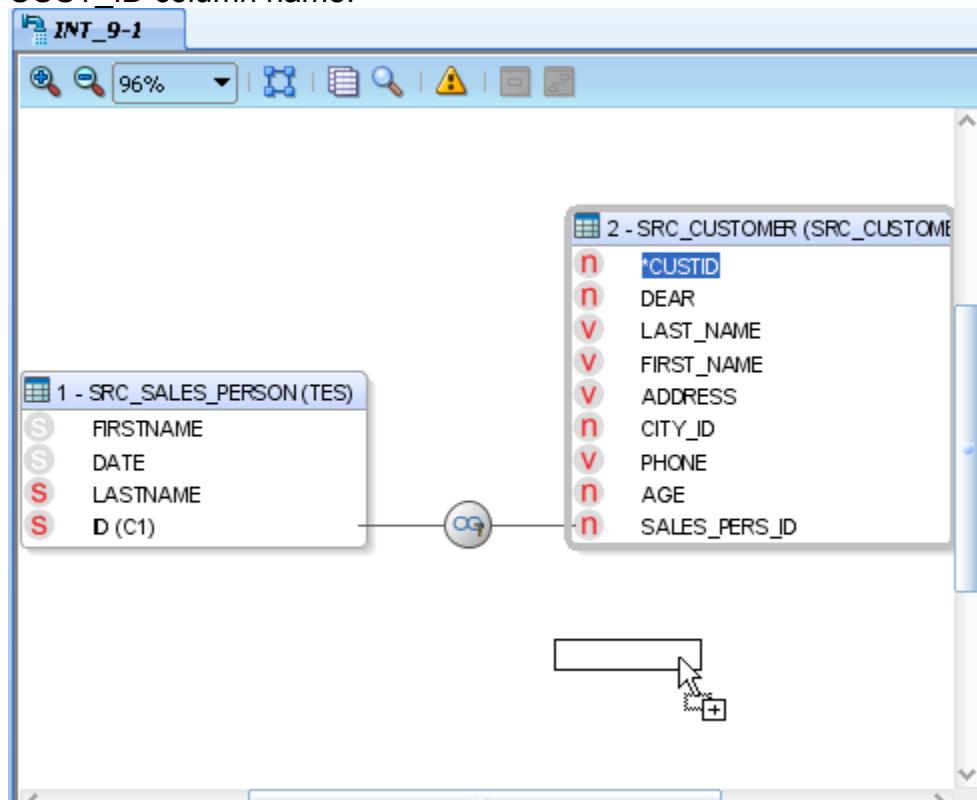
Note: Ensure that the *Active Mapping* check box is selected for the UPD_DATE column of the target data store.



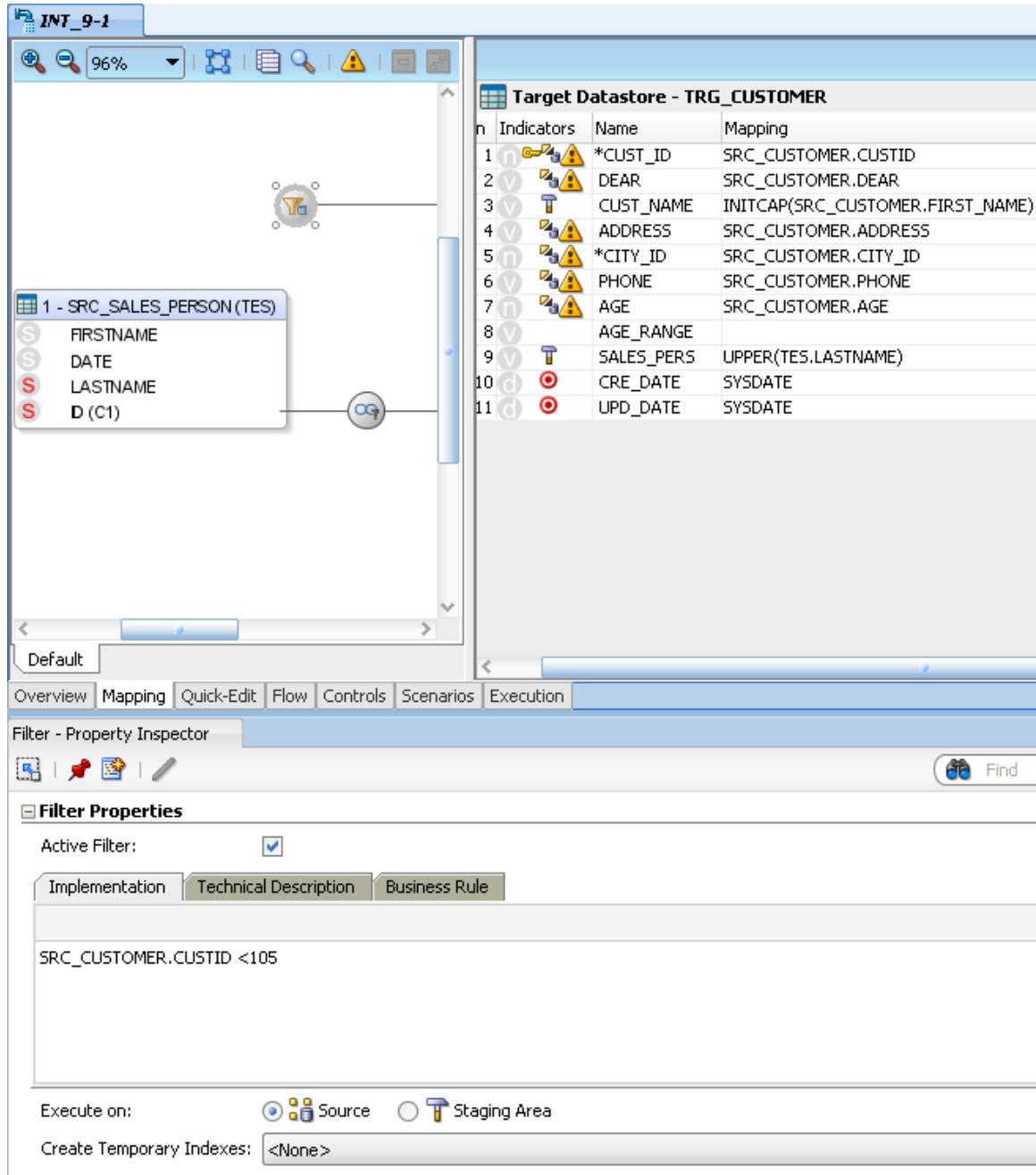
- i) Do the same for the CRE_DATE column. Since the value of CRE_DATE column should not be changed later, *deselect* the **Update** check box. Make sure the **Target** option is selected from the Execute on panel.



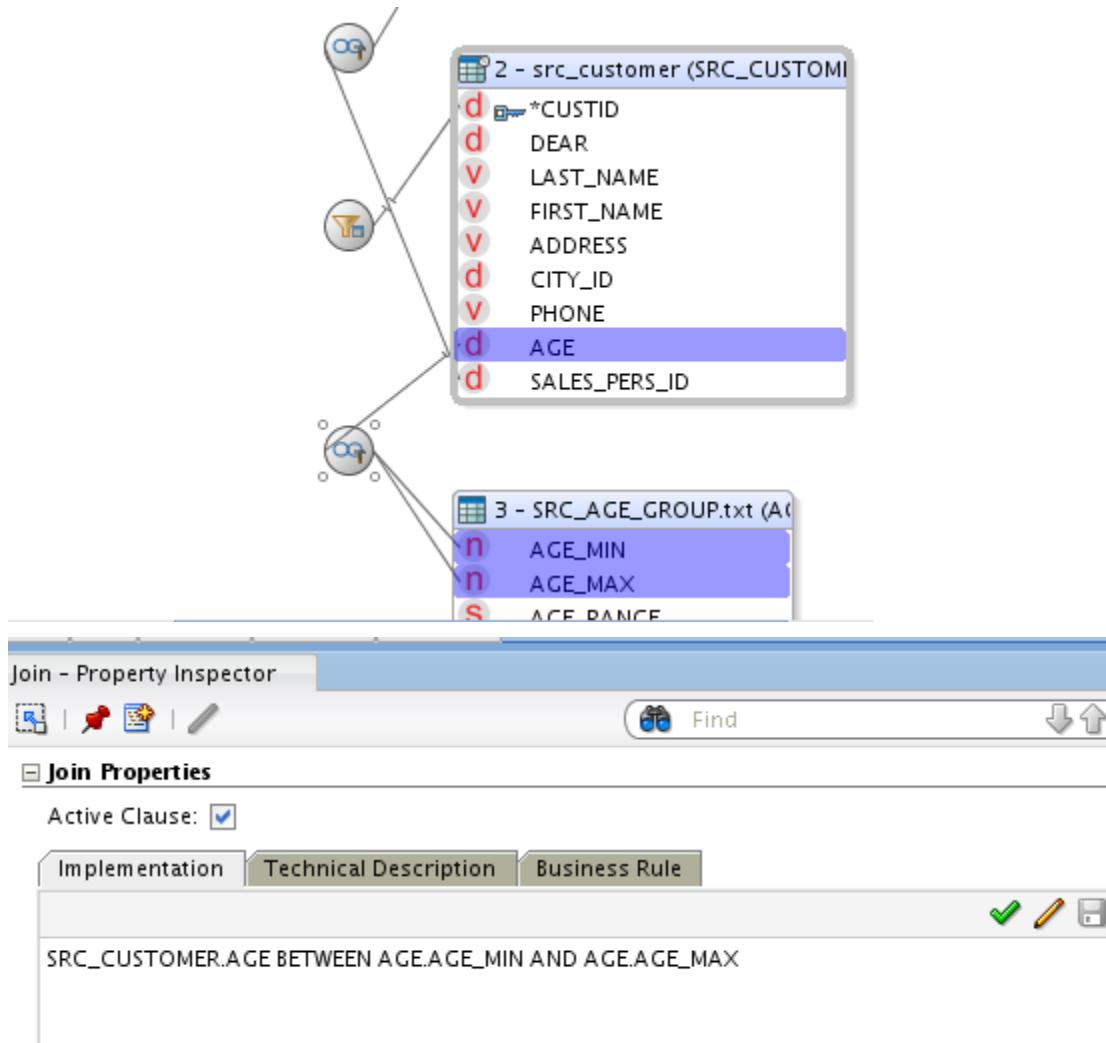
- m) In the diagram, drag the CUSTID column from the SRC_CUSTOMER source to the workbench (the gray background). A filter appears with the CUST_ID column name.



- n) Edit the filter expression to have SRC_CUSTOMER.CUSTID < 105. Scroll down and make sure that the Source option from the Execute on panel is selected.



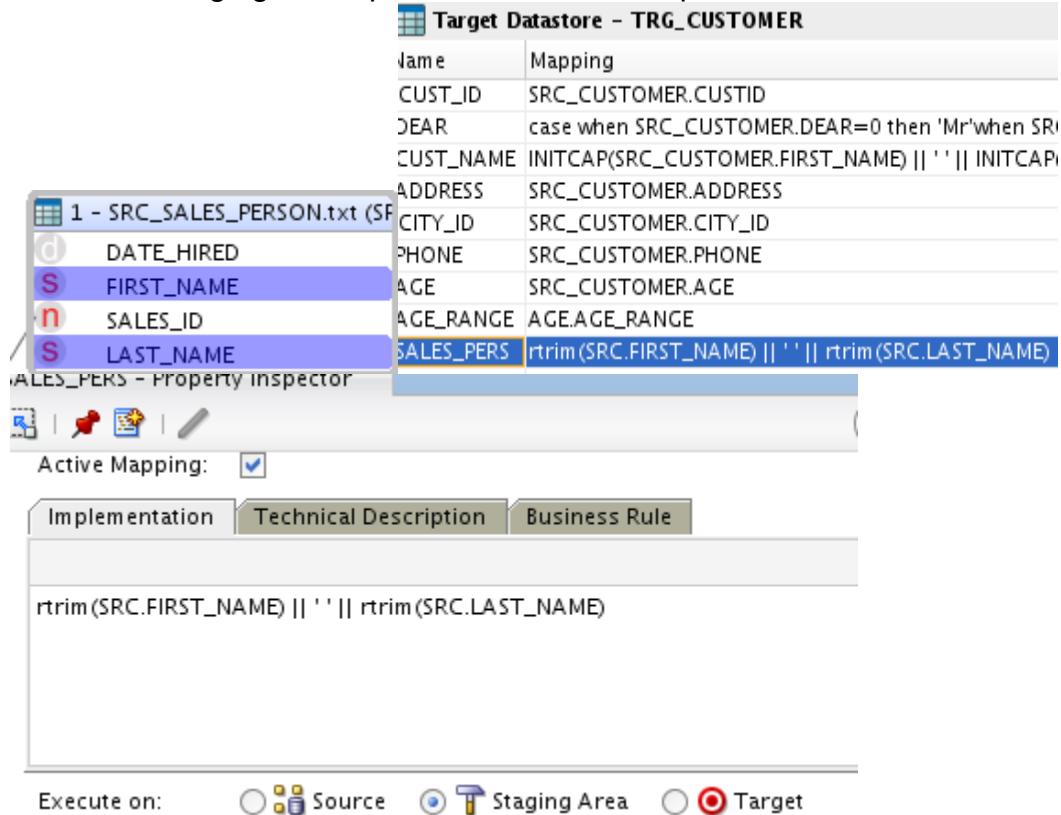
- o) In the Design Navigator Models tab, expand the Flat Files Sources model. Drag the SRC_AGE_GROUP.txt datastore to the Sources zone of your diagram. Join SRC_CUSTOMER with SRC_AGE_GROUP.txt with the AGE between the AGE_MIN and the AGE_MAX.



- p) Drag the AGE_RANGE from source SRC_AGE_GROUP.txt and drop it on the AGE_RANGE field of target TRG_CUSTOMER.

Target Datastore - TRG_CUSTOMER		
	Name	Mapping
Indicators	*CUST_ID	SRC_CUSTOMER.CUS
	DEAR	case when SRC_CUST
	CUST_NAME	INITCAP(SRC_CUSTO
	ADDRESS	SRC_CUSTOMER.ADD
	*CITY_ID	SRC_CUSTOMER.CIT
	PHONE	SRC_CUSTOMER.PHO
	AGE	SRC_CUSTOMER.AGE
	AGE_RANGE	AGE.AGE_RANGE

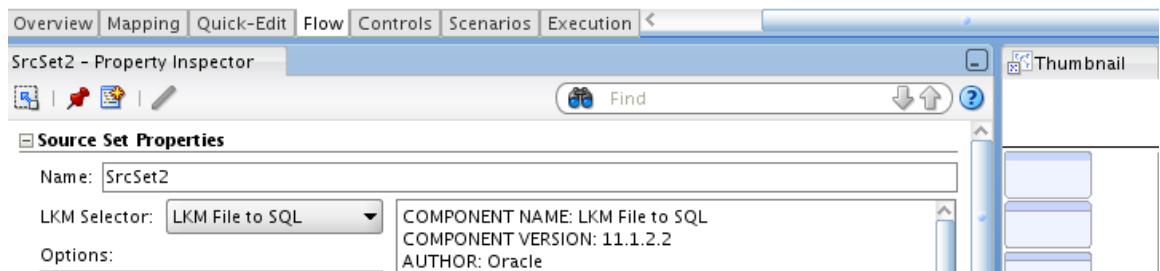
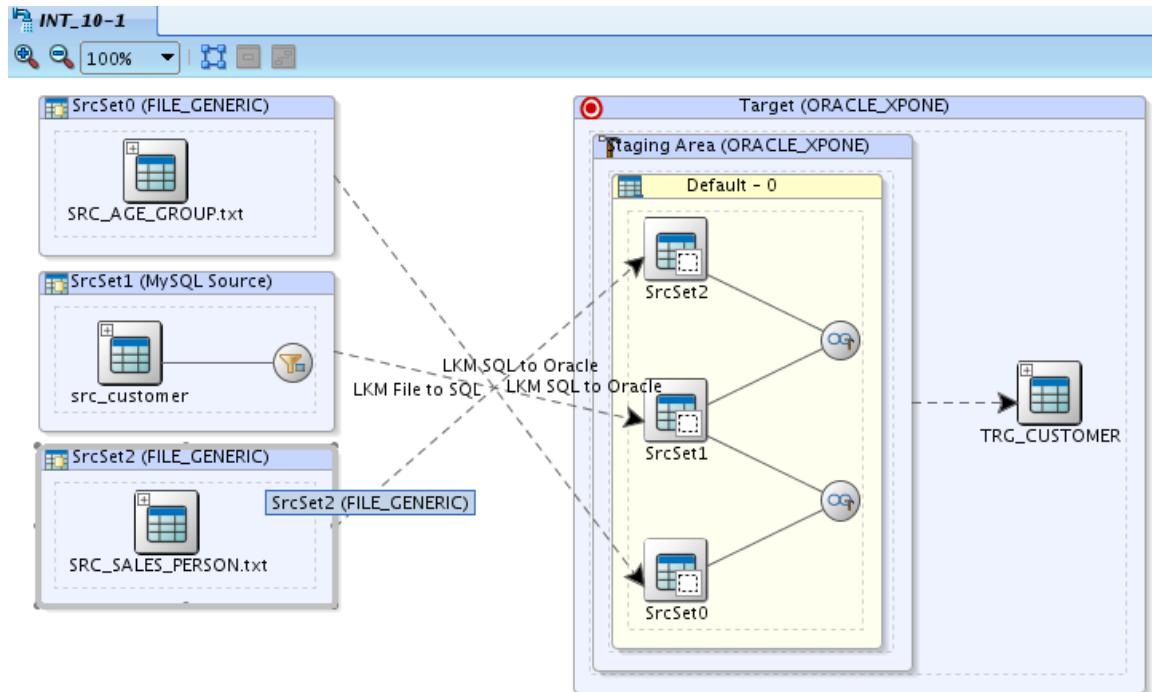
- q) Modify the SALES_PERS of the target data store, Drag the FIRST_NAME and LAST_NAME field of source SRC_SALES_PERSON.txt and drop it on the property window of the SALES_PERS target field and modify the expression to rtrim(SRC.FIRST_NAME) || ' ' || rtrim(SRC.LAST_NAME). Select the Staging area option for Execute on option.



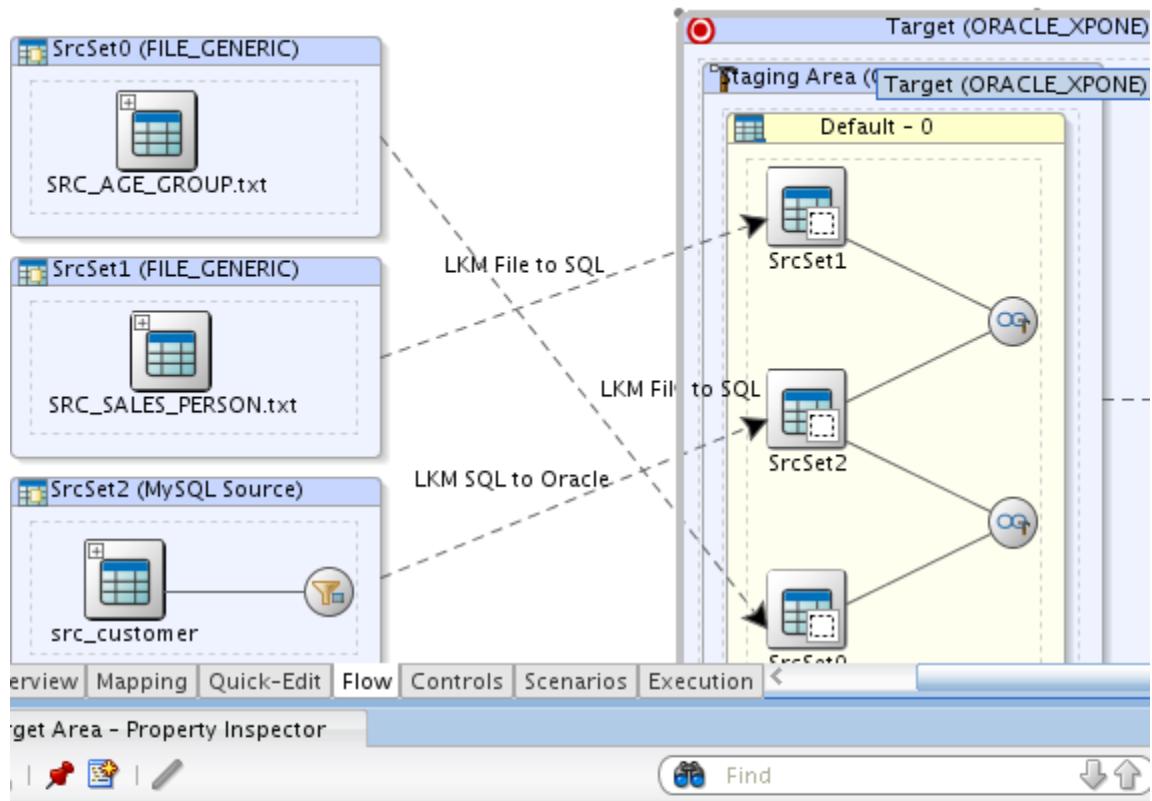
- r) Similar to Lab 3, transform the DEAR of the target data store as follows:

MySQL server column	Oracle Column	Transformation	SQL Syntax
Dear	Dear	Change 0 into Mr Change 1 into Mrs Change 2 into Ms	Case when <expression> then <value> when <expression> then <value> else <value> End

- s) Click the Flow tab. Click SrcSet0 (FILE_GENERIC). In the Source Set Properties, select LKM File to SQL. Repeat the same for SrcSet2 (FILE_GENERIC).



- t) Click Target (ORACLE_XPONE) , Change the value of the FLOW_CONTROL to **false** and DELETE_ALL to **true** for the IKM Oracle Incremental update. Click Save button to save the interface.



Target Properties

Distinct Rows:

IKM Selector: **IKM Oracle Incremental Update**

Options:

Name	Value
COMMIT	<default>:true
SYNC_JRN_DELETE	<default>:true
FLOW_CONTROL	false
RECYCLE_ERRORS	<default>:false
STATIC_CONTROL	<default>:false
TRUNCATE	<default>:false
DELETE_ALL	true

COMPONENT NAME: IKM Oracle Incremental Update

COMPONENT VERSION: 11.1.2.22

AUTHOR: Oracle

COMPATIBILITY: ODI 11.1.1.1.0 and above

DESCRIPTION:

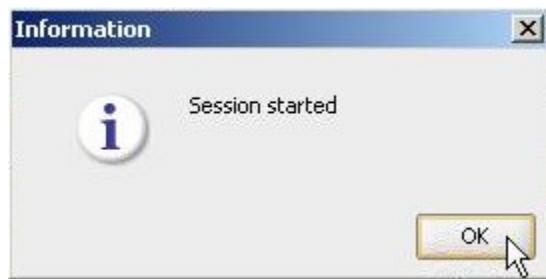
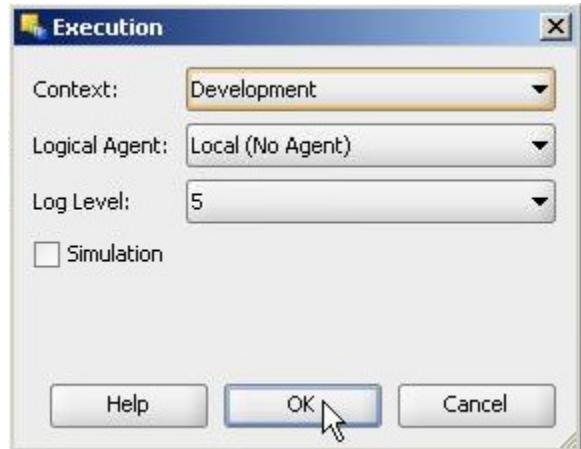
- Integrates data into an Oracle target table in incremental update mode.
- Inexistent rows are inserted; already existing rows are updated.
- Data can be controlled. Invalid data is isolated in the Error Table and can be recycled.

2) Run this interface, and check the content of the TRG_CUSTOMER table.

a) In the Projects tab, select interface INT_10-1. Click the Execute button 

- b) Click OK in the Execution window that appears, and then click OK when the Session Started message appears. Open Operator, and verify that your Interface executed successfully.

Note: In ODI Operator Navigator, you may need to click the Refresh icon



- 1 - INT_10-1 - Sep 17, 2010 4:04:39 PM
 - + 1 - Loading - SrcSet1 - Drop work table
 - + 2 - Loading - SrcSet1 - Create work table
 - + 3 - Loading - SrcSet1 - Load data
 - + 5 - Loading - SrcSet2 - Drop work table
 - + 6 - Loading - SrcSet2 - Create work table
 - + 7 - Loading - SrcSet2 - Load data
 - + 8 - Loading - SrcSet2 - Analyze work table
 - + 10 - Loading - SrcSet0 - Drop work table
 - + 11 - Loading - SrcSet0 - Create work table
 - + 12 - Loading - SrcSet0 - Load data
 - + 14 - Integration - INT_10-1 - Drop flowtable
 - + 15 - Integration - INT_10-1 - Create flowtable I\$
 - + 16 - Integration - INT_10-1 - Delete target table
 - + 17 - Integration - INT_10-1 - Insert flow into I\$ table
 - + 18 - Integration - INT_10-1 - Create Index on flowtable
 - + 19 - Integration - INT_10-1 - Analyze integration table
 - + 20 - Integration - INT_10-1 - Flag rows for update
 - + 21 - Integration - INT_10-1 - Flag useless rows
 - + 22 - Integration - INT_10-1 - Update existing rows
 - + 23 - Integration - INT_10-1 - Insert new rows
 - + 24 - Integration - INT_10-1 - Commit transaction
 - + 25 - Integration - INT_10-1 - Drop flowtable
 - + 1000004 - Loading - SrcSet1 - Drop work table
 - + 1000009 - Loading - SrcSet2 - Drop work table
 - + 1000013 - Loading - SrcSet0 - Drop work table

- c) Return to Designer, click the Mapping tab, select the TRG_CUSTOMER target datastore (click the name of the datastore). Right-click and select the Data option. A window appears with the loaded data.

The screenshot shows two windows from the Oracle Data Integrator interface:

- Target Datastore - TRG_CUSTOMER**: This window lists 11 rows in a table. Row 4 is selected and has a context menu open. The menu items are: Open, Add Column, Delete, Data..., Number Of Rows..., Sort, and Redo Auto Mapping. The "Data..." option is highlighted with a blue selection bar.
- Data Editor**: This window displays a table with three columns: CUST_ID, DEAR, and CUST_NAME. The data is as follows:

	CUST_ID	DEAR	CUST_NAME
1	101	Mr	Rodney Maute
2	103	Ms	Phyllis Reuschel
3	104	Mr	Lewis Checca
4	100	Ms	Janette Saide

Below the table, the "Executed Query:" field contains the SQL statement: `select * from CUST_DW_DEV.TRG_CUSTOMER`. The status bar at the bottom left says "Record 1 of 5". A "Close" button is located at the bottom right of the Data Editor window.

- 3) Close this window, and then close your interface.

Lab 11: Advanced Interface – Using Datasets with ODI Interface

Introduction

A **dataset** represents the data flow coming from a group of datastores. Several datasets can be merged into the interface target datastore using set-based operators such as Union and Intersect. The support for datasets as well as the set-based operators supported depends on the capabilities of the staging area's technology.

You can add, remove, and order the datasets of an interface and define the operators between them in the DataSets Configuration dialog. Note that the set-based operators are always executed on the staging area.

In previous labs, you learned how to create an interface and execute it by an agent. Oracle Data Integrator generates the code for the execution in the form of a *session*. The run-time agent processes this code and connects to the sources and targets to perform the data integration.

In Oracle Data Integrator you have the possibility at design-time to simulate an execution. Simulating an execution generates and displays the code corresponding to the execution without running this code. Execution **simulation** provides reports suitable for code review. Note: No session is created in the log when the execution is started in simulation mode

Process Overview

In this practice, you add a new dataset to an interface. You will add a bulk feed of customer data from another system with some filters by adding another dataset.

Now you create a new dataset, naming it BULK_CUST_DATA. For its source, you specify the ARCHIVE_CUST data store from the Oracle Sales Warehouse model.

You specify the UNION operator to unite this new source of data with the default dataset. Next, you perform execution **simulation** to preview the code that will be used for this new dataset with UNION operator.

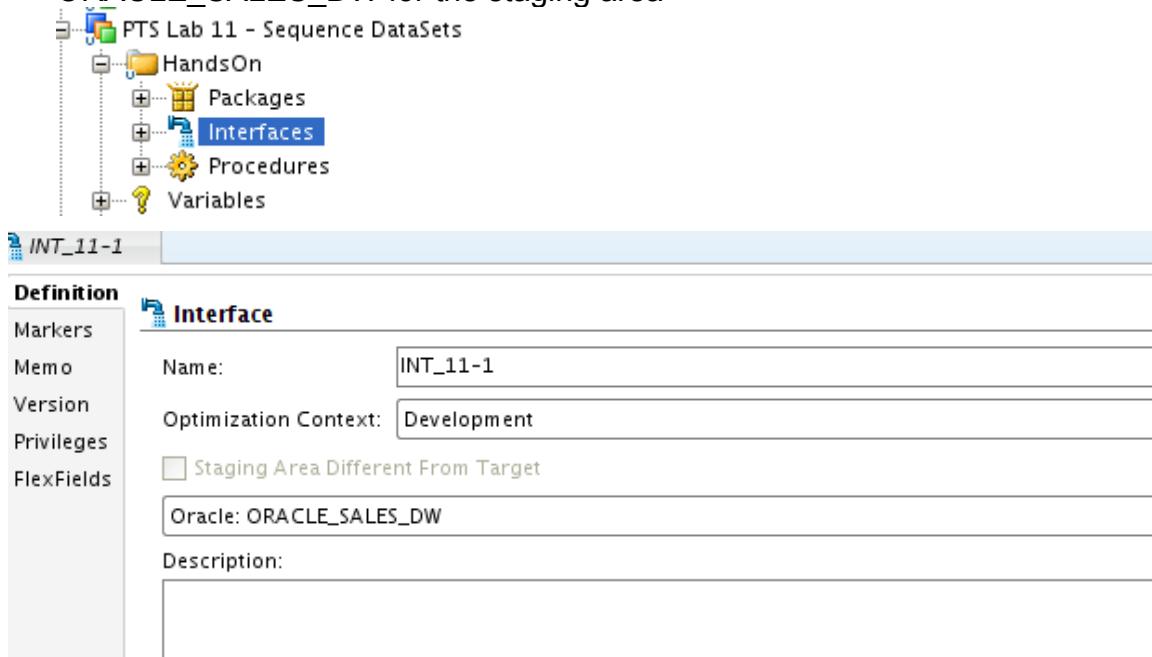
Finally, you execute the interface and verify the rows inserted into the target table.

Scenario

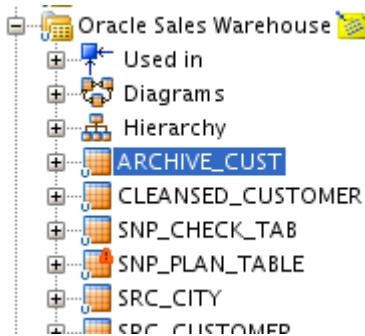
You created interface to load data in the target model from different sources and perform data transformation, filtering, and provided a date when data was updated. Now you create an interface with multiple datasets to add in a bulk feed of customer data from another system.

Instructions

- 1) Create ODI interface with multiple datasets.
 - a) In the Designer, Expand the project PTS Lab 11 – Datasets. Right-click interface and create a new interface INT_11-1. Choose the Oracle: ORACLE_SALES_DW for the staging area



- b) Click the Mapping tab of the Interface. Expand the Model Oracle Sales Warehouse from the Model Designer; Drag the table TRG_CUSTOMER on to the target area. Drag the table ARCHIVE_CUST on to the source area (If the table is not visible, right click the Model Oracle Sales Warehouse and select Reverse Engineer option. This will update model). Drag the CITY_ID field to the open area to add a filter and give the condition ARCHIVE_CUST.CITY_ID < 20



Target Datastore - TRG_CUSTOMER

Position	Indicators	Name	Mapping
1	N	*CUST_ID	ARCHIV...
2	V	DEAR	ARCHIV...
3	V	CUST_NAME	ARCHIV...
4	V	ADDRESS	ARCHIV...
5	N	*CITY_ID	ARCHIV...
6	V	PHONE	ARCHIV...
7	N	AGE	ARCHIV...
8	V	AGE_RANGE	ARCHIV...
9	V	SALES_PERS	ARCHIV...
10	D	CRE_DATE	ARCHIV...
11	D	UPD_DATE	ARCHIV...

Filter Properties

Active Filter:

Implementation: ARCHIVE_CUST.CITY_ID < 20

- c) Add a new data set by selecting the data set icon as show in the below screen shot

- d) Enter new dataset name: BULK_CUST_DATA. Select Operator: UNION. Close DataSets Configuration window.

Order	DataSet Name	Operator
0	Default	<N/A>
10	BULK_CUST_DATA	UNION

- e) In the Mapping tab, select BULK_CUST_DATA dataset tab. In the Models tab, drag ARCHIVE_CUST data store from the Oracle Sales Warehouse to the Source area of the interface. Click Yes to perform automatic mapping. Drag the CITY_ID field to the open area to add a filter and give the condition ARCHIVE_CUST.CITY_ID > 40

The screenshot displays the Oracle Data Integrator (ODI) interface with two main panels: a dataset mapping view and a filter configuration view.

Dataset Mapping View:

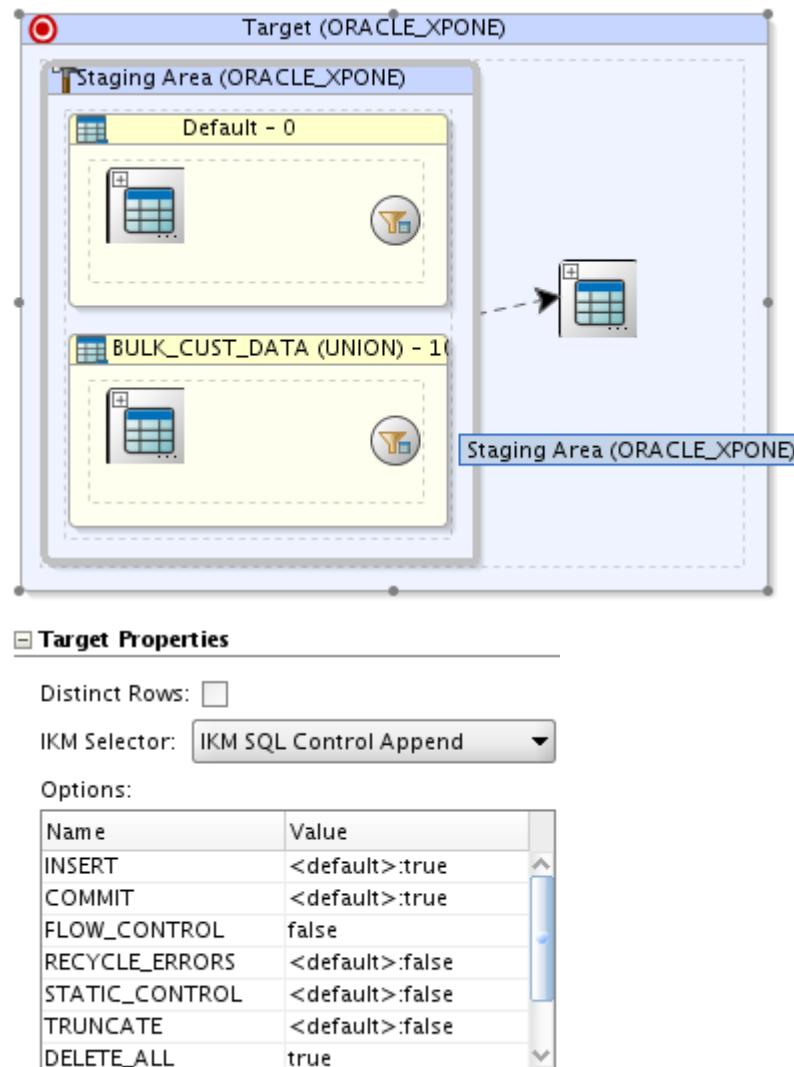
- Source Dataset:** "1 - ARCHIVE_CUST (ARCHIVE CU)" containing columns: *CUST_ID, DEAR, CUST_NAME, ADDRESS, *CITY_ID, PHONE, AGE, AGE_RANGE, SALES_PERS, CRE_DATE, and UPD_DATE.
- Target Datastore:** "Target Datastore - TRG_CUSTOMER" with 11 mappings listed in a table:

Position	Indicators	Name	Mapping
1	N	*CUST_ID	ARCHIV...
2	V	DEAR	ARCHIV...
3	V	CUST_NAME	ARCHIV...
4	V	ADDRESS	ARCHIV...
5	N	*CITY_ID	ARCHIV...
6	V	PHONE	ARCHIV...
7	N	AGE	ARCHIV...
8	V	AGE_RANGE	ARCHIV...
9	V	SALES_PERS	ARCHIV...
10	d	CRE_DATE	ARCHIV...
11	d	UPD_DATE	ARCHIV...

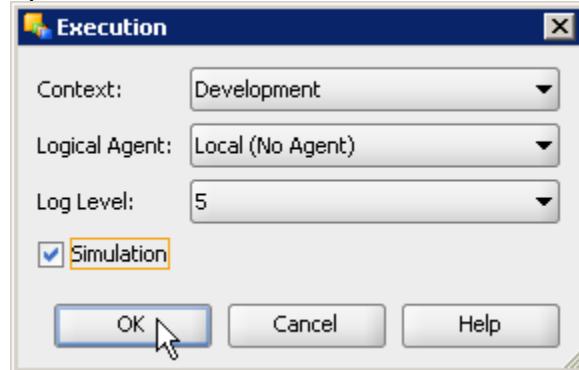
Filter Configuration View:

- Title:** Filter - Property Inspector
- Active Filter:**
- Implementation:** ARCHIVE_CUST.CITY_ID > 40

- f) Open Flow tab and verify the flow of your interface. Click the Target datastore. Select the IKM SQL Control Append , Set FLOW_CONTROL option to false and DELETE_ALL option to true. Save the Interface



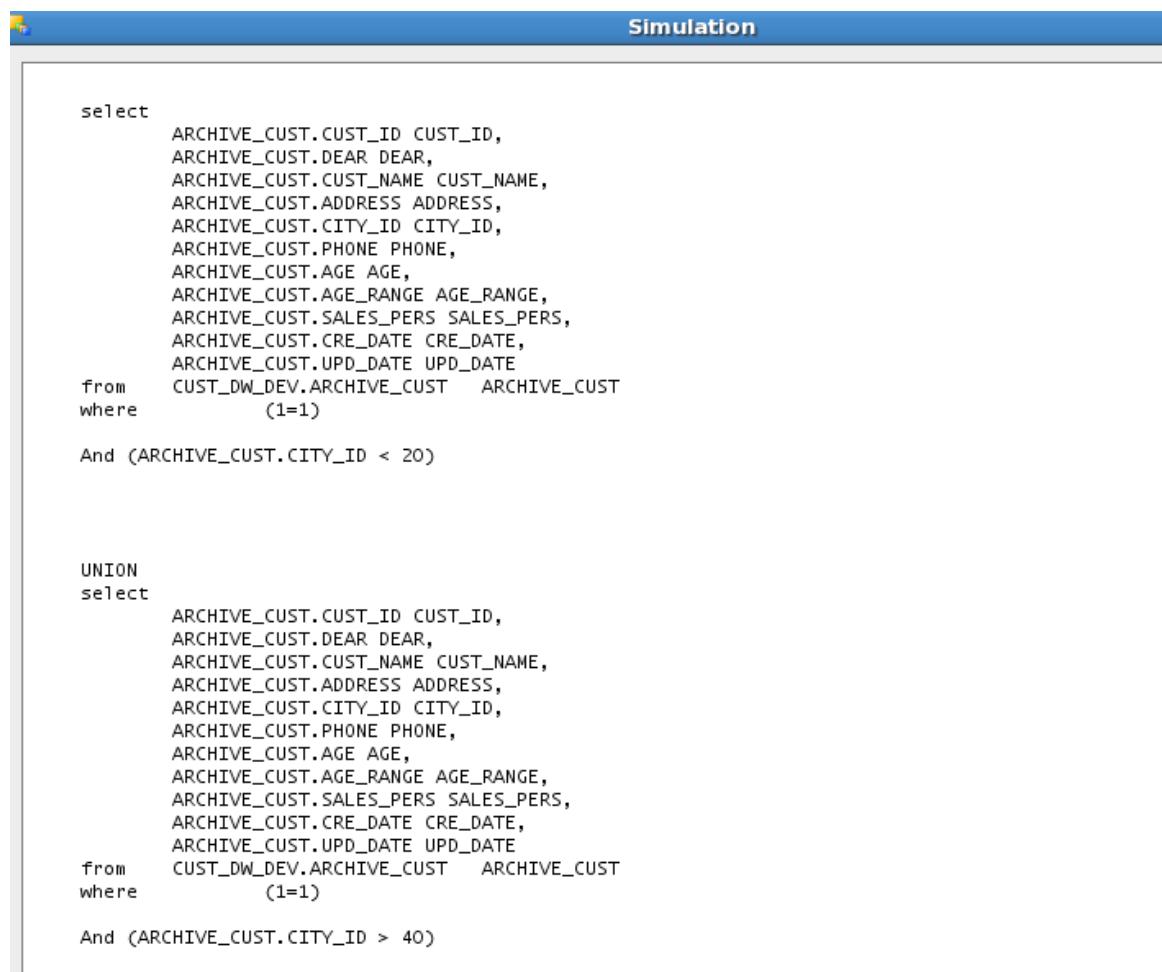
- g) Perform execution simulation. In the Simulation window, scroll down and view simulated code for the dataset ARCHIVE_CUST with UNION operator as shown below.



The Execution dialog box shows the following settings:

- Context: Development
- Logical Agent: Local (No Agent)
- Log Level: 5
- Simulation

Buttons: OK (highlighted with a cursor), Cancel, Help.



The Simulation window displays the following SQL query code:

```

select
    ARCHIVE_CUST.CUST_ID CUST_ID,
    ARCHIVE_CUST.DEAR DEAR,
    ARCHIVE_CUST.CUST_NAME CUST_NAME,
    ARCHIVE_CUST.ADDRESS ADDRESS,
    ARCHIVE_CUST.CITY_ID CITY_ID,
    ARCHIVE_CUST.PHONE PHONE,
    ARCHIVE_CUST.AGE AGE,
    ARCHIVE_CUST.AGE_RANGE AGE_RANGE,
    ARCHIVE_CUST.SALES_PERS SALES_PERS,
    ARCHIVE_CUST.CRE_DATE CRE_DATE,
    ARCHIVE_CUST.UPD_DATE UPD_DATE
from CUST_DW_DEV.ARCHIVE_CUST ARCHIVE_CUST
where (1=1)

And (ARCHIVE_CUST.CITY_ID < 20)

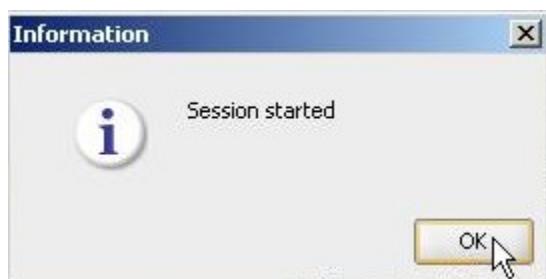
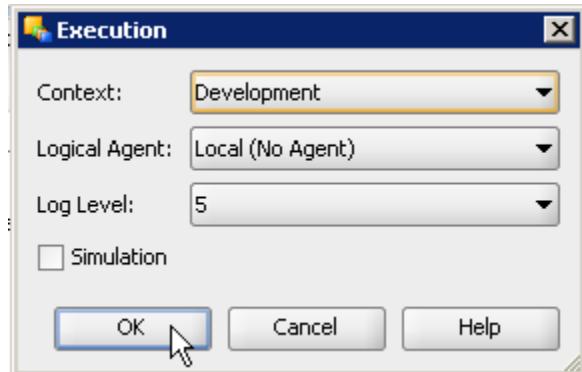
UNION
select
    ARCHIVE_CUST.CUST_ID CUST_ID,
    ARCHIVE_CUST.DEAR DEAR,
    ARCHIVE_CUST.CUST_NAME CUST_NAME,
    ARCHIVE_CUST.ADDRESS ADDRESS,
    ARCHIVE_CUST.CITY_ID CITY_ID,
    ARCHIVE_CUST.PHONE PHONE,
    ARCHIVE_CUST.AGE AGE,
    ARCHIVE_CUST.AGE_RANGE AGE_RANGE,
    ARCHIVE_CUST.SALES_PERS SALES_PERS,
    ARCHIVE_CUST.CRE_DATE CRE_DATE,
    ARCHIVE_CUST.UPD_DATE UPD_DATE
from CUST_DW_DEV.ARCHIVE_CUST ARCHIVE_CUST
where (1=1)

And (ARCHIVE_CUST.CITY_ID > 40)

```

- 2) Execute interface INT_11-1 and verify the execution results.

a) Execute interface INT_11-1.



