

## **Oracle® Fusion Middleware**

Getting Started with Oracle Data Integrator 12c

December 2018

---

---

# Contents

<b>Preface</b> .....	<b>6</b>
Audience .....	6
Documentation Accessibility .....	6
Related Documents .....	6
Conventions .....	7
<b>1 Oracle Data Integrator Overview</b> .....	<b>8</b>
<b>1.1 Introduction to Oracle Data Integrator</b> .....	<b>8</b>
1.1.1 The Business Problem .....	8
1.1.2 A Unique Solution .....	8
<b>1.2 ODI Component Architecture</b> .....	<b>9</b>
1.2.1 Repositories .....	10
1.2.2 ODI Studio and User Interfaces .....	10
1.2.3 Run-Time Agent.....	11
1.2.4 Management Pack for Oracle Data Integrator .....	12
1.2.5 Oracle Data Integrator Console .....	12
1.2.6 Oracle Enterprise Manager Fusion Middleware Control.....	12
<b>1.3 Get Started with Oracle Data Integrator</b> .....	<b>12</b>
<b>2 Working with the ETL Project</b> .....	<b>13</b>
<b>21 The Example Environment</b> .....	<b>13</b>
<b>22 The Data Models</b> .....	<b>14</b>
2.2.1 Orders Application.....	14
2.2.2 Parameters .....	14
2.2.3 Sales Administration – Oracle.....	15
<b>23 Integration Challenges</b> .....	<b>16</b>
<b>3 Introduction to Using Oracle Data Integrator Studio</b> .....	<b>17</b>
<b>31 Using the ODI Studio Navigators</b> .....	<b>17</b>
3.1.1 Starting Oracle Data Integrator Studio .....	17
<b>32 Designer Navigator</b> .....	<b>18</b>
<b>33 Operator Navigator</b> .....	<b>20</b>
<b>4 Working with Mappings</b> .....	<b>21</b>

<b>41</b>	<b><i>Load TRG_CUSTOMER Mapping Example .....</i></b>	<b><i>21</i></b>
4.1.1	Purpose and Integration Requirements.....	21
4.1.2	Mapping Definition.....	22
4.1.3	Creating the Mapping .....	24
<b>42</b>	<b><i>Load TRG_SALES Mapping Example.....</i></b>	<b><i>38</i></b>
4.2.1	Purpose and Integration Requirements.....	38
4.2.2	Mapping Definition.....	39
4.2.3	Creating the Mapping .....	40
<b>5</b>	<b>Implementing Data Quality Control .....</b>	<b>48</b>
<b>51</b>	<b><i>Introduction to Data Integrity Control.....</i></b>	<b><i>48</i></b>
<b>52</b>	<b><i>SRC_CUSTOMER Control Example.....</i></b>	<b><i>49</i></b>
5.2.1	Objective.....	49
5.2.2	Interpreting the Problem .....	50
5.2.3	Creating Constraints.....	50
5.2.4	Run the Static Control .....	53
5.2.5	Follow the Execution of the Control in Operator Navigator .....	54
5.2.6	Interpreting the Results in Operator Navigator .....	55
<b>6</b>	<b>Working with Packages .....</b>	<b>58</b>
<b>61</b>	<b><i>Introduction.....</i></b>	<b><i>58</i></b>
6.1.1	Automating Data Integration Flows .....	58
6.1.2	Packages.....	58
<b>62</b>	<b><i>Load Sales Administration Package Example .....</i></b>	<b><i>59</i></b>
6.2.1	Purpose.....	59
6.2.2	Mappings Provided with Oracle Data Integrator .....	59
6.2.3	Problem Analysis.....	60
6.2.4	Creating the Package.....	61
<b>7</b>	<b>Executing Your Developments and Reviewing the Results .....</b>	<b>64</b>
<b>71</b>	<b><i>Executing the Load Sales Administration Package.....</i></b>	<b><i>64</i></b>
7.1.1	Run the Package.....	64
7.1.2	Follow the Execution of the Package in Operator Navigator .....	64
7.1.3	Interpreting the Results of the Load TRG_CUSTOMER Session Step .....	65
<b>8</b>	<b>Deploying Integrated Applications .....</b>	<b>68</b>
<b>81</b>	<b><i>Introduction.....</i></b>	<b><i>68</i></b>
<b>82</b>	<b><i>Scenario Creation .....</i></b>	<b><i>68</i></b>
<b>83</b>	<b><i>Run the Scenario .....</i></b>	<b><i>69</i></b>
8.3.1	Executing a Scenario from ODI Studio .....	70
<b>84</b>	<b><i>Follow the Execution of the Scenario.....</i></b>	<b><i>70</i></b>