- b) Open the Operator. Verify the session INT_11-1 executed successfully. Scroll down and open task “Integration – INT_11-1- Insert new rows”. Verify number of inserts in the target table.

The screenshot shows the Oracle Data Integrator interface. At the top, there is a tree view of sessions and tasks:

- 265020 - INT_11-1 - Sep 17, 2010 6:29:55 PM
 - Variables
 - 1 - INT_11-1 - Sep 17, 2010 6:29:55 PM
 - 1 - Integration - INT_11-1 - Delete target table
 - 2 - Integration - INT_11-1 - Insert new rows**
 - 3 - Integration - INT_11-1 - Commit transaction

Below the tree view is a detailed session summary for "Session Task: Integration".

Definition	
Code	Task Name: Integration
Connection	Status: Done
Privileges	Type: Interface
	<input type="checkbox"/> Ignore Errors
Order:	2
	Log Level: 3
Record Statistics	
No. of Inserts:	30
No. of Deletes:	0
No. of Rows:	30
No. of Updates:	0
No. of Errors:	0
Execution Statistics	
Start:	Sep 17, 2010 6:29:55 PM
Duration (seconds):	0
End:	Sep 17, 2010 6:29:55 PM
Return Code:	0
Message:	

Lab 12: Advanced Interface – Temporary Interface with Lookup

Introduction

The source datastores contain data used to load the target datastore. Two types of datastores can be used as an interface source: datastores from the models and temporary datastores that are the target of an interface.

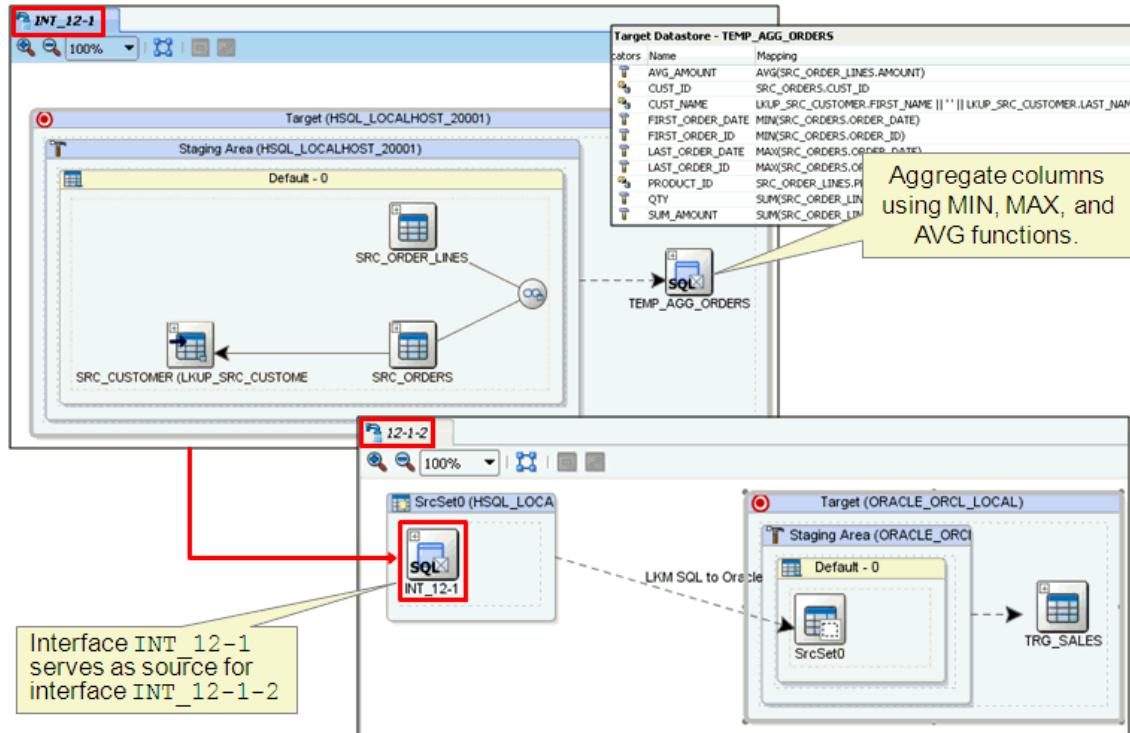
Derived Select for Temporary Interfaces - When using a temporary datastore that is the target of another interface as a source or as a lookup table, you can choose not to persist the target of the temporary interface, and generate instead a Derived Select (sub-select) statement corresponding to the loading of the temporary datastore. Consequently, the temporary interface no longer needs to be executed to load the temporary datastore. The code generated for the sub-select is either a default generated code, or a customized syntax defined in an IKM. This feature eliminates the need for complex packages handling temporary interfaces and simplifies the execution of cascades of temporary interfaces.

Lookups - A lookup is a datastore (from a model or the target datastore of an interface) – called the lookup table – associated to a source datastore – the driving table – via a join expression and from which data can be fetched and used into mappings. Lookups are used to simplify the design and readability of interfaces and allow for optimized code for execution.

Process Overview

In this lab, you perform the following steps:

- 1) Create a temporary interface INT_12-1.
 - a) Use datastores SRC_ORDERS and SRC_ORDER_LINES from the HSQL Source Data model, joined on ORDER_ID.
 - b) Use SRC_CUSTOMER as a lookup table.
- 2) Create a temporary target table TEMP_AGG_ORDERS.
 - a) Aggregate some of its columns using the MIN, MAX, and AVG functions.
- 3) Create interface INT_12-2.
 - a) Use the temporary interface INT_12-1 as a source.
 - b) Use datastore TRG_SALES as the target.
- 4) Execute INT_12-2 and examine the rows inserted into TRG_SALES.



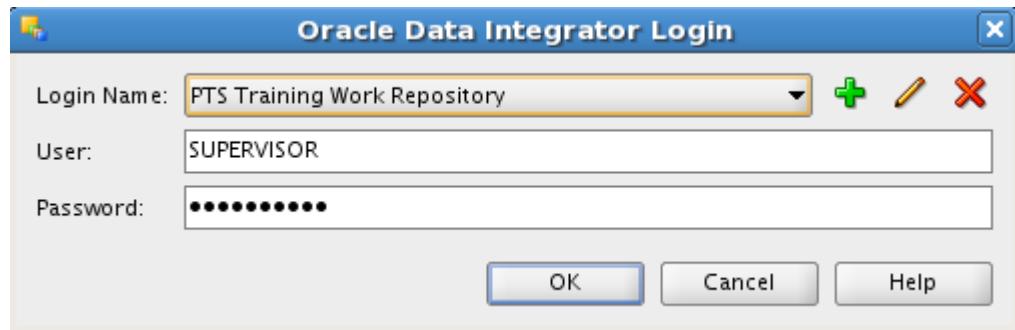
Scenario

You created interfaces to load data in the target models from different sources and performed data transformation, filtering, and implemented a lookup to fetch additional information for the data loading in the target. You also created an interface with multiple datasets to add in a bulk feed of customer data from another system.

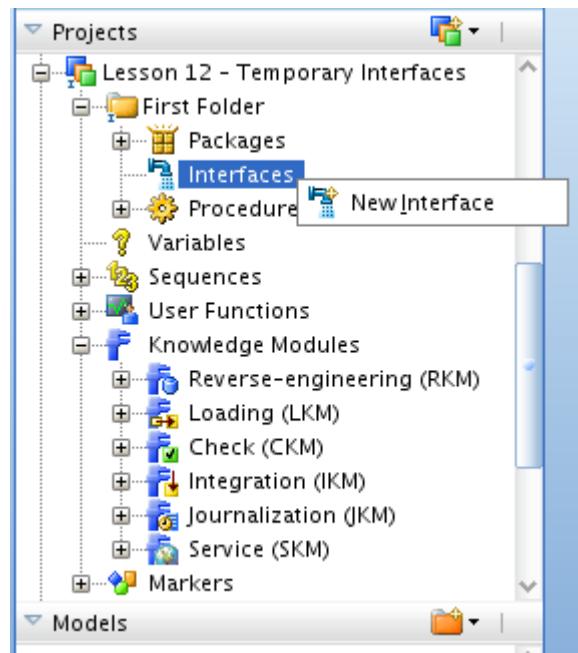
Now, you need to create a temporary interface, which is used for loading data to sales target datastore.

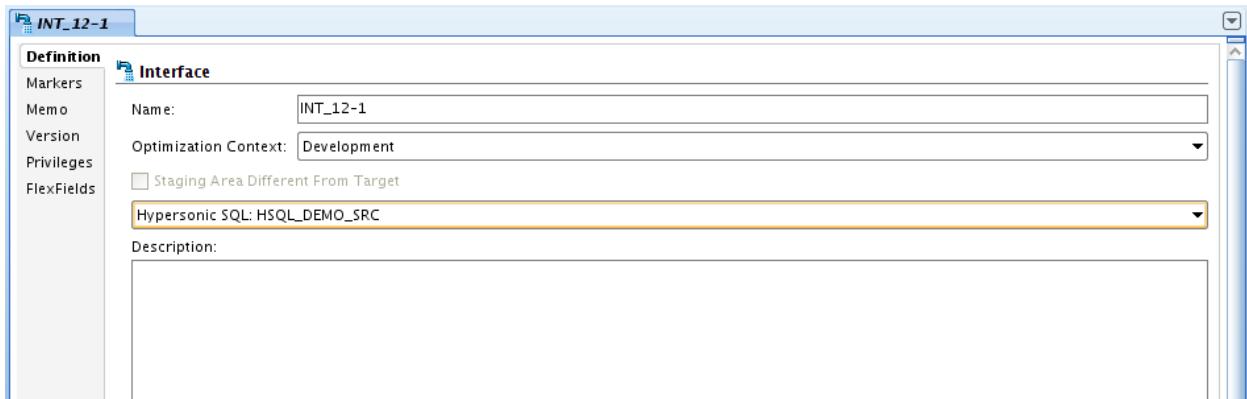
Instructions

- 1) Start HSQL, on terminal window, type > startHSQL
- 2) Develop ODI Temporary interface.
 - a) If not connected, connect to PTS Training Work Repository (User: SUPERVISOR, Password: SUPERVISOR). Click Designer tab.

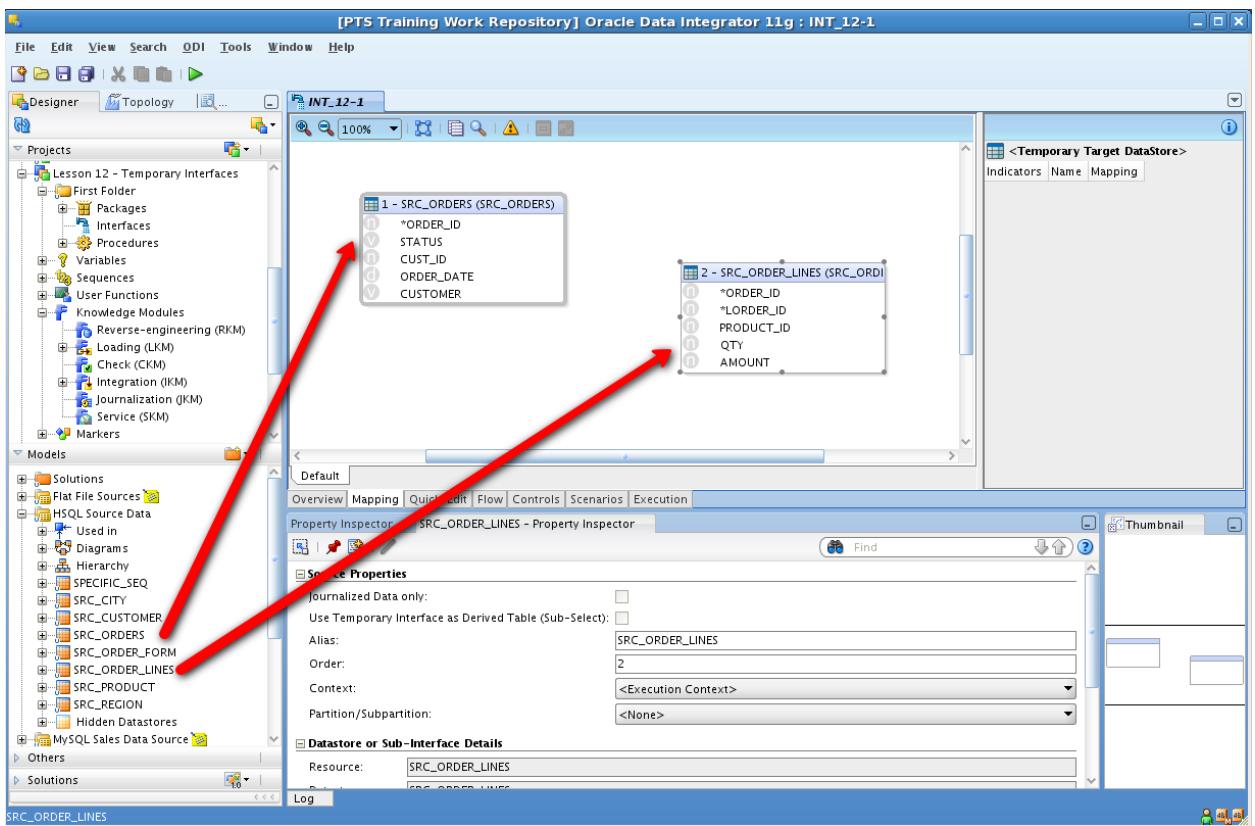


- b) In the Projects tab, expand: **Lesson 12 – Temporary Interfaces**. Right-click Interfaces, and then select New Interface. Name the new interface INT_12-1. For Staging Area, select HYPERSONIC SQL: HSQL SOURCE DATA. Click Mapping tab.

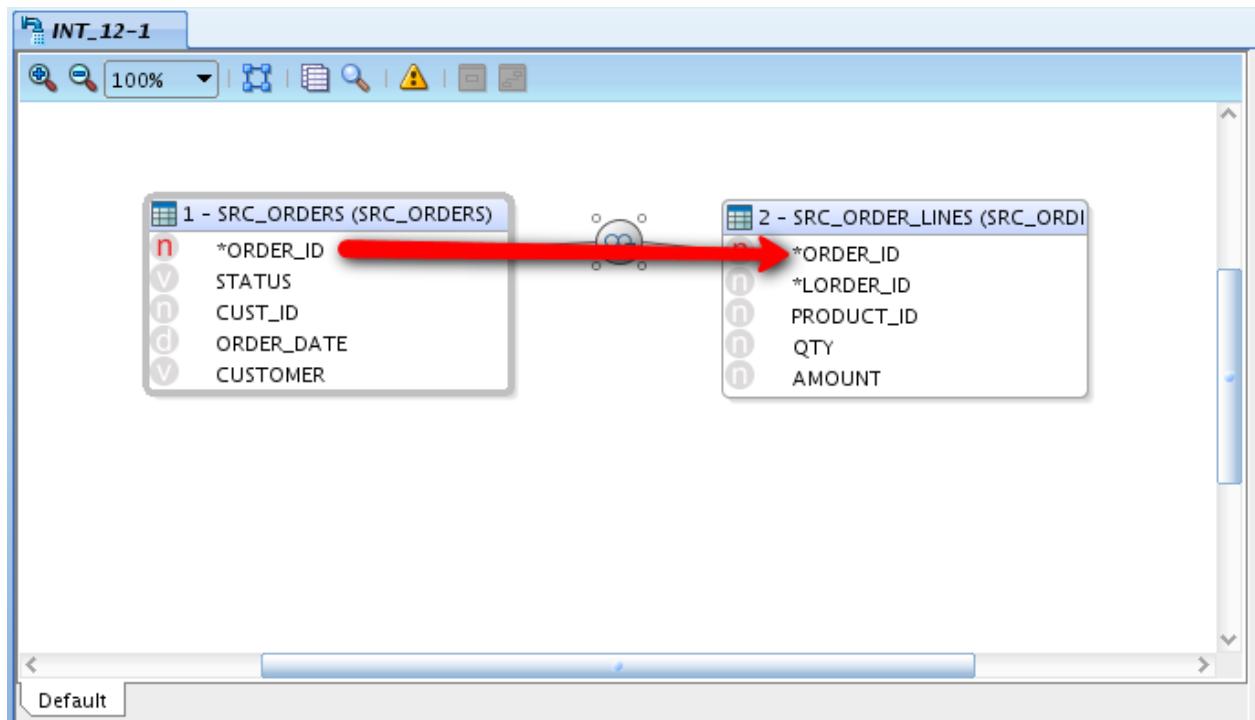




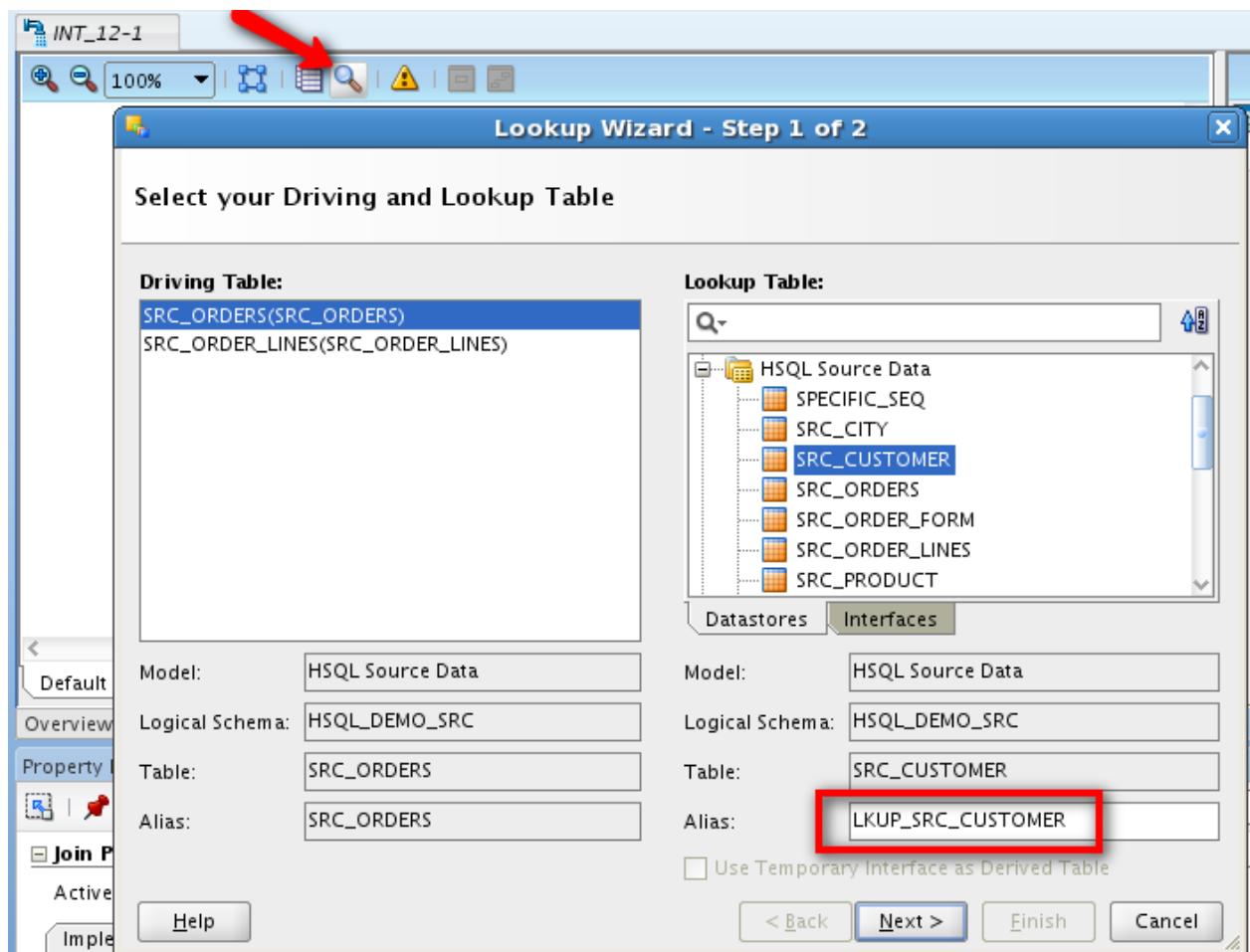
- c) Open Models tab. Expand HSQL Source Data model, and drag SRC_ORDERS and SRC_ORDER_LINES datastores from the model to the Source area.



- d) Drag ORDER_ID column from datastore SRC_ORDERS to datastore SRC_ORDER_LINES to create a Join.



- e) Click Lookup icon to start Lookup Wizard  . Make sure your Driving table is SRC_ORDERS. Select Lookup table SRC_CUSTOMER in HSQL Source Data model. For Lookup table, edit the Alias to read: LKUP_SRC_CUSTOMER as shown below. Click Next. Select CUST_ID column in the source table and CUSTID column in the lookup table, and then click Join. Click Finish.



Lookup Wizard - Step 2 of 2

The following information defines your Lookup Table settings

Source: SRC_ORDERS(SRC_ORDERS)	Lookup: SRC_CUSTOMER(LKUP_SRC_CUSTO...)
CUSTOMER	ADDRESS
CUST_ID	AGE
ORDER_DATE	CITY_ID
ORDER_ID	CUSTID
STATUS	DEAR

Join

Lookup condition:
SRC_ORDERS.CUST_ID=LKUP_SRC_CUSTOMER.CUSTID

Options

Execute on: Source Staging

Lookup type: SQL left-outer join in the from clause SQL expression in the select clause

[Help](#) [Back](#) [Next >](#) [Finish](#) [Cancel](#)

INT_12-1

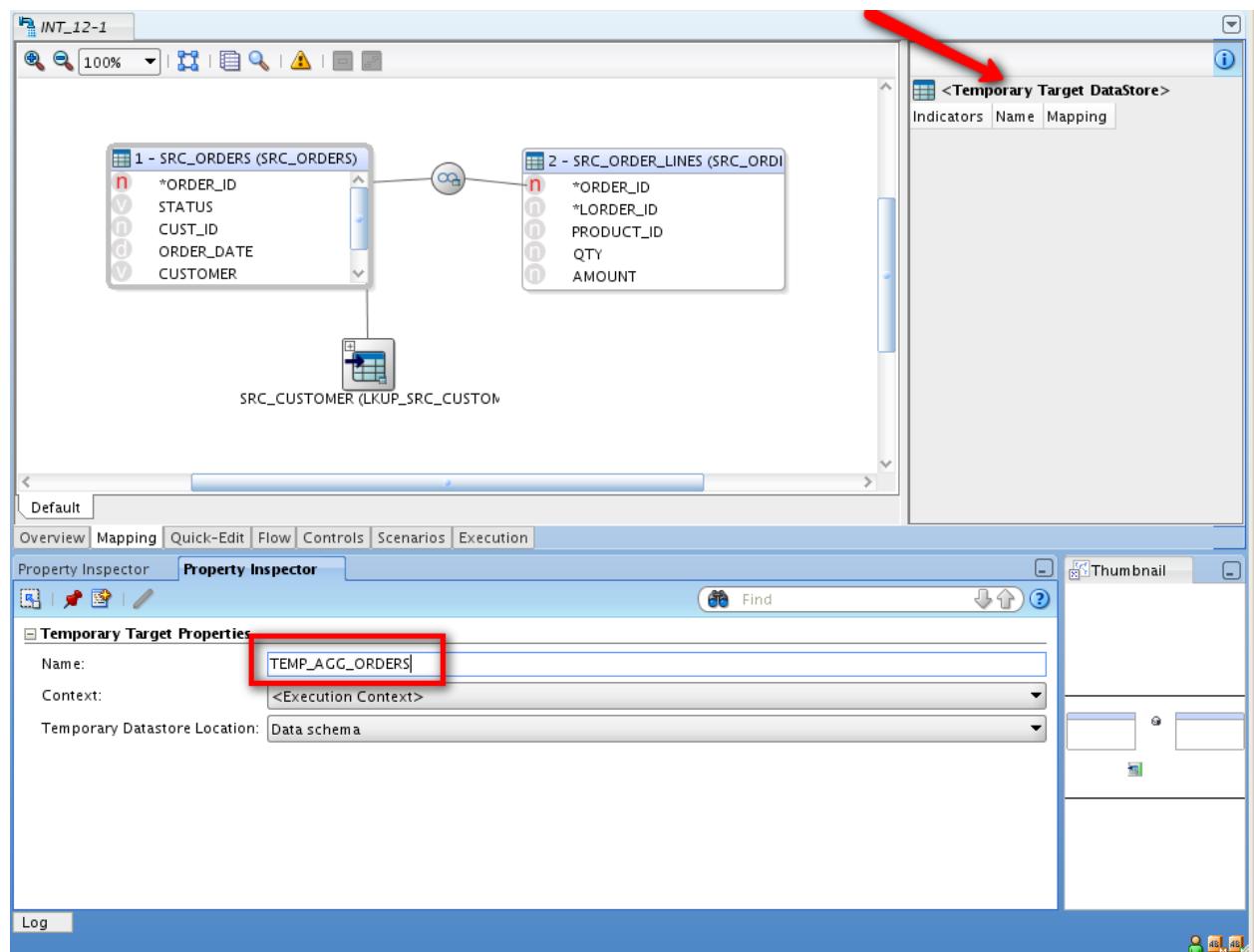
```

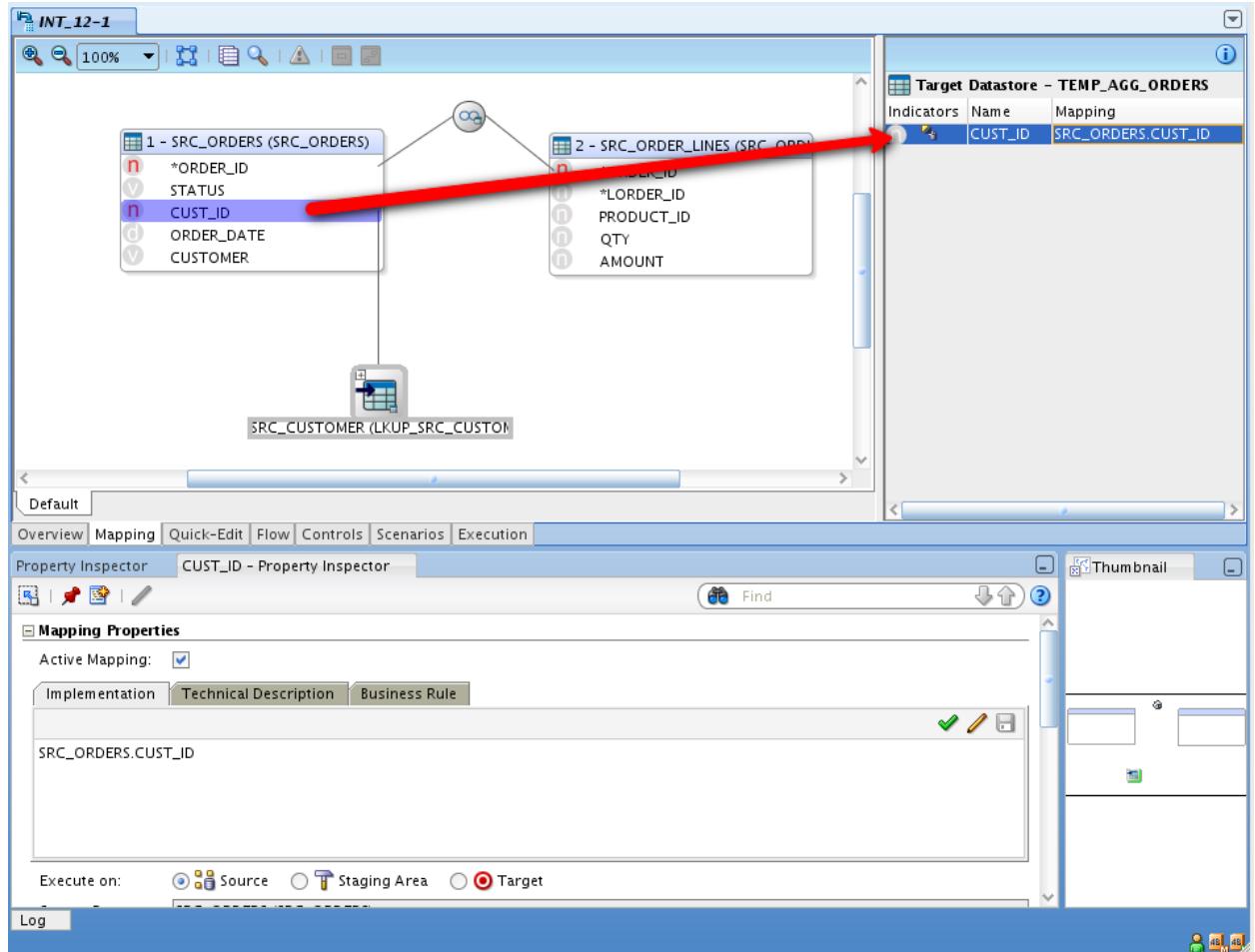
graph LR
    subgraph INT_12_1 [INT_12-1]
        direction TB
        T1["1 - SRC_ORDERS (SRC_ORDERS)"]
        T2["2 - SRC_ORDER_LINES (SRC_ORDERS)"]
        T1 --- J(( ))
        J --- T2
        T1 --- L["SRC_CUSTOMER (LKUP_SRC_CUSTO...)"]
        L --- T2
    end

```

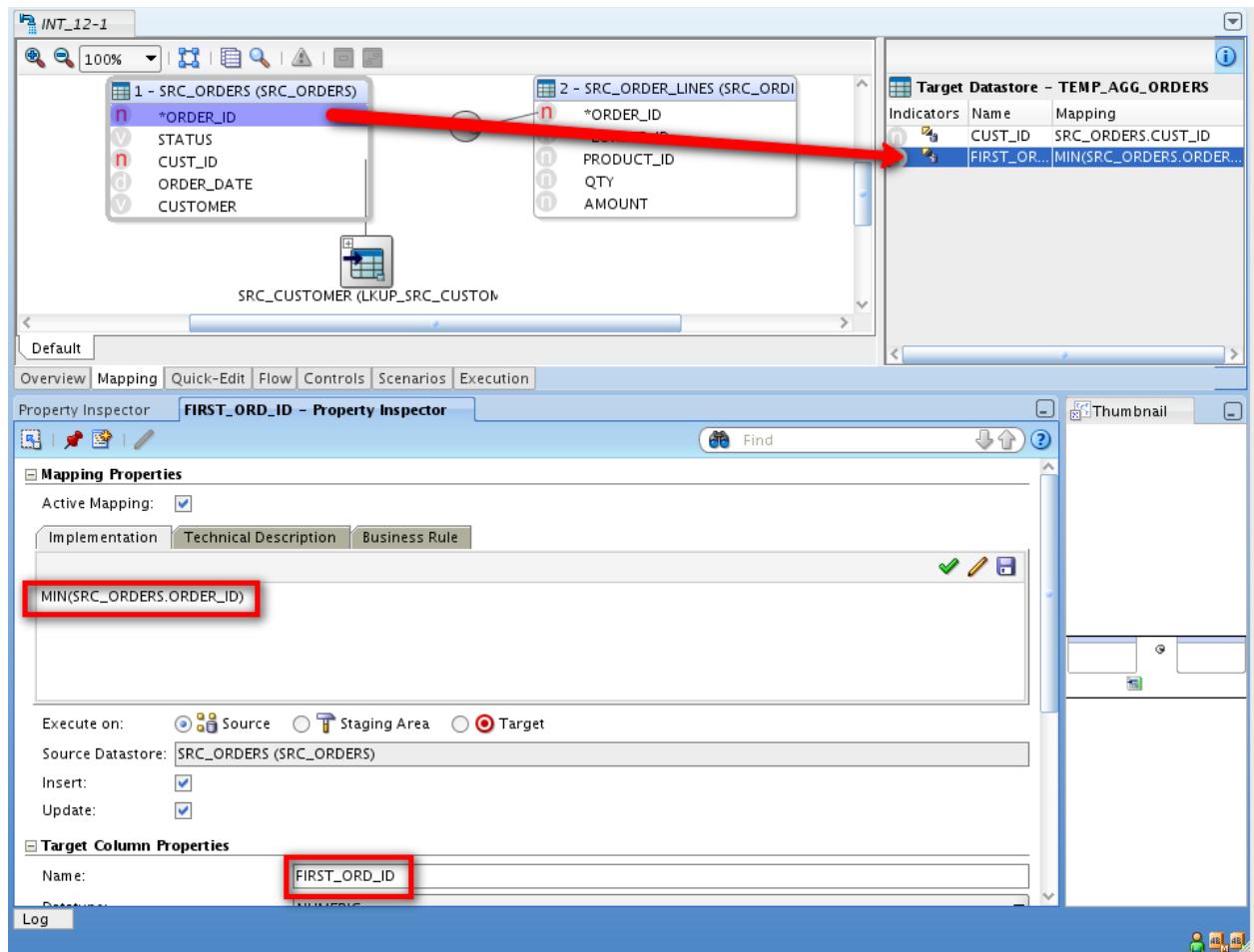
The diagram illustrates a data model interface named 'INT_12-1'. It contains two tables: '1 - SRC_ORDERS (SRC_ORDERS)' and '2 - SRC_ORDER_LINES (SRC_ORDERS)'. A relationship line connects the 'CUST_ID' field in the first table to the 'CUSTID' field in the second table. Below this, another line connects the 'SRC_CUSTOMER (LKUP_SRC_CUSTO...)' table to the 'CUSTID' field in the second table, indicating a lookup relationship.

- f) Now you need to create a temporary target datastore. Drag column CUST_ID from SRC_ORDERS table to the Target area. Click <Temporary Target DataStore>. In the Temporary Target Properties, enter Name: TEMP_AGG_ORDERS as shown below.

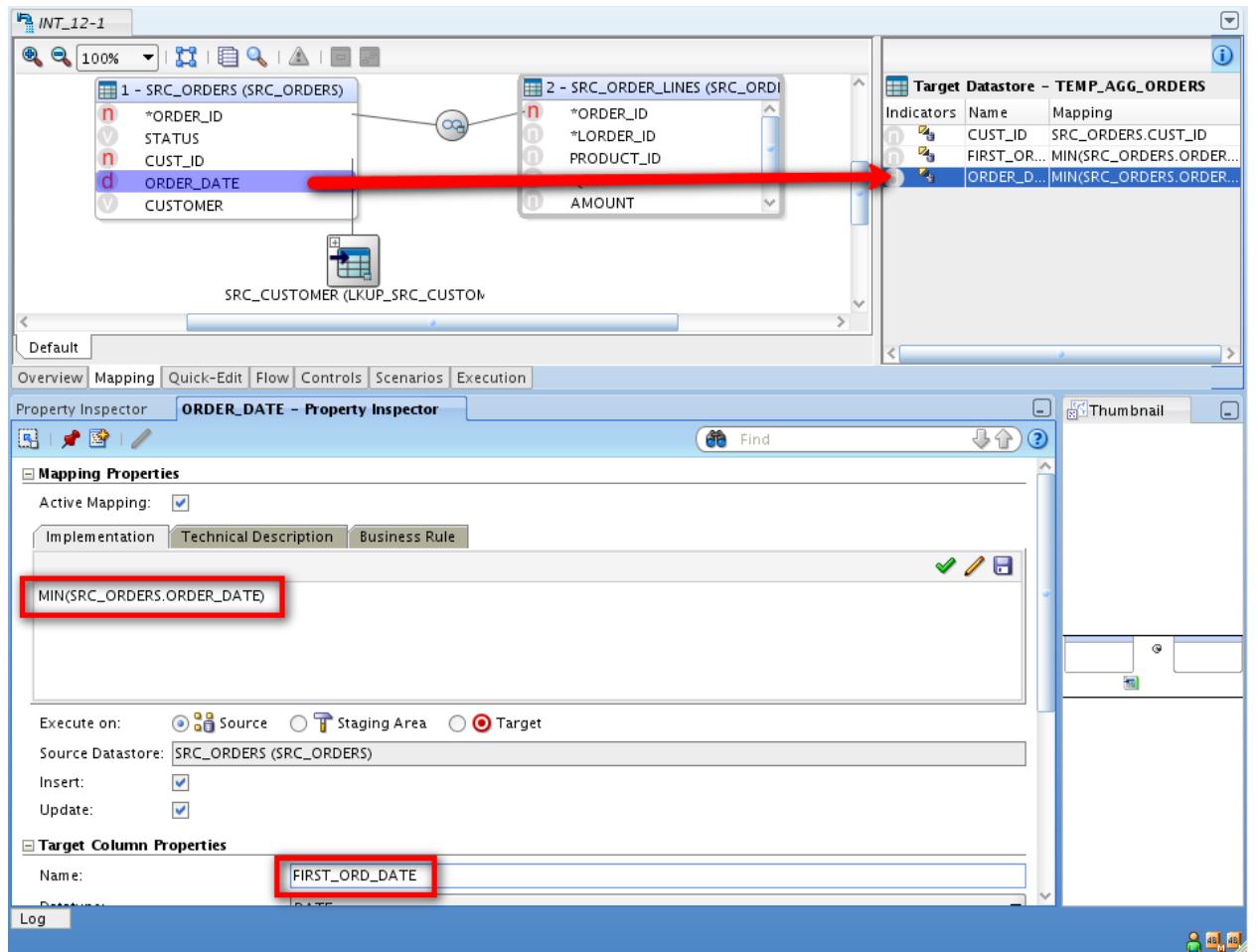




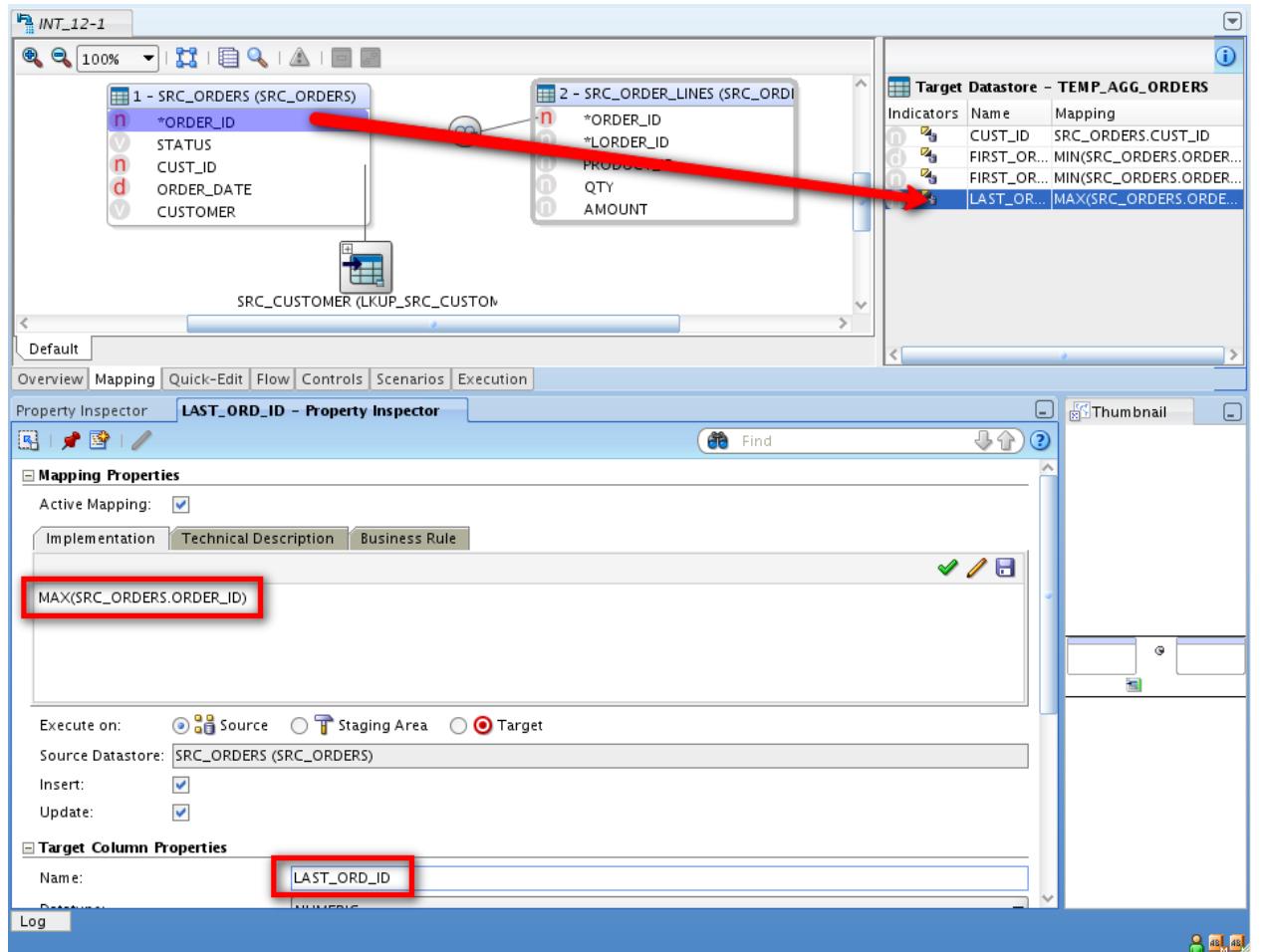
- g) Drag column ORDER_ID from SRC_ORDERS to the Target area. Edit mapping to read: MIN(SRC_ORDERS.ORDER_ID). Scroll down to the Target Column Properties section and rename this column to FIRST_ORD_ID. Click the interface tab and view the result.



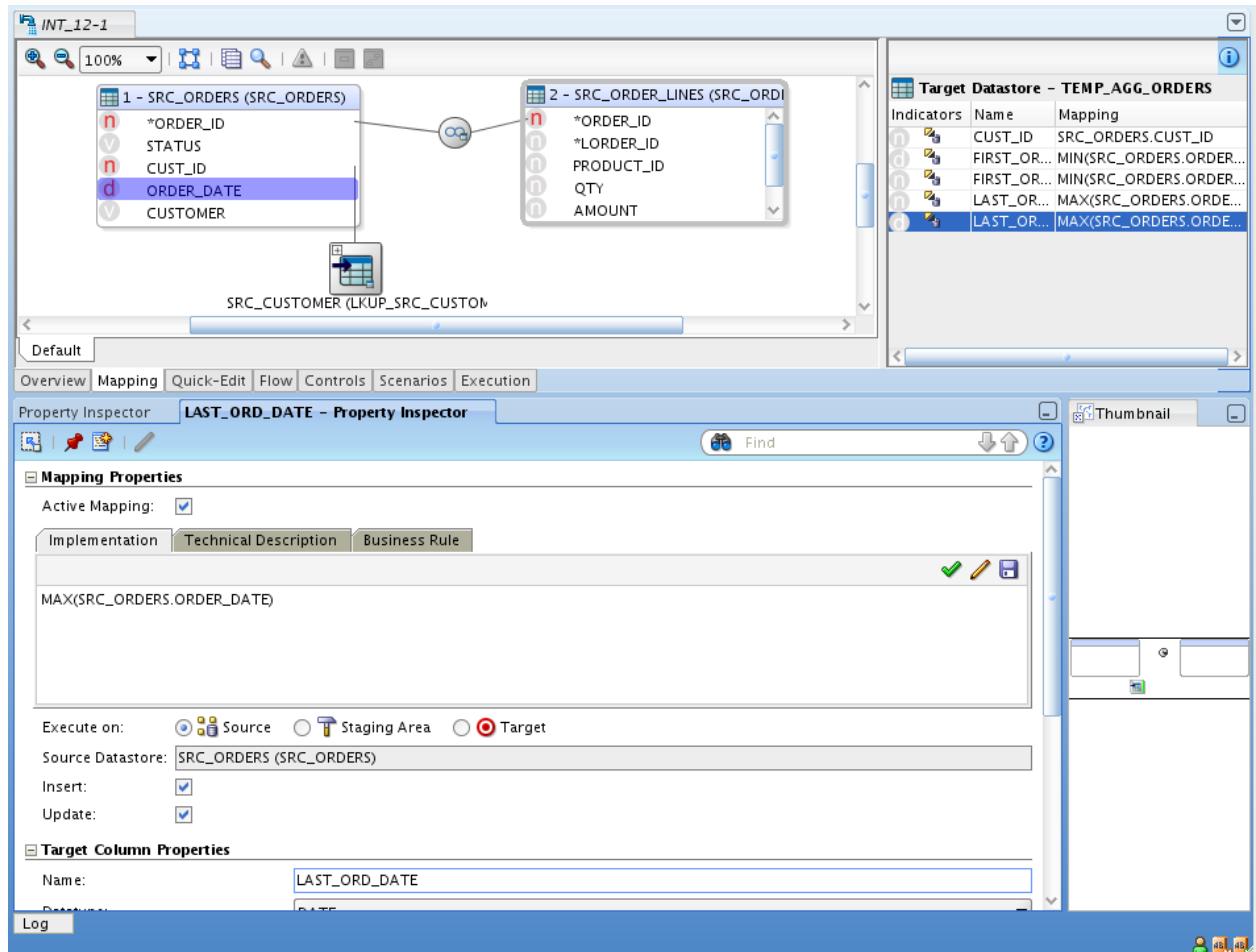
- h) Drag column ORDER_DATE from SRC_ORDERS to the Target area. Edit mapping to read: MIN(SRC_ORDERS.ORDER_DATE). Scroll down and rename this column to FIRST_ORD_DATE.



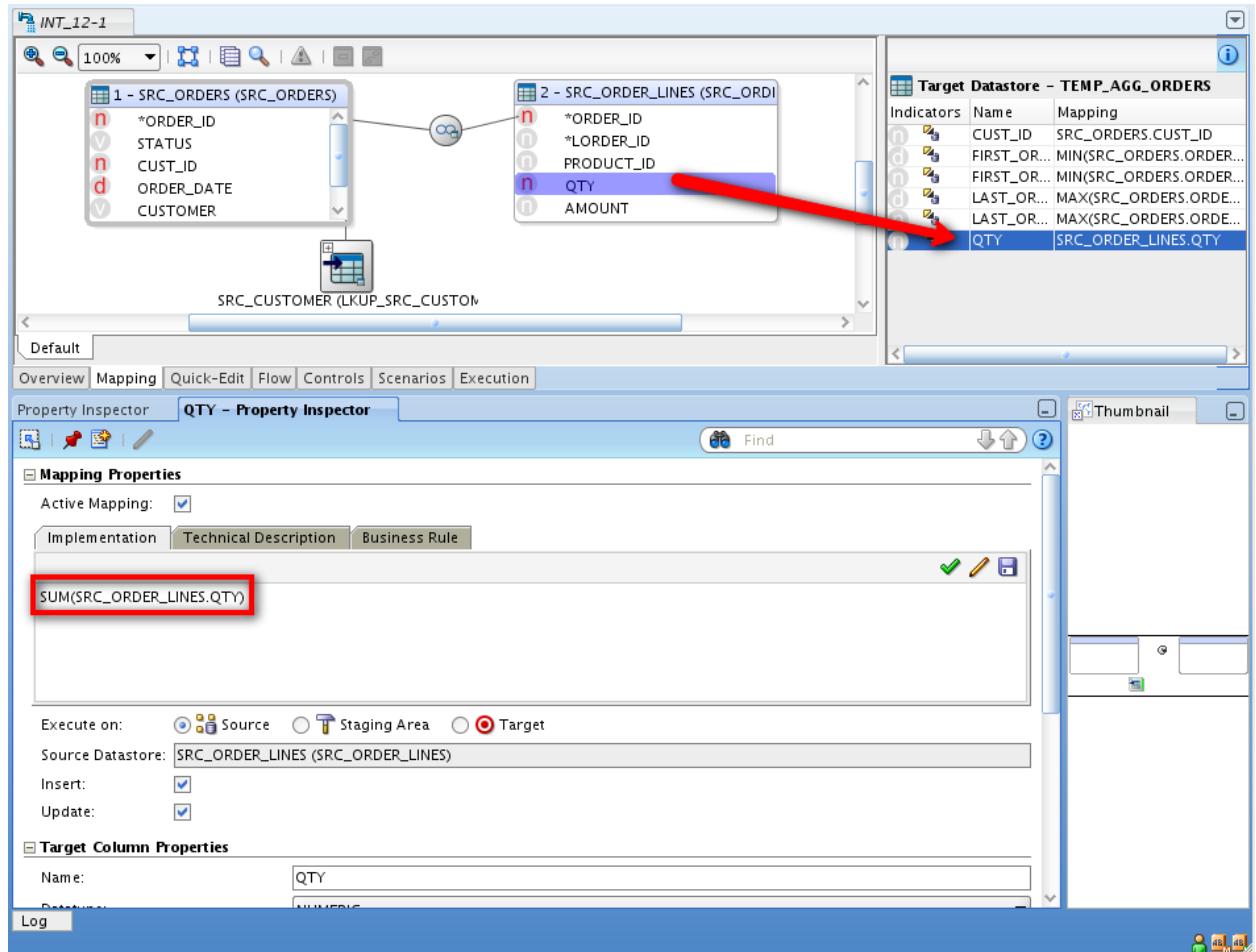
- i) Again, drag column Order_ID from SRC_ORDERS to the Target area. Edit mapping to read: MAX(SRC_ORDERS.ORDER_ID). Scroll down and rename this column to LAST_ORD_ID.



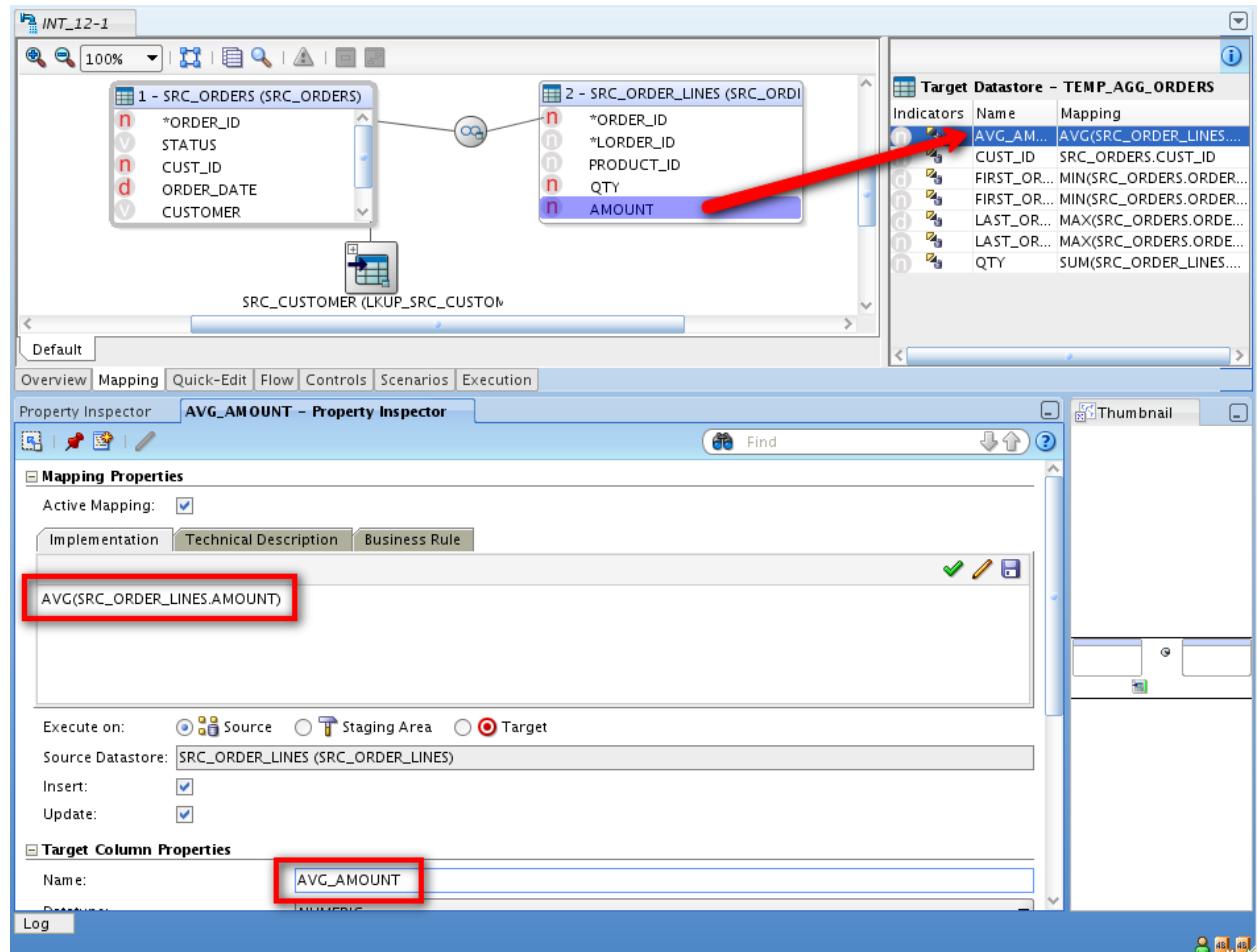
- j) Similarly, drag column ORDER_DATE again from SRC_ORDERS to the Target area. Edit the mapping to read: MAX(SRC_ORDERS.ORDER_DATE). Scroll down and rename this column to LAST_ORD_DATE.



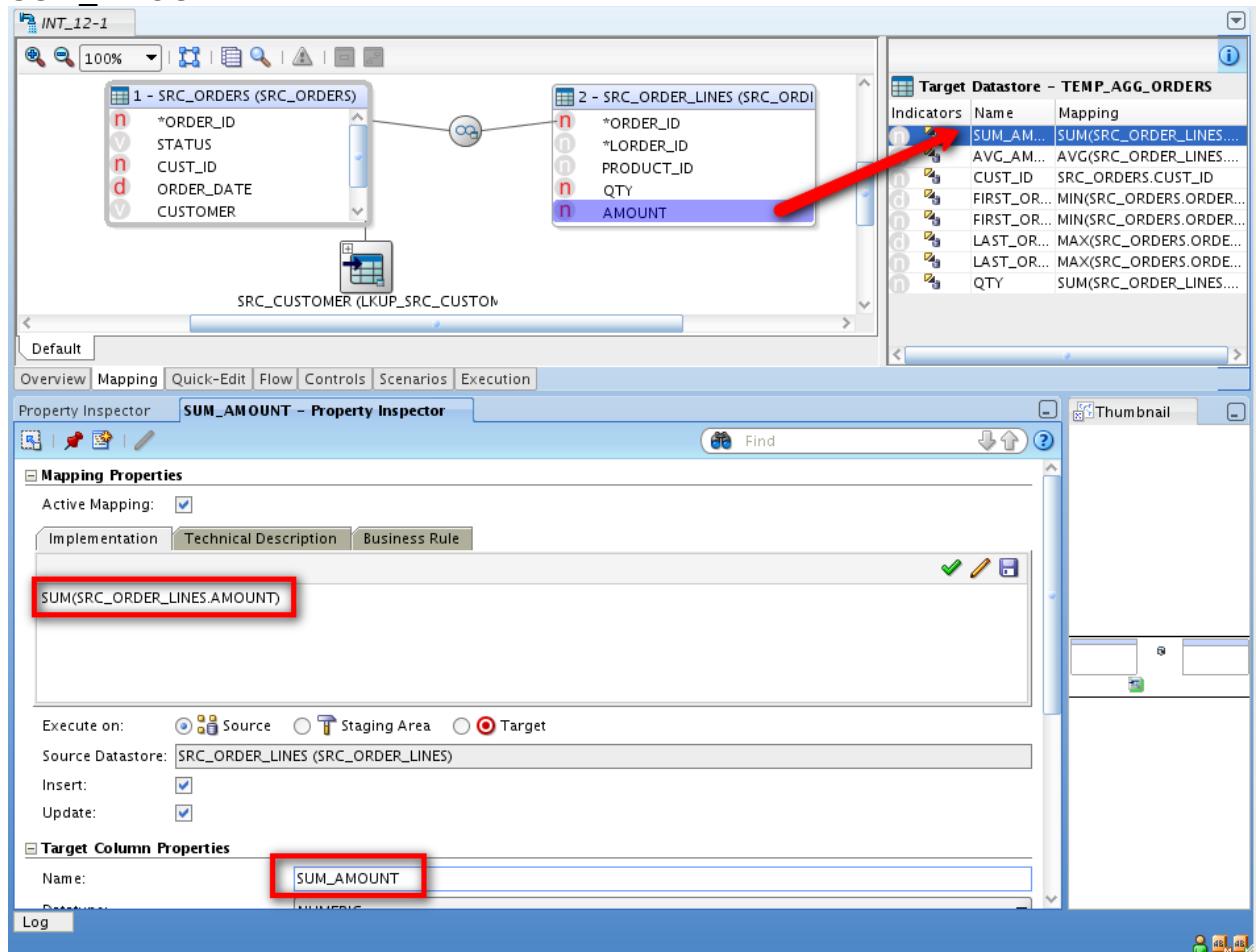
- k) Drag column QTY from SRC_ORDER_LINES source datastore to the Target area. Edit mapping expression to read: SUM(SRC_ORDER_LINES.QTY).



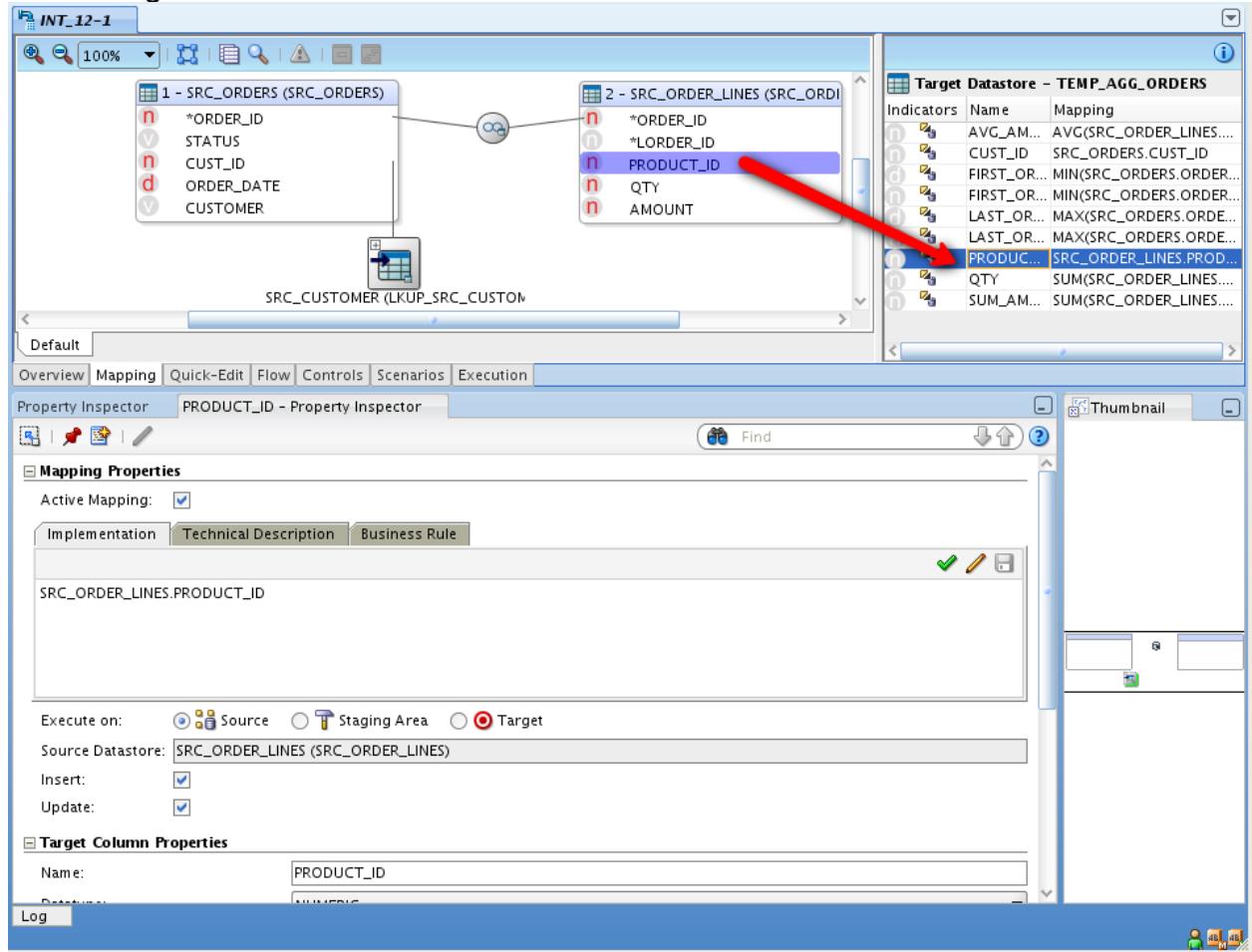
- I) Drag column AMOUNT from SRC_ORDER_LINES source datastore to the Target area. Edit mapping expression to read: AVG(SRC_ORDER_LINES.AMOUNT). In the Target datastore, rename this column to AVG_AMOUNT.



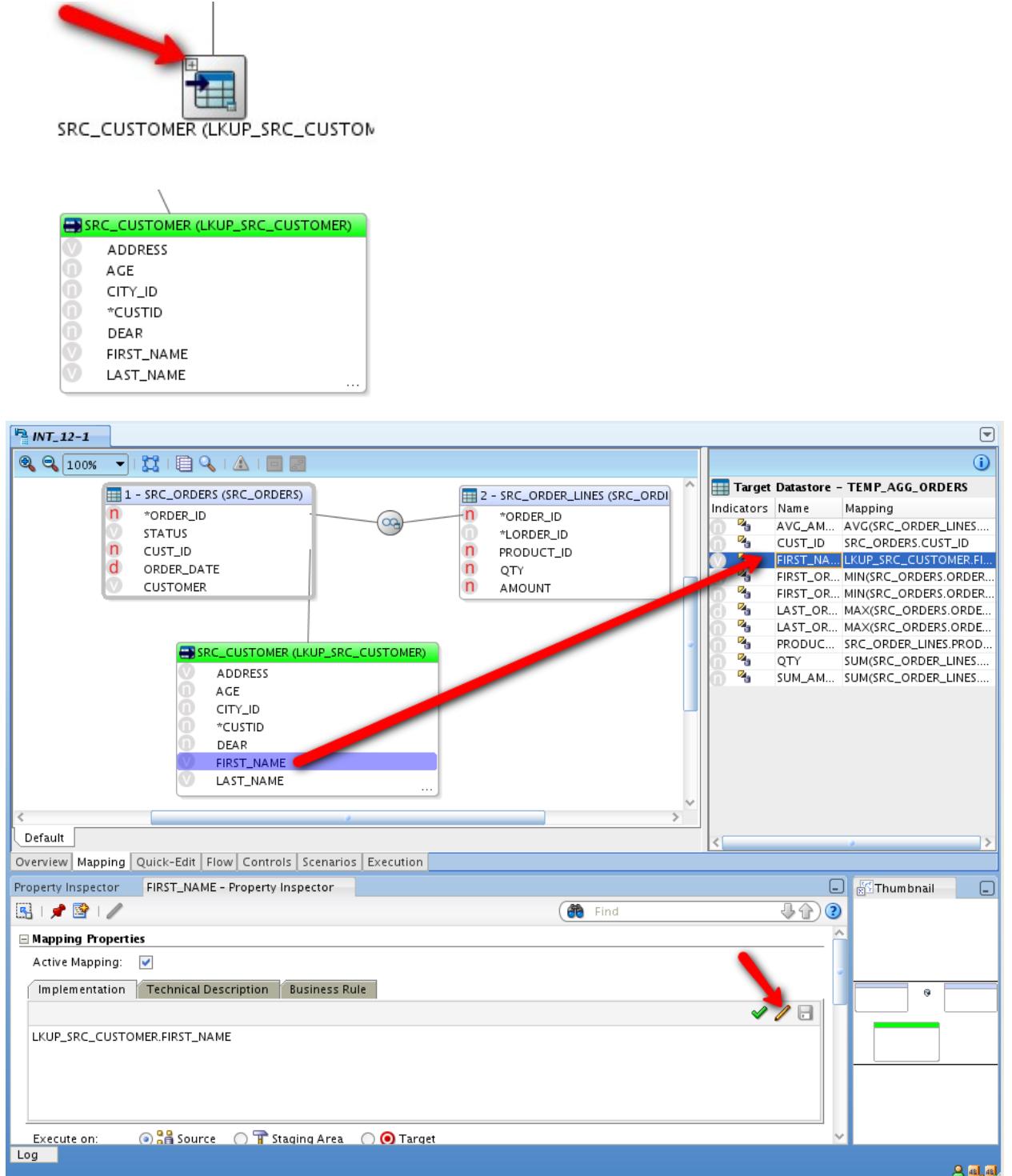
- m) Drag again column AMOUNT from SRC_ORDER_LINES source datastore to the Target area. Edit mapping expression to read: SUM(SRC_ORDER_LINES.AMOUNT). Rename this column to SUM_AMOUNT.



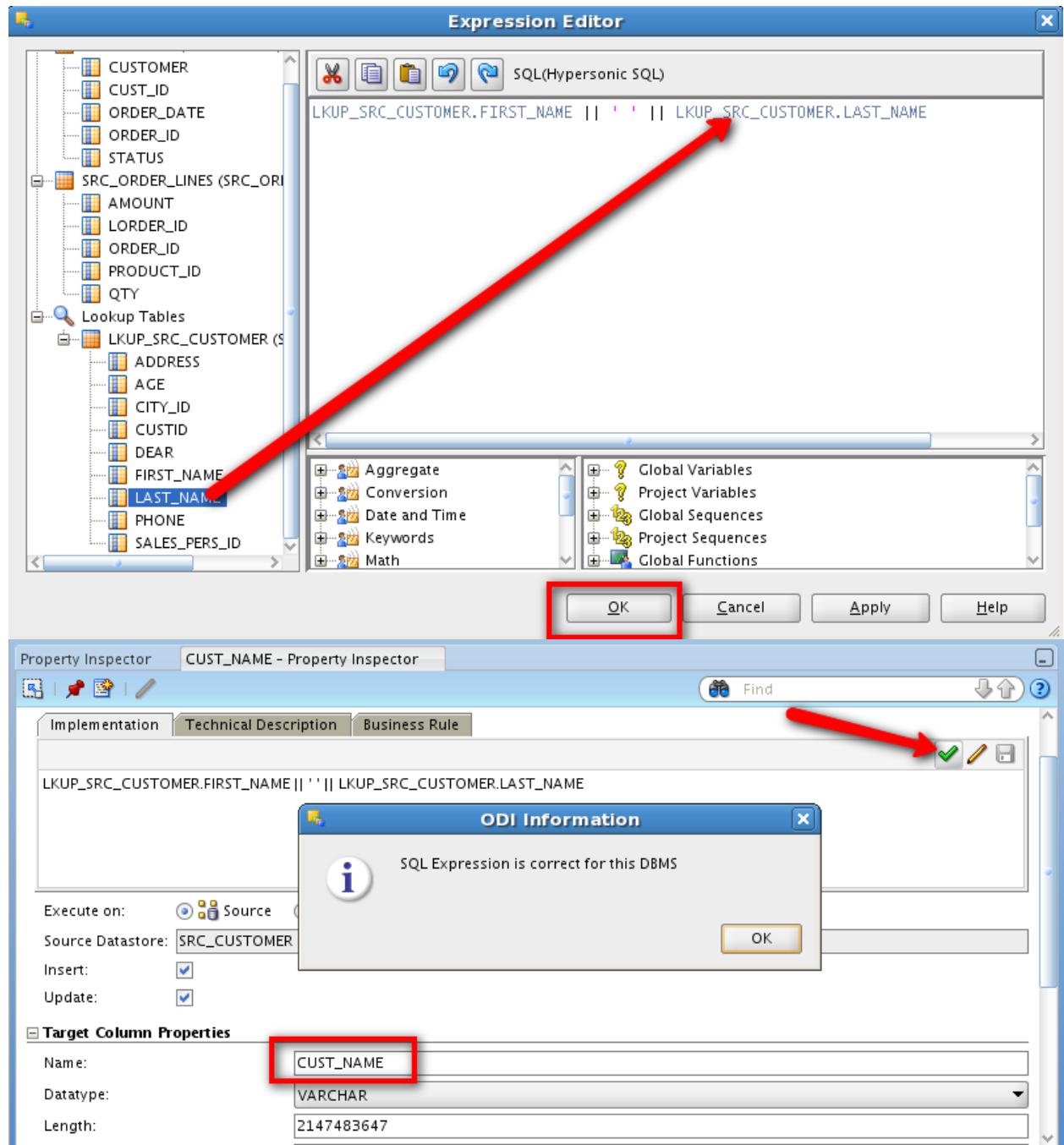
- n) Drag column PRODUCT_ID from SRC_ORDER_LINES source datastore to the Target area.



- o) In the Source area, expand lookup datastore. Drag FIRST_NAME column from the lookup table to the Target area. In the Target area, click the FIRST_NAME column. Start Expression editor , and then drag the LAST_NAME column from the left panel of Expression Editor to the expression



- p) Edit the expression to read: LKUP_SRC_CUSTOMER.FIRST_NAME || '' || LKUP_SRC_CUSTOMER.LAST_NAME as shown below. Click OK.
- Click button to validate the expression. Rename the column to CUST_NAME



- q) In the Target datastore, click AVG_AMOUNT column. In the Execute on section, select Staging Area. Repeat this step for all other columns, which have a function in mapping expression. Refer to the table below to verify execution location. Your Target datastore should now look as shown on the screenshot. Click Flow tab.

The screenshot shows the SAP Data Services interface for a process named INT_12-1. It displays the mapping between three datastores:

- 1 - SRC_ORDERS (SRC_ORDERS)**: Contains columns *ORDER_ID, STATUS, CUST_ID, ORDER_DATE, and CUSTOMER.
- 2 - SRC_ORDER_LINES (SRC_ORDER_LINES)**: Contains columns *ORDER_ID, *LORDER_ID, PRODUCT_ID, QTY, and AMOUNT.
- SRC_CUSTOMER (LKUP_SRC_CUSTOMER)**: Contains columns ADDRESS, AGE, CITY_ID, *CUSTID, DEAR, FIRST_NAME, and LAST_NAME.

The **Target Datastore - TEMP_AGG_ORDERS** table lists the indicators and their mappings:

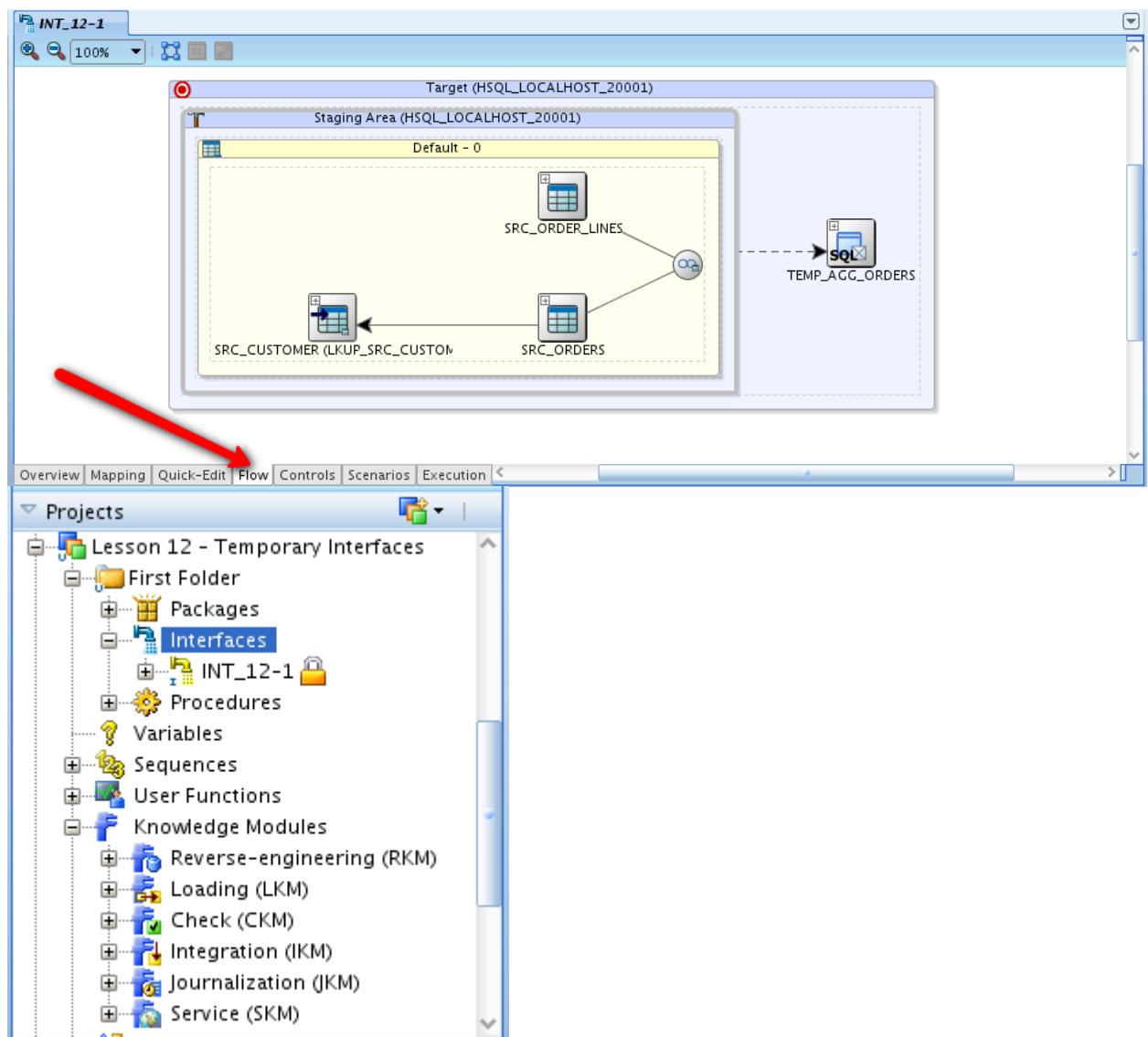
Indicators	Name	Mapping
T	AVG_AMOUNT	AVG(SRC_ORDER_LINES.AMOUNT)
U	CUST_ID	SRC_ORDERS.CUST_ID
U	CUST_NAME	LKUP_SRC_CUSTOMER.FIRST_N...
U	FIRST_ORD_DATE	MIN(SRC_ORDERS.ORDER_DATE)
U	FIRST_ORD_ID	MIN(SRC_ORDERS.ORDER_ID)
U	LAST_ORD_DATE	MAX(SRC_ORDERS.ORDER_DATE)
U	LAST_ORD_ID	MAX(SRC_ORDERS.ORDER_ID)
U	PRODUCT_ID	SRC_ORDER_LINES.PRODUCT_ID
U	QTY	SUM(SRC_ORDER_LINES.QTY)
U	SUM_AMOUNT	SUM(SRC_ORDER_LINES.AMOU...

In the **Property Inspector** for the AVG_AMOUNT indicator, the "Execute on" dropdown is set to "Staging Area". A red arrow points from the "Name" column in the target table to this dropdown.

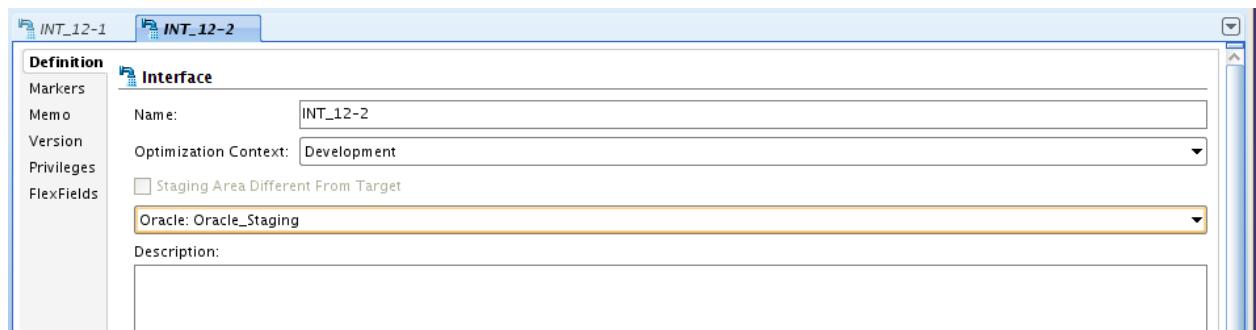
The bottom part of the screenshot shows the **Target Datastore - TEMP_AGG_ORDERS** table again, with the first seven rows highlighted by a red box. The columns are Indicators, Name, and Mapping.

Column	Execution location
SUM_AMOUNT	Staging Area
AVG_AMOUNT	Staging Area
CUST_ID	Source
CUST_NAME	Source
FIRST_ORD_DATE	Staging Area
FIRST_ORD_ID	Staging Area
LAST_ORD_DATE	Staging Area
LAST_ORD_ID	Staging Area
PRODUCT_ID	Source
QTY	Staging Area

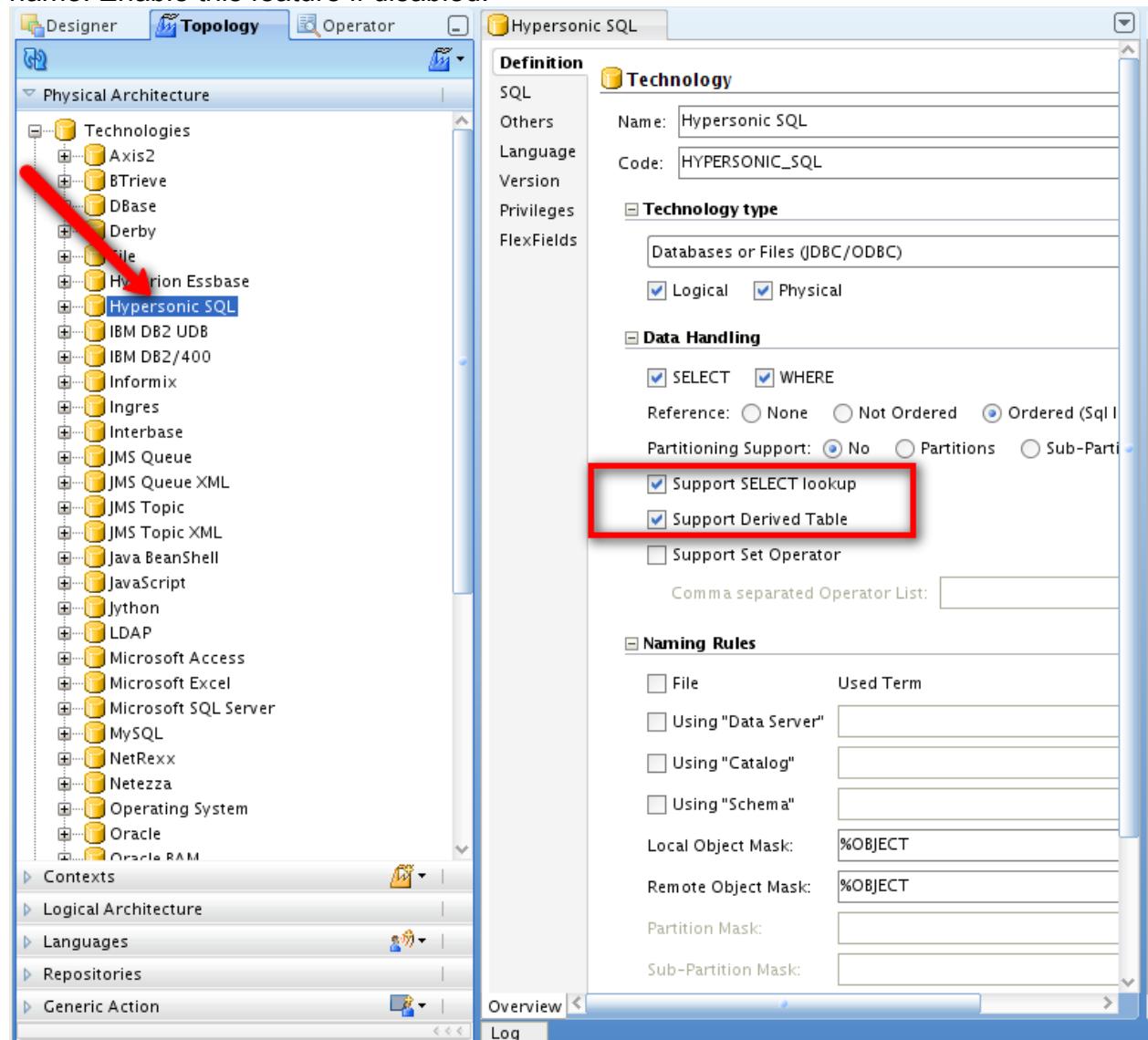
- r) Verify the flow, and then save your temporary interface. Verify that your temporary interface appeared in the tree view. Close your interface tab. Note the color of your newly created temporary interface in the objects tree view.



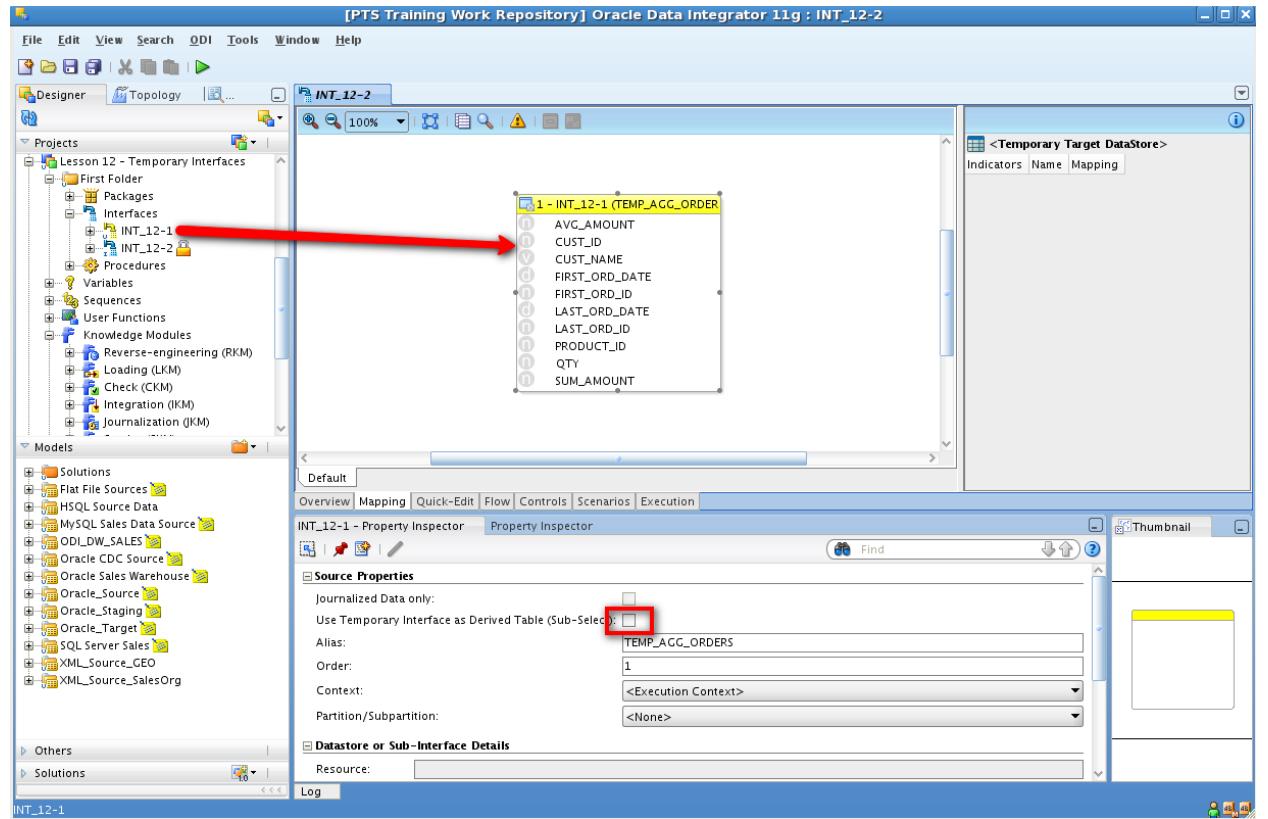
- 3) Create new interface, which use the newly created temporary interface as a source.
- a) In Lesson 12 – Temporary Interface project, start creating new ODI interface INT_12-2 as shown below. Open Mapping tab.