<b>9</b>	<b>Using Oracle Data Integrator with Oracle GoldenGate .....</b>	<b>71</b>
<b>91</b>	<b><i>Introduction.....</i></b>	<b>71</b>
9.1.1	Connect to the ODI Work Repository .....	72
<b>92</b>	<b><i>Reviewing the Oracle GoldenGate JAgent configuration in ODI Studio .....</i></b>	<b>75</b>
<b>93</b>	<b><i>Initial load .....</i></b>	<b>76</b>
<b>931</b>	<b><i>Starting the ODI and OGG Demo Client.....</i></b>	<b>76</b>
<b>932</b>	<b><i>Running the Mappings .....</i></b>	<b>77</b>
<b>94</b>	<b><i>Setting up Changed Data Capture .....</i></b>	<b>81</b>
<b>95</b>	<b><i>Synchronizing the changed data .....</i></b>	<b>88</b>
<b>951</b>	<b><i>Load TRG_CUSTOMER Mapping.....</i></b>	<b>88</b>
<b>952</b>	<b><i>Sync Data Package .....</i></b>	<b>91</b>
<b>10</b>	<b>Going Further with Oracle Data Integrator.....</b>	<b>94</b>
<b>101</b>	<b><i>Summary .....</i></b>	<b>94</b>
10.1.1	Getting Started Tutorial Solution .....	94
<b>102</b>	<b><i>What else can you do with Oracle Data Integrator? .....</i></b>	<b>96</b>
<b>103</b>	<b><i>Learn More .....</i></b>	<b>97</b>

---

# Preface

This manual describes how to get started with Oracle Data Integrator. It provides general background information and detailed examples to help you learn how to use Oracle Data Integrator.

This preface contains the following topics:

- Audience
- Documentation Accessibility
- Related Documents
- Conventions

## Audience

This document is intended for users interested in learning how to use Oracle Data Integrator as a development tool for their integration processes.

## Documentation Accessibility

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

### Access to Oracle Support

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

## Related Documents

For more information, see the following Oracle resources:

- *Oracle Fusion Middleware Developer's Guide for Oracle Data Integrator*
- *Oracle Fusion Middleware Installation Guide for Oracle Data Integrator*
- *Oracle Fusion Middleware Upgrade Guide for Oracle Data Integrator*
- *Oracle Fusion Middleware Connectivity and Knowledge Modules Guide for Oracle Data Integrator*
- *Oracle Fusion Middleware Knowledge Module Developer's Guide for Oracle Data Integrator*
- *Oracle Data Integrator 12c Online Help*
- *Oracle Data Integrator 12c Release Notes, included with your Oracle Data Integrator 12c installation, and on Oracle Technology Network*

## Conventions

The following text conventions are used in this document:

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

---

# 1 Oracle Data Integrator Overview

This chapter provides an introduction to Oracle Data Integrator, the technical architecture, and the contents of this Getting Started guide.

This chapter includes the following sections:

- Section 1.1, "Introduction to Oracle Data Integrator"
- Section 1.2, "Oracle Data Integrator Component Architecture"
- Section 1.3, "Get Started with Oracle Data Integrator"

## 1.1 Introduction to Oracle Data Integrator

A widely used data integration software product, Oracle Data Integrator provides a new declarative design approach to defining data transformation and integration processes, resulting in faster and simpler development and maintenance. Based on a unique *E-LT architecture (Extract - Load Transform)*, Oracle Data Integrator not only guarantees the highest level of performance possible for the execution of data transformation and validation processes but is also the most cost-effective solution available today.

Oracle Data Integrator provides a unified infrastructure to streamline data and application integration projects.

### 1.1.1 The Business Problem

In today's increasingly fast-paced business environment, organizations need to use more specialized software applications; they also need to ensure the coexistence of these applications on heterogeneous hardware platforms and systems and guarantee the ability to share data between applications and systems. Projects that implement these integration requirements need to be delivered on-spec, on-time and on-budget.

### 1.1.2 A Unique Solution

Oracle Data Integrator employs a powerful declarative design approach to data integration, which separates the declarative rules from the implementation details. Oracle Data Integrator is also based on a unique E-LT (Extract - Load Transform) architecture which eliminates the need for a standalone ETL server and proprietary engine, and instead leverages the inherent power of your RDBMS engines. This combination provides the greatest productivity for both development and maintenance, and the highest performance for the execution of data transformation and validation processes.

Here are the key reasons why companies choose Oracle Data Integrator for their data integration needs:

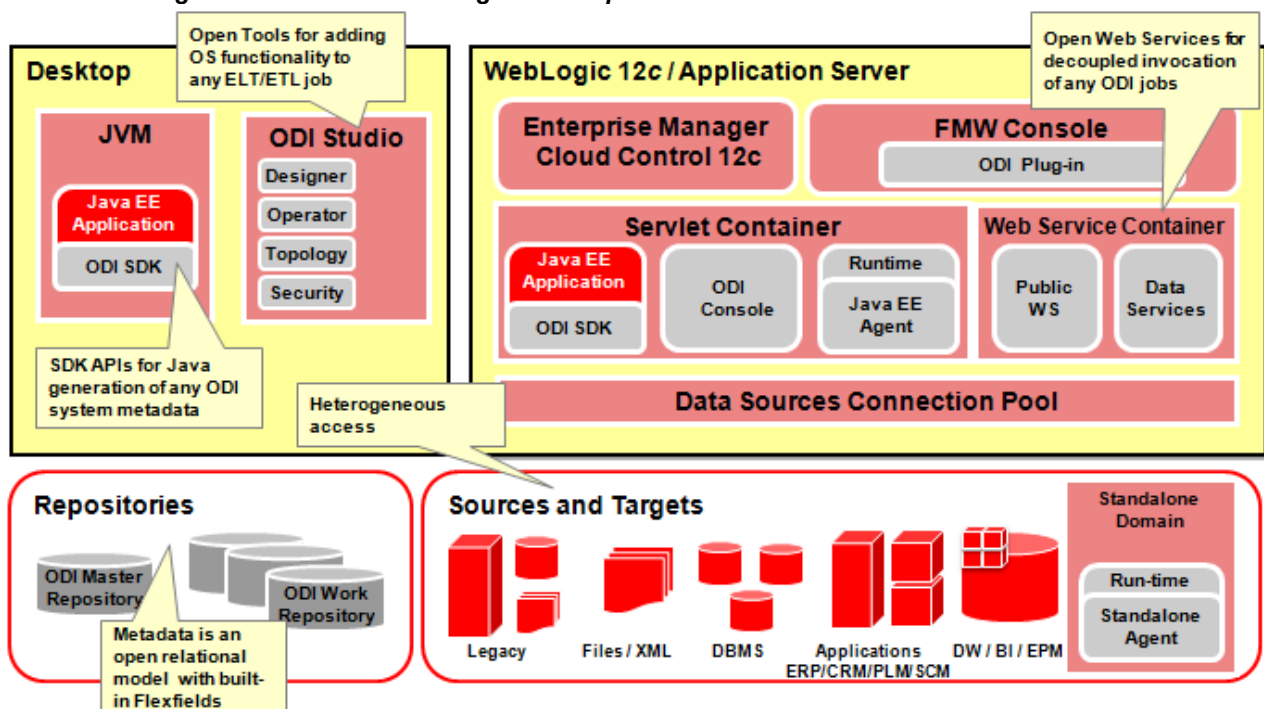
- **Faster and simpler development and maintenance:** The declarative rules driven approach to data integration greatly reduces the learning curve of the product and increases developer productivity while facilitating ongoing maintenance. This approach separates the definition of the processes from their actual implementation, and separates the declarative rules (the "what") from the data flows (the "how").
- **Data quality firewall:** Oracle Data Integrator ensures that faulty data is automatically detected and recycled before insertion in the target application. This is performed without the need for programming, following the data integrity rules and constraints defined both on the target application and in Oracle Data Integrator.

- **Better execution performance:** traditional data integration software (ETL) is based on proprietary engines that perform data transformations row by row, thus limiting performance. By implementing an E-LT architecture, based on your existing Big Data or database engines, you are capable of executing data transformations on the target server at a set-based level, giving you much higher performance.
- **Simpler and more efficient architecture:** the E-LT architecture removes the need for an ETL Server sitting between the sources and the target server. It utilizes the source and target servers to perform complex transformations, most of which happen in batch mode when the server is not busy processing end-user queries.
- **Platform Independence:** Oracle Data Integrator supports many platforms, hardware and OSs with the same software.
- **Data Connectivity:** Oracle Data Integrator supports many Big Data technologies and databases including Spark, Spark Streaming, Hive and Kafka or leading Data Warehousing platforms such as Oracle, Exadata, Teradata, IBM DB2, Netezza and numerous other technologies such as Big Data, flat files, ERPs, LDAP, XML.
- **Cost-savings:** the elimination of the ETL Server and ETL engine reduces both the initial hardware and software acquisition and maintenance costs. The reduced learning curve and increased developer productivity significantly reduce the overall labor costs of the project, as well as the cost of ongoing enhancements.

## 1.2 ODI Component Architecture

The Oracle Data Integrator platform integrates in the broader Fusion Middleware platform and becomes a key component of this stack. Oracle Data Integrator provides its run-time components as Java EE applications, enhanced to fully leverage the capabilities of the Oracle WebLogic Application Server. Oracle Data Integrator components include exclusive features for Enterprise-Scale Deployments, high availability, scalability, and hardened security. Figure 1–1 shows the ODI component architecture.

**Figure 1–1 Oracle Data Integrator Component Architecture**





## 1.2.1 Repositories

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, for example *Development*, *QA*, *User Acceptance*, and *Production*. 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 is composed of one *Master Repository* and several *Work Repositories*. Objects developed or configured through the user interfaces are stored in one of these repository types.

There is usually only one master repository that stores the following information:

- Security information including users, profiles and rights for the ODI platform
- Topology information including technologies, server definitions, schemas, contexts, languages and so forth.
- Versioned and archived objects.

The work repository is the one that contains actual developed objects. Several work repositories may coexist in the same ODI installation (for example, to have separate environments or to match a particular versioning life cycle). A Work Repository stores information for:

- Models, including schema definition, datastores structures and metadata, fields and columns definitions, data quality constraints, cross references, data lineage and so forth.
- Projects, including business rules, packages, procedures, folders, Knowledge Modules, variables and so forth.
- Scenario execution, including scenarios, scheduling information and logs.