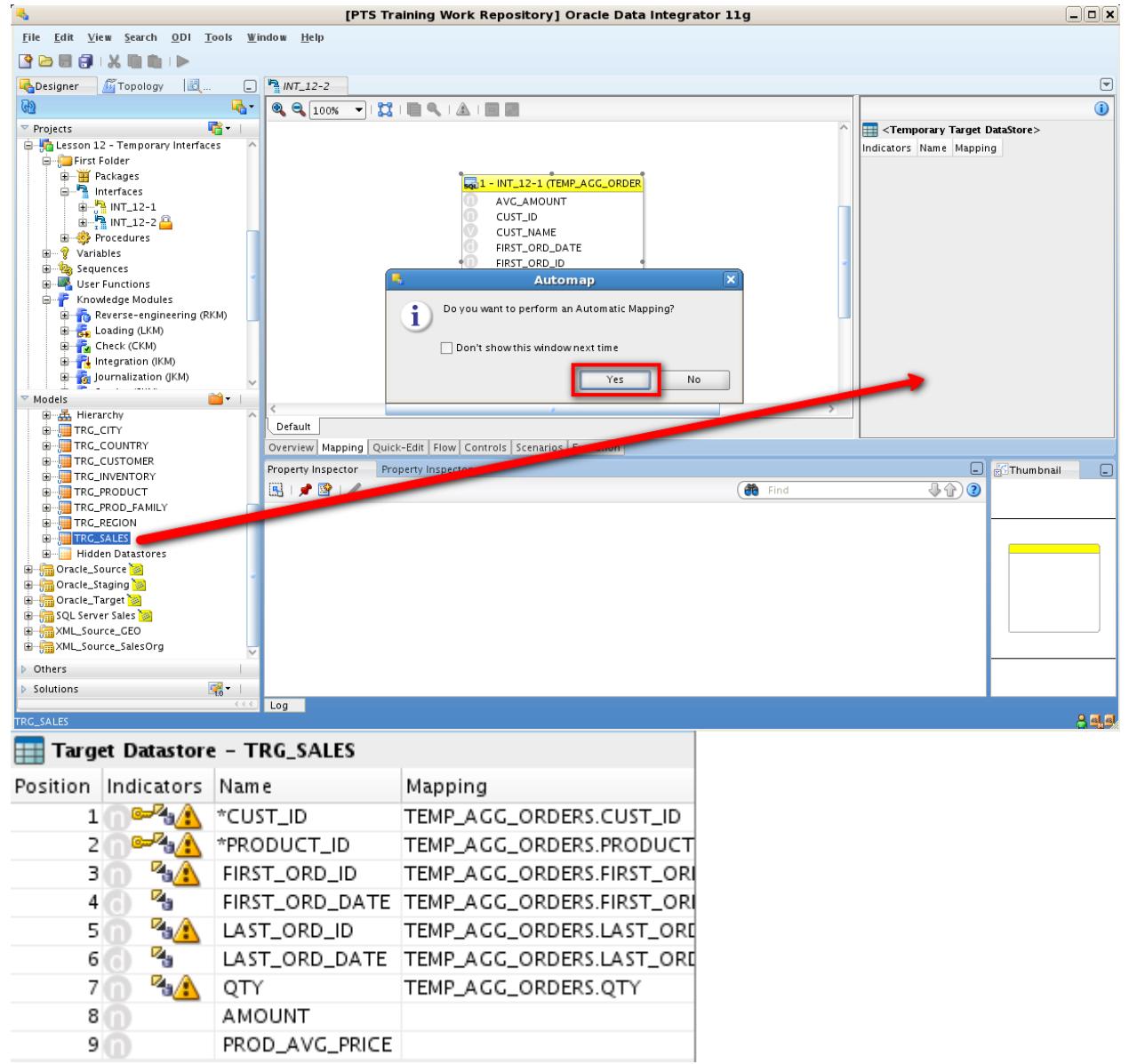
- b) Check if the Source Technology has derived table support. Go to Topology, expand Physical Architecture and then double-click technology name. Enable this feature if disabled.



- c) Drag temporary interface INT_12-1 from Projects tab to the Source area.



- d) Drag datastore TRG_SALES from Oracle Sales Warehouse model to the Target Datastore area. Click Yes to perform automatic mapping.



In the target datastore, select AMOUNT column. Drag SUM_AMOUNT column from the temporary interface in the source area to the Implementation tab in Mapping Properties area.

The screenshot shows the SAP Data Services interface with two panes. The left pane displays the contents of the temporary interface 'sq_1 - INT_12-1 (TEMP_AGG_ORDER)' with columns: AVG_AMOUNT, CUST_ID, CUST_NAME, FIRST_ORD_DATE, FIRST_ORD_ID, LAST_ORD_DATE, LAST_ORD_ID, PRODUCT_ID, QTY, and SUM_AMOUNT. The right pane shows the 'Target Datastore - TRG_SALES' with a mapping table:

Indicators	Name	Mapping
L	*CUST_ID	TEMP_AGG_ORDERS.CUST_ID
N	*PRODUCT_ID	TEMP_AGG_ORDERS.PRODUCT_ID
D	FIRST_ORD_ID	TEMP_AGG_ORDERS.FIRST_ORD_ID
D	FIRST_ORD_DATE	TEMP_AGG_ORDERS.FIRST_ORD_DATE
D	LAST_ORD_ID	TEMP_AGG_ORDERS.LAST_ORD_ID
D	LAST_ORD_DATE	TEMP_AGG_ORDERS.LAST_ORD_DATE
N	QTY	TEMP_AGG_ORDERS.QTY
N	AMOUNT	TEMP_AGG_ORDERS.SUM_AMOUNT
D	PROD_AVG_PRICE	

A red arrow points from the 'AMOUNT' column in the temporary interface to the empty mapping cell for 'PROD_AVG_PRICE' in the target datastore.

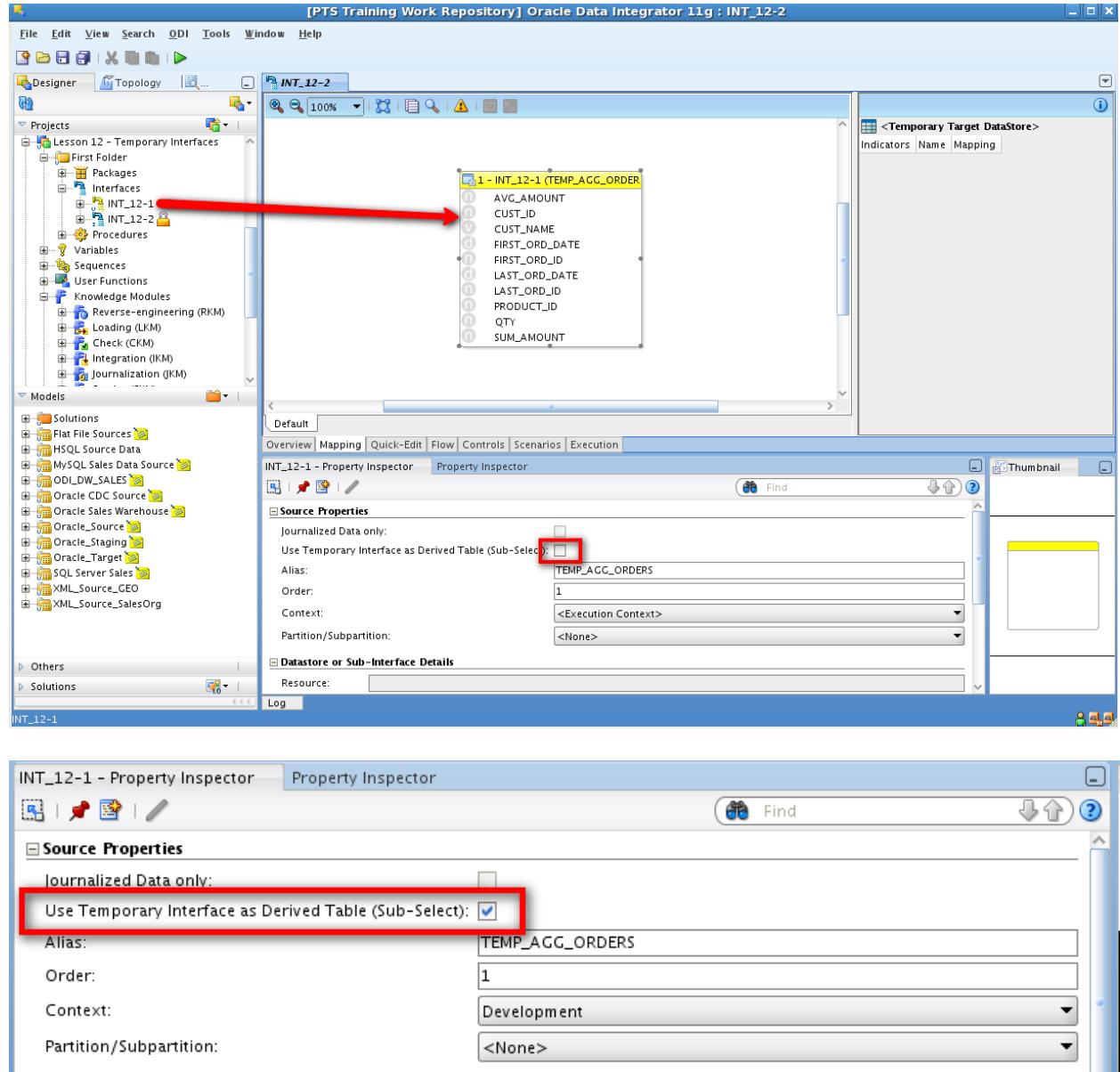
- e) Repeat previous step to map PROD_AVG_PRICE to AVG_AMOUNT column in the temporary interface as shown below.

The screenshot shows the SAP Data Services interface with the same two panes. The left pane now includes the 'PROD_AVG_PRICE' column. The right pane's mapping table now includes a row for 'PROD_AVG_PRICE':

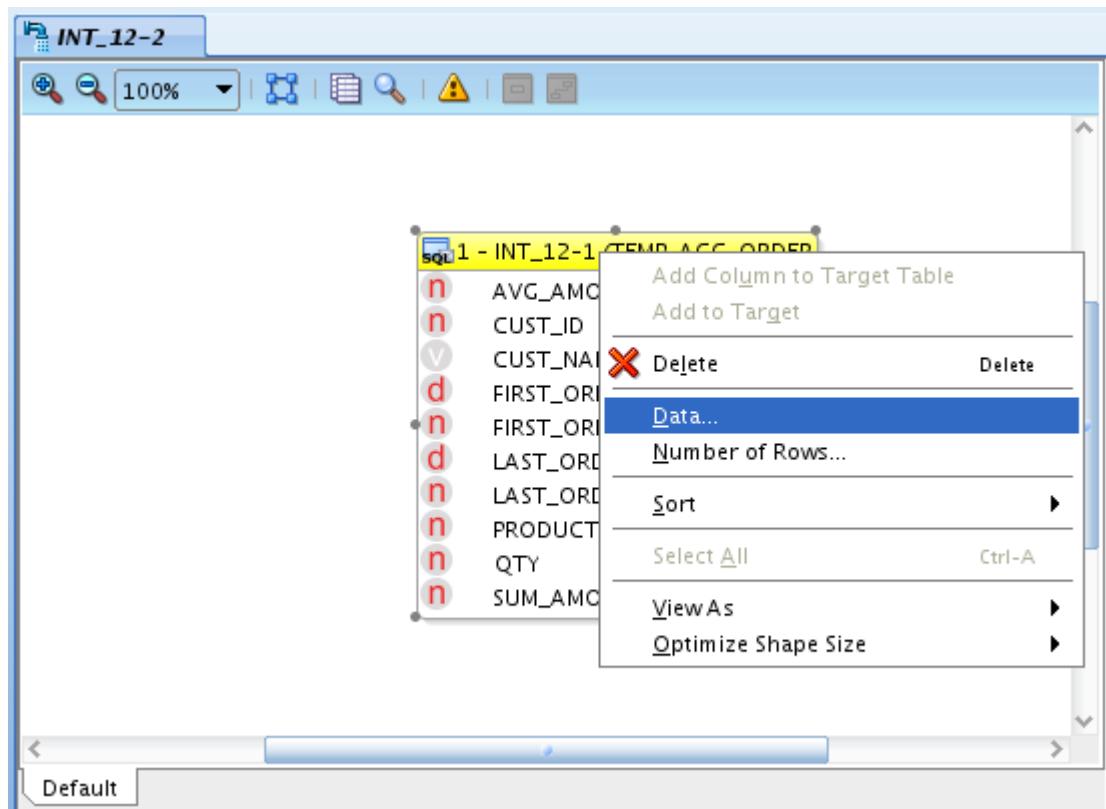
Indicators	Name	Mapping
L	*CUST_ID	TEMP_AGG_ORDERS.CUST_ID
N	*PRODUCT_ID	TEMP_AGG_ORDERS.PRODUCT_ID
D	FIRST_ORD_ID	TEMP_AGG_ORDERS.FIRST_ORD_ID
D	FIRST_ORD_DATE	TEMP_AGG_ORDERS.FIRST_ORD_DATE
D	LAST_ORD_ID	TEMP_AGG_ORDERS.LAST_ORD_ID
D	LAST_ORD_DATE	TEMP_AGG_ORDERS.LAST_ORD_DATE
N	QTY	TEMP_AGG_ORDERS.QTY
N	AMOUNT	TEMP_AGG_ORDERS.SUM_AMOUNT
D	PROD_AVG_PRICE	TEMP_AGG_ORDERS.AVG_AMOUNT

A red arrow points from the 'PROD_AVG_PRICE' column in the temporary interface to its mapped column in the target datastore.

- f) In the Source panel, click INT_12-1 (TEMP_AGG_ORDERS), scroll-down to the Source properties section, and then select “Use Temporary Interface as Derived Table (Sub-Select)” checkbox.



- g) To view data derived from the temporary interface, right-click the temporary interface and select Data. Close Data Editor window.



Data Editor

	FIRST_ORD_DATE	CUST_NAME	AVG_AMOUNT	PRODUCT_ID	CUST_ID	QTY
16	1998-06-23	William Baker	30879	7	106	34
17	1998-06-23	William Baker	5683	13	106	62
18	1998-06-23	William Baker	19934	14	106	49
19	1990-08-26	Jack Swenson	29526	3	107	49
20	1990-08-26	Jack Swenson	142821	5	107	45
21	1990-08-26	Jack Swenson	28990	6	107	41

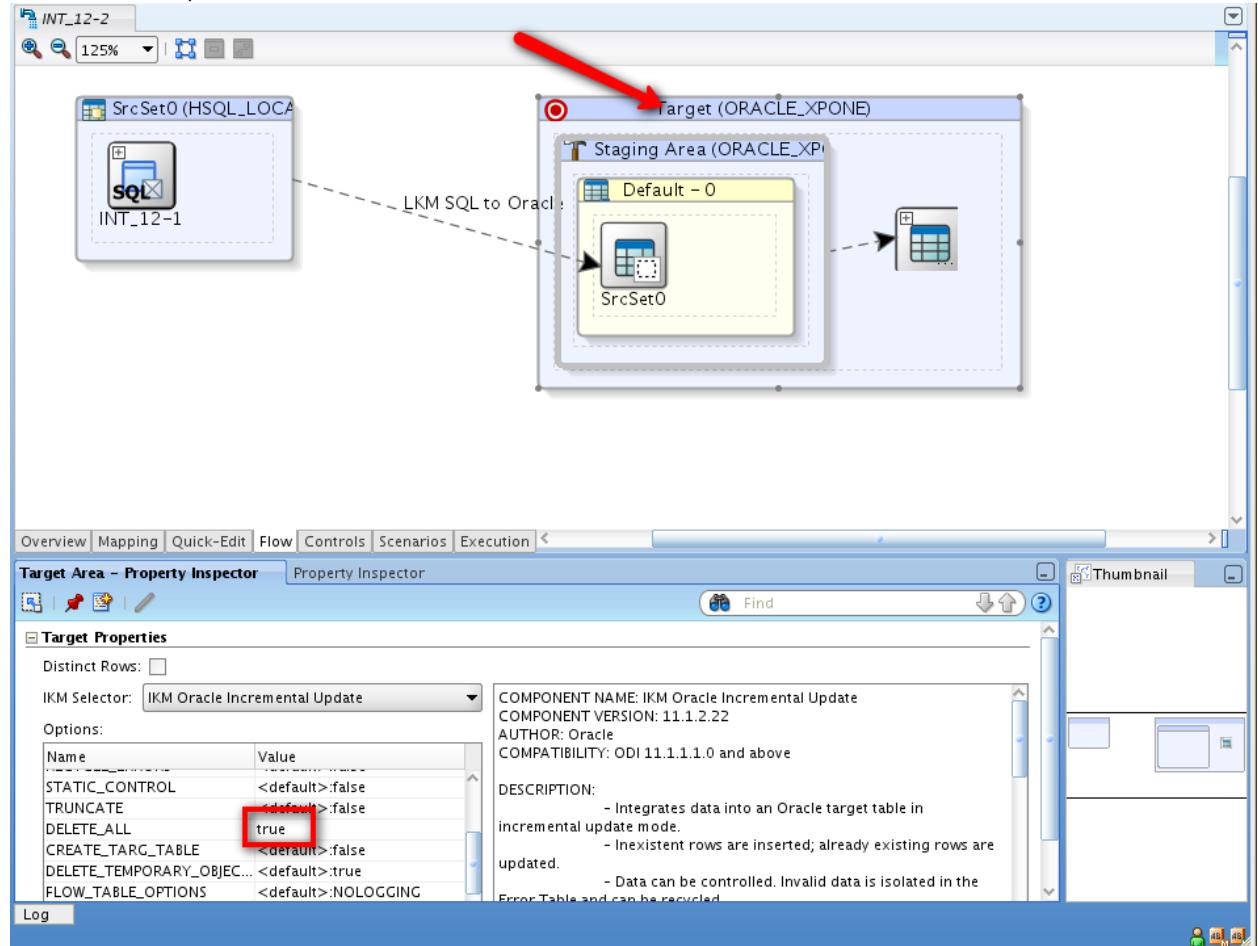
Executed Query:

```
select * from (
  select
    MIN(SRC_ORDERS.ORDER_DATE) FIRST_ORD_DATE,
    CUST_NAME,
    AVG(SRC_ORDER_LINES.AMOUNT) AVG_AMOUNT,
    LKUP_SRC_CUSTOMER.FIRST_NAME || ' ' || LKUP_SRC_CUSTOMER.LAST_NAME
    SRC_ORDER_LINES.PRODUCT_ID PRODUCT_ID,
```

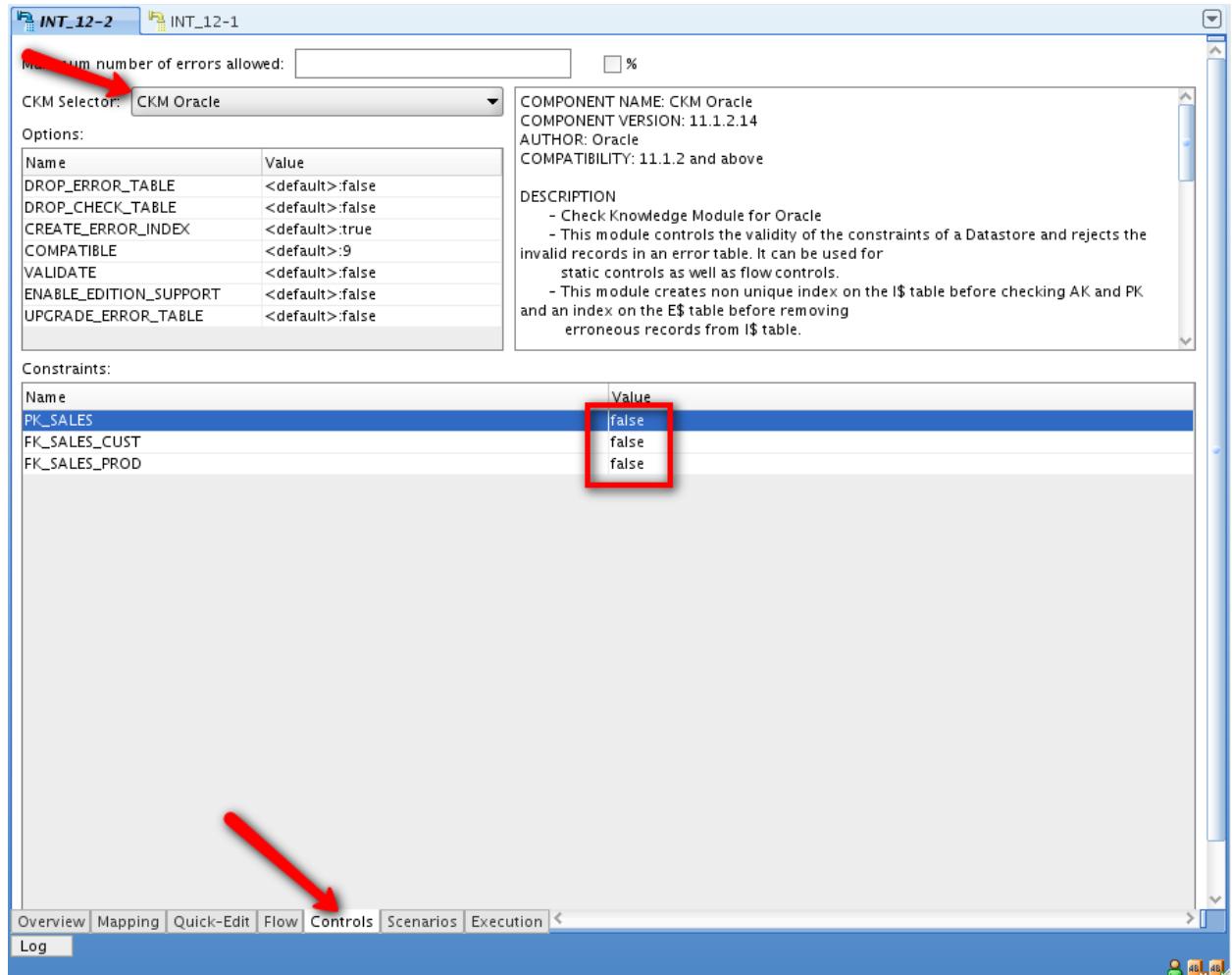
Record 1 of 50

Close

- h) Select Flow tab, click Target datastore, and verify the flow of your interface. In the Target Properties, set DELETE_ALL option to true. Save the interface, and then close the interface tab.

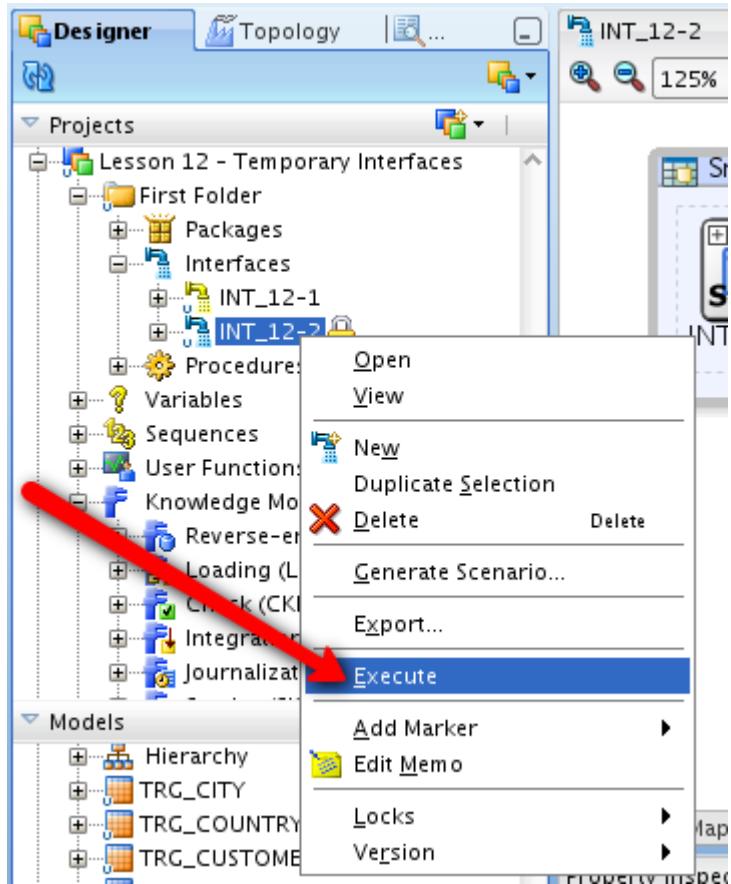


- i) Disable check constraints on Target datastore, since there are no corresponding foreign keys. Click on Controls Tab and set all the constraints to false.

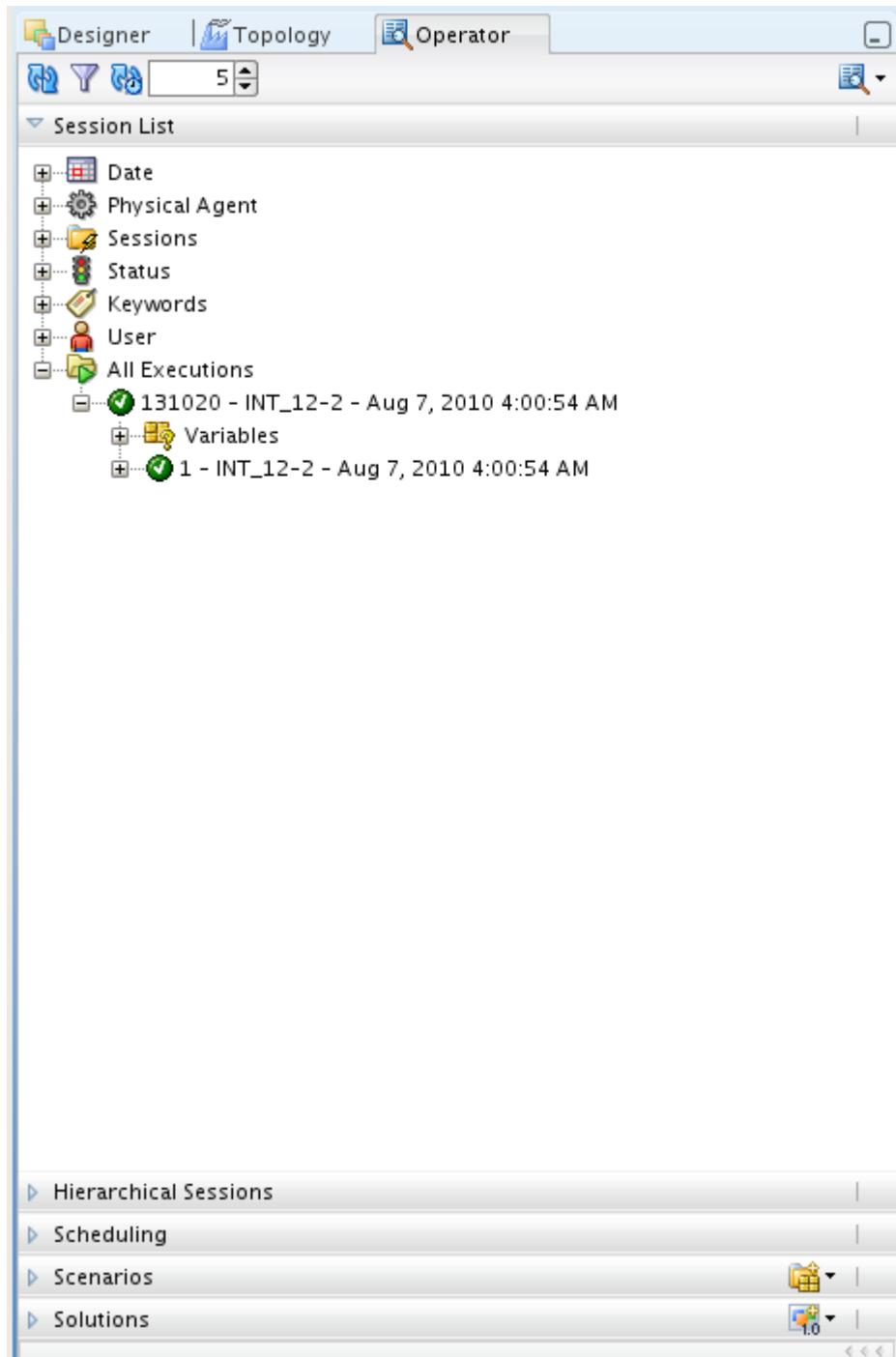


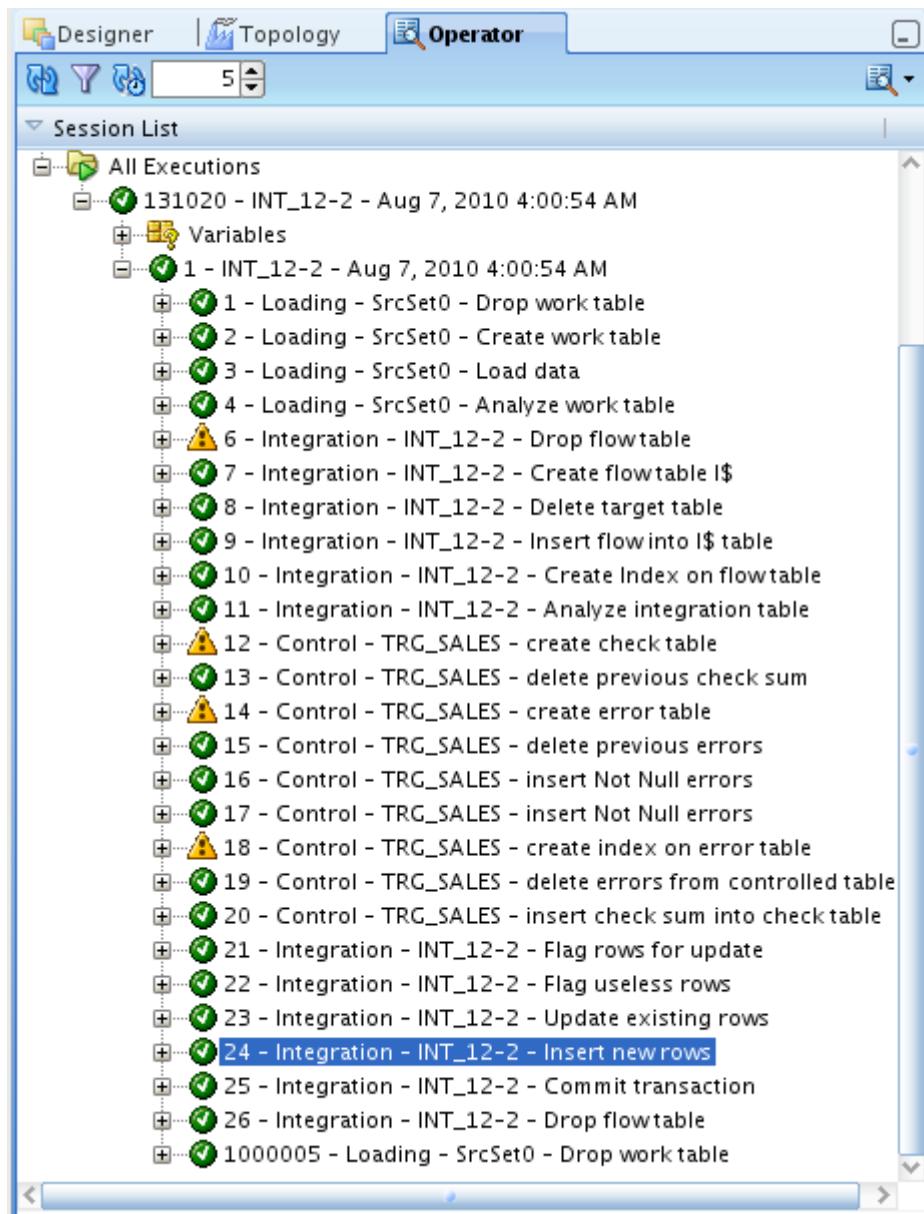
4) Execute interface INT_12-2 and verify execution results.

a) Execute interface INT_12-2.



- b) Open ODI Operator Navigator and verify that interface INT_12-2 executed successfully. Scroll down and open the task Integration- INT_12-2 – Insert new rows. Verify number of inserts.





INT_12-2 Session Task: Integration INT_12-1

Definition

Code
Connection
Privileges

Session Task

Task Name:	Integration	Status:	Done
	INT_12-2	Type:	Interface
	Insert new rows	<input type="checkbox"/> Ignore Errors	
Order:	24	210	Log Level: 3

Record Statistics

No. of Inserts:	149	No. of Updates:	0
No. of Deletes:	0	No. of Errors:	0
No. of Rows:	149		

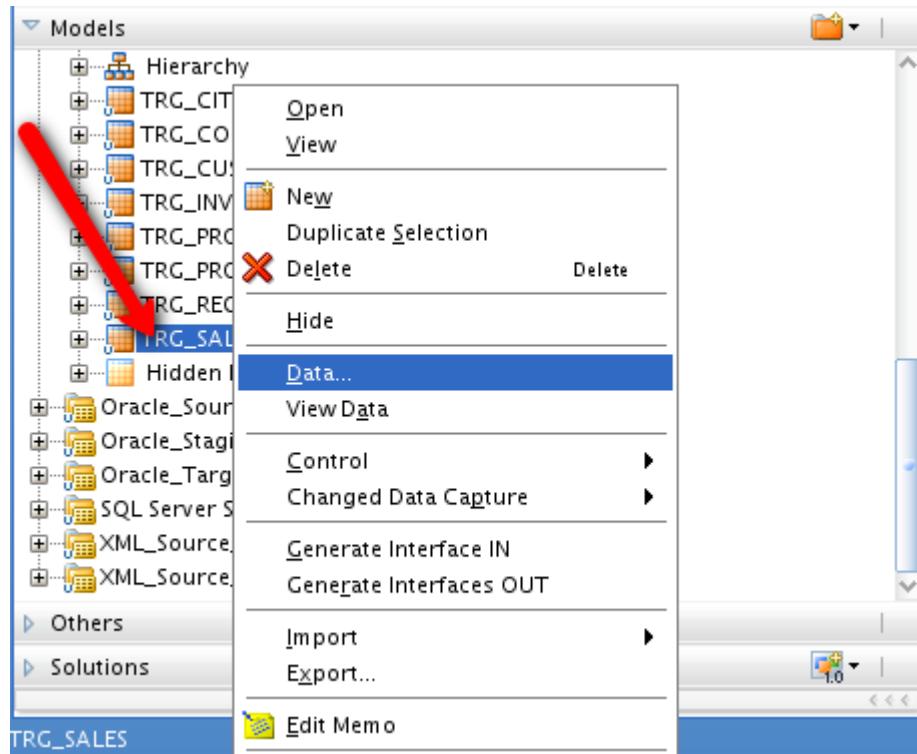
Execution Statistics

Start:	Aug 7, 2010 4:00:57 AM	End:	Aug 7, 2010 4:00:57 AM
Duration (seconds):	0	Return Code:	0

Message:

Overview < >

- c) In the Models tab in Oracle Sales Warehouse model, right-click TRG_SALES datastore, and select View Data. View rows inserted in the target datastore.



As an alternative, right-click the target datastore and select Data...

The screenshot shows the SAP Data Integration interface with two main windows:

- Top Window (Session Task: Integration):** Displays a mapping between a source table ('INT_12-1 (TEMP_AGG_ORDER)') and a target datasource ('TRG_SALES'). The target datasource has 9 rows. A context menu is open over the 8th row, with 'Data...' selected. The menu also includes options like Open, Add Column, Delete, Number Of Rows..., Sort, and Redo Auto Mapping.
- Bottom Window (Data Editor):** Shows the data for the 'TRG_SALES' table. The table has columns: CUST_ID, PRODUCT_ID, FIRST_ORD_ID, FIRST_ORD_DATE, and LAST_ORD_ID. The data is as follows:

	CUST_ID	PRODUCT_ID	FIRST_ORD_ID	FIRST_ORD_DATE	LAST_ORD_ID
21	107	6	7	08-26 00:00:00.0	80
22	201	2	8	08-18 00:00:00.0	40
23	201	4	8	08-18 00:00:00.0	8
24	201	10	8	08-18 00:00:00.0	81

The 'Executed Query:' field contains the SQL statement: `select * from CUST_DW_DEV.TRG_SALES`. The status bar at the bottom left says 'Record 1 of 50'. A 'Close' button is at the bottom right of the Data Editor window.

Lab 13: Change Data Capture with ODI

Introduction

Changed Data Capture (CDC), also referred to as *Journalizing*, allows you to trap changes occurring on the source data. CDC is used in Oracle Data Integrator to eliminate the transfer of unchanged data. This feature can be used for example for data synchronization and replication. Journalizing can be applied to models, sub-models or datastores based on certain type of technologies.

Changed Data Capture is performed by journalizing models. Journalizing a model consists of setting up the infrastructure to capture the changes (inserts, updates and deletes) made to the records of this model's datastores.

Oracle Data Integrator supports two journalizing modes:

- Simple Journalizing tracks changes in individual datastores in a model.
- Consistent Set Journalizing tracks changes to a group of the model's datastores, taking into account the referential integrity between these datastores. The group of datastores journalized in this mode is called a **Consistent Set**.

Process Overview

A common task that is performed using ODI is to implement Changed Data Capture. In this practice, students implement Changed Data Capture to detect changes in the source environment for the interface. You will capture changes of data in the SRC_CITY table in MYSQL DataSource model. Then, you modify the interface to process changed records only.

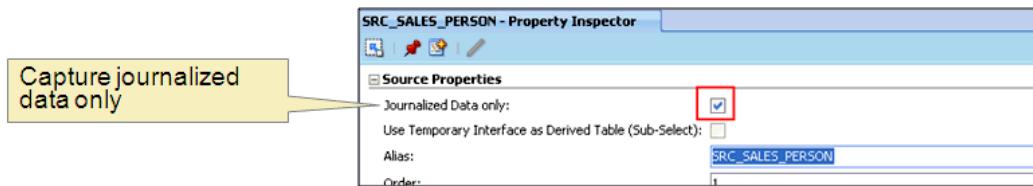
In this practice, you perform changed data capture in an interface, using the JKM MySql Simple knowledge module.

Below are the steps that have been completed:

1. Import the JKM MySql Simple knowledge module.
2. Create a model Oracle CDC Source, specifying this knowledge module in the Journalizing tab.
3. Reverse engineer the model, verifying the resulting structure.

Below are the steps that have to be completed:

1. Add the models' SRC_CITY table to CDC, and start the Journal using the default subscriber SUNOPSIS.
2. Use Data viewer to change data in the table and verify in Journal Data, that the data change was captured.
3. Create am interface to process changed data only.

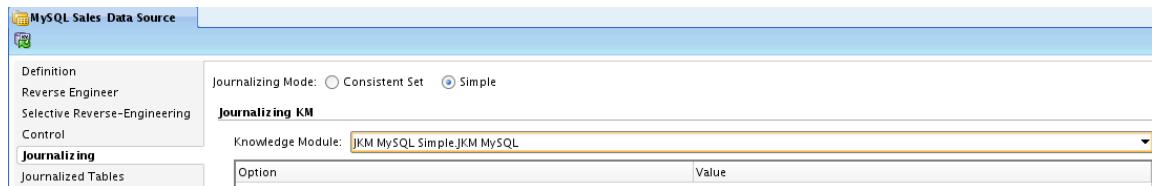


Scenario

As a database administrator you are responsible for data loading, transformation, and validation. You want to be able to detect changes in your source environment. For that purpose you need to implement Changed Data Capture to your data integration project.

Instructions

- 1) Open the MYSQL Sales DataSource from the Model tab. Verify if the Journalizing is set to Simple and has the corresponding Knowledge Module. If not set the mode to simple. Select JKM MySQL Knowledge Module as shown.



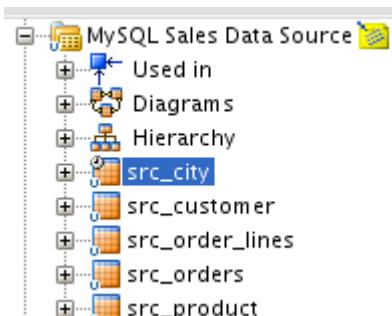
- 2) Open the Oracle CDC Source from the Model tab. Select the logical schema as ORACLE_SALES_DW. Go to Reverse Engineer tab and select the context as Development and save the model.

The screenshot shows the Oracle CDC Source configuration interface. On the left, there is a navigation pane with options like Reverse Engineer, Selective Reverse-Engineering, Control, Journalizing, Journalized Tables, Markers, Services, Memo, and Version. The main area is titled 'Model' and contains the following fields:

Name:	Oracle CDC Source
Code:	COPY OF ORA_CDC_SRC
Technology:	Oracle
Logical Schema:	ORACLE_SALES_DW
Action Group:	<Generic Action>
Default Folder:	(empty)

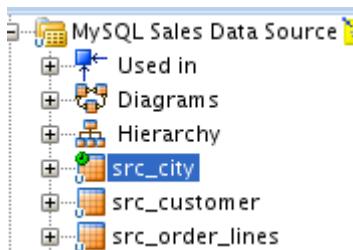
- 3) Set up the CDC Infrastructure. You will start the CDC on the SRC_CITY table in MYSQL Sales DataSource model.

- a) To add the table to CDC, expand the MYSQL Sales DataSource model, right-click the SRC_CITY table and select Change Data Capture > Add to CDC. Click Yes to confirm.

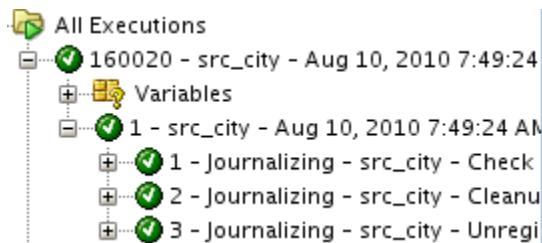


- b) Click the Refresh icon. The small yellow clock icon is added to the table. Right-click SRC_CITY table again and select Changed Data Capture > Start Journal.

- c) In this practice, you use the default subscriber SUNOPSIS. For that reason, you do not have to add another subscriber. Click OK to confirm that your subscriber is SUNOPSIS. Click OK. In the Information window, click OK again. Click Refresh  and verify that the tiny clock icon at SRC_CITY table is green now. This means that your journal is properly started.



- d) Click the ODI Operator icon to open the Operator. Click Refresh. Select All Executions and verify that the SRC_CITY session executed successfully.



4) View data and changed data.

- a) In the Designer window, open models tab. Right-click the SRC_CITY datastore and select Data.

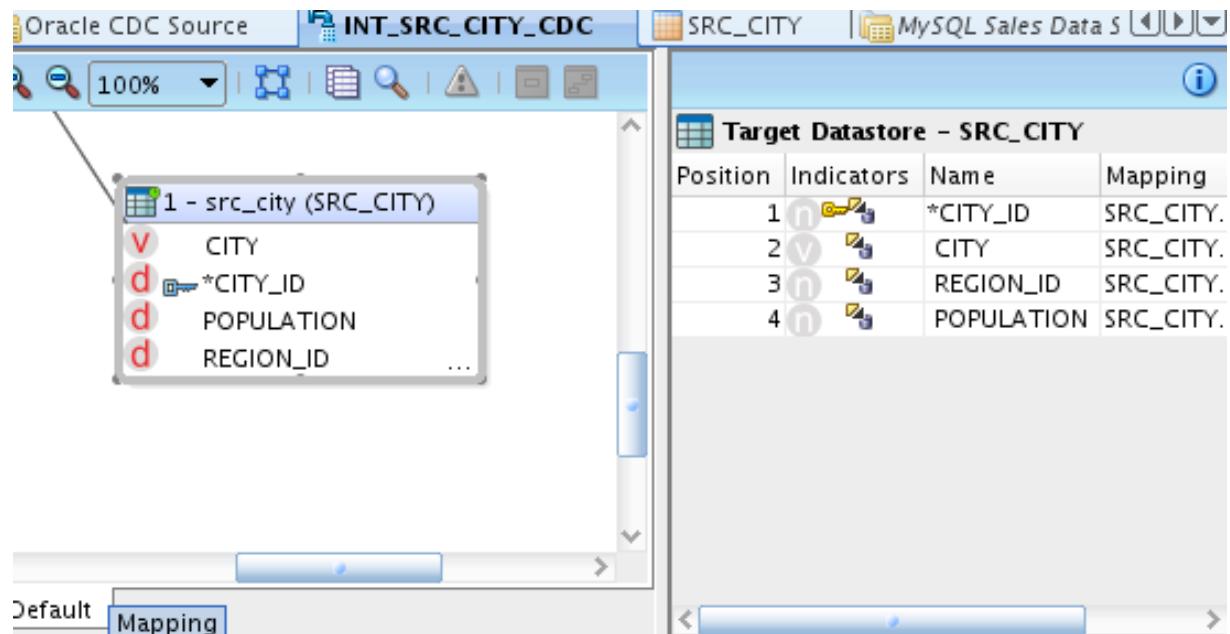
- b) Select the row with CITY_ID = 11. Change the value of the Population column to “122345”. Select the row with CITY_ID = 22. Change the value of the CITY column to “PHOENIX”. Similarly, select the row with CITY_ID = 27, and then change the REGION_ID to “23”. Save your changes and close the tab.