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

## 1.2.2 ODI Studio and User Interfaces

Administrators, Developers and Operators use the Oracle Data Integrator Studio to access the repositories. This user interface is used for administering the infrastructure (security and topology), reverse-engineering the metadata, developing projects, scheduling, operating and monitoring executions.

ODI Studio provides four Navigators for managing the different aspects and steps of an ODI integration project:

- *Designer Navigator* is used to design data integrity checks and to build transformations such as for example:
  - Automatic reverse-engineering of existing applications or databases
  - Graphical development and maintenance of transformation mappings
  - Visualization of data flows in the mappings
  - Automatic documentation generation
  - Customization of the generated code

- *Operator Navigator* is the production management and monitoring tool. It is designed for IT production operators. Through Operator Navigator, you can manage your mapping executions in the sessions, as well as the scenarios in production.
- *Topology Navigator* is used to manage the data describing the information system's physical and logical architecture. Through Topology Navigator you can manage the topology of your information system, the technologies and their datatypes, the data servers linked to these technologies and the schemas they contain, the contexts, the languages and the agents, as well as the repositories. The site, machine, and data server descriptions will enable Oracle Data Integrator to execute the same mappings in different physical environments.
- *Security Navigator* is the tool for managing the security information in Oracle Data Integrator. Through Security Navigator you can create users, roles and profiles and assign user rights for methods (edit, delete, etc) on generic objects (data server, datatypes, etc), and fine-tune these rights on the object instances (Server 1, Server 2, etc).

Oracle Data Integrator also provides a Java API for performing all these run-time and design-time operations. This *Oracle Data Integrator Software Development Kit (SDK)* is available for standalone Java applications and application servers.

### 1.2.3 Run-Time Agent

At design time, developers generate scenarios from the business rules that they have designed. The code of these scenarios is then retrieved from the repository by the Run-Time Agent. This agent then connects to the data servers and orchestrates the code execution on these servers.

It retrieves the return codes and messages for the execution, as well as additional logging information – such as the number of processed records, execution time and so forth – in the Repository.

The Agent comes in three different flavors:

- Java Enterprise Edition (Java EE) Agents are deployed on Oracle WebLogic Server and can benefit from the application server layer features such as clustering for High Availability requirements. Java EE Agents can be managed using Oracle Enterprise Manager.
- Standalone Agents can be installed on the source or target systems and require a Java Virtual Machine.
- Colocated Standalone Agents can be installed on the source or target systems as well. They can be managed using Oracle Enterprise Manager and must be configured with an Oracle WebLogic domain. Colocated Standalone Agents can run on a separate machine from the Oracle WebLogic Administration Server

These agents are multi-threaded java programs that support load balancing and can be distributed across the information system. The Agent holds its own execution schedule which can be defined in Oracle Data Integrator, and can also be called from an external scheduler. It can also be invoked from a Java API or a web service interface.

To manage and monitor the Java EE and Colocated Standalone Agents as well as the ODI Console, Oracle Data Integrator provides a plug-in that integrates with Oracle Enterprise Manager Cloud Control as well as Oracle Fusion Middleware Control Console.

## 1.2.4 Management Pack for Oracle Data Integrator

The Management Pack for Oracle Data Integrator leverages Oracle Enterprise Manager Cloud Control best-in-class application performance management, service level management and configuration management capabilities to provide a centralized management solution for Oracle Data Integrator Enterprise Edition.

For more information about the Management Pack for Oracle Data Integrator, please visit the following link: <http://www.oracle.com/us/products/middleware/data-integration/management-pack-for-odi/overview/index.html>

## 1.2.5 Oracle Data Integrator Console

Business users (as well as developers, administrators and operators), can have read access to the repository, perform topology configuration and production operations through a web based UI called *Oracle Data Integrator Console*. This web application can be deployed in Oracle WebLogic Server.

## 1.2.6 Oracle Enterprise Manager Fusion Middleware Control

Fusion Middleware Control organizes a wide variety of performance data and administrative functions into distinct, Web-based home pages for the farm, cluster, domain, servers, components, and applications. The Fusion Middleware Control home pages make it easy to locate the most important monitoring data and the most commonly used administrative functions all from your Web browser.

## 1.3 Get Started with Oracle Data Integrator

Table 1–1 summarizes the contents of this guide.

**Table 1–1 Content Summary**

This chapter	Describes how to...
Chapter 2, "Working with the ETL Project"	Provides an introduction to the demonstration environment delivered with Oracle Data Integrator Studio
Chapter 3, "Introduction to Using Oracle Data Integrator Studio"	Start the demonstration environment and Oracle Data Integrator Studio
Chapter 4, "Working with Mappings"	Create and work with Mappings in Oracle Data Integrator
Chapter 5, "Implementing Data Quality Control"	Implement data quality control
Chapter 6, "Working with Packages"	Create and work with Packages in Oracle Data Integrator
Chapter 7, "Executing Your Developments and Reviewing the Results"	Execute your developments, follow the execution, and interpret the execution results
Chapter 8, "Deploying Integrated Applications"	Run an ODI Package automatically in a production environment
Chapter 9, " Using Oracle Data Integrator with Oracle GoldenGate "	Configure and use Changed Data Capture (CDC) with Oracle GoldenGate and Oracle Data Integrator
Chapter 10, "Going Further with Oracle Data Integrator"	Perform advanced tasks with Oracle Data Integrator

---

## 2 Working with the ETL Project

This chapter provides an introduction to the *ETL* (Extract Transform Load) project that is delivered in the demonstration environment with Oracle Data Integrator Studio.

This chapter includes the following sections:

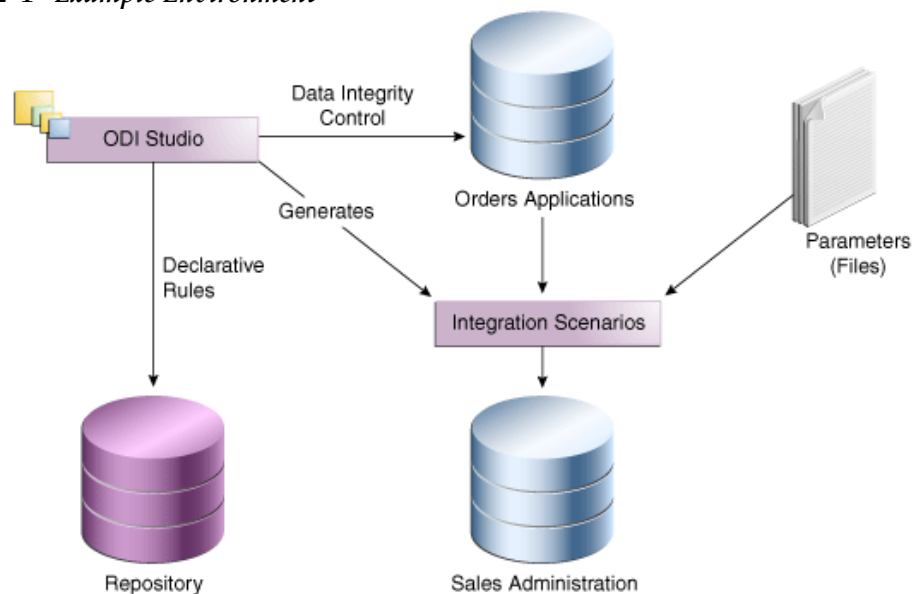
- Section 2.1, "The Example Environment"
- Section 2.2, "The Data Models"
- Section 2.3, "Integration Challenges"

### 2.1 The Example Environment

The *Demo* project is an example to help you understand how to transform and check the integrity of the data in your information systems.

The examples in this getting started guide track sales from various heterogeneous data sources issued from the production systems. Figure 2–1 shows the example environment.

*Figure 2–1 Example Environment*



The example environment uses the following elements:

- *The Repository*: The Repository contains all of the metadata required for the getting started demo examples.
- *Orders Application*: An application for tracking customer orders, hosted in the Oracle database (the "srcdemo" sample).
- *Parameters*: Flat files (ASCII) issued from the production system containing a list of sales representatives and the segmentation of ages into age ranges.

- *Sales Administration*: The administration or tracking of sales, hosted in another Oracle database (the "trgdemo" sample). This data warehouse is populated with our transformations.

## 2.2 The Data Models

The demonstration environment includes three ODI data models:

- Orders Application
- Parameters
- Sales Administration

This section provides the schema diagrams for these data models.

### 2.2.1 Orders Application

The *Orders Application* data model is based on the Oracle RDBMS technology and includes six datastores:

- SRC\_CITY
- SRC\_CUSTOMER
- SRC\_ORDERS
- SRC\_ORDER\_LINES
- SRC\_PRODUCT
- SRC\_REGION

Figure 2–2 shows the schema diagram of this data model.

Note that this data model does not enforce any foreign key constraints, even if some functional relations exist between the data.

**Figure 2–2 Orders Application Schema Diagram**

SRC_REGION				
REGION_ID	NUMERIC(10)	<pk>	not null	
REGION	VARCHAR(50)		null	
COUNTRY_ID	NUMERIC(10)		null	
COUNTRY	VARCHAR(50)		null	

SRC_CITY				
CITY_ID	NUMERIC(10)	<pk>	not null	
CITY	VARCHAR(50)		null	
REGION_ID	NUMERIC(10)		null	
POPULATION	NUMERIC(10)		null	

SRC_ORDER_LINES				
ORDER_ID	NUMERIC(10)	<pk>	not null	
ORDER_ID	NUMERIC(10)	<pk>	not null	
PRODUCT_ID	NUMERIC(10)		null	
QTY	NUMERIC(10)		null	
AMOUNT	NUMERIC(10,2)		null	