	CITY_ID	CITY	REGION_ID	POPULATION
1	10	Houston	20	743113
2	11	Dallas	20	822416
3	12	San Francisco	21	157574
4	13	Los Angeles	21	743878
5	14	San Diego	21	840689
6	15	Chicago	23	616472
7	16	Memphis	23	580075
8	18	Boston	22	275581
9	19	Washington D.C.	22	688002
10	21	Versailles	30	625097
11	22	Fontainebleau	30	285559
12	23	Lyon	31	689978
13	24	Grenoble	31	488825
14	25	Albertville	31	959909
15	27	Nice	32	482282

- c) Right-click the table again and select View Data. Scroll down, and verify that the rows are modified. Close the tab.
- d) To verify that your changed data is captured, right-click SRC_CITY, and select Change Data Capture > Journal Data. Find the captured changed records in the journal data. Close the tab. You should see three records
- 5) Create an interface that processes captured data changes. The interface loads the SRC_CITY datastore in the Oracle CDC Source model with the content of the SRC_CITY table from the MYSQL Sales DataSource model. The purpose of this interface is to process and load only changed data.
- In the Projects tab, expand Lesson -13 CDC> HandsOn > Interfaces. Right-click Interfaces, and then select New Interface.
 - In the Interface: NEW window, enter INT_SRC_CITY_CDC as the name of the interface. Ensure that the Optimization Context is Development. Click the Mapping tab.

- c) Open the Models tab. In the tree view, expand the Oracle CDC Source model. Drag the SRC_CITY datastore from the tree view to the Target Datastore zone.

Expand the MYSQL Sales DataSource model and drag the SRC_CITY datastore from the model tree to the Sources zone of your diagram. Click Yes to perform Automatic Mapping. Click the SRC_CITY (Source) datastore, and then select Journalized Data Only checkbox. Save you interface.



- d) Right-click the caption of source datastore and select Data. Scroll down to preview the current source data. Close the Data window.
- e) Click Flow tab, click the Staging Area caption and verify that selected IKM is IKM Oracle Incremental Update. Select Distinct Rows.
- f) Save you interface and close the tab. Execute the interface to process only journalized records.
- 6) Verify the execution results.
 - a) Open the Operator. Click Refresh. Expand the All Executions node and verify that you session INT_SRC_CITY_CDC executed successfully.
 - b) View data from model SRC_CITY Oracle CDC data store and verify that only three changed rows are updated in the target datastore.
- 7) In the Models tab, right-click SRC_CITY MYSQL Sales DataSource datastore and select Drop Journal. Click OK, and then click OK again. Right-click SRC_CITY datastore again, and then select Remove from CDC.

Lab 14: Version, Solution and Migration

Introduction

Oracle Data Integrator provides a comprehensive system for managing and safeguarding changes. The version management system allows **flags** on developed objects (such as projects, models, etc) to be automatically set, to indicate their status, such as new or modified. It also allows these objects to be backed up as stable checkpoints, and later restored from these checkpoints.

A **version** is a backup copy of an object. It is checked in at a given time and may be restored later. Versions are saved in the Master repository. The following objects in ODI can be versioned: Projects, Folders, Packages, Interfaces, Procedures, Sequences, Variables, User Functions, Models, and Solutions. Creating a version creates the XML definition of the object, and then stores it in compressed form in the Master repository version tables. Versions can then be restored into any connected Work repository. ODI provides a Version Browser to display versions. Versions of ODI objects can be compared by using the Version Comparison Tool.

A **solution** is a comprehensive and consistent set of interdependent versions of objects. Like other objects, it can be checked in at a given time as a version, and may be restored at a later date. Solutions are saved into the master repository. A solution assembles a group of versions called the solution's **elements**. A solution is automatically assembled using cross-references. By scanning cross-references, a solution automatically includes all dependent objects required for a particular object. You can also manually add or remove elements into and from the solution. The following objects may be added into solutions: Projects, Models, Scenarios, Global Variables, User Functions and Sequences.

Another important aspect of organizing and managing integration projects is the ability to **import** and **export** objects to share them with other projects. Exporting and importing Oracle Data Integrator objects means transferring objects between different repositories. Oracle Data Integrator uses internal identifiers (ID) to ensure object uniqueness across several work repositories. Oracle Data Integrator stores all objects in a relational database schema (the Repository) with dependencies between objects. Repository tables that store these objects maintain these dependencies as references using the IDs. Because object IDs are transferred as part of the export and import process, you need to understand the implications of the operation you are performing. Using solutions and

versioning is the recommended way of maintaining the dependencies automatically when doing export/import.

Prerequisites

Completion of **Lab 4-1: Creating Master and Work Repository**.

Process Overview

ODI provides easy to use built in version control and solution management mechanism to manage the ODI objects. In typical development cycle, developers will be uploading ODI projects/scenarios to the work repository. Developer and/or admin can version the projects or can create solutions to group related objects to better version control the solutions. This allows for easy management of ODI metadata for version control and migration.

In this practice, you will experience simple version creation of ODI project, create solution to understand grouping of objects, view version control reports and, perform simple import/export.

For this lab you can use any of the previously completed projects and perform activities mentioned for this lab. We will use *PTS Lab 03 – Building First Interface – Solved* as an example.

Below are the steps that have to be completed for this lab

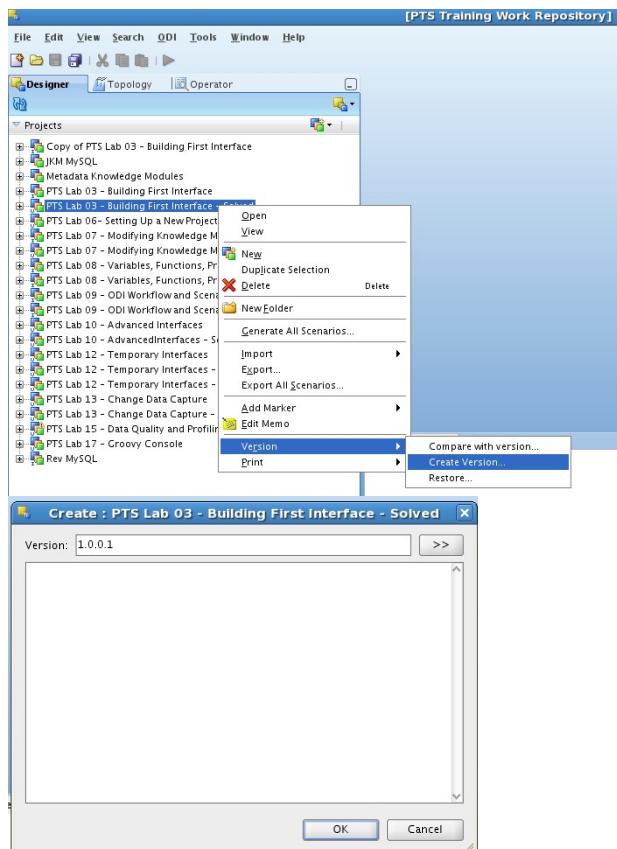
- Create New Version for the project
- Create New Solution to group the related objects for the project
- View version comparison report
- Export and import Operation.

Scenario

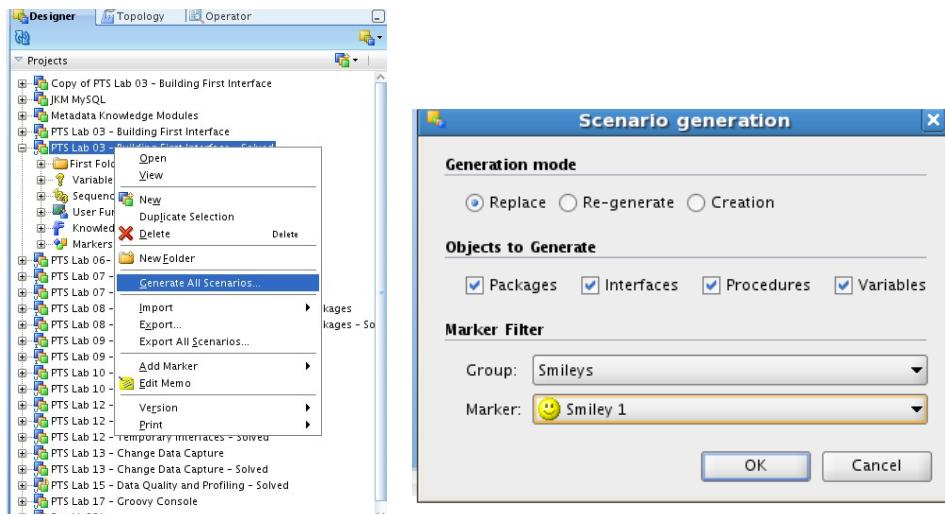
As a Data Integration Lead, you are also responsible for managing ODI project and versioning ODI objects for migration purposes as well managing the development to production cycle. You want to be able to track changes to ODI objects, version them and migrate them in a manageable group of objects.

Instructions

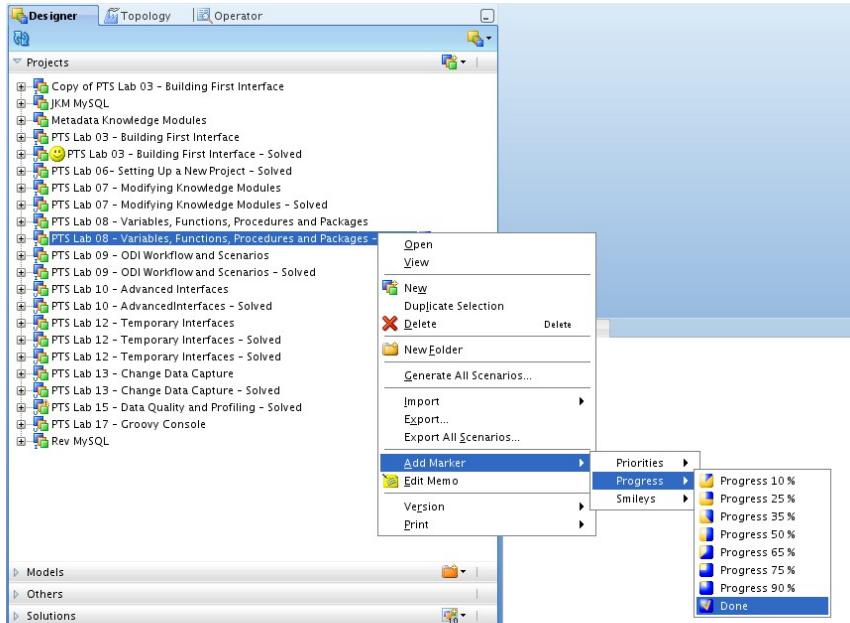
- 1) Right Click on Project and select Version -> Create Version Menu. A Version Creation window will show up. One can explicitly provide version number based on development standard of the projects or can continue with system provided versions. We will use auto generated version number for the lab. Click OK with default version number.



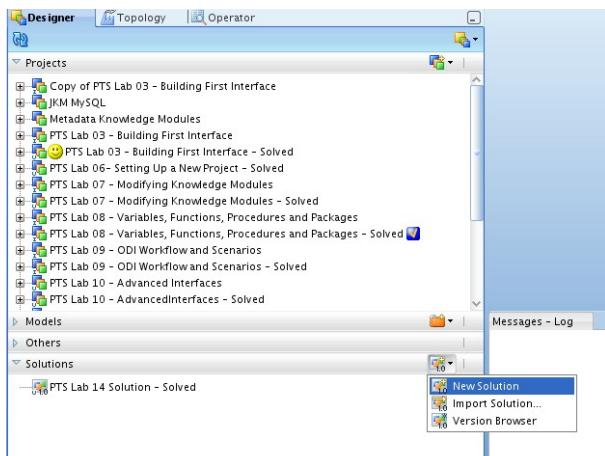
- 2) Generate all scenarios by right clicking on the project and selecting Generate All Scenarios. You can keep 'Replace' as a Generation Mode and keep all objects selected as part of scenario. A new feature is enabled to provide better grouping and search option for the metadata within ODI. You can select Group Smileys and Marker as Smiley 1. Click OK.



Note: Markers may not display this way. You can also add marker by right clicking on objects. See example below and add markers to couple of projects to see them ☺.



- 3) We will create Solution to group all the related objects for project PTS Lab 03 so that when we migrate solution to different repository, model, KM, interface, etc. will all move together. Expand Solutions section from left pane and click on icon to create new Solution.

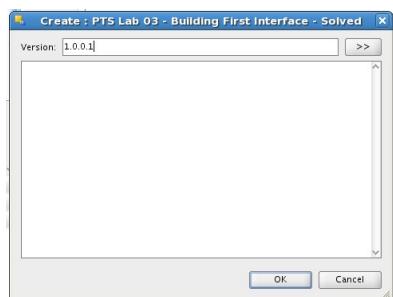
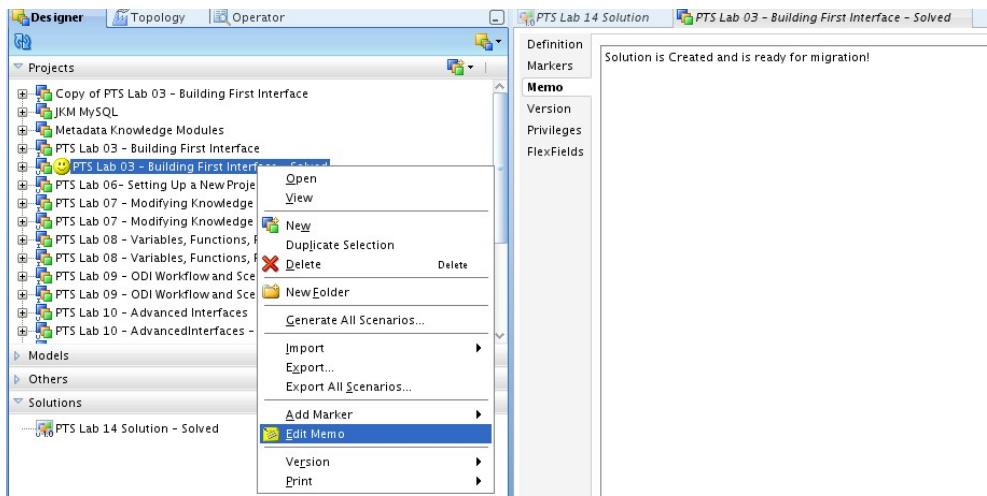


On Definition tab for Solution, give it a name PTS Lab 14 Solution. Drag a PTS Lab 03 –Building First Interface Project from project pane and drop it on the right side within Principal Elements section. You will see how all related objects are automatically imported into solution's Required Elements section. You can also add more components if needed as a part of solution. Click Save All to save your solution.

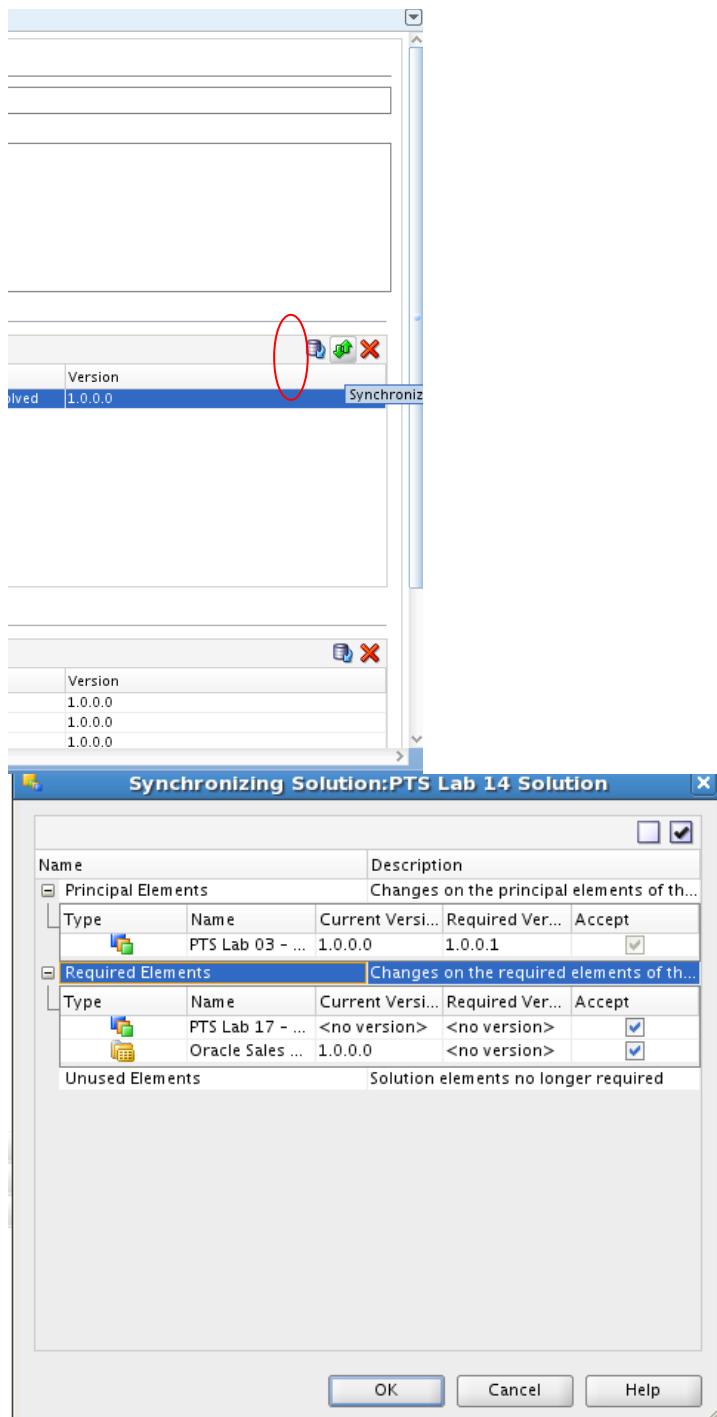
Type	Name	Version
Project	PTS Lab 03 - Building First Interface - Solved	1.0.0.0

Type	Name	Version
Project	Metadata Knowledge Modules	1.0.0.0
Project	JKM MySQL	1.0.0.0
Model	Oracle Sales Warehouse	1.0.0.0

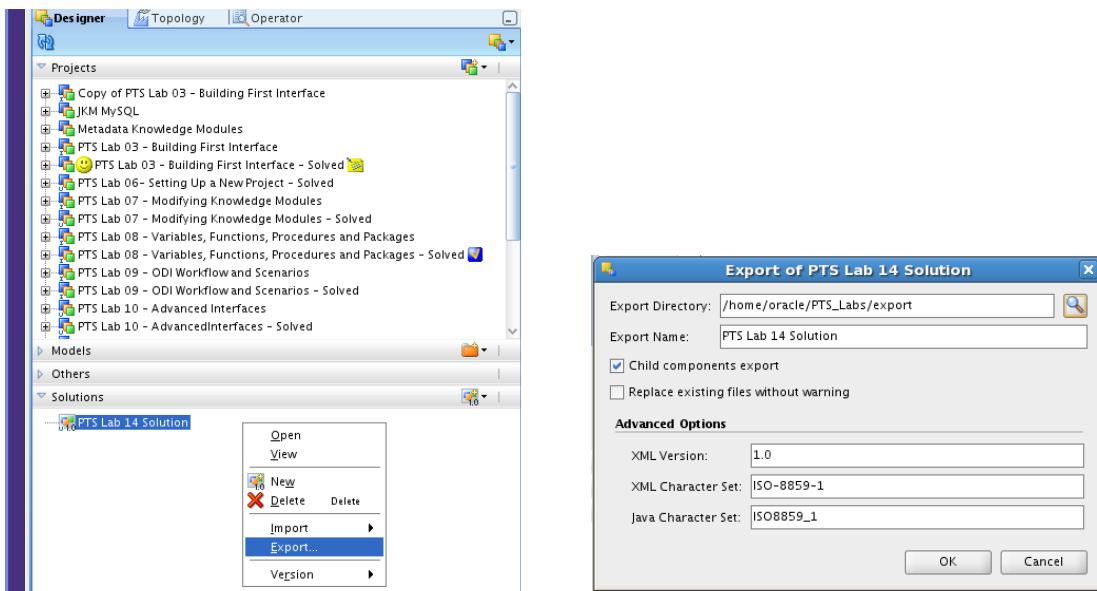
- 4) Update Lab 3 with notes and create new version. Right click in PTS Lab 3 on project and Edit Memo. Enter Some description on the right side memo box: 'Solution is created and ready for migration!'. Save all your changes. Generate new version for the Project.



- 5) Open your Lab 14 Solution if not already open. Click on synchronize button to update solution with new project version. You will be shown synchronizing update window to verify the changes. Click OK. You should see new project version for Principal Elements section.



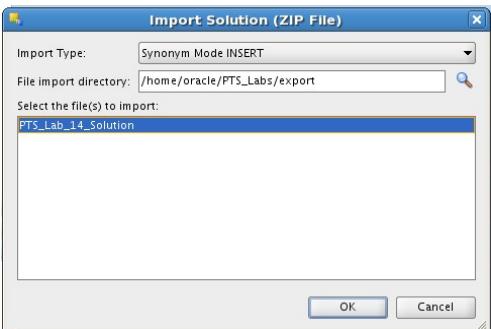
- 6) We will export solution and import it into different repository. Right click in solution and select export. Select /home/oracle/PTS_Labs/export as export directory and click OK for the export wizard.



Export zip file will be saved into specified directory. You can verify that from terminal window. Alternatively, you can double-click Oracle's home folder on the Desktop, then drill down from PTS_Labs to open up export folder to see contents.

```
oracle@pts:~/PTS_Labs/export$ pwd
/home/oracle/PTS_Labs/export
oracle@pts:~/PTS_Labs/export$ ls
SOL_PTS_Lab_14_Solution.zip
oracle@pts:~/PTS_Labs/export$
```

- 7) Disconnect from current repository and log into work repository you created as a part of Lab 4. Go to solution section and select Import Solution. Make sure you have /home/oracle/PTS_Labs/export directory selected and then select the Lab 14 Solution, click OK.



Lab 15: Data Quality and Profiling

Introduction

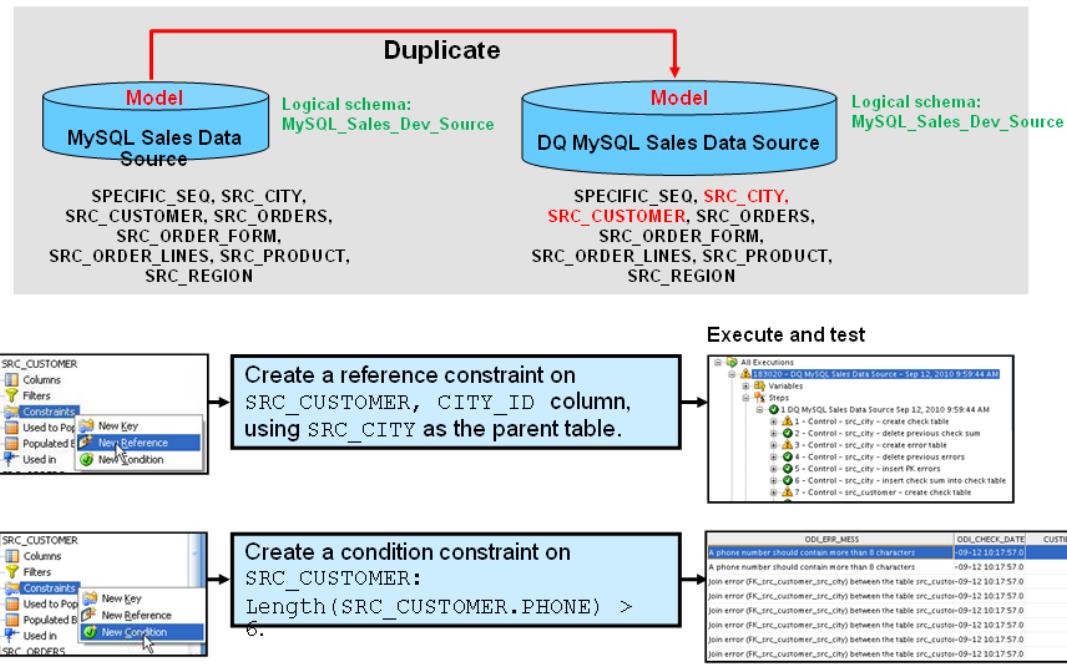
Data Quality control is essential in ensuring the overall consistency of the data in your information system's applications. Quality control can happen before data is integrated, when it is called **flow control**. It can also happen after the integration, on the datastores, when it is called **static control**.

In either case, the data is checked against a set of constraints you have defined. These are similar to database constraints: primary keys, foreign keys, and data conditions. If the data does not pass the quality control, it is moved to a separate table. You can then review and fix the problems in ODI, and recycle the data through the interface.

Process Overview

Once the models are defined, you need to check the quality of the data in these models. In this practice, you check the quality of data in the models and define constraints on models for the given sample application.

Note: Completing this practice is critical for all the following practice sessions.



Lab 15.1: Enforcing Data Quality in the Model (on the Source)

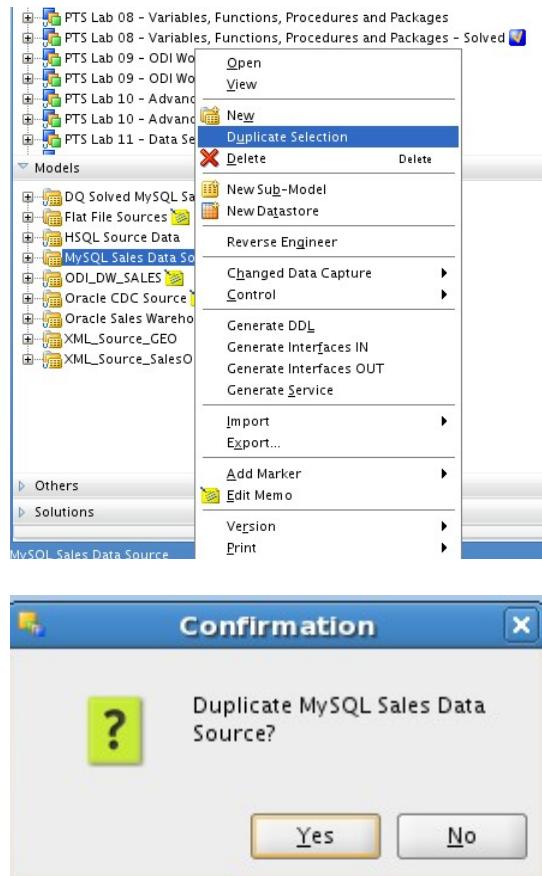
Scenario

You created new ODI Models and reverse engineered all tables and files in the models. Now, you need to check the quality of data in the models, define the constraints on the models, and detect possible errors in data.

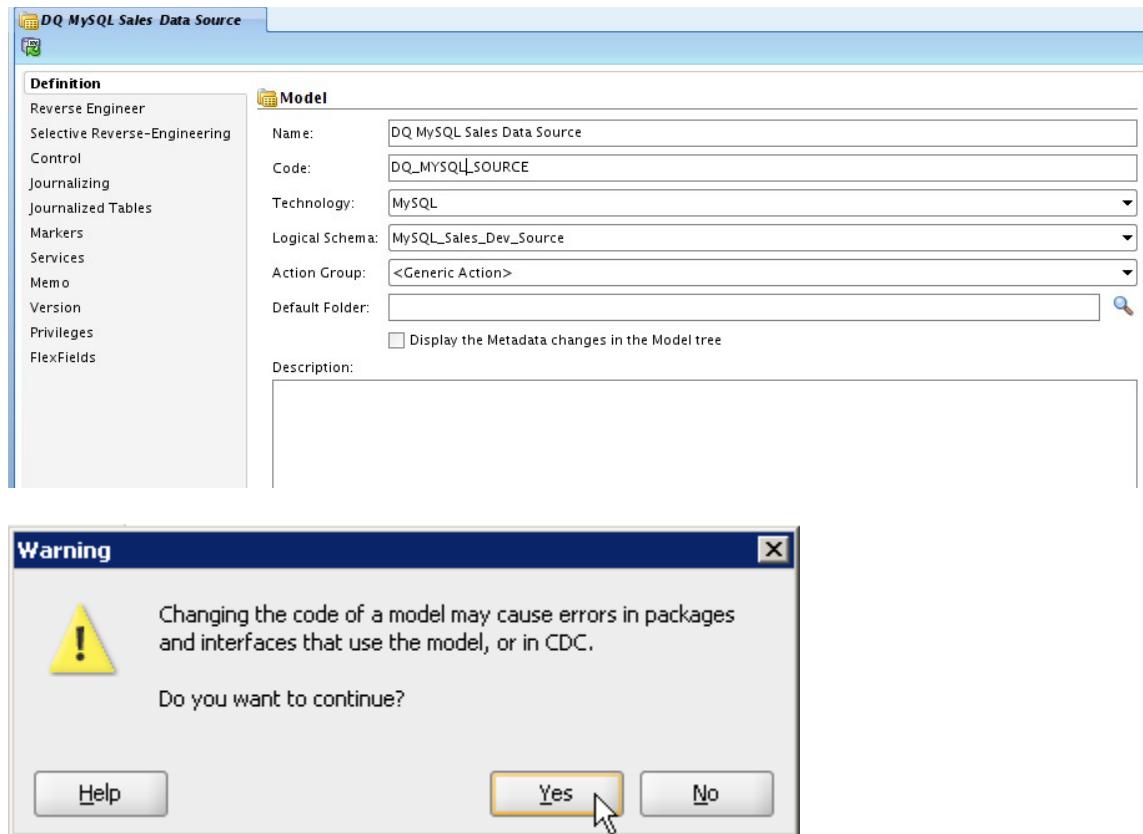
Instructions

- 1) Create a Reference constraint between the SRC_CUSTOMER and SRC_CITY datastores in a new Orders Application model. This reference is on the CITY_ID column.

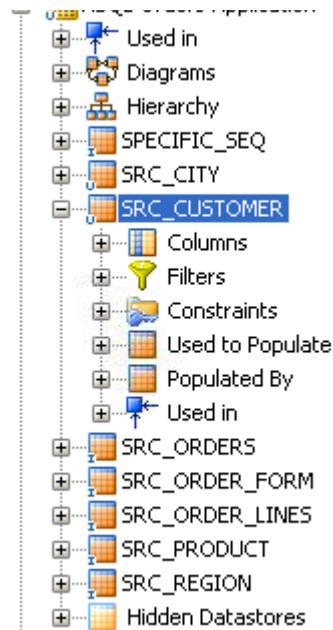
- a) In Designer, open the Models tab. Right-click MySQL Sales Data Source model and select Duplicate selection to duplicate the datastore. Click Yes.



- 2) Open the Copy of MySQL Sales Data Source model and rename it as DQ MySQL Sales Data Source. For Code, enter DQ_MYSQL_SOURCE. Click Save to save the model. If you get a warning message, click Yes to finish saving.



- a) Expand the DQ MySQL Sales Data Source model, and expand the SRC_CUSTOMER datastore.



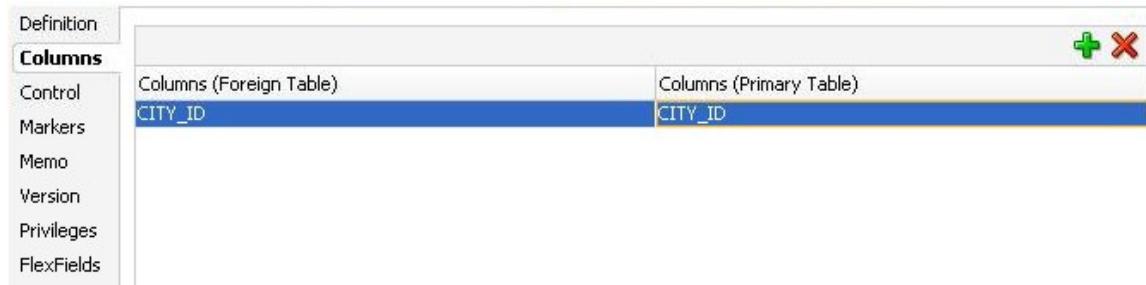
- b) Select the Constraints node, right-click and select the New Reference option.



- c) From the Table drop-down list, select the SRC_CITY table. The name of the constraint is automatically generated.

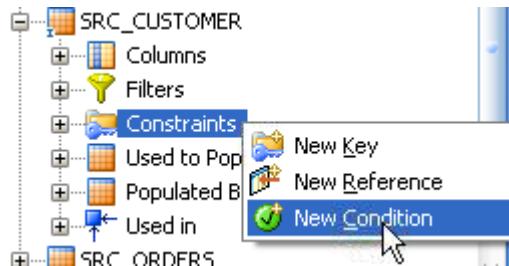
The screenshot shows the 'FK_src_customer_src_city' constraint definition dialog. The 'Definition' tab is selected. The 'Name' field contains 'FK_src_customer_src_city'. The 'Type' field is set to 'User Reference'. In the 'Parent Model/Table' section, the 'Model' is 'DQ MySQL Sales Data Source' and the 'Table' is 'src_city'. In the 'External Table' section, the 'Catalog' is 'sales_dev' and the 'Table' is 'src_city'. A checkbox at the bottom left is labeled 'Active on the Database'.

- d) Click the Columns tab. Click the  Add icon, and use drop-down lists to select the CITY_ID column for both tables of reference. (Note that the Primary Table column is City, which you want to also change to CITY_ID.) Click the Save button.



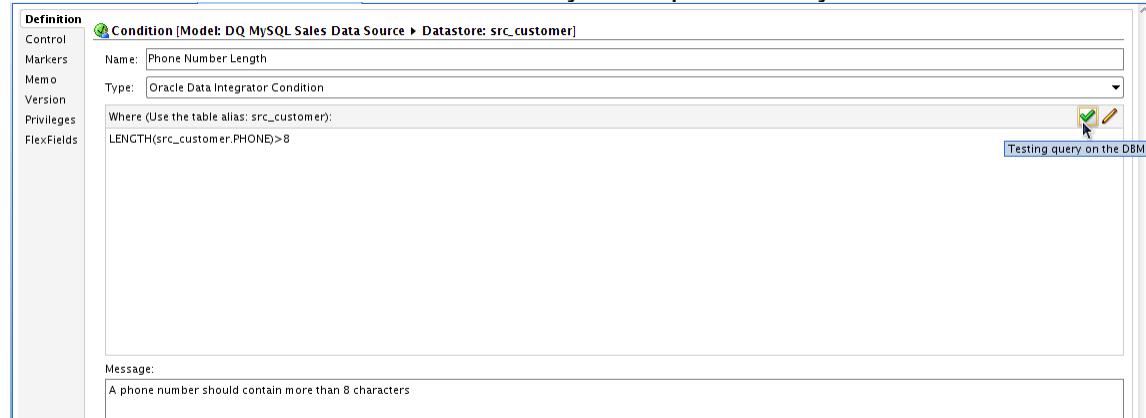
Columns (Foreign Table)	Columns (Primary Table)
CITY_ID	CITY_ID

- 3) Create a Condition constraint on SRC_CUSTOMER to check that the phone number contains nine or more characters.
- a) Select the Constraints node again for SRC_CUSTOMER, right-click and select the New Condition option.

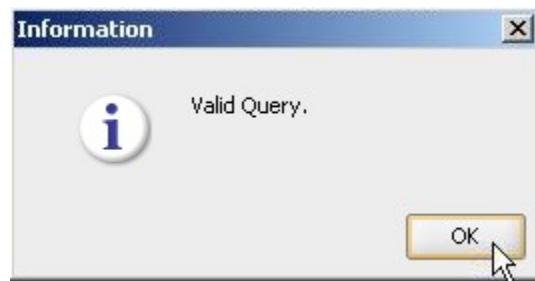


- b) Enter “Phone Number Length” in the Name field.
- c) Select Oracle Data Integrator Condition from the Type drop-down list.
- d) Enter the following expression in the Where field:
 $\text{LENGTH}(\text{SRC_CUSTOMER.PHONE}) > 8$. In the Message field, enter “A phone number should contain more than 8 characters”.
- Note:** You can also use the Expression Editor icon  to graphically edit this expression.

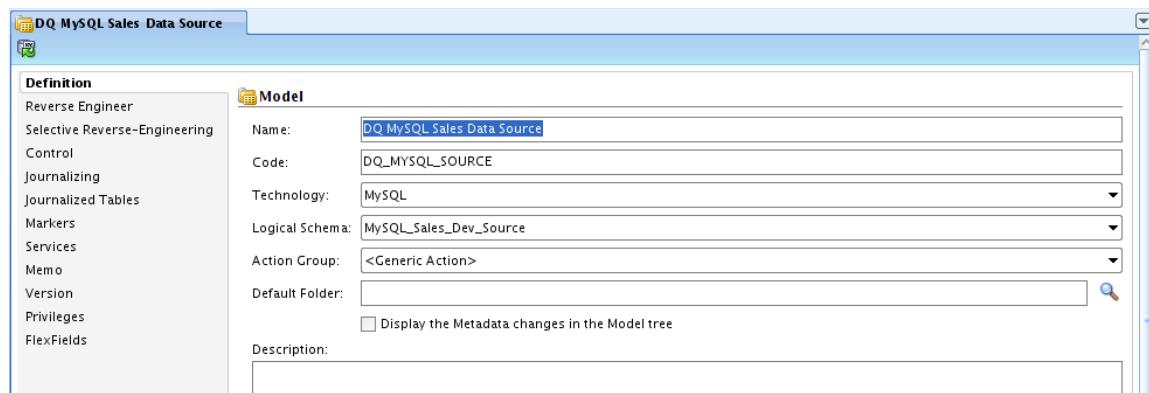
- e) Click the Validation icon  to validate your expression syntax.

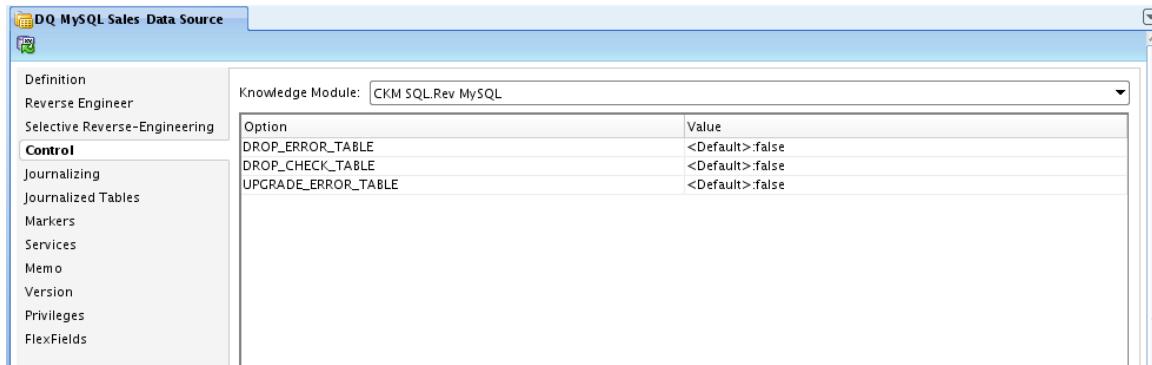


- f) On Oracle Data Integration Information screen, click OK.



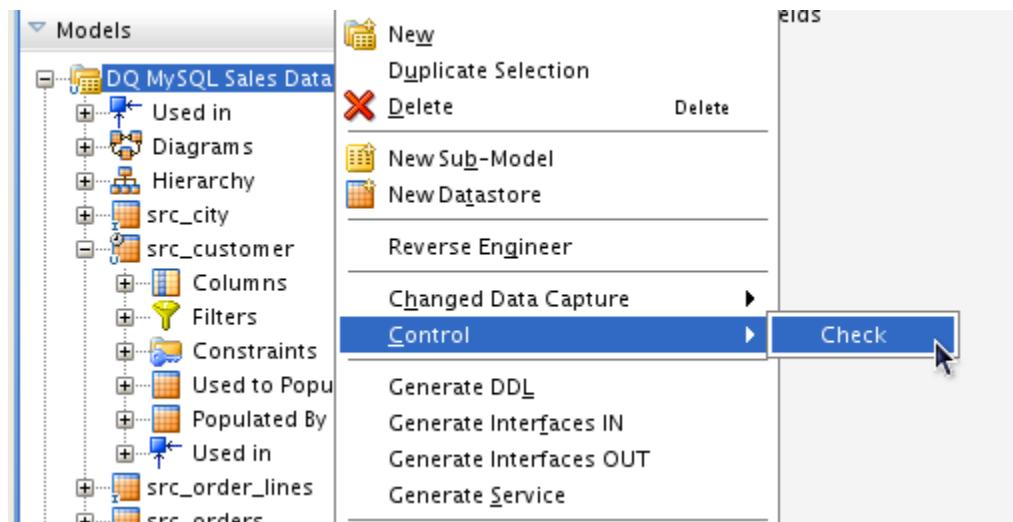
- g) Click Save button to add the condition, and then close the tab.
 4) Run a static check on the DQ MySQL Sales Data Source model.
 a) Open DQ MySQL model, click Control tab and select Knowledge Module CKM SQL Rev MySQL. Click Save button. Close DQ MySQL Sales Data Source model tab.



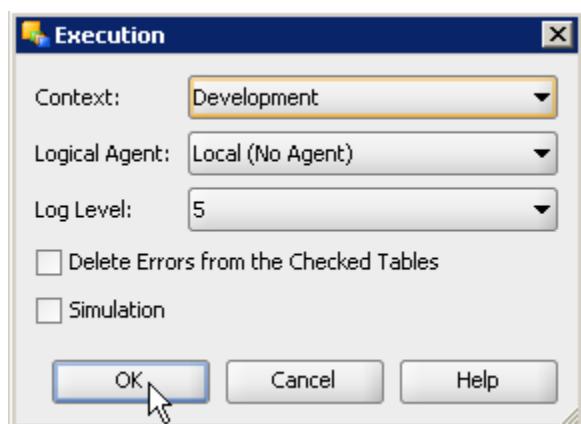


- b) Select the DQ MySQL Sales Data Source model in the tree view, right-click and select Control > Check.

c)



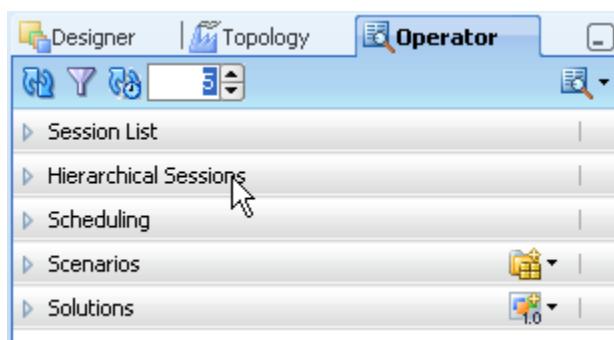
- d) In the Execution dialog box that appears, select Development context and then click OK.



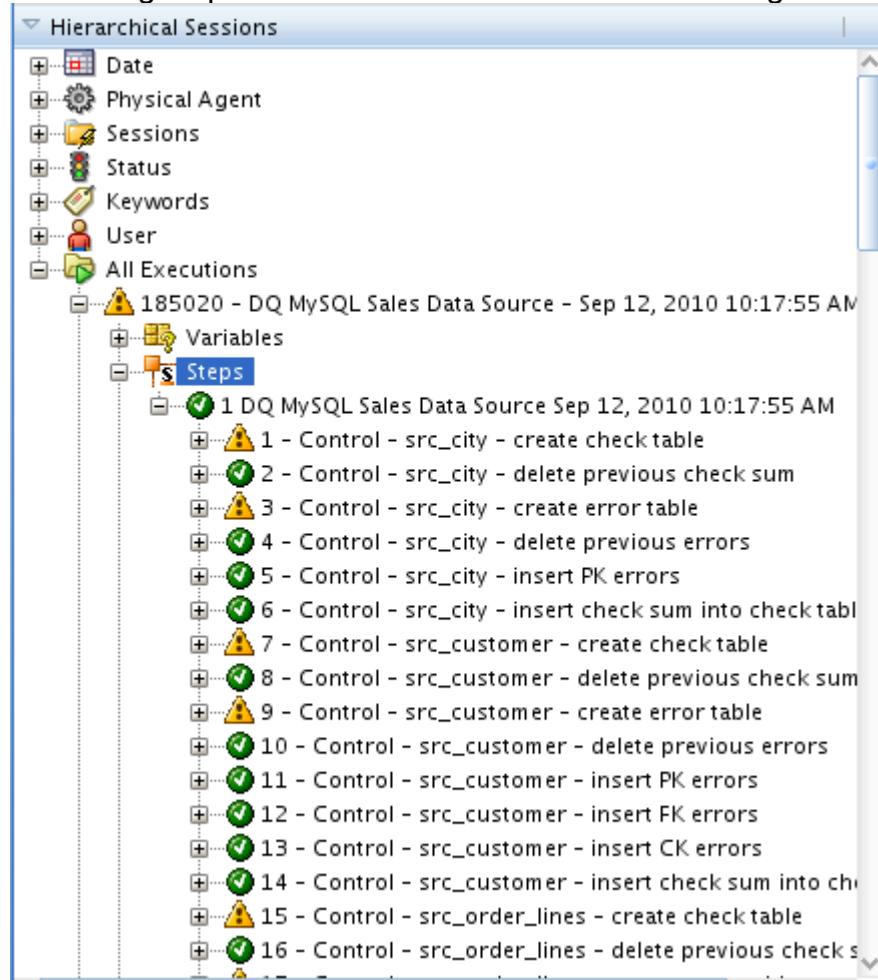
- e) Click OK when an Information dialog box appears saying that the session is started.