SRC_CUSTOMER				
CUSTID	NUMERIC(10)	<pk>	not null	
DEAR	NUMERIC(1)		null	
LAST_NAME	VARCHAR(50)		null	
FIRST_NAME	VARCHAR(50)		null	
ADDRESS	VARCHAR(100)		null	
CITY_ID	NUMERIC(10)		null	
PHONE	VARCHAR(50)		null	
AGE	NUMERIC(3)		null	
SALES_PERS_ID	NUMERIC(10)		null	

SRC_ORDERS				
ORDER_ID	NUMERIC(10)	<pk>	not null	
STATUS	VARCHAR(3)		null	
CUST_ID	NUMERIC(10)		null	
ORDER_DATE	DATE		null	
CUSTOMER	VARCHAR(35)		null	

SRC_PRODUCT				
PRODUCT_ID	NUMERIC(10)	<pk>	not null	
PRODUCT	VARCHAR(50)		null	
PRICE	NUMERIC(10,2)		null	
FAMILY_NAME	VARCHAR(50)		null	

### 2.2.2 Parameters

The *Parameters* data model is based on the File technology and includes two datastores:

- SRC\_SALES\_PERSON
- SRC\_AGE\_GROUP

Figure 2–3 shows the schema diagram of this data model.

**Figure 2–3 Parameters Schema Diagram**

SRC_SALES_PERSON				SRC_AGE_GROUP			
SALES_PERSON_ID	NUMERIC(10)	<pk>	not null	AGE_MIN	NUMERIC(3)	<pk>	not null
FIRST_NAME	VARCHAR(50)		null	AGE_MAX	NUMERIC(3)	<pk>	not null
LAST_NAME	VARCHAR(50)		null	AGE_RANGE	VARCHAR(50)		null
HIRE_DATE	DATE		null				

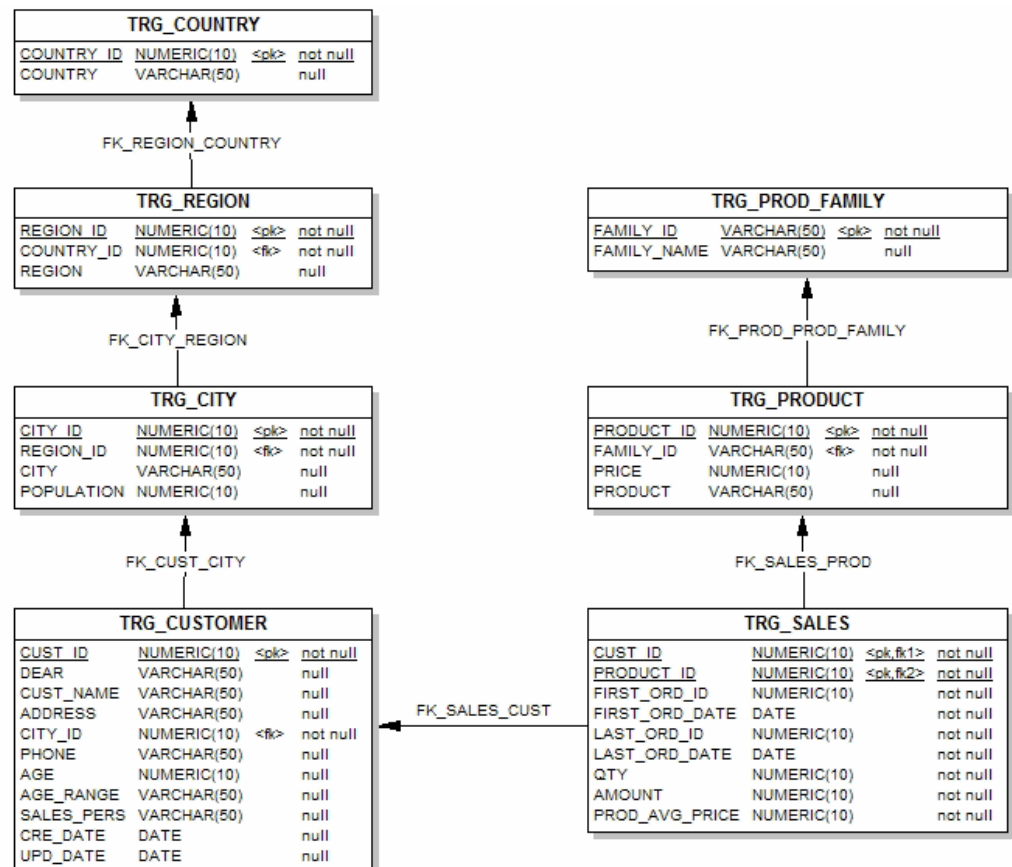
## 2.2.3 Sales Administration – Oracle

The *Sales Administration* data model is based on the Oracle RDBMS technology and includes seven datastores:

- TRG\_CITY
- TRG\_COUNTRY
- TRG\_CUSTOMER
- TRG\_PRODUCT
- TRG\_PROD\_FAMILY
- TRG\_REGION
- TRG\_SALES

Figure 2–4 shows the schema diagram of this data model.

**Figure 2–4 Sales Administration Schema Diagram**



## 23 Integration Challenges

The challenges common to all data integration and transformation projects are:

- Accurately and easily exchanging data between your applications while respecting the business rules of your information system
- Automate end to end process flows
- Visibility over the entire set of data integration processes

The examples used in this guide illustrate how to address these issues. During this getting started guide, you will learn how to:

- **Create mappings to move and transform data**

Two simple examples will show you how to improve productivity by loading the data from Orders Application and Parameters into the Sales Administration data warehouse.

- **Automate the execution of these mappings into packages**

This part of the Getting Started guide will show you how to automate your Oracle Data Integrator processes. The aim of this exercise is to load the entire *Sales Administration* data warehouse with a single click.

- **Execute the package and review the execution results**

You will learn how to execute the Load Sales Administration package and the mappings Load TRG\_CUSTOMER and Load TRG\_SALES you have created and how to review the results of these executions.

- **Prepare the developed components for deployment**

You will learn how to run the Load Sales Administration package automatically in a production environment.

- **Implement Data Quality Control to check data in a database**

By implementing two examples, you will learn how Oracle Data Integrator enables you to ensure the quality of the data in your applications while segregating invalid rows. The *Orders Application* tables contain a number of data inconsistencies that you will detect.

---

**Note:** In this guide, we will be looking at processes that focus on ETL. While it is beyond the scope of this document, implementing different integration patterns (real-time, for example) can be carried out in the same fashion. For more information on this, see the Oracle Data Integrator documentation after completing this guide.

---

Now that you have been introduced to the concepts of the Demo Project and its components, you can move on to Introduction to using ODI Studio.

---

## 3 Introduction to Using Oracle Data Integrator Studio

This chapter describes the first steps towards using Oracle Data Integrator Studio.

### 3.1 Using the ODI Studio Navigators

ODI Studio provides four Navigators for managing the different aspects and steps of an ODI integration project:

- Designer Navigator
- Operator Navigator
- Topology Navigator
- Security Navigator

The tasks performed in this getting started guide take place in Designer Navigator (to create and execute your developments) and in Operator Navigator (to monitor the execution of your developments). This section only describes the Navigators that are used in this getting started guide. See the *Oracle Fusion Middleware Developer's Guide for Oracle Data Integrator* for information about the Topology and Security Navigators.

#### 3.1.1 Starting Oracle Data Integrator Studio

**This section describes how to start Oracle Data Integrator Studio. You can skip to section 3.2 if you have already started ODI Studio.**

To launch ODI Studio:

- On Unix operating systems:  
ODI\_HOME/odi/studio/odi.sh
- On Windows operating systems:  
ODI\_HOME\odi\studio\odi.exe

---

**Note:** On Windows, you can launch ODI Studio from the **Start** menu:

On the **Start** menu, select **All Programs > Oracle > Oracle Data Integrator > ODI Studio**.

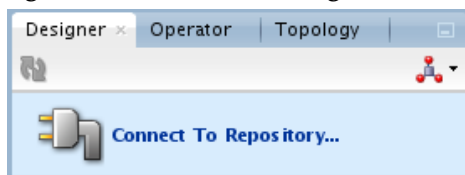
---

Upon launching Studio the first time, you will be prompted with an Import Preferences screen. Proceed to click **No**.

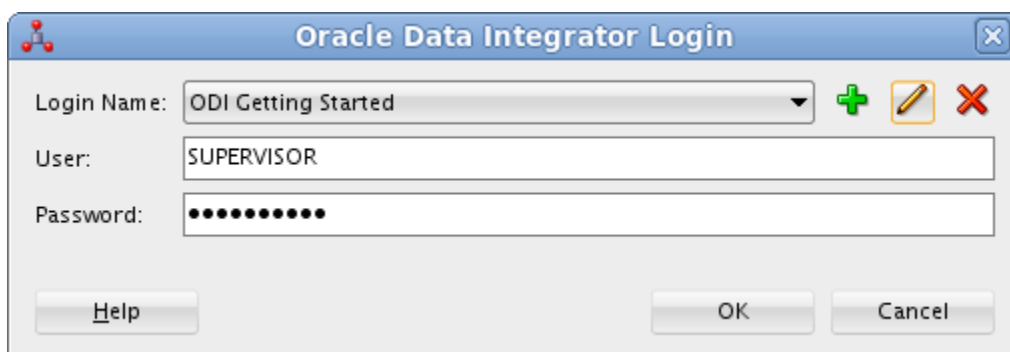


The ODI Studio client will continue to load. Click on Connect to Repository then ensure the Login Name is set to ODI Getting Started and click OK.