- f) Click the Operator Navigator icon tab . The Operator window appears. Click Hierarchical Sessions tab.



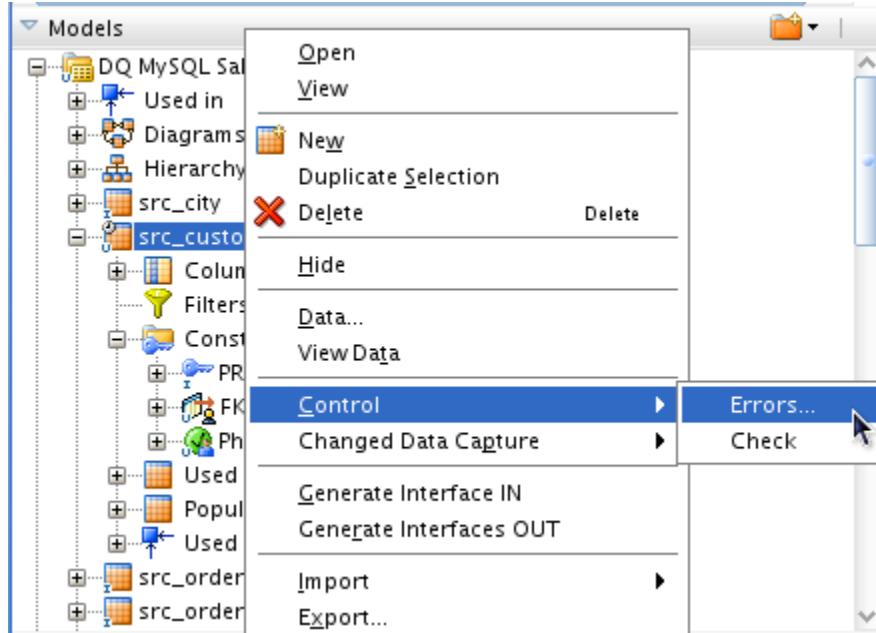
- g) Expand the All Executions node. Expand the Sales Data Source session. Expand its Steps node. The session should appear as complete, containing steps marked with checkmarks and warnings.



Note: You can optionally review the content of this session and see the different SQL commands issued to perform the check operations.

- 5) Review the errors detected in the SRC_CUSTOMER datastore.

- a) Click the Designer tab. In the Designer window, expand the DQ MySQL model, select the SRC_CUSTOMER datastore, right-click and select Control > Errors.



- b) A table appears listing the errors detected in your table. You have one join error and seven invalid phone numbers. Each line also contains the content of the invalid record.

Data					
	ODI_ERR_TYPE	ODI_ERR_MESS	ODI_CHECK_DATE	CUSTID	D
1	S	A phone number should contain more than 8 characters	-09-12 10:17:57.0	129	
2	S	A phone number should contain more than 8 characters	-09-12 10:17:57.0	124	
3	S	Join error (FK_src_customer_src_city) between the table src_customer and src_city	-09-12 10:17:57.0	141	
4	S	Join error (FK_src_customer_src_city) between the table src_customer and src_city	-09-12 10:17:57.0	128	
5	S	Join error (FK_src_customer_src_city) between the table src_customer and src_city	-09-12 10:17:57.0	125	
6	S	Join error (FK_src_customer_src_city) between the table src_customer and src_city	-09-12 10:17:57.0	118	
7	S	Join error (FK_src_customer_src_city) between the table src_customer and src_city	-09-12 10:17:57.0	115	
8	S	Join error (FK_src_customer_src_city) between the table src_customer and src_city	-09-12 10:17:57.0	112	
9	S	Join error (FK_src_customer_src_city) between the table src_customer and src_city	-09-12 10:17:57.0	111	
10	S	Join error (FK_src_customer_src_city) between the table src_customer and src_city	-09-12 10:17:57.0	106	

- How many customers are with an invalid CITY_ID? (8)
 - How many customers have a possibly invalid phone number? (2)
- c) Close this window.

Lab 15.2: Enforcing Data Quality in the Interface

Prerequisites

Replace SRC_SALES_PERSON.txt file with Lab15_SRC_SALES_PERSON.txt. to SRC_SALES_PERSON_copy.txt. Open terminal window and go to dir /app/Oracle/Middleware/Oracle_ODI/oracledi/demo/file

```
cp Lab15_SRC_SALES_PERSON.txt SRC_SALES_PERSON.txt
```

Note: You can copy Clean_SRC_SALES_PERSON.txt if need the original source file.

Process Overview

In the previous practices, you learned how to create ODI Interfaces, which transform data from one or more relational tables in the source to a relational table in the target and to export data from a flat file to a relational table. In this practice, you perform data quality control for the interface created for you for the DQ Lab.

First, for a certain datastore, you verify that the primary key constraint is marked static, and 2 columns are marked mandatory with static control.

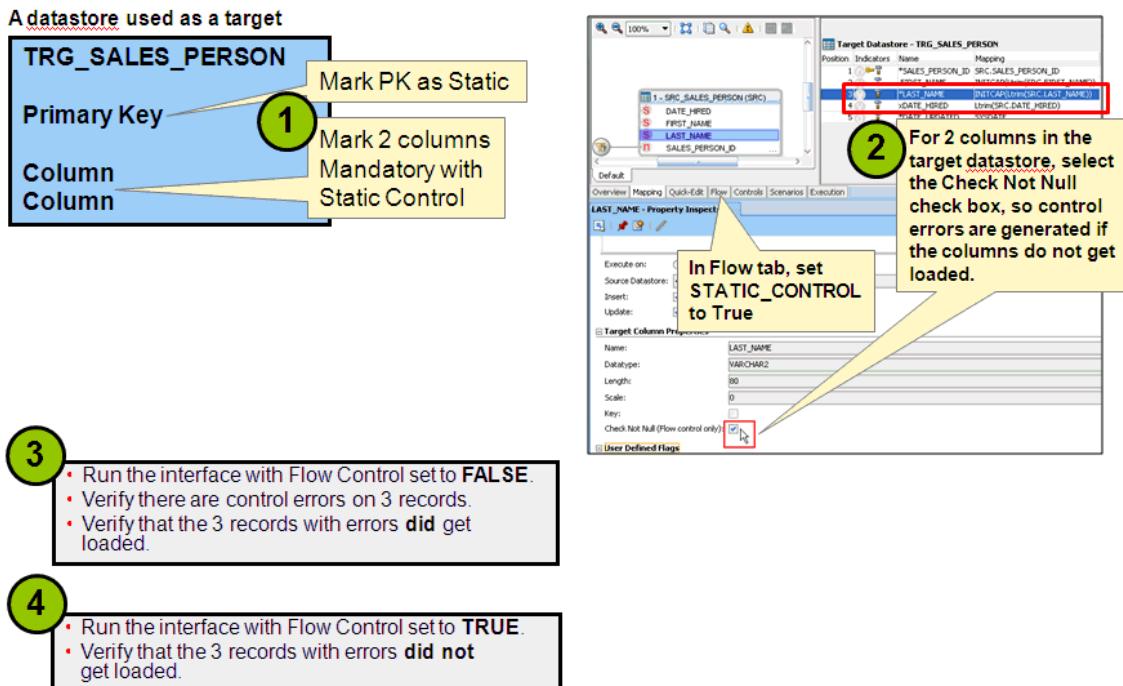
You then create an interface using that datastore as a target.

In the interface's Flow tab, you set STATIC_CONTROL to True. In the Controls tab, you ensure the knowledge module is set to CKM SQL (a knowledge module that you will import) and you set maximum allowed errors to different thresholds to see impact on results.

For 2 columns in the target datastore, you will select the Check Not Null check box, so that control errors will generate if these columns do not get loaded.

You will run the interface with Flow Control set to FALSE. You will verify that there are control errors on 7 records, and that the 7 records with errors **did** get loaded into the target datastore.

Finally, you will re-run the interface, with Flow Control set to TRUE, and verify that errors are **excluded** from the target datastore.



Scenario

You are responsible for data loading, transformation, and validation. You created a project and an interface to export data from a flat file and load this data into a relational table. Now you need to verify the quality of data loaded in the table.

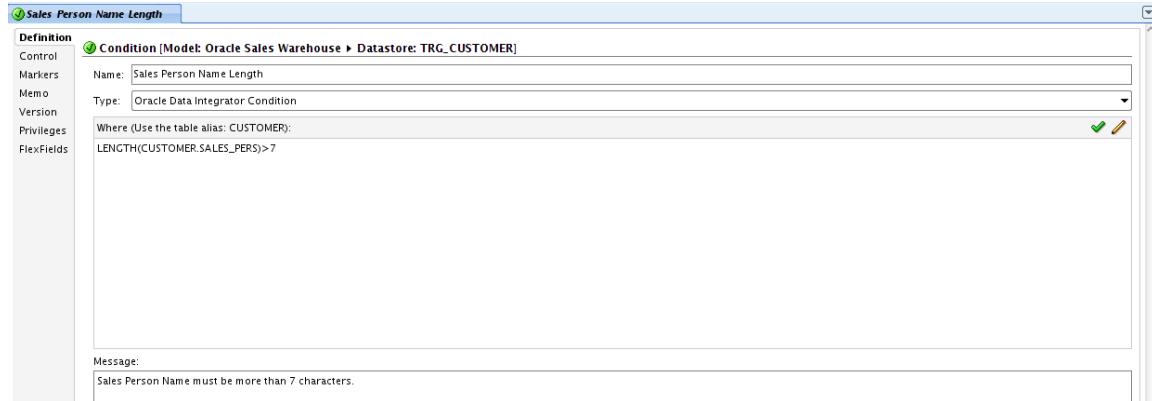
Instructions

If not connected, connect to work repository PTS Training Work Repository.

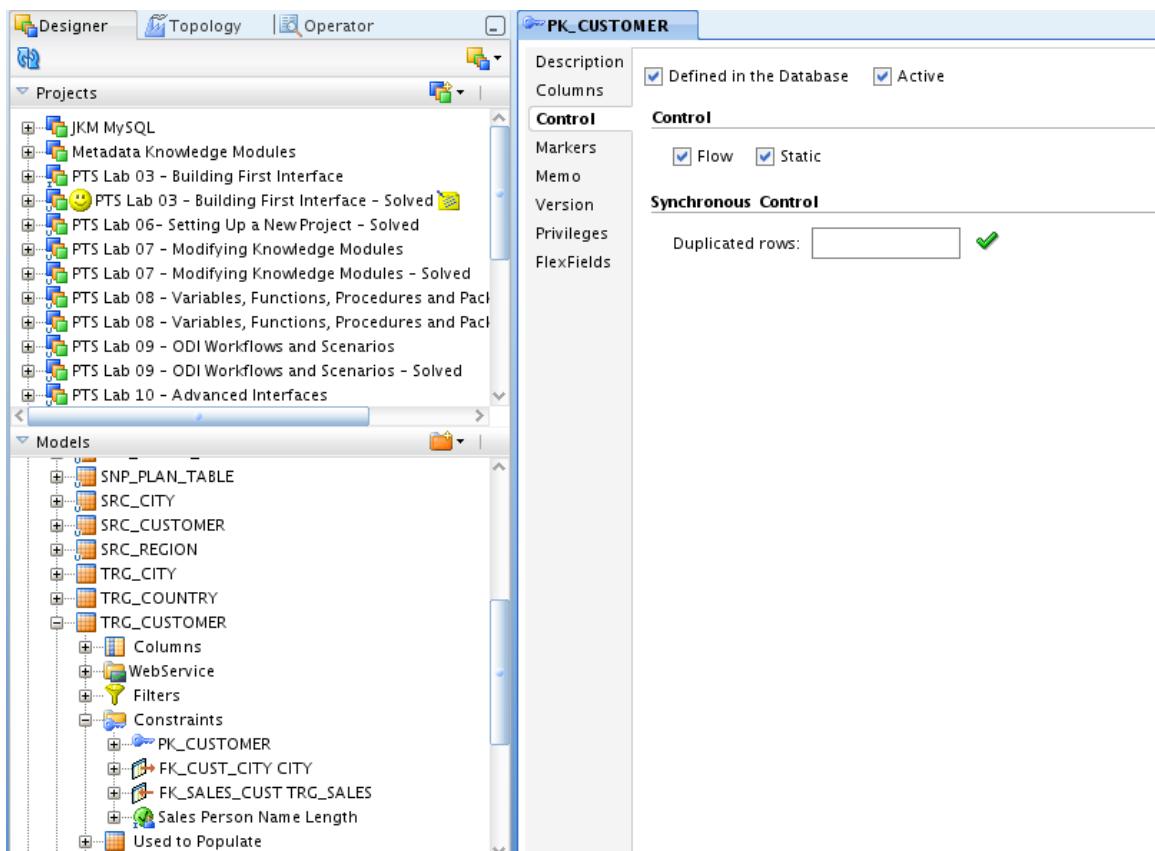
- 1) Create a Condition constraint on TRG_CUSTOMER target datastore to check that the sales person name contains eight or more characters.
 - a) Open the Oracle Sales Warehouse model and expand the TRG_CUSTOMER datastore. Select the Constraints node, right-click and select the New Condition option.
 - b) Enter “Sales Person Name Length” in the Name field.
 - c) Select Oracle Data Integrator Condition from the Type drop-down list.
 - d) Enter the following expression in the Where field:
LENGTH(CUSTOMER.SALES_PERS) > 7. In the Message field, enter “Sales Person Name must be more than 7 characters.”

Note: You can also use the Expression Editor icon to graphically edit this expression.

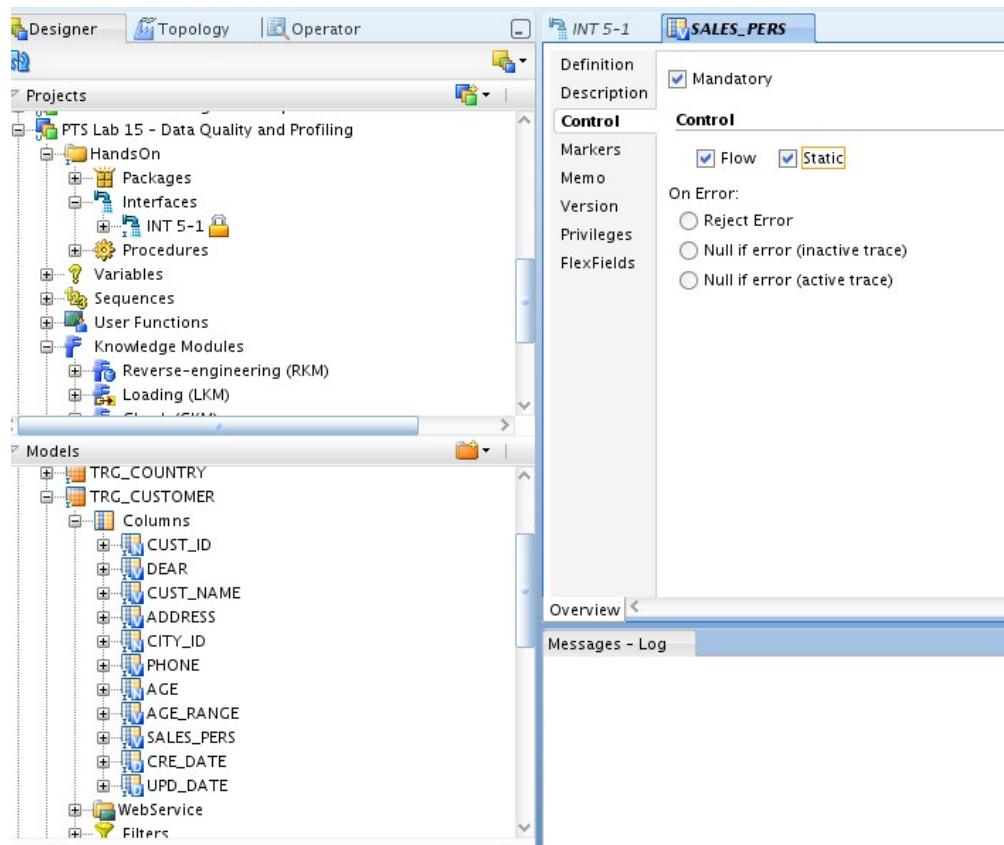
- e) Click the Validation icon  to validate your expression syntax.



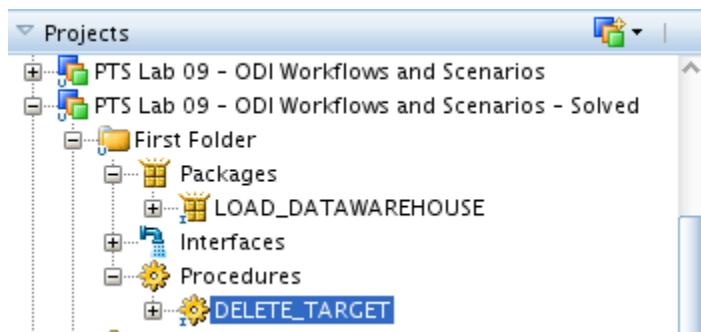
- f) Click Save button to add the condition, and then close the tab.
- 2) In the Models tab, expand Oracle Sales Warehouse > TRG_CUSTOMER> Constraints, and then double-click PK_CUSTOMER. Click the Control tab and verify that Static is selected.



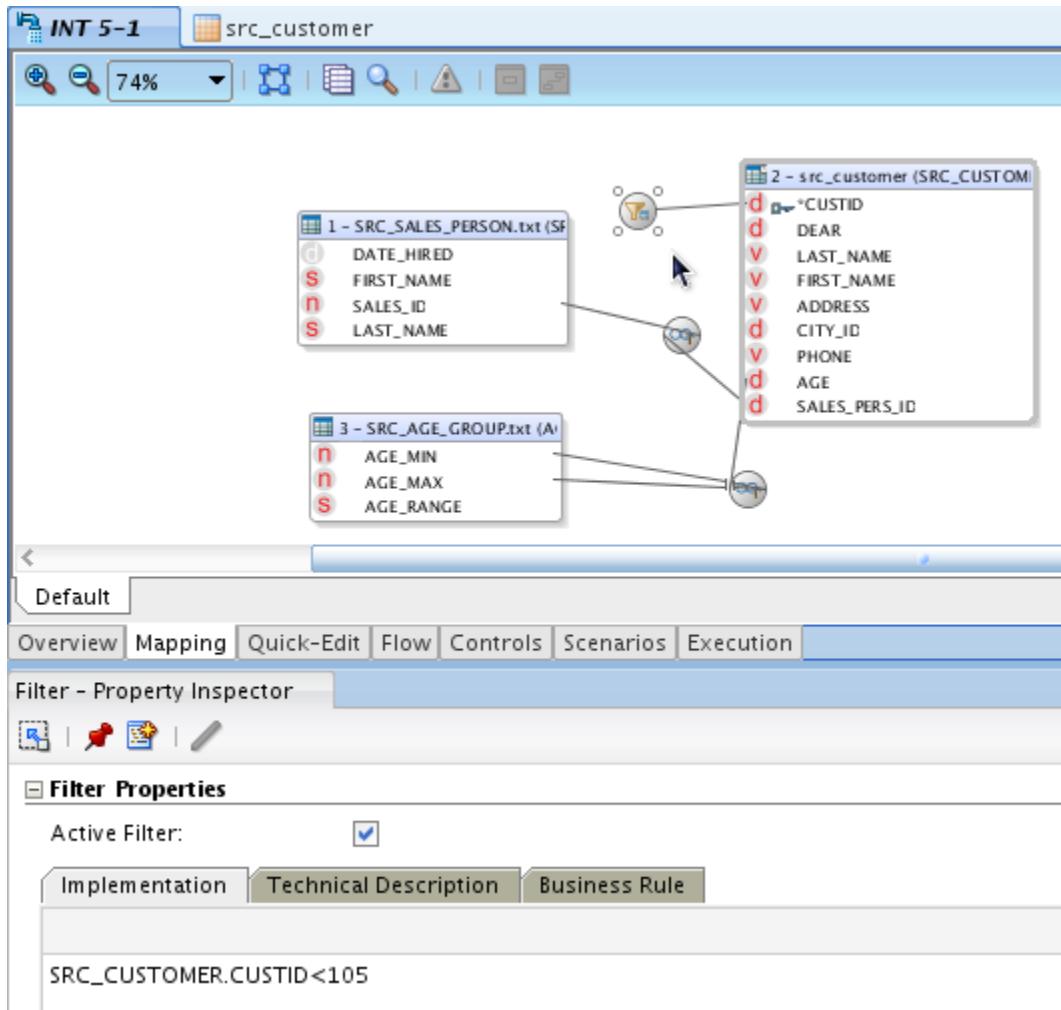
- 3) In the model Oracle Sales Warehouse, expand TRG_CUSTOMER > Columns, double-click column SALES_PERS, and then click the Control tab. Select Static to enable static control. Select the Mandatory check box. Click Save button.



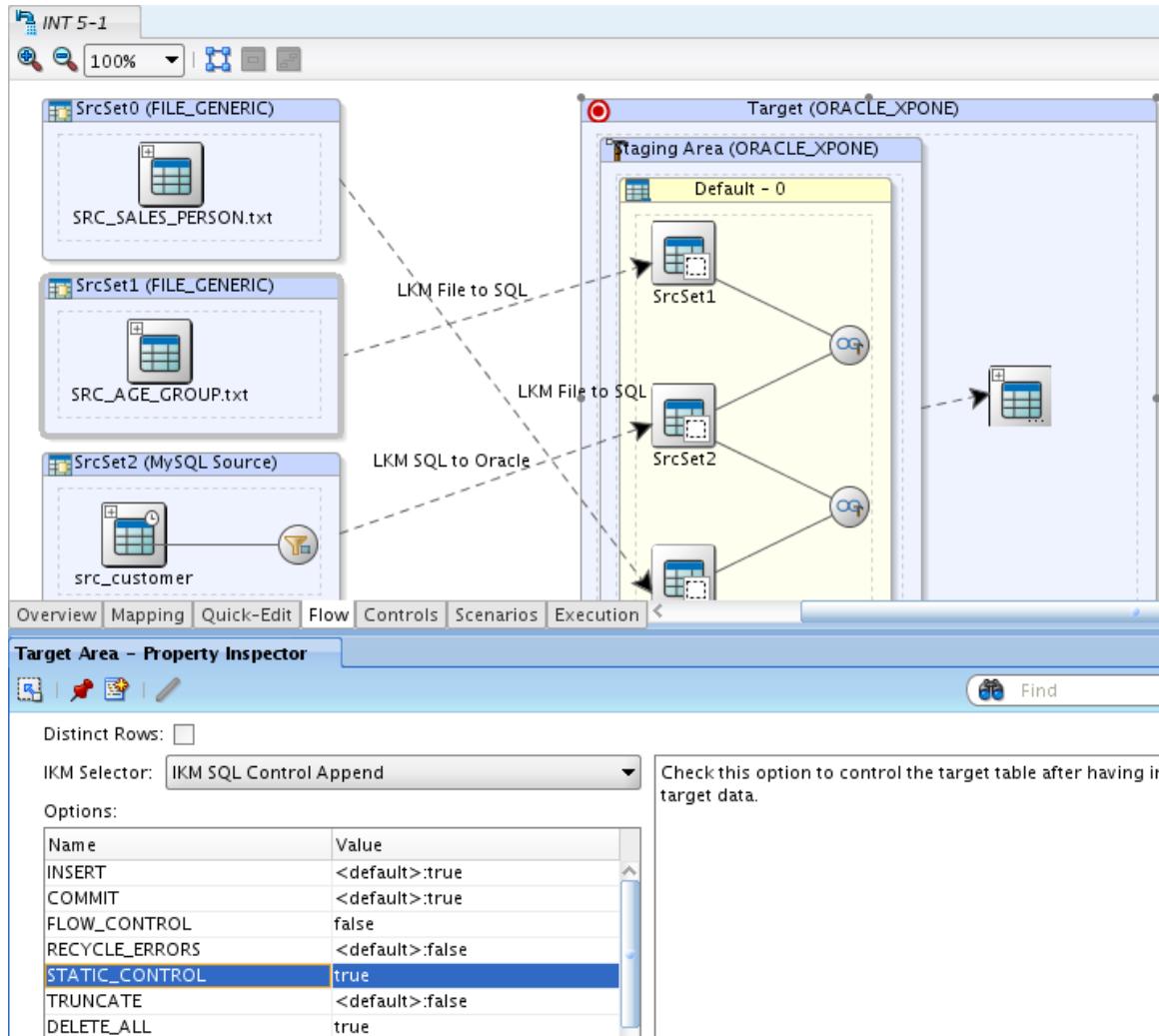
- 4) In the Projects tab, expand Procedures for PTS Lab 09 - ODI Workflows and Scenarios – Solved. Execute DELETE_TARGET to clear TRG_CUSTOMER table.



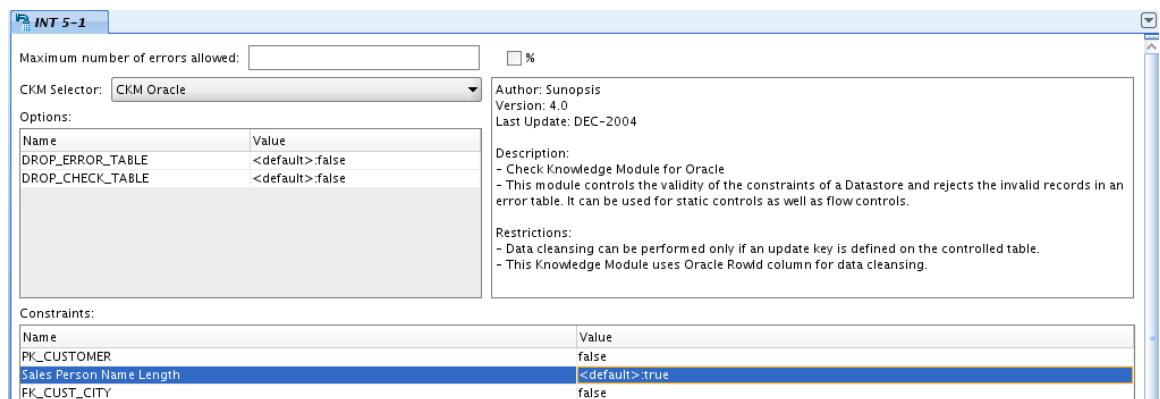
- 5) In the Projects tab, open interface for the PTS Lab 15.
a) In the Mapping tab, remove the CUST_ID filter.



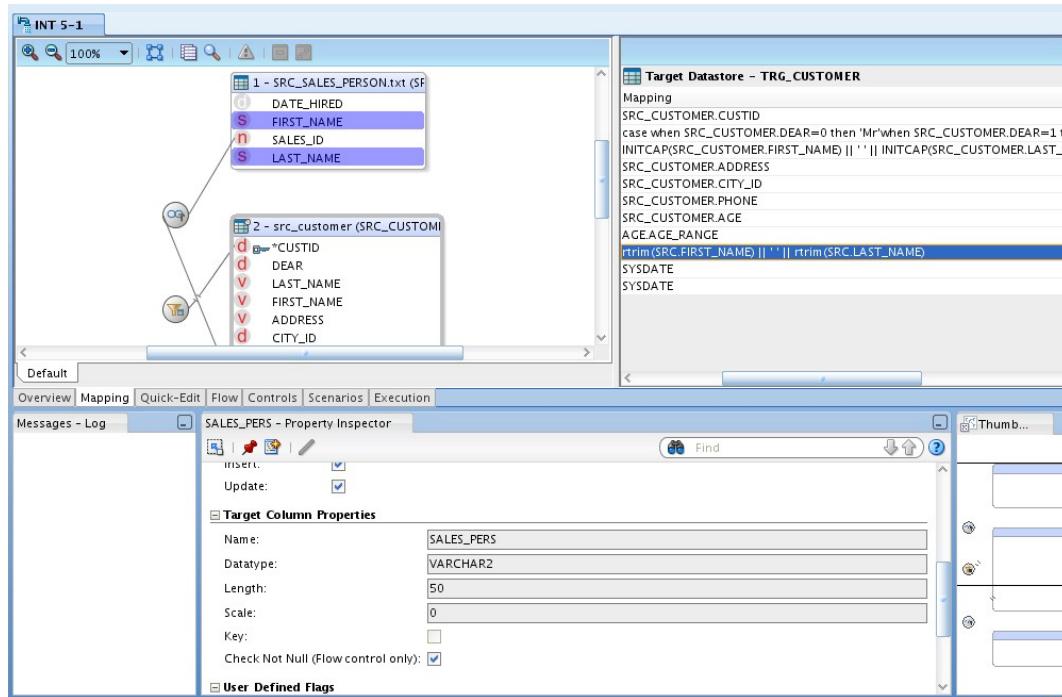
- b) Click the Flow tab. In the Flow tab, Click Target + Staging Area (ORACLE_XPONE) to open the options section. Set the static control option to true. Click the Controls tab.



- c) On the Controls tab, make sure your KM is set to CKM Oracle. Clear the maximum number of errors allowed. Set PK_CUSTOMER and FK_CUST_CITY to false. Click the Mapping tab.



- d) In Target Datastore, select the SALES_PERS column, and then select the Check Not Null check box. Click the Execute button to execute the Interface. Click Save button to save your changes.



- 6) Execute your Interface . Click the Execution tab to view execution results with the number of errors as shown in the screenshot.

Note: You may need to click the Refresh icon to view your last execution.

INT 5-1 - TRG_CUSTOMER												
Direct Executions			Scenario Execution									
Agent	Context	Status	Start	End	Duration	Return Code	Message	Rows	Inserts	Updates	Deletes	Errors
Internal	GLOBAL		Sep 27, 2010 ...	Sep 27, 2010 ...	2:0			134	51	0	0	7

- 7) On the Models tab, right-click the TRG_CUSTOMER target datastore and select Control > Errors. View the records with errors as shown below:

The screenshot shows two instances of the Oracle Data Integrator 11g Designer interface. Both windows have the title bar "[PTS Training Work Repository] Oracle Data Integrator 11g".

The left pane of both windows displays the Project tree, which includes various projects like "PTS Lab 13 - Change Data Capture - Solved", "PTS Lab 15 - Data Quality and Profiling", and "PTS Lab 15 - Data Quality and Profiling - Solved". The "Models" section is expanded, showing datastores such as "TRG_CUSTOMER", "TRG_COUNT", and "TRG_CITY".

The right pane is titled "Data" and contains a table of error records. The columns are "ROW_ID", "...", "ERR_MESS", "CHECK_DATE", "CUST_ID", and "DEAR". The table lists 7 rows of errors, all related to the "Sales Person Name must be more than 7 characters" constraint.

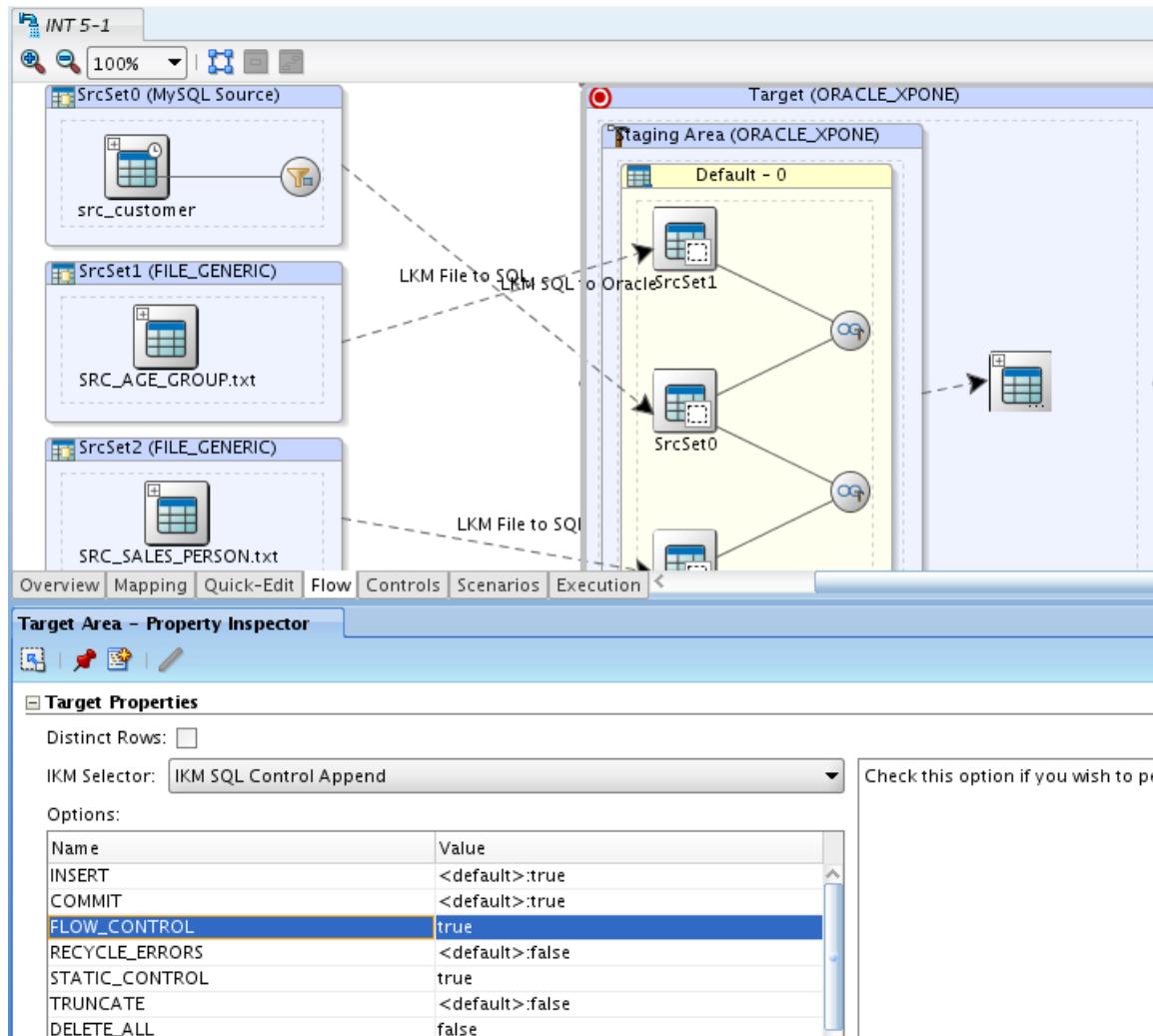
ROW_ID	ERR_MESS	CHECK_DATE	CUST_ID	DEAR
1	AAAAF+AAEAAAS Sales Person Name must be more than 7 characters.	9-24 19:35:48.0	112 Mr	Emilio Coward
2	AAAAF+AAEAAAS Sales Person Name must be more than 7 characters.	9-24 19:35:48.0	131 Mr	Theodore Washin
3	AAAAF+AAEAAAS Sales Person Name must be more than 7 characters.	9-24 19:35:48.0	103 Ms	Phyllis Reuschel
4	AAAAF+AAEAAAS Sales Person Name must be more than 7 characters.	9-24 19:35:48.0	122 Ms	Marybelle Revenc
5	AAAAF+AAEAAAS Sales Person Name must be more than 7 characters.	9-24 19:35:48.0	124 Mr	Thad Pinelli
6	AAAAF+AAEAAAS Sales Person Name must be more than 7 characters.	9-24 19:35:48.0	134 Mrs	Luz Woytek
7	AAAAF+AAEAAAS Sales Person Name must be more than 7 characters.	9-24 19:35:48.0	130 Ms	Camie Huenergar

In the second window, the "Models" section is expanded to show the "TRG_CUSTOMER" model in more detail, including its constraints and relationships.

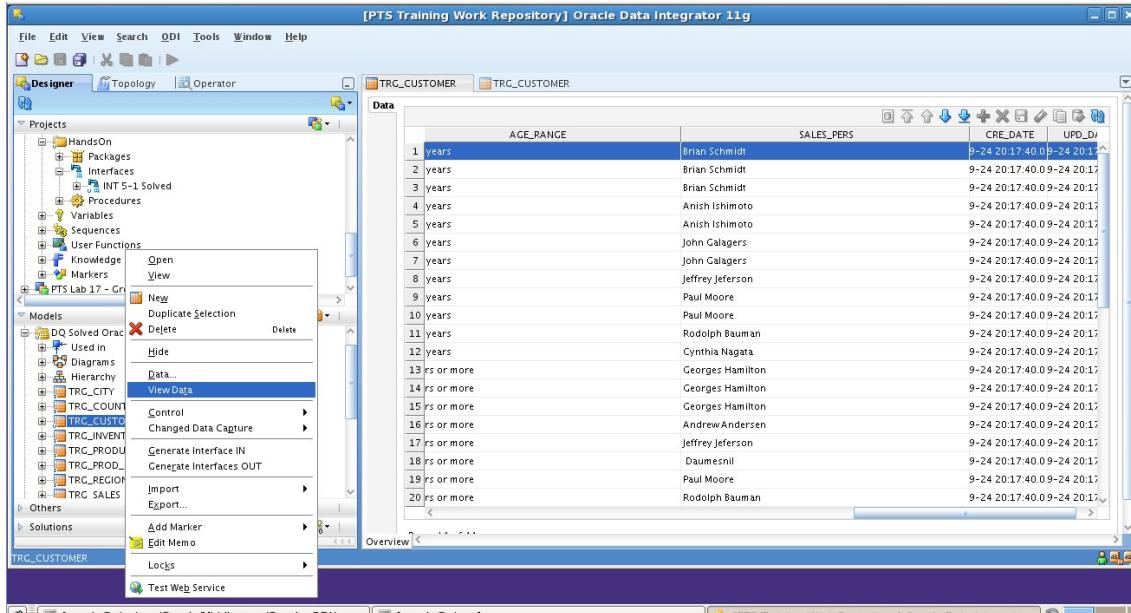
- 8) Right-click TRG_CUSTOMER datastore and select View Data. Verify that the rows with errors are still inserted in the target datastore. Close TRG_CUSTOMER data tabs.

IGE	AGE_RANGE	SALES_PERS	CRE_DATE
1	18 Less than 20 years	John Galagors	9-24 19:35:48.0
2	19 Less than 20 years	Paul Edwood	9-24 19:35:48.0
3	19 Less than 20 years	Anish Ishimoto	9-24 19:35:48.0
4	20 20-29 years	Georges Hamilton	9-24 19:35:48.0
5	20 20-29 years	Andrew Andersen	9-24 19:35:48.0
6	24 20-29 years	Jeffrey Jefferson	9-24 19:35:48.0
7	27 20-29 years	Barr	9-24 19:35:48.0
8	20 20-29 years	Paul Moore	9-24 19:35:48.0
9	20 20-29 years	Megan Keegan	9-24 19:35:48.0
10	27 20-29 years	Anish Ishimoto	9-24 19:35:48.0
11	21 20-29 years	Anish Ishimoto	9-24 19:35:48.0
12	36 30-39 years	John Galagors	9-24 19:35:48.0
13	31 30-39 years	Jeffrey Jefferson	9-24 19:35:48.0
14	32 30-39 years	Carl	9-24 19:35:48.0
15	38 30-39 years	Carl	9-24 19:35:48.0
16	30 30-39 years	Megan Keegan	9-24 19:35:48.0
17	35 30-39 years	Megan Keegan	9-24 19:35:48.0
18	36 30-39 years	Brian Schmidt	9-24 19:35:48.0
19	31 30-39 years	Cynthia Nagata	9-24 19:35:48.0
20	46 40-49 years	Jeffrey Jefferson	9-24 19:35:48.0

- 9) Click the INT 5-1 tab. On the Interface screen, click the Flow tab. Set Flow Control to true. Save changes, and then execute the interface .



- 10) On the Models tab, right-click TRG_CUSTOMER and select View Data. Verify that the rows with errors are excluded from data in the target datastore. Close the tabs.



- 11) See what happens when you change the threshold for errors allowed. Repeat Step 4 to Execute DELETE_TARGET to clear TRG_CUSTOMER table. Set the Maximum number of errors allowed to 5. Save and execute the interface. How many records are inserted?

INT 5-1

Maximum number of errors allowed: %

CKM Selector: CKM Oracle

Options:

Name	Value
DROP_ERROR_TABLE	<default>.false
DROP_CHECK_TABLE	<default>.false

Lab 16: ODI and GoldenGate

Introduction

Oracle Data Integrator and Oracle GoldenGate combined enable you to rapidly move transactional data between enterprise systems. By using a Knowledge Module approach, Oracle Data Integrator and Oracle GoldenGate can be connected to provide greater optimization and extensibility. Specifically the ODI Knowledge Module for Oracle GoldenGate leverages the power of Oracle GoldenGate for its real-time log-based CDC. This gives users the following technical benefits:

- Reduces the invasiveness of extracts on the source systems via log-based change data capture
- Minimizes the overhead on source systems
- Automates Oracle GoldenGate deployment directly from using a single UI, eliminates manual entry of config files
- Provides a common best practice pattern loading operational stores or data warehouses in ‘mini-batch’ to remove batch window dependency
- Enables transformations at the row-level during data capture or delivery, as well as high performance set-based transformations within the target database
- Improves recoverability of data by persisting changed data

Prerequisites

Basic knowledge of Oracle Data Integrator and GoldenGate, Installed Oracle Data Integrator, Oracle GoldenGate and Oracle Database instance.

Process Overview

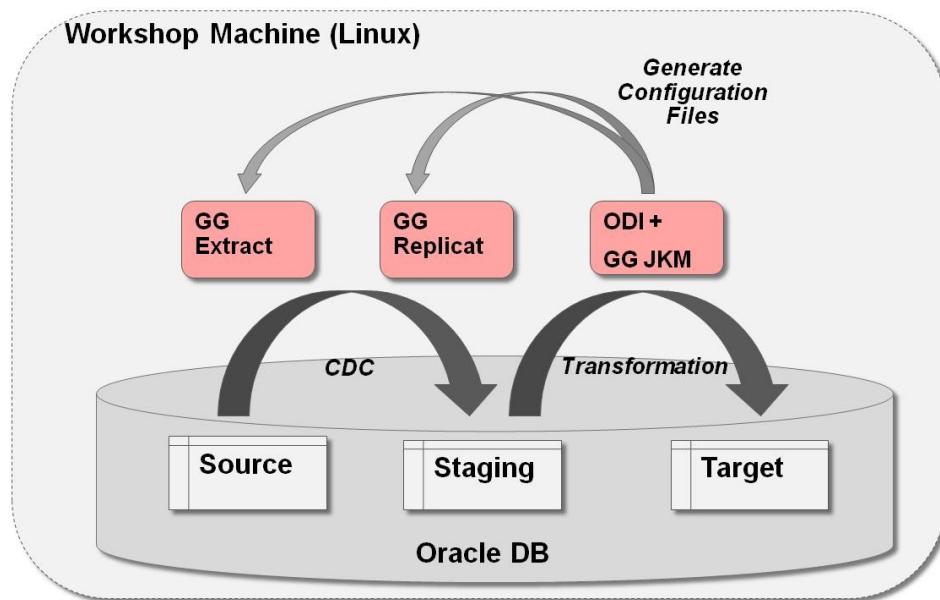
Data integration is more than simply about ETL (extract, transform, load); it's also about unifying information for improved business agility. This hands-on lab—for developers, DBAs, and architects—highlights best practices using Oracle Data Integrator (ODI) and Oracle GoldenGate (OGG) together to perform real-time ETL. In the lab, participants will use ODI's Journalizing Knowledge Module for OGG to generate OGG scripts based on ODI models and interfaces.

Scenario

In this lab we are going to add OGG CDC to an existing ODI project. The project is mostly identical to the ODI Hands-on Lab that covers ODI CDC mechanisms.

For simplicity the entire setup is on one machine using a single database. Real-life use cases would most likely use source and target on different servers, but the fundamental setup of OGG and ODI are very similar.

The demo machine has two separate installations of OGG, one in /app/GoldenGate as a source set up and one in /apps/GoldenGate_stg as a staging area set up. These two locations simulate OGG installations on two separate machines.



Part 0: Preparation Steps

In these steps you will clean and setup the environment for this exercise

- 1) Open terminal and go to directory /home/oracle/Desktop/Reset_Workshop
Run following two sh file from command line:

```
> ./reset_for_goldengate.sh
```

```
> ./reset_for_gg2.sh
```

- 2) Delete dir ODIS_to_ODIT1 from /tmp folder, if exists:

```
/tmp> rm -r ODIS_to_ODIT1
```

- 3) Delete any contents in **dirdat** and **dirdef** folders from source and staging on GG installation, /app/GoldenGate and /app/GoldenGate_stg directories

Part 1: Enable Staging Model for CDC

In this exercise you will review the existing models and projects in ODI

- 1) Start GoldenGate manager for source as well as for staging area

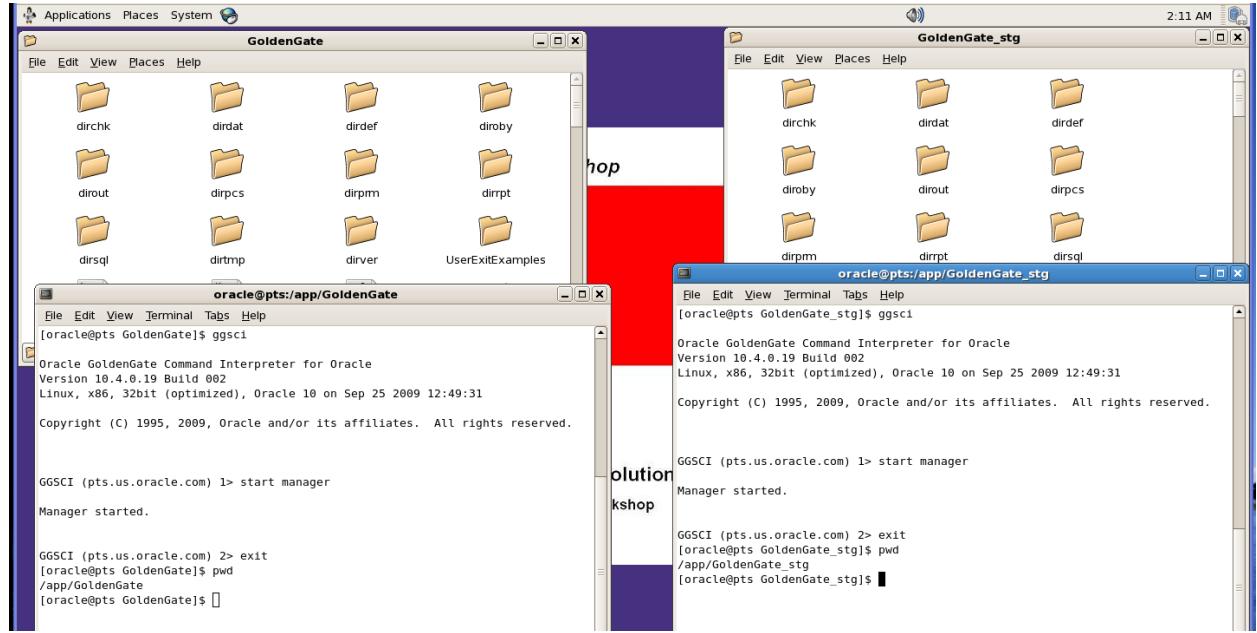
Open terminal window and go to directory /app/GoldenGate and run command as:

```
> ggsci
```

```
GGSCI (pts.us.oracle.com) 1> start manager
```

```
GGSCI (pts.us.oracle.com) 2> exit
```

Repeat the same for /app/GoldenGate_stg directory, which is our staging area. Running manager program starts GoldenGate manager process which is parent process to start/manage other processes like extract, replication, etc. Following image shows manager process started for source and stage servers.

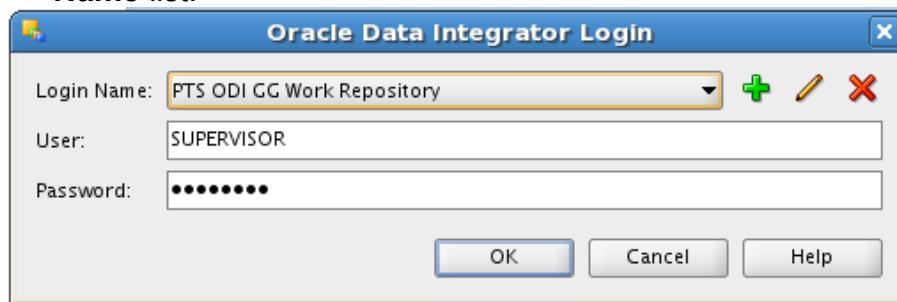


- 2) Open the Oracle Data Integrator Studio using the command **odistudio** within any terminal window.

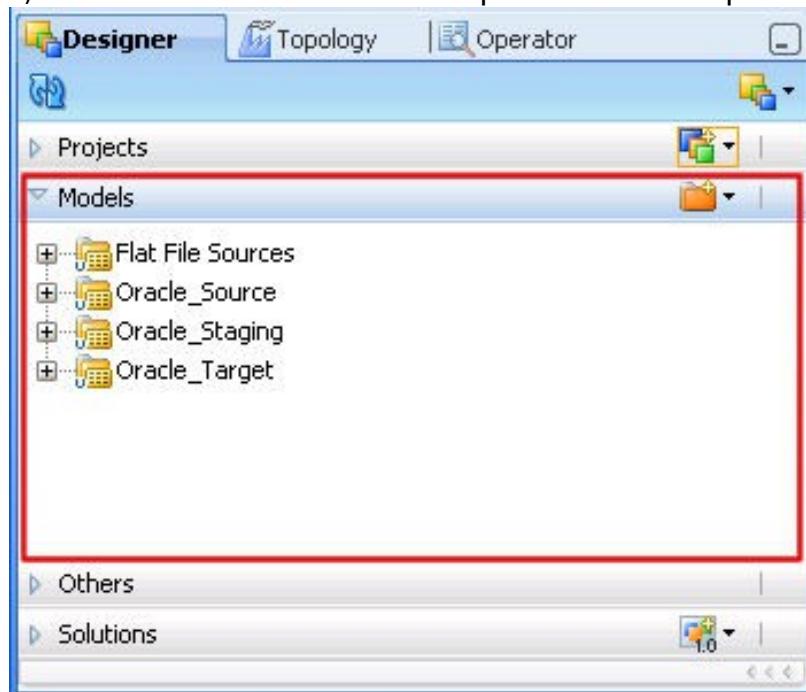
Choose the **Designer** tab and click on **Connect To Repository...**



In the Login dialog, choose **PTS ODI GG Work Repository** from the **Login Name** list.

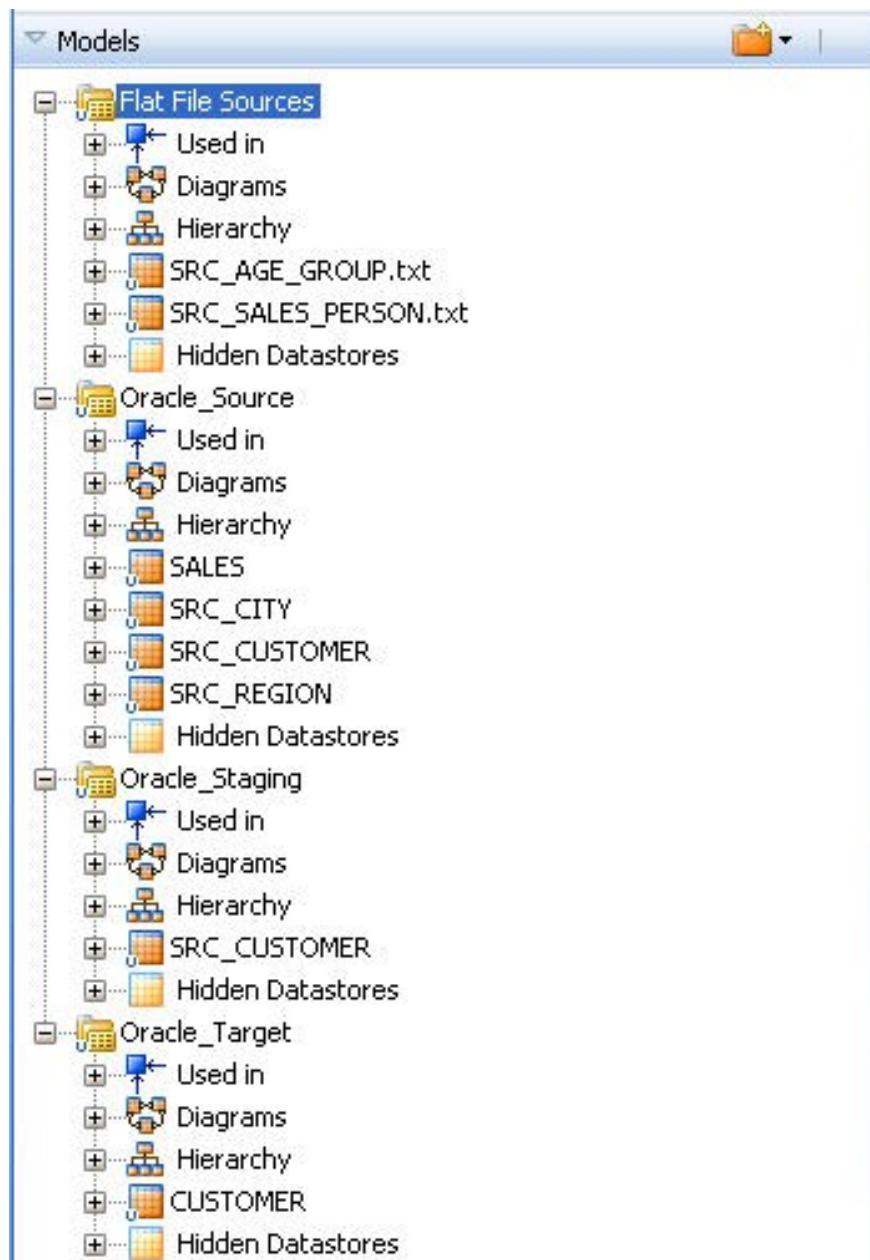


3) Click on the **Models** title to expand the models pane.



4) Review the models present in the repository:

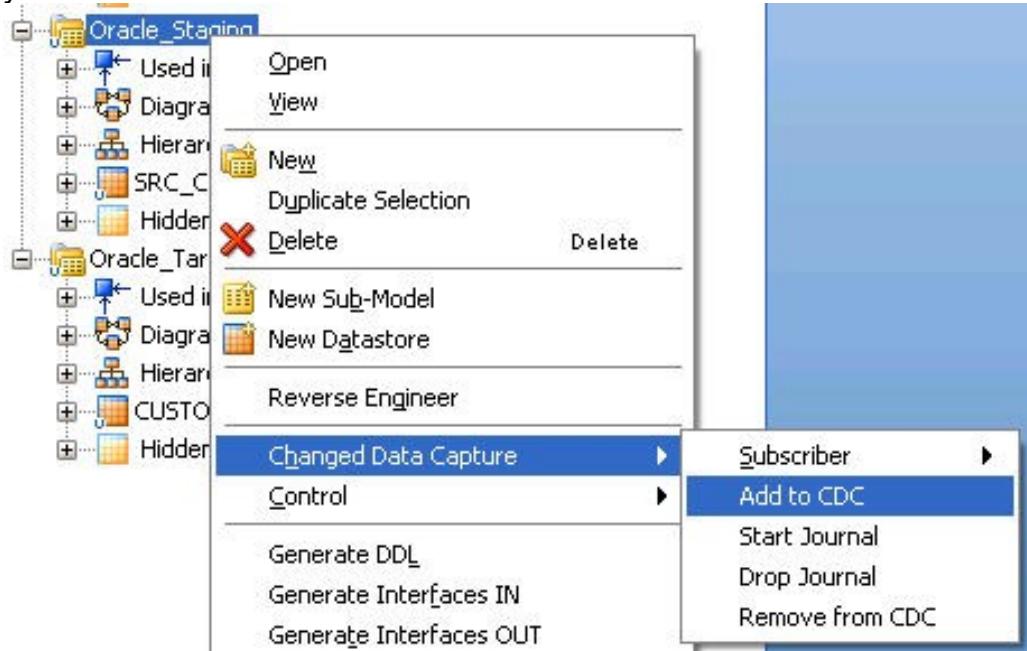
Expand each one of the 4 models Oracle_Source, Oracle_Staging, Oracle_Target, and Flat File Sources. The **Oracle_Source** model contains the tables in the original source location. **Oracle_Staging** contains copies of tables from the source that need to be replicated by Oracle GoldenGate in real-time. The **Oracle_Target** model contains transformed tables in the final target location. The **Flat File Sources** model contains flat file datastores that will participate in the transformation to the target table.



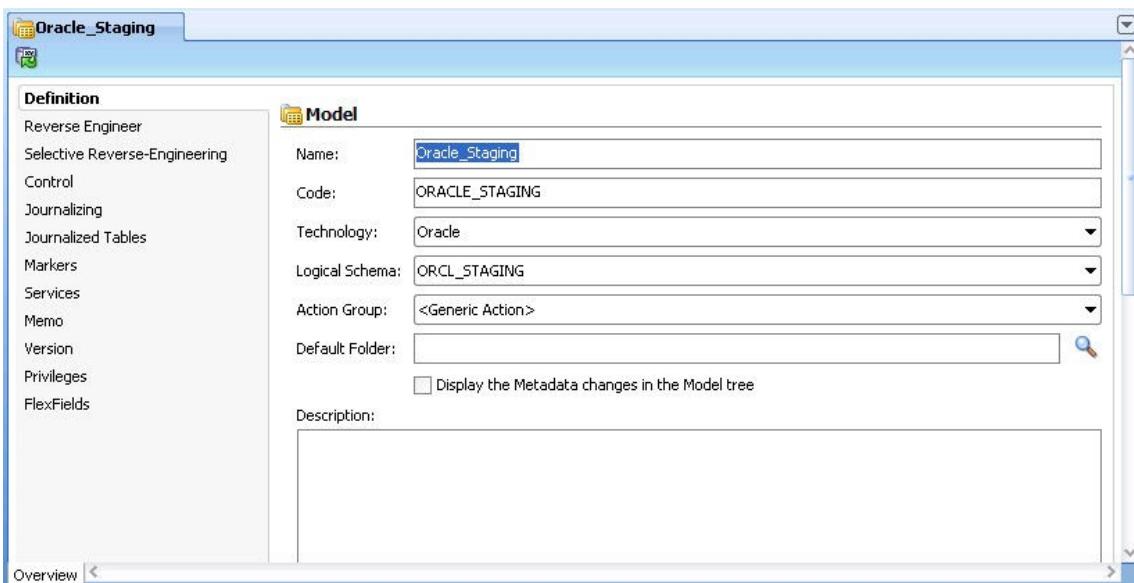
- 5) **Enable the staging model for ODI CDC:** The JKM for Oracle GoldenGate performs a 2-stage CDC; OGG detects changes in the source and replicates

them to the staging tables; ODI is informed of changes arriving in the staging table through its standard CDC framework; therefore the staging tables are enabled for CDC and generate OGG scripts to perform replication from source to staging.

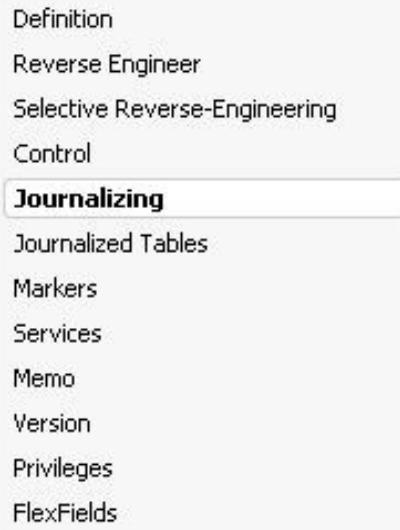
Right-click on the model **Oracle_Staging**, go into the submenu **Changed Data Capture**, and select **Add to CDC**. Press **Yes** when asked “Are you sure you want to add all the model’s tables to CDC?”



- 6) Double-click on the model **Oracle_Staging** to open the model editor



- 7) Go to the **Journalizing** tab.



- 8) In Journalizing Mode pick **Consistent Set**, press Ok in the Alert dialog, and in Journalizing KM pick “**JKM Oracle to Oracle Consistent (OGG)**”.



- 9) Enter the following options (highlighted in red) to the JKM. Leave options starting with <Default> unchanged.

NOTE PASSWORD for SRC_DB_PASSWORD is oracle

Knowledge Module: JKM Oracle to Oracle Consistent (OGG).GG_HOL

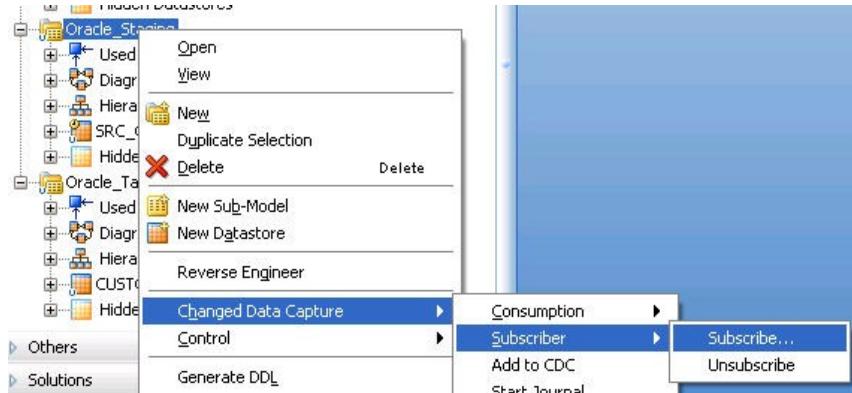
Option	Value
VALIDATE	<Default>:false
LOCAL_TEMP_DIR	/tmp
NB_APPLY_PROCESS	<Default>:1
TRAIL_FILE_SIZE	<Default>:100
SRC_OGG_OBJECT_GROUP	<Default>:ODIS
SRC_LSCHEMA	ORCL_SOURCE
SRC_DB_USER	system
SRC_DB_PASSWORD	oracle
SRC_OGG_PATH	/app/GoldenGate
SRC_SETUP_OGG_PROCESSES	<Default>:true
STG_OGG_OBJECT_GROUP	<Default>:ODIT1
STG_HOSTNAME	pts.us.oracle.com
STG_MANAGER_PORT	7909
STG_OGG_PATH	/app/GoldenGate_stg
STG_SETUP_OGG_PROCESSES	<Default>:true
COMPATIBLE	10
CHECKPOINT_TABLE_NAME	<Default>:ODIOGGCKPT
ENABLE_ODI_CDC	<Default>:true
USE_OGG_FOR_INIT	<Default>:false
CLUSTER_DATABASE_INSTANCES	<Default>:
CHARSET_ENCODING	<Default>:ISO8859-1

Values:

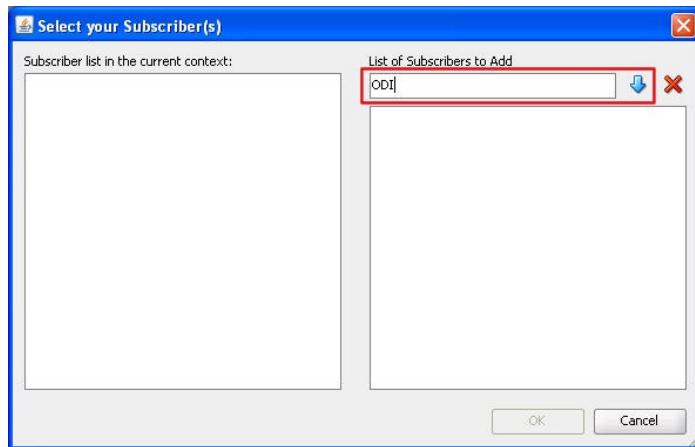
LOCAL_TEMP_DIR	/tmp
SRC_LSCHEMA	ORCL_SOURCE
SRC_DB_USER	system
SRC_DB_PASSWORD	oracle
SRC_OGG_PATH	/app/GoldenGate
STG_HOSTNAME	pts.us.oracle.com
STG_MANAGER_PORT	7909
STG_OGG_PATH	/app/GoldenGate_stg
COMPATIBLE	10

Press Save() to save the model; close the model editor.

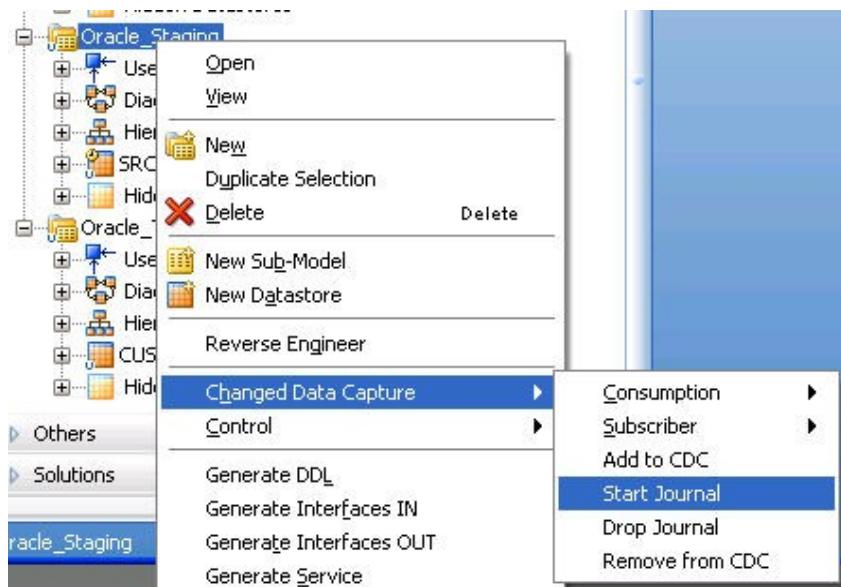
- 10) Right-click on model **Oracle_Staging**, go into the submenu **Changed Data Capture > Subscriber**, and select **Subscribe....**



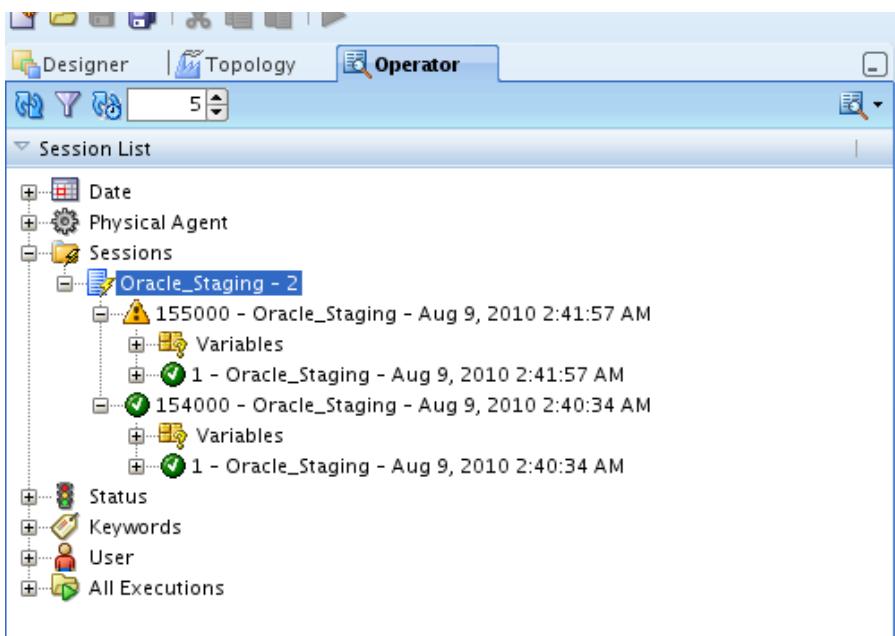
Enter **ODI** as the name of a new subscriber and press the red Arrow Down button to add to the list.



- 11) Press OK to close the dialog. Press **Ok** on the **Execution** and then **Information** dialogs that appear.
- 12) Right-click on **Oracle_Staging**, go into the submenu **Changed Data Capture**, and select **Start Journal**. Press **Ok** on the **Execution** and then **Information** dialogs that appear.

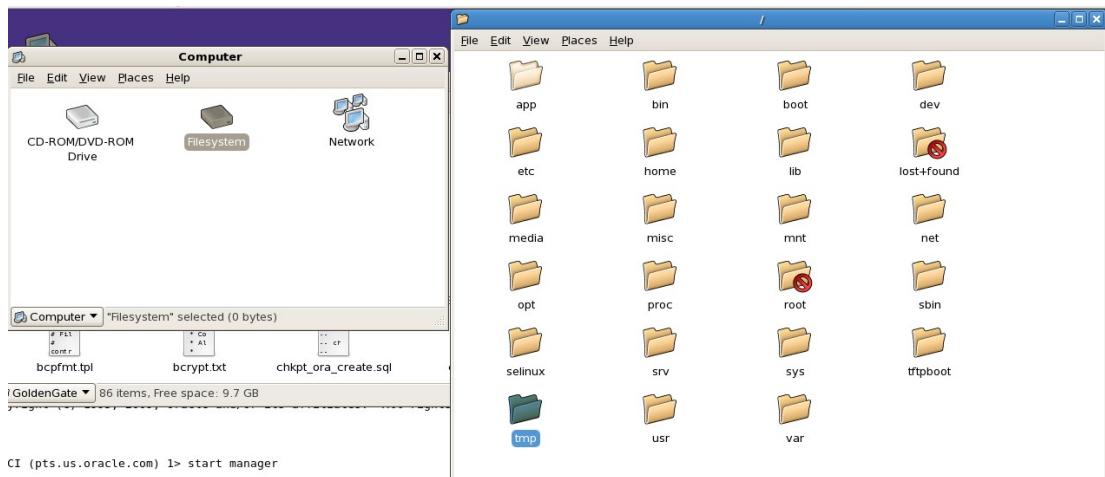


Verify on the Operator tab that both the session have successfully completed.

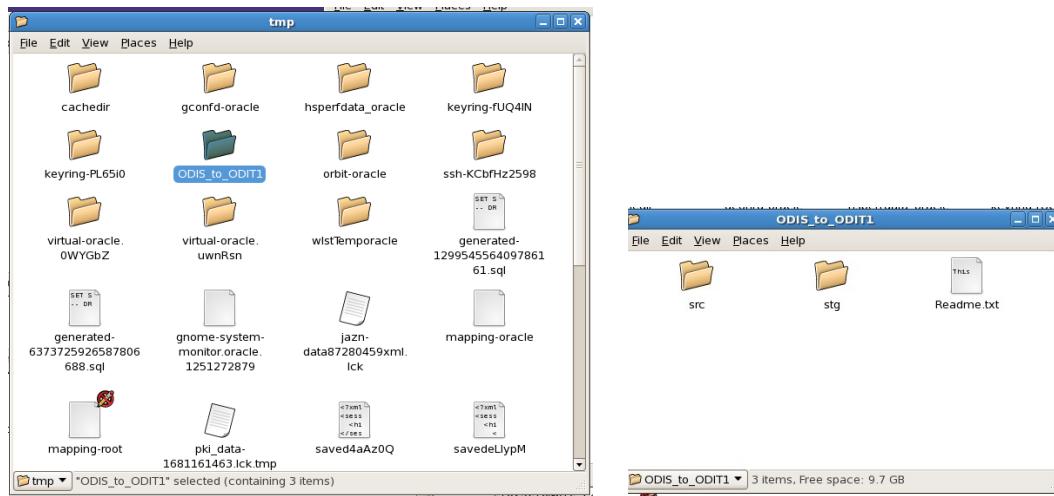


Part 2: Configure Oracle GoldenGate using generated files

- 1) Open a File Explorer window by opening **Computer** icon on the desktop and change directories to **/tmp**. A new directory called **ODIS_to_ODIT1** should be visible. This directory contains the OGG configuration files that were generated during the **Start Journal** step earlier.

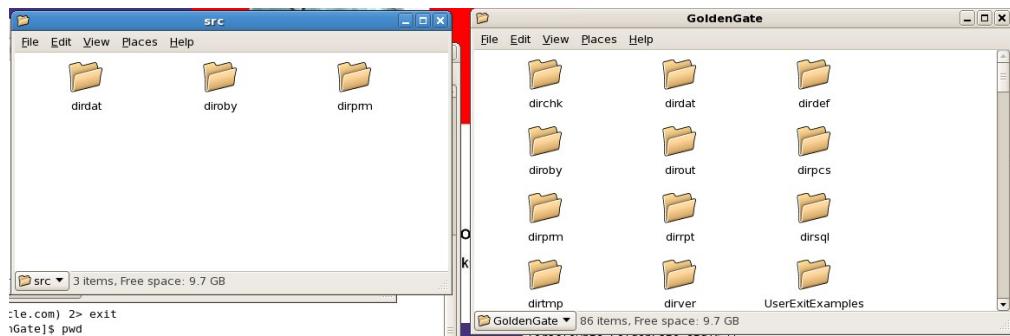


Double-click on **ODIS_to_ODIT1** to see src and stg directories

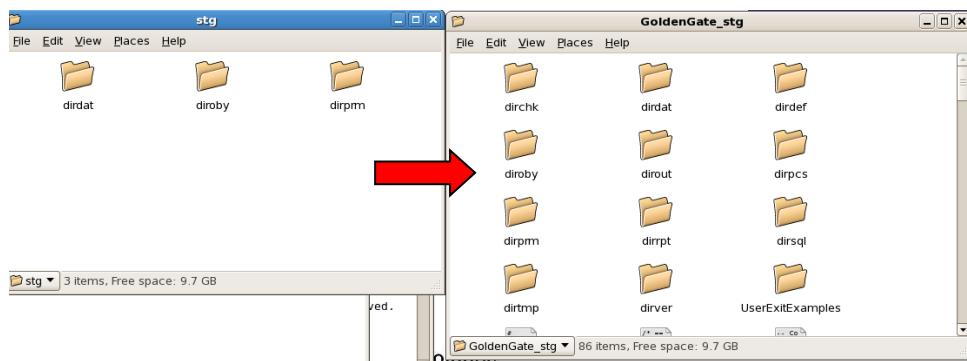


- 2) Change into the **ODIS_to_ODIT1** directory. You can see two directories containing the configuration files for the source and staging OGG installations, as well as a **Readme.txt** file explaining the configuration of OGG using these files.
- 3) **Readme.txt** file can be reviewed for the configuration steps needed as part of the GG set up. We will perform the 6 actions listed in the **Readme** in the coming steps.
- 4) **UPLOAD FILES TO SOURCE MACHINE:** Change directories into **/tmp/ODIS_to_ODIT1/src**. It contains 3 subdirectories dirdat, diroby, and

dirprm which mirror the common OGG directory structure. Open a second file explorer to the directory /app/GoldenGate and copy those 3 subdirectories into it.



- 5) **UPLOAD FILES TO STAGING MACHINE:** Change directories into **/tmp/ODIS_to_ODIT1/stg**. It contains 3 subdirectories dirdat, diroby, and dirprm which mirror the common OGG directory structure. Open a second file explorer to the directory **/app/GoldenGate_stg** and copy those 3 subdirectories into it.



6) **RUN THE SOURCE OBEY FILE**

Enter the following command (Same as in the Readme.txt file):

From /app/GoldenGate directory

> app/GoldenGate/ggsci paramfile /app/GoldenGate/diroby/ODISS.oby

The following output should be shown:

```
oracle@pts:/app/GoldenGate
File Edit View Terminal Tabs Help
[oracle@pts GoldenGate]$ ggsci paramfile /app/GoldenGate/diroby/ODISS.oby

Oracle GoldenGate Command Interpreter for Oracle
Version 10.4.0.19 Build 002
Linux, x86, 32bit (optimized), Oracle 10 on Sep 25 2009 12:49:31

Copyright (C) 1995, 2009, Oracle and/or its affiliates. All rights reserved.

GGSCI (pts.us.oracle.com) 1> dblogin userid system, password oracle
Successfully logged into database.

GGSCI (pts.us.oracle.com) 2> add trandata CUST_DW_DEV.SRC_CUSTOMER, COLS (CUSTID), NOKEY
Logging of supplemental redo data enabled for table CUST_DW_DEV.SRC_CUSTOMER.

GGSCI (pts.us.oracle.com) 3> add extract ODISC, tranlog, begin now
ERROR: EXTRACT ODISC already exists.

GGSCI (pts.us.oracle.com) 4>

GGSCI (pts.us.oracle.com) 5> add exttrail /app/GoldenGate/dirdat/ODISoc/oc, extract ODISC,
megabytes 100
ERROR: TARGETEXTTRAIL already exists.

GGSCI (pts.us.oracle.com) 6>
GGSCI (pts.us.oracle.com) 6> stop extract ODISC
EXTRACT ODISC is already stopped.

GGSCI (pts.us.oracle.com) 8> add extract ODIT1P, exttrailsource /app/GoldenGate/dirdat/ODI
Soc/oc
ERROR: EXTRACT ODIT1P already exists.

GGSCI (pts.us.oracle.com) 9> add rmttrail /app/GoldenGate_stg/dirdat/ODIT1op/op, extract O
DIT1P, megabytes 100
ERROR: TARGETEXTTRAIL already exists.

GGSCI (pts.us.oracle.com) 10> stop extract ODIT1P
EXTRACT ODIT1P is already stopped.

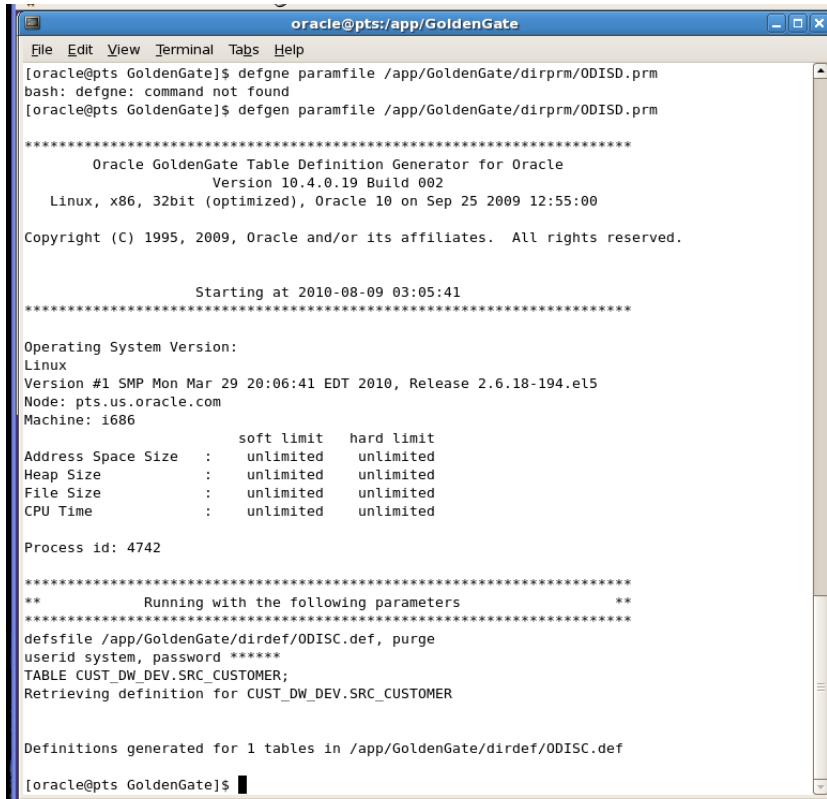
GGSCI (pts.us.oracle.com) 11> start extract ODIT1P

Sending START request to MANAGER ...
EXTRACT ODIT1P starting

[oracle@pts GoldenGate]$
```

- 7) **GENERATE THE DEFINITION FILE:** Enter the following command in /app/GoldenGate directory from terminal window

```
> app/GoldenGate/defgen paramfile /app/GoldenGate/dirprm/ODISD.prm
```



The screenshot shows a terminal window titled "oracle@pts:/app/GoldenGate". The window displays the output of the "defgen" command. It starts with an error message "defgne: command not found", followed by the Oracle GoldenGate Table Definition Generator for Oracle version information. It then shows system configuration details like operating system version, memory limits, and process ID. The generator starts at 2010-08-09 03:05:41 and runs with parameters: defsfile /app/GoldenGate/dirdef/ODISC.def, purge userid system, password *****. It retrieves the definition for TABLE CUST_DW_DEV.SRC_CUSTOMER and generates definitions for 1 table. The command concludes with "[oracle@pts GoldenGate]\$".

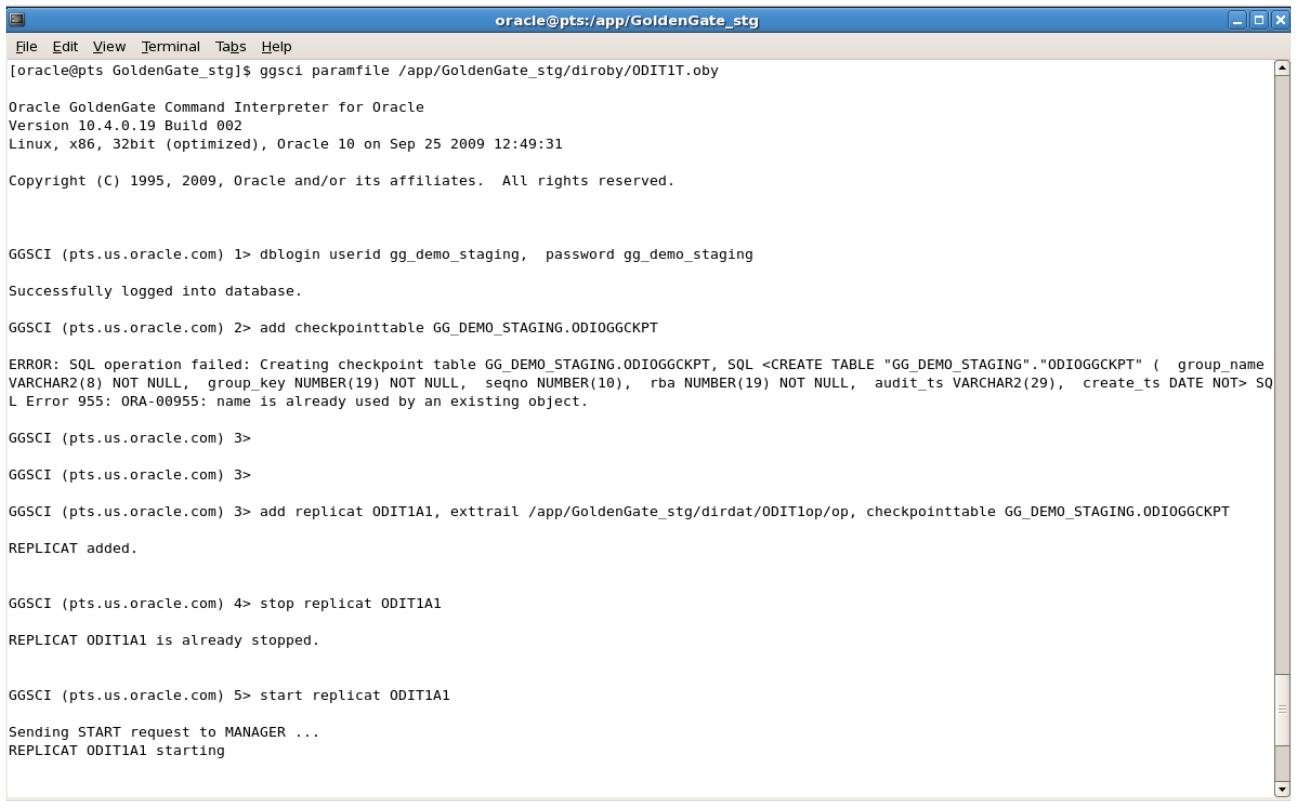
- 8) **COPY DEFINITION FILE:**

Run following command to copy definition file from source to staging area:

```
cp /app/GoldenGate/dirdef/ODISC.def /app/GoldenGate_stg/dirdef/
```

- 9) **RUN THE STAGING OBEY FILE:** Enter the following command in a terminal window from /app/GoldenGate_stg directory

```
> ggsci paramfile /app/GoldenGate_stg/diroby/ODIT1T.oby
```



The screenshot shows a terminal window titled "oracle@pts:/app/GoldenGate_stg". The window contains the following command-line session:

```
[oracle@pts GoldenGate_stg]$ ggsci paramfile /app/GoldenGate_stg/diroby/ODIT1T.oby
Oracle GoldenGate Command Interpreter for Oracle
Version 10.4.0.19 Build 002
Linux, x86, 32bit (optimized), Oracle 10 on Sep 25 2009 12:49:31
Copyright (C) 1995, 2009, Oracle and/or its affiliates. All rights reserved.

GGSCI (pts.us.oracle.com) 1> dblogin userid gg_demo_staging, password gg_demo_staging
Successfully logged into database.

GGSCI (pts.us.oracle.com) 2> add checkpointtable GG_DEMO_STAGING.ODIOGGCKPT
ERROR: SQL operation failed: Creating checkpoint table GG_DEMO_STAGING.ODIOGGCKPT, SQL <CREATE TABLE "GG_DEMO_STAGING"."ODIOGGCKPT" ( group_name
VARCHAR2(8) NOT NULL, group_key NUMBER(19) NOT NULL, seqno NUMBER(10), rba NUMBER(19) NOT NULL, audit_ts VARCHAR2(29), create_ts DATE NOT> SO
L Error 955: ORA-00955: name is already used by an existing object.

GGSCI (pts.us.oracle.com) 3>
GGSCI (pts.us.oracle.com) 3>
GGSCI (pts.us.oracle.com) 3> add replicat ODIT1A1, exttrail /app/GoldenGate_stg/dirdat/ODIT1lop/op, checkpointtable GG_DEMO_STAGING.ODIOGGCKPT
REPLICAT added.

GGSCI (pts.us.oracle.com) 4> stop replicat ODIT1A1
REPLICAT ODIT1A1 is already stopped.

GGSCI (pts.us.oracle.com) 5> start replicat ODIT1A1
Sending START request to MANAGER ...
REPLICAT ODIT1A1 starting
```

If you see error on message for add checkpoint table in log above, you can ignore it since the error can occur if table already exists.

Important:

After you finished above steps, from /app/GoldenGate dir, open ggsci shell by typing ggsci at command line. > **ggsci** . Once in ggsci shell, you can check status of gg processes by typing command **info all** , which should show three processes running on source side; one manager and two extract processes. You can check the same for staging server from /app/GoldenGate_stg dir, which should show two processes running; one manager and one replication process. If any of the process is not running or shows abended status, please check with your instructor before proceeding.

Part 3: Initialize Load to Staging and Target tables

In this exercise you will initialize both staging and target areas with the pre-existing source data.

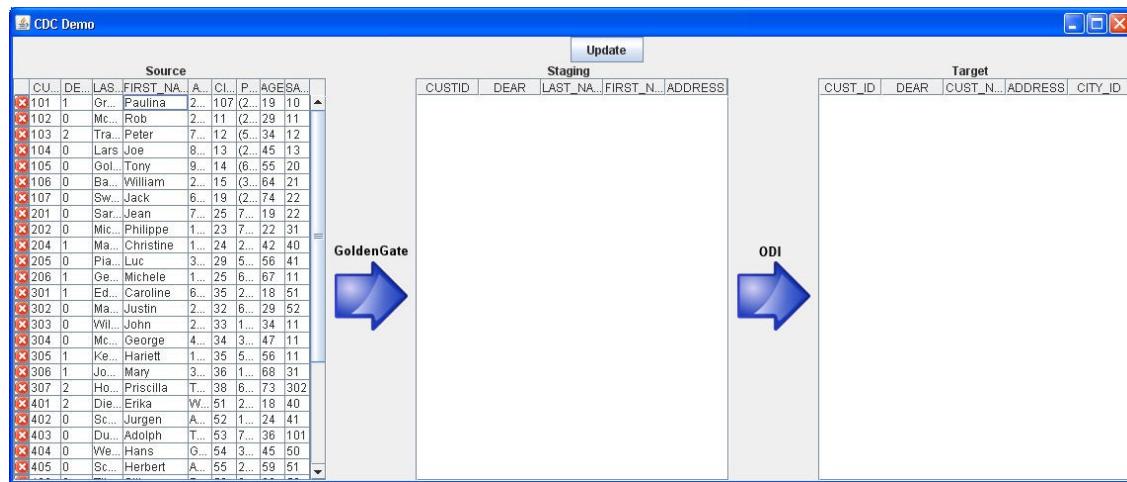
1) Review the data in source, staging, and target table:

Run following command on terminal window to open demo GUI tool to visualize data flow from source to target as data moves.

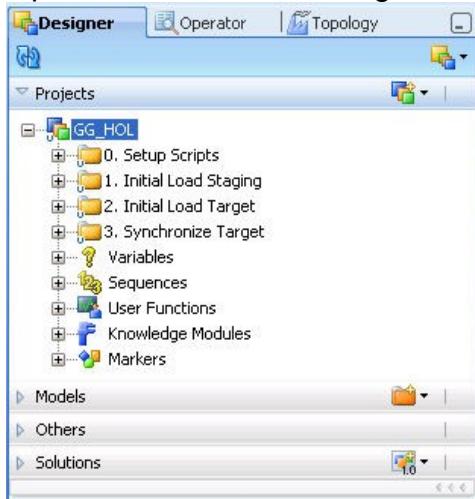
```
> startGGDemo
```

GoldenGate will capture the changes from source and will replicate the change in staging table from which ODI CDC module for goldengate picks up the change and moves the data to target.

This client is a custom Java program to display changes in the 3 tables as they occur. At this point only the source table has data. Keep this client tool running during next set of steps to see data at each stage as it moves. This tool is for demo purpose only to visualize and understand ODI+GoldenGate Lab data flow.



- 2) Open the ODI Studio and go to Designer > Projects.



- 3) **Review folders:** The relevant folders for this GG_HOL project are:

1. **Initial Load Staging:** Performs a 1-to-1 copy from the source tables to the staging tables.

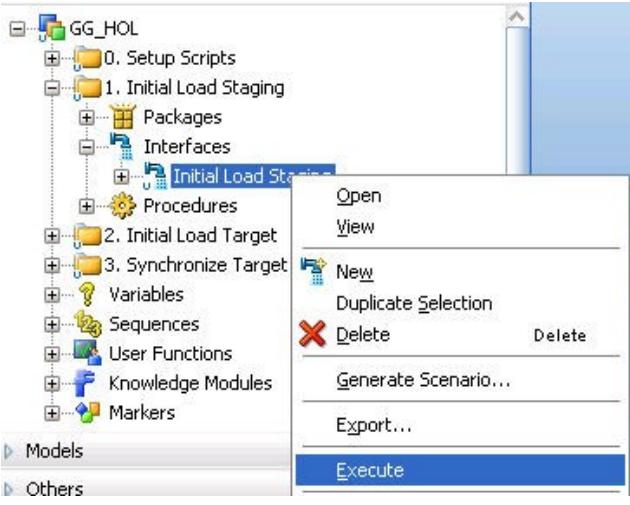
2. **Initial Load Target:** Does an bulk load & transformation from staging to target tables

3. **Synchronize Target:** Performs a CDC-triggered load&transformation from staging to target.

- 4) Expand the folder **1. Initial Load Staging** and its **Interfaces** subfolder:



- 5) Right click on interface **Initial Load Staging** and select **Execute**. Press **Ok** on the **Execution** and then **Information** dialogs that appear.



- 6) Open the ETL Demo Client to review the loading of the staging data:

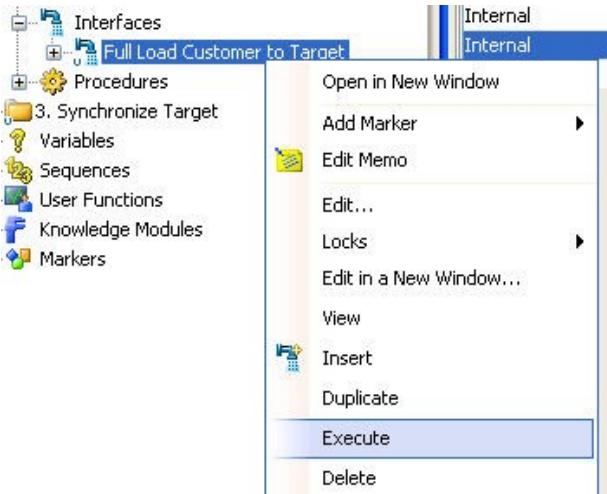
CUST_ID	DEAR	LAS	FIRST_NA	A	CL	P	AGE	SOURCE
101	1	Gr...	Paulina	2...	107	(2...	19	10
102	0	Mc...	Rob	2...	11	(2...	29	11
103	2	Tra...	Peter	7...	12	(5...	34	12
104	0	Lars	Joe	8...	13	(2...	45	13
105	0	Gol...	Tony	9...	14	(6...	55	20
106	0	Ba...	William	2...	15	(3...	64	21
107	0	Sw...	Jack	6...	19	(2...	74	22
201	0	Sar...	Jean	7...	25	7...	18	22
202	0	Mic...	Philippe	1...	23	7...	22	31
204	1	Ma...	Christine	1...	24	2...	42	40
205	0	Pia...	Luc	3...	29	5...	56	41
206	1	Ge...	Michele	1...	25	6...	67	11
301	1	Ed...	Caroline	6...	35	2...	18	51
302	0	Ma...	Justin	2...	32	6...	28	52
303	0	Wil...	John	2...	33	1...	34	11
304	0	Mc...	George	4...	34	3...	47	11
305	1	Ke...	Hariett	1...	35	5...	58	11
306	1	Jo...	Mary	3...	36	1...	68	31
307	2	Ho...	Priscilla	T...	38	6...	73	302
401	2	Die...	Erika	W...	51	2...	18	40
402	0	Sc...	Jurgen	A...	52	1...	24	41
403	0	Du...	Adolph	T...	53	7...	38	101
404	0	We...	Hans	G...	54	3...	45	50
405	0	Sc...	Herbert	A...	55	2...	58	51

CUSTID	DEAR	LAST_N...	FIRST_N...	ADDRESS
101	1	Green	Paulina	29 Jisper...
102	0	McCarthy	Rob	27 Pasa...
103	2	Travis	Peter	7935 Ha...
104	0	Lars	Joe	87 Carm...
105	0	Goldsch...	Tony	91 Torre...
106	0	Baker	William	2890 Or...
107	0	Swenson	Jack	64 Imagin...
201	0	Sartois	Jean	71 rue R...
202	0	Michaud	Philippe	197 imp...
204	1	Martin	Christine	12 allee...
205	0	Piageto	Luc	38 allee ...
206	1	Gentil	Michele	17 mont...
301	1	Edwards	Caroline	68 Dow...
302	0	Marlow	Justin	290 York...
303	0	Wilson	John	28 Sutto...
304	0	McCartney	George	45 Glent...
305	1	Keegan	Hariett	10 Hami...
306	1	Jones	Mary	34 Apple...
307	2	Hopkins	Priscilla	The Gab...
401	2	Diemers	Erika	Wiesen...
402	0	Schiller	Jurgen	Auf dem ...
403	0	Durnstein	Adolph	Thomas...
404	0	Weimar	Hans	Goetherl...
405	0	Schultz	Herbert	Am Holz...

- 7) Expand the folder 2. Initial Load Target and its Interfaces subfolder:



- 8) Right click on interface **Full Load Customer to Target** and select **Execute**. Press **Ok** on the **Execution** and then **Information** dialogs that appear.



9) Open the ETL Demo Client to review the loading of the target data:

Source								
CUST_ID	DEAR	LAST_N.	FIRST_N.	ADDRESS	CITY_ID	AGESA...	CI...	P...
101	1	Green	Paulina	20 Jasper...	107		10	
102	0	Mc...	Rob	27 Pasa...	11		11	
103	2	Tr...	Peter	7835 Ha...	12		12	
104	0	Lars	Joe	87 Carm...	13		13	
105	0	Gol...	Tony	91 Torre...	14		14	
106	0	Ba...	William	2890 Gr...	15		15	
107	0	Sw...	Jack	64 Imagi...	16		16	
201	0	Sar...	Jean	71 rue R...	22		22	
202	0	Mic...	Philippe	197 Imp...	23		23	
204	1	Ma...	Christine	12 allee ...	24		24	
205	0	Pia...	Luc	38 allee ...	25		25	
206	1	Ge...	Michel	17mont...	26		26	
301	1	Ed...	Caroline	68 Dow...	27		27	
302	0	Ma...	Justin	290 York...	28		28	
303	0	Wi...	John	28 Sutton...	29		29	
304	0	Mc...	George	45 Glent...	30		30	
305	1	Ke...	Hanett	10 Hamil...	31		31	
306	1	Jo...	Mary	34 Apple...	32		32	
307	2	Ho...	Priscilla	The Gab...	33		33	
401	2	Die...	Erika	Wiesen...	34		34	
402	0	Sc...	Jurgen	Auf dem ...	35		35	
403	0	Du...	Adolph	Thomas...	36		36	
404	0	We...	Hans	Goetheri...	37		37	
405	0	Sc...	Herbert	Am Holz...	38		38	

Staging								
CUSTID	DEAR	LAST_N.	FIRST_N.	ADDRESS	CITY_ID	AGESA...	CI...	P...
101	1	Green	Paulina	20 Jasper...	107		10	
102	0	McCarthy	Rob	27 Pasa...	11		11	
103	2	Travis	Peter	7835 Ha...	12		12	
104	0	Lars	Joe	87 Carm...	13		13	
105	0	Goldsch...	Tony	91 Torre...	14		14	
106	0	Baker	William	2890 Gr...	15		15	
107	0	Swenson	Jack	64 Imagi...	16		16	
201	0	Sartois	Jean	71 rue R...	22		22	
202	0	Michaud	Philippe	197 Imp...	23		23	
204	1	Martin	Christine	12 allee ...	24		24	
205	0	Piaget	Lu	38 allee ...	25		25	
206	1	Gentil	Michele	17mont...	26		26	
301	1	Edwards	Caroline	68 Dow...	27		27	
302	0	Marlow	Justin	290 York...	28		28	
303	0	Wilson	John	28 Sutton...	29		29	
304	0	McCartney	George	45 Glent...	30		30	
305	1	Keegan	Hariett	10 Hamil...	31		31	
306	1	Jones	Mary	34 Apple...	32		32	
307	2	Hopkins	Priscilla	The Gab...	33		33	
401	2	Diemers	Erika	Wiesen...	34		34	
402	0	Schiller	Jurgen	Auf dem ...	35		35	
403	0	Durnstein	Adolph	Thomas...	36		36	
404	0	Weimar	Hans	Goetheri...	37		37	
405	0	Schultz	Herbert	Am Holz...	38		38	

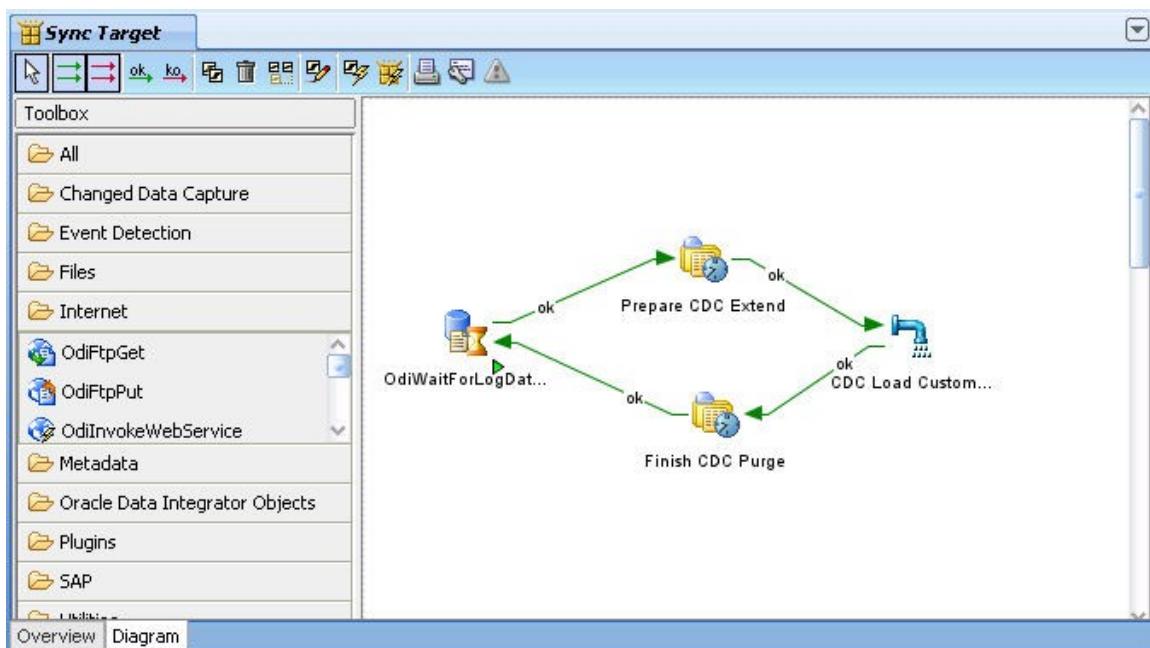
Target								
CUST_ID	DEAR	CUST_N.	ADDRESS	CITY_ID	AGESA...	CI...	P...	
101	1	Paulina	20 Jasper...	107		10		
102	0	Rob Mc...	27 Pasa...	11		11		
103	2	Peter Tr...	7835 Ha...	12		12		
104	0	Joe Lars	87 Carm...	13		13		
105	0	Tony Gol...	91 Torre...	14		14		
106	0	William	2890 Gr...	15		15		
107	0	Jack Sw...	64 Imagi...	16		16		
201	0	Jean Sa...	71 rue R...	25		25		
202	0	Phillippe...	197 Imp...	26		26		
204	1	Christin...	12 allee ...	27		27		
205	0	Luc Piag...	38 allee ...	28		28		
206	1	Michèle ...	17mont...	29		29		
301	1	Caroline...	68 Dow...	30		30		
302	0	Justin M...	290 York...	31		31		
303	0	John Will...	28 Sutton...	32		32		
304	0	George ...	45 Glent...	33		33		
305	1	Hanett K...	10 Hamil...	34		34		
306	1	Mary Jon...	34 Apple...	35		35		
307	2	Priscilla ...	The Gab...	36		36		
401	2	Erika Diem...	Wiesen...	37		37		
402	0	Jurgen S...	Auf dem ...	38		38		
403	0	Adolph D...	Thomas...	39		39		
404	0	Hans Weim...	Goetheri...	40		40		
405	0	Herbert Sc...	Am Holz...	41		41		

Part 4: Initialize CDC process and perform change operations

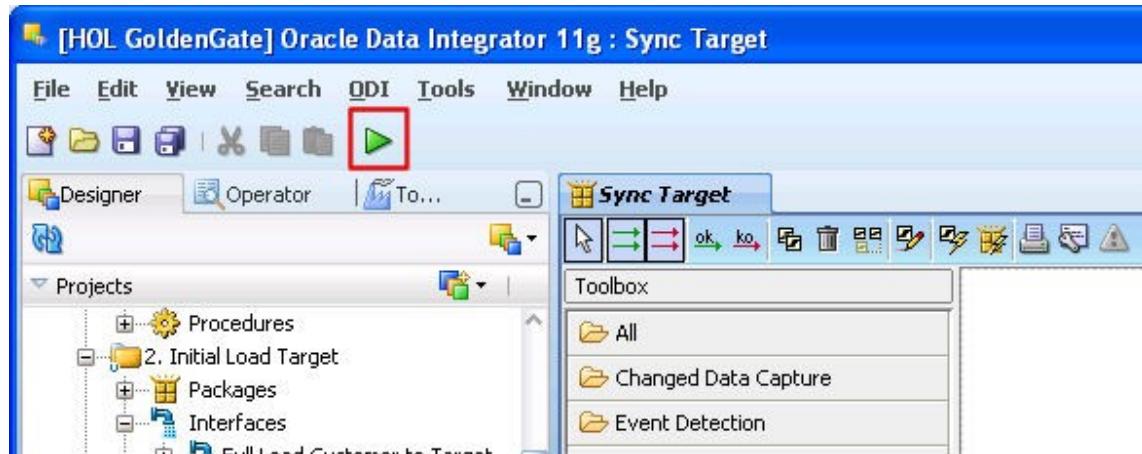
- 1) Open the ODI Studio and go to Designer > Projects.
- 2) Expand the folder **3. Synchronize Target** and its **Packages** folder.



- 3) Double-Click on the **Sync Target** package to open it. Click on the **Diagram** tab. The interface consists of a perpetual loop that is triggered by a new journal log entry and processes new changes with the CDC Load Customer interface.



- 4) Press the Execute button (in the toolbar to start the interface.



- 5) Open the Operator from the toolbar to review the process of the Sync Target interface.



- 6) Open the **Date > Today > Sync Target** tree branch in the operator. The interface is currently running and waiting for a new journal entry at OdiWaitForLogData 1.



- 7) Open the **ETL Demo Client** and double-click on a **FIRST_NAME** of an entry in the Source table. The entry is editable.

Source									
	CU...	DE...	LAS...	FIRST_NA...	A...	CI...	P...	AGE	SA...
	101	1	Gr...	Paulina	2...	107	(2...	19	10
	102	In	Mr	Rnh	?	11	72	29	11

- 8) Edit the name to something else and press Enter. Watch the Staging and Target tables. After a few seconds the **FIRST_NAME** in the Staging table flashes yellow and after that the concatenated **CUST_NAME** in the Target

table.

Source

CUST_ID	DEAR	LAS	FIRST_NA	A...	CL...	P...	AGE_S...
101	1	Gr...	Paulinus	2...	107	(2...	19
102	0	Mc...	Rob	2...	11	(2...	29
103	2	Tra...	Peter	7...	12	(5...	34
104	0	Lars	Joe	8...	13	(2...	45
105	0	Gol...	Tony	9...	14	(6...	55
106	0	Ba...	William	2...	15	(3...	64

Staging

CUSTID	DEAR	LAST_N	FIRST_N	ADDRESS
101	1	Green	Paulinus	20 Jisper...
102	0	McCarthy	Rob	27 Pasa...
103	2	Travis	Peter	7835 Ha...
104	0	Lars	Joe	87 Carm...
105	0	Goldsch...	Tony	91 Torre...
106	0	Baker	William	2890 Gr...

Target

CUST_ID	DEAR	CUST_NAME	ADD...	CITY...
101	1	Paulina Green	20 J...	107
102	0	Rob McCarthy	27 P...	11
103	2	Peter Travis	7835...	12
104	0	Joe Lars	87 C...	13
105	0	Tony Goldschm...	91 T...	14
106	0	William Baker	2890...	15

Source

CUST_ID	DEAR	LAS	FIRST_NA	A...	CL...	P...	AGE_S...
101	1	Gr...	Paulinus	2...	107	(2...	19
102	0	Mc...	Rob	2...	11	(2...	29
103	2	Tra...	Peter	7...	12	(5...	34
104	0	Lars	Joe	8...	13	(2...	45
105	0	Gol...	Tony	9...	14	(6...	55
106	0	Ba...	William	2...	15	(3...	64

Staging

CUSTID	DEAR	LAST_N	FIRST_N	ADDRESS
101	1	Green	Paulinus	20 Jisper...
102	0	McCarthy	Rob	27 Pasa...
103	2	Travis	Peter	7835 Ha...
104	0	Lars	Joe	87 Carm...
105	0	Goldsch...	Tony	91 Torre...
106	0	Baker	William	2890 Gr...

Target

CUST_ID	DEAR	CUST_NAME	ADD...	CITY...
101	1	Paulinus Green	20 J...	107
102	0	Rob McCarthy	27 P...	11
103	2	Peter Travis	7835...	12
104	0	Joe Lars	87 C...	13
105	0	Tony Goldschm...	91 T...	14
106	0	William Baker	2890...	15

Lab 17: Web Services and Extending ODI with SDK

A : Data Services

A.1. Introduction

So far, we have mostly processed data in a batch oriented approach. ODI SOA architecture will now allow us to expose data and changed data through Web Services for a real time access to the information.

There are two types of Web Services:

Data Services are specialized Web Services that provide access to data in datastores, and to changes captured for these datastores using Changed Data Capture. These Web Services are automatically generated by Oracle Data Integrator and deployed to a Web Services container in an application server.

Run-Time Services are Web Services that enable users to leverage Oracle Data Integrator features in a service-oriented architecture (SOA). These Web Services are invoked by a third-party application to perform different Oracle Data Integrator execution tasks such as executing a scenario, restarting a session, listing execution contexts and scenarios.

- The **Public Web Service** connects to the repository to retrieve a list of context and scenarios. This web service is deployed in a Java EE application server.
- The **Agent Web Service** commands the Oracle Data Integrator Agent to start and monitor a scenario and to restart a session. Note that this web service is built-in the Java EE or Standalone Agent.

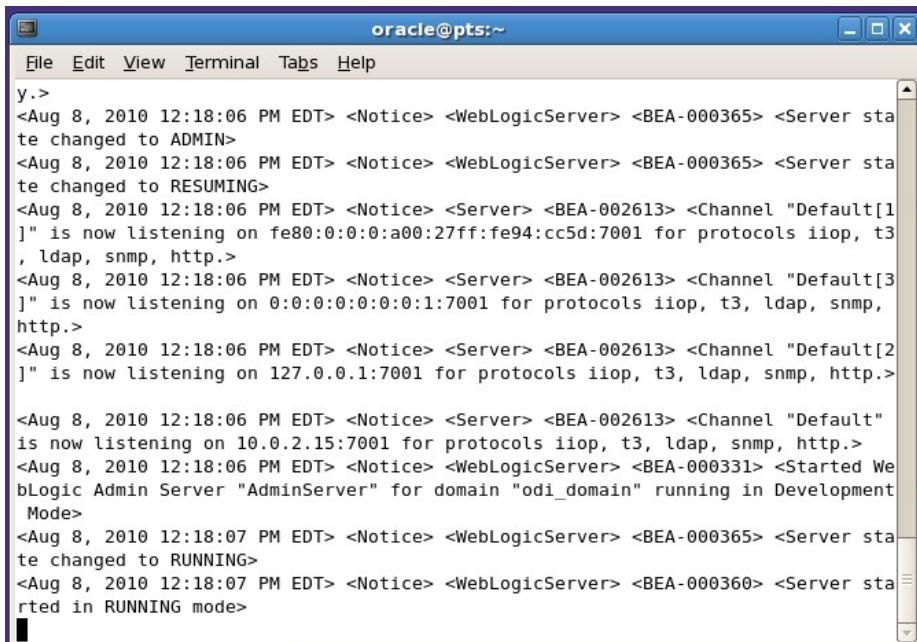
A.2. Web Services Server Setup

The code and runtime components for Web Services will be generated by ODI, but the Web Services themselves will be provided by a Web Services server that will host these components. For this exercise we use WebLogic Server 11g.

In our environment, WebLogic has been installed to serve as our Web Services Server. To start WebLogic, simply execute following command on terminal window:

> startWLS

Once WebLogic server starts, terminal window will show server in running mode. Do not close the window, you can minimize it.



A screenshot of a terminal window titled "oracle@pts:~". The window displays a log of messages from a WebLogic server. The log includes various notices about server state changes, channel listening, and deployment status.

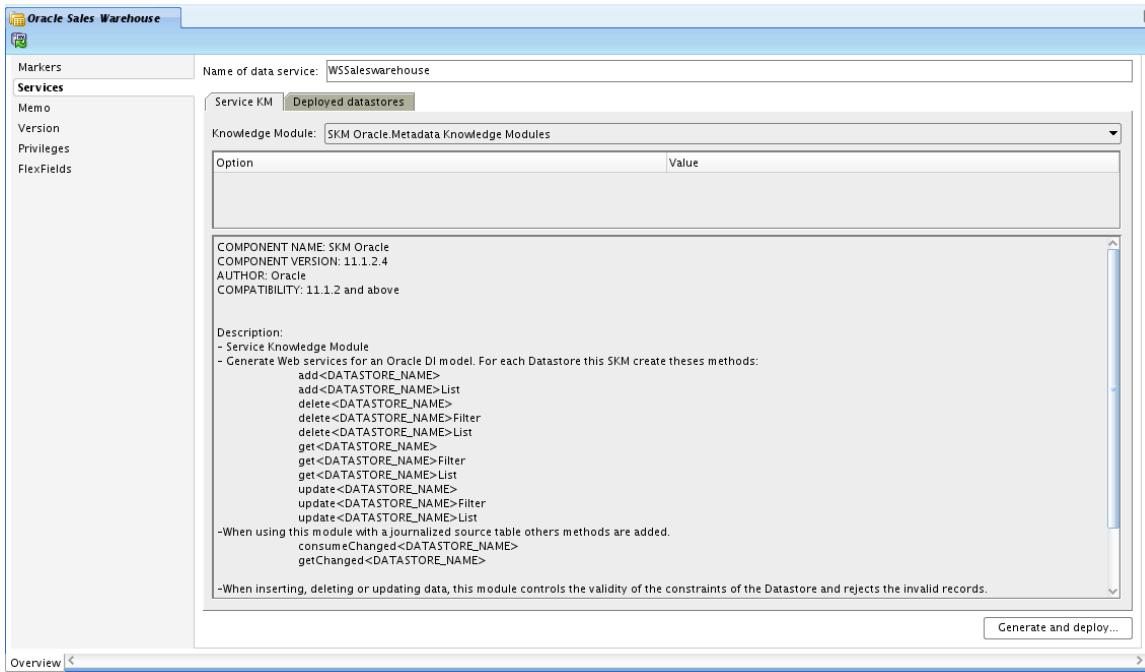
```
y.>
<Aug 8, 2010 12:18:06 PM EDT> <Notice> <WebLogicServer> <BEA-000365> <Server state changed to ADMIN>
<Aug 8, 2010 12:18:06 PM EDT> <Notice> <WebLogicServer> <BEA-000365> <Server state changed to RESUMING>
<Aug 8, 2010 12:18:06 PM EDT> <Notice> <Server> <BEA-002613> <Channel "Default[1]" is now listening on fe80:0:0:0:a00:27ff:fe94:cc5d:7001 for protocols iiop, t3, ldap, snmp, http.>
<Aug 8, 2010 12:18:06 PM EDT> <Notice> <Server> <BEA-002613> <Channel "Default[3]" is now listening on 0:0:0:0:0:0:1:7001 for protocols iiop, t3, ldap, snmp, http.>
<Aug 8, 2010 12:18:06 PM EDT> <Notice> <Server> <BEA-002613> <Channel "Default[2]" is now listening on 127.0.0.1:7001 for protocols iiop, t3, ldap, snmp, http.>
<Aug 8, 2010 12:18:06 PM EDT> <Notice> <Server> <BEA-002613> <Channel "Default" is now listening on 10.0.2.15:7001 for protocols iiop, t3, ldap, snmp, http.>
<Aug 8, 2010 12:18:06 PM EDT> <Notice> <WebLogicServer> <BEA-000331> <Started WebLogic Admin Server "AdminServer" for domain "odi_domain" running in Development Mode>
<Aug 8, 2010 12:18:07 PM EDT> <Notice> <WebLogicServer> <BEA-000365> <Server state changed to RUNNING>
<Aug 8, 2010 12:18:07 PM EDT> <Notice> <WebLogicServer> <BEA-000360> <Server started in RUNNING mode>
```

If you want to look into more details as to how the connection to WebLogic is defined in ODI, look in the Topology navigator for the connection information under the technology “WebLogic”. You will find there the necessary information for ODI to store the ODI Web Services web application into WebLogic’s autodeploy directory.

The JDBC datasource for this exercise is defined and deployed to the local WebLogic server.

A.3. Generate and Deploy

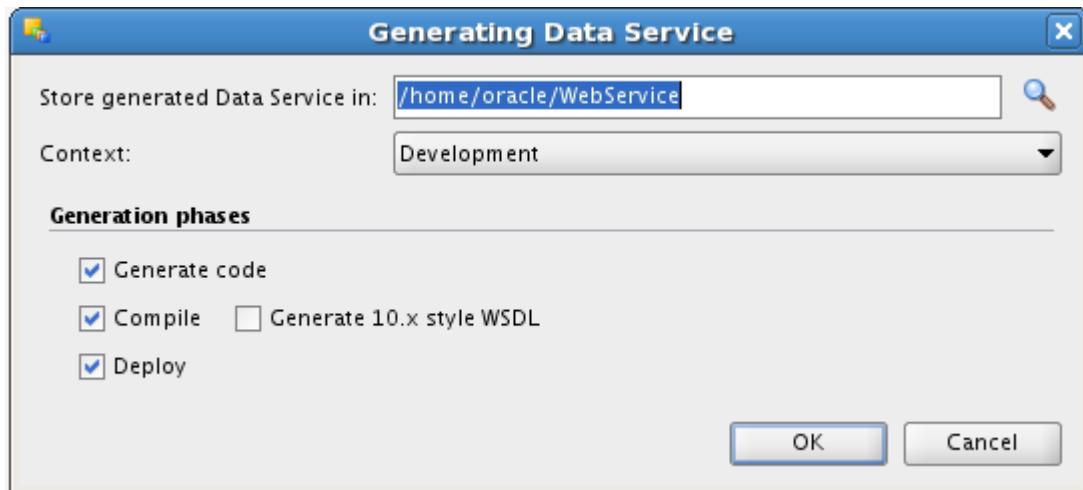
Edit the *Oracle Sales Warehouse* model, go to the *Services* tab and select the appropriate SKM: *SKM Oracle*.



Then click on the *Deployed Datastores* tab, and select the *TRG_CUSTOMER* table.

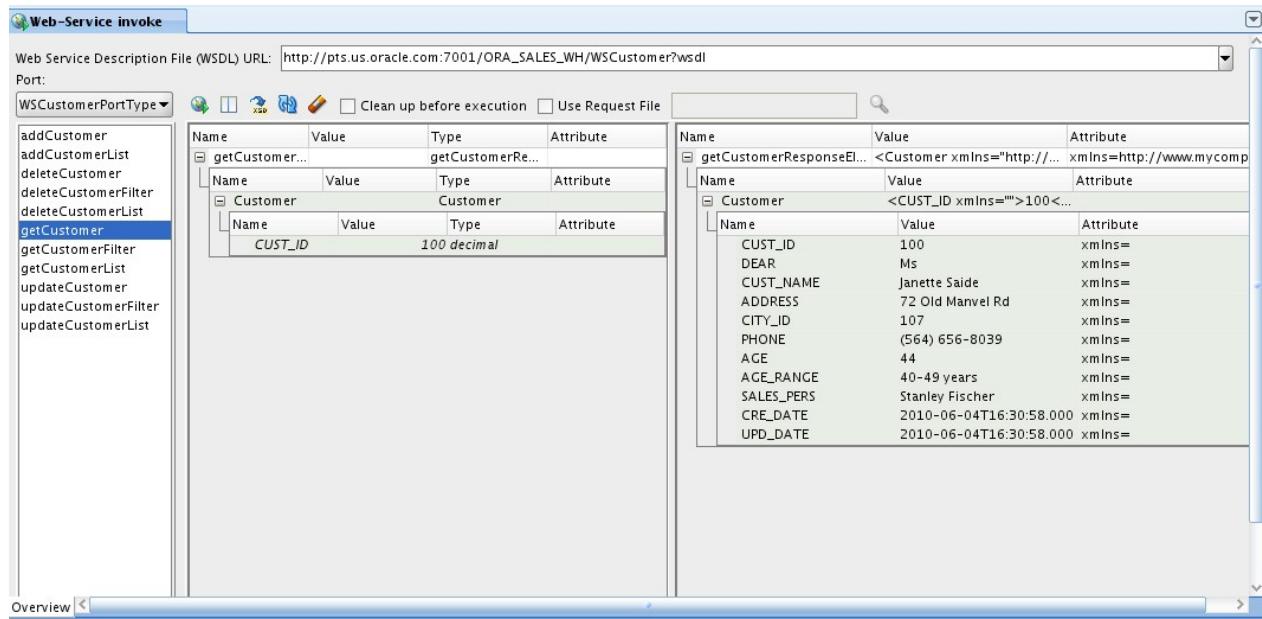
Act...	Datastore name	Web Service name	Published entity
<input type="checkbox"/>	ARCHIVE_CUSTOMER	WSCustomer	Customer
<input type="checkbox"/>	TRG_CITY	WSTcity	Tcity
<input type="checkbox"/>	TRG_COUNTRY	wstrgCountry	TrgCountry
<input checked="" type="checkbox"/>	TRG_CUSTOMER	WSCustomer	Customer
<input type="checkbox"/>	TRG_INVENTORY	WSTrgInventory	TrgInventory
<input type="checkbox"/>	TRG_PRODUCT	WSTrgProduct	TrgProduct
<input type="checkbox"/>	TRG_PROD_FAMILY	WSTrgProdFamily	TrgProdFamily
<input type="checkbox"/>	TRG_REGION	WSTrgRegion	TrgRegion
<input type="checkbox"/>	TRG_SALES	WSTrgSales	TrgSales

Finally, click *Generate and Deploy*. Select all the options on the following window, and click OK:



A.4. Test the Web Service

- Check that the Web service is properly listed by your app server
- Right-click on the TRG_CUSTOMER table, click on Test Web Service
- Select getCustomer method and enter 100 as Cust_id
- Remove rest of the blank column by clicking button
- Click on invoke icon to test Web service
- Test Web Service will retrieve the data and show it in SOAP Test GUI.



You can stop Weblogic Server, if not needed, by running following command:

> stopWLS

B : Use SDK to Manage ODI

B.1. Introduction

With ODI 11g, SDK has been improved to provide better access to ODI methods programmatically. SDK provides API to create dynamic mapping or create and execute ODI scenario programmatically. SDK also allows for embedding ODI into third party GUI tool and manage ODI objects through APIs.

B.2. SDK Examples

In workshop Image, we have placed few examples Java program which leverages ODI API to create Agent, Context and execute Scenario. You can run any or all of these examples to understand SDK use and gain understanding of using SDK within Java program.

B.3. Running Java Program using SDK

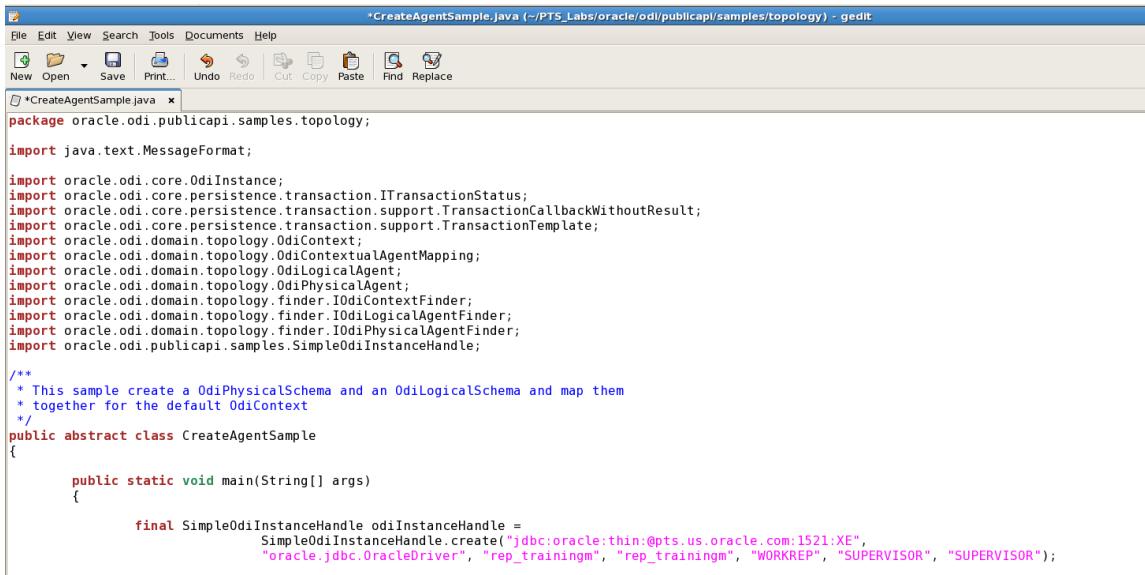
- 1) Open terminal window and go to dir /home/oracle/PTS_Lab

```
cd /home/oracle/PTS_Labs
```

- 2) Make sure of connection information, repository information, etc used in the program. Please open java file that creates agent programmatically. From PTS_Lab dir, open java file using gedit command:

```
gedit oracle/odi/publicapi/samples/topology/CreateAgentSample.java
```

- 3) Make sure repository information, username, pwd, and other parameters are correct. You should change the repository information, connection etc. to a desired repository (e.g. rep_trainingm or newly created SNPM1 repository from Lab 4) so that you can view the execution of program easily. Type in Master repository username and pwd; ODI work repository name; default ODI username and pwd.



```

*CreateAgentSample.java (~/PTS_Labs/oracle/odi/publicapi/samples/topology) - gedit
File Edit View Search Tools Documents Help
New Open Save Print... Undo Redo Cut Copy Paste Find Replace
* *CreateAgentSample.java x
package oracle.odi.publicapi.samples.topology;

import java.text.MessageFormat;

import oracle.odi.core.OdiInstance;
import oracle.odi.core.persistence.transaction.ITransactionStatus;
import oracle.odi.core.persistence.transaction.support.TransactionCallbackWithoutResult;
import oracle.odi.core.persistence.transaction.support.TransactionTemplate;
import oracle.odi.domain.topology.OdiContext;
import oracle.odi.domain.topology.OdiContextualAgentMapping;
import oracle.odi.domain.topology.OdiLogicalAgent;
import oracle.odi.domain.topology.OdiPhysicalAgent;
import oracle.odi.domain.topology.finder.IDiContextFinder;
import oracle.odi.domain.topology.finder.IDiLogicalAgentFinder;
import oracle.odi.domain.topology.finder.IDiPhysicalAgentFinder;
import oracle.odi.publicapi.samples.SimpleOdiInstanceHandle;

/**
 * This sample create a OdiPhysicalSchema and an OdiLogicalSchema and map them
 * together for the default OdiContext
 */
public abstract class CreateAgentSample
{
    public static void main(String[] args)
    {
        final SimpleOdiInstanceHandle odiInstanceHandle =
            SimpleOdiInstanceHandle.create("jdbc:oracle:thin:@pts.us.oracle.com:1521:XE",
                                         "oracle.jdbc.OracleDriver", "rep_trainingm", "rep_trainingm", "WORKREP", "SUPERVISOR", "SUPERVISOR");
    }
}

```

4) Save changes made by clicking save button and close the gedit window.

5) To compile java file just created/modified, run following command from PTS_Labs directory:

**/app/Oracle/Middleware/jdk160_18/bin/javac -cp \$SDKDEMO_CLASSPATH
oracle.odi/publicapi/samples/topology/CreateAgentSample.java**

You can copy/paste the commands from run_sdk_demo.txt file within PTS_Lab directory.

6) To run and test the Java program just compiled, run following command:

a) First make sure you add current dir into class path by running following command:

export SDKDEMO_CLASSPATH=.:\$SDKDEMO_CLASSPATH

b) To run java program that creates agent, use following command:

**/app/Oracle/Middleware/jdk160_18/bin/java -cp \$SDKDEMO_CLASSPATH
oracle.odi.publicapi.samples.topology.CreateAgentSample**

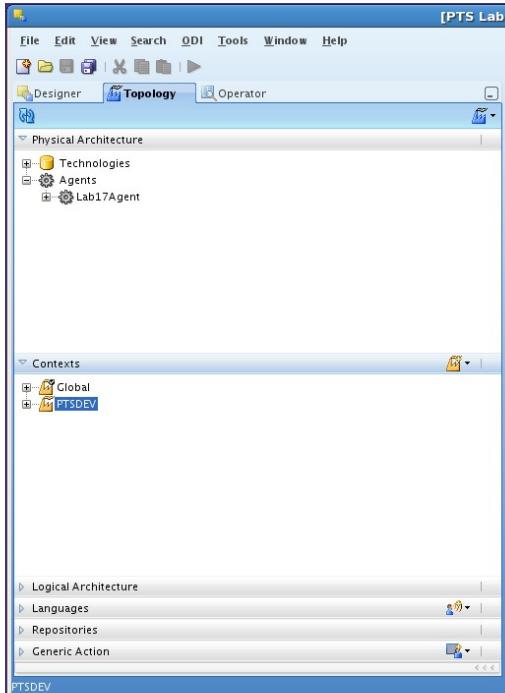
The image shows two terminal windows side-by-side. The left window displays a command-line interface with the following session:

```
[oracle@pts PTS_Labs]$ ls
Models oracle run_sdk_demo.txt
[oracle@pts PTS_Labs]$ vi run_sdk_demo.txt
[oracle@pts PTS_Labs]$
```

The right window shows the output of running a Java program:

```
[oracle@pts PTS_Labs]$ gedit oracle/odi/publicapi/samples/topology/CreateAgentSample.java
[oracle@pts PTS_Labs]$ /app/Oracle/Middleware/jdk160_18/bin/javac -cp $SDKDEMO_CLASSPATH oracle/odi/publicapi/samples/topology/CreateAgentSample.java
[oracle@pts PTS_Labs]$ /app/Oracle/Middleware/jdk160_18/bin/java -cp $SDKDEMO_CLASSPATH oracle.odi.publicapi.samples.topology.CreateAgentSample
Aug 8, 2010 10:45:35 PM org.eclipse.persistence.persistance.default
INFO: work-session logout successful
Aug 8, 2010 10:45:35 PM org.eclipse.persistence.persistance.default
INFO: master-session logout successful
[oracle@pts PTS_Labs]$
```

- 7) You can run CreateContextSample.java file by modifying it to appropriate parameter, compile and run it to see how it creates context programmatically.
- 8) Verify within ODI studio that Agent and Context you specified in java programs get created:



- 9) There are many example java files available if you wish to explore them. These sample java files comes with installation under demo folder:

/app/Oracle/Middleware/Oracle_ODI/oracledi/demo/SDK

Oracle Corporation
World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065
USA

Worldwide inquiries:
Phone: +1 650 506 7000
Fax: +1 650 506 7200

www.oracle.com

Oracle is the information company

Oracle is a registered trademark of Oracle Corporation.
Various product and service names referenced herein may be trademarks of Oracle Corporation. All other product and service names mentioned may be trademarks of their respective owners.

Copyright © 2010 Oracle Corporation

All rights reserved.

ORACLE®