**Figure 3-1 Oracle Data Integrator Studio 12c**



When prompted for a wallet password enter *welcome1*



## 3.2 Designer Navigator

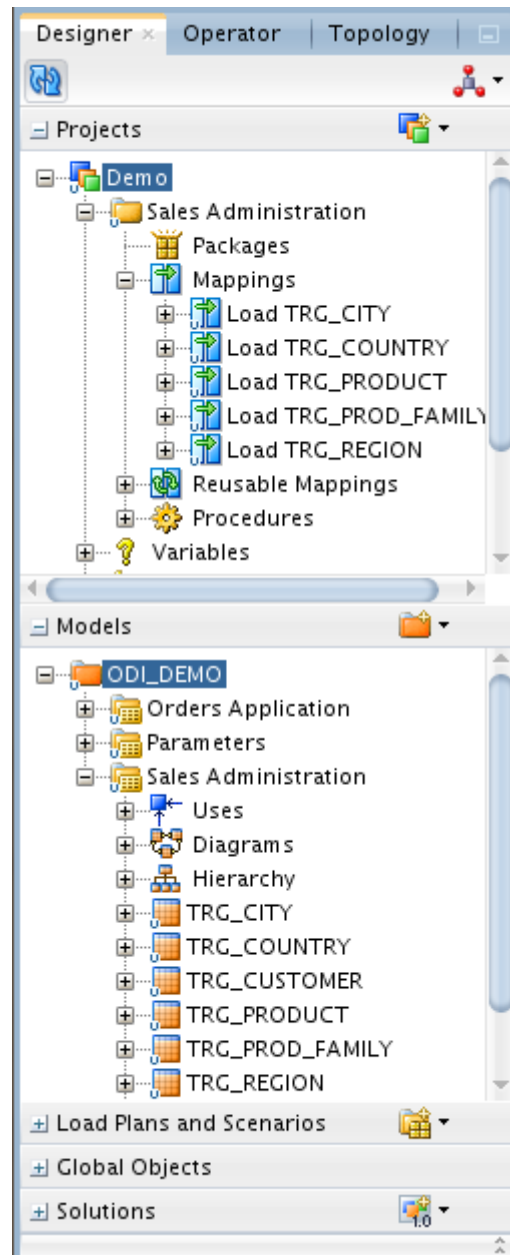
Designer Navigator is used to manage metadata, to design data integrity checks, and to build transformations.

The main objects you handle through Designer Navigator are *models* and *projects*.

- The data models for your applications contain all of the metadata in your data servers (tables, columns, constraints, descriptions, cross-references, etc.)
- The projects contain all of the loading and transformation rules for your data servers (mappings, procedures, variables, etc.)

The Designer Navigator appears as shown in Figure 3–2.

**Figure 3–2 Designer Navigator**



The Designer Navigator has the following accordions:

Projects	The Projects accordion contains the developments made with Designer Navigator.
Models	The Models accordion contains the descriptions of the data and applications structures.
Load Plans and Scenarios	The Load Plan and Scenarios accordion contains generated code and executable objects.
Global Objects	The Global Objects accordion contains the Global User Functions, Variables, Markers, and Sequences.

Solutions	The Solutions accordion contains the Solutions that have been created when working with version management.
-----------	---

The demonstration environment provides the objects you will need in this Getting Started guide:

- In the Models accordion, you will find all the data models corresponding to the *Orders Application*, *Parameters* and *Sales Administration* applications.
- In the Projects accordion, you will find the *Demo* project and the *Sales Administration* folder which already contains several mappings. You will develop your new mappings in this folder.

The necessary Knowledge Modules (KM) are already imported in the Demo Project:

- LKM SQL to SQL (Built-In)
- CKM Oracle
- IKM Oracle Incremental Update

### 3.3 Operator Navigator

Operator Navigator is the management and monitoring tool. It is designed for IT operators and can be used by developers to check code execution and perform debugging operations. Through Operator Navigator, you can manage your development executions in the sessions, as well as the scenarios.

The Operator Navigator has the following accordions:

<b>Session List</b>	The Session List accordion displays all sessions organized per date, physical agent, status, keywords, and so forth.
<b>Hierarchical Sessions</b>	The Hierarchical Sessions accordion displays the execution sessions organized in a hierarchy with their child sessions.
<b>Load Plan Executions</b>	The Load Plan Executions displays the Load Plan Runs of the Load Plan instances
<b>Scheduling</b>	The Scheduling accordion displays the list of physical agents and schedules.
<b>Load Plans and Scenarios</b>	The Scenarios accordion displays the list of scenarios available
<b>Solutions</b>	The Solutions accordion contains the Solutions that have been created when working with version management.

---

## 4 Working with Mappings

This chapter describes how to work with mappings in Oracle Data Integrator. The demonstration environment includes several example mappings. In this chapter you will learn how to create the following mappings:

- Load TRG\_CUSTOMER: This mapping loads the data from the SRC\_CUSTOMER table in the *Orders Application* model into the TRG\_CUSTOMER target table in the *Sales Administration* model.
- Load TRG\_SALES: This mapping loads the data from the SRC\_ORDERS table and from the SRC\_ORDER\_LINES table in the *Orders Application* model into the TRG\_SALES target table in the *Sales Administration* model.

This chapter includes the following sections:

- Section 4.1, "Load TRG\_CUSTOMER Mapping Example"
- Section 4.2, "Load TRG\_SALES Mapping Example"

### 4.1 Load TRG\_CUSTOMER Mapping Example

This section contains the following topics:

- Purpose and Integration Requirements
- Mapping Definition
- Creating the Mapping

#### 4.1.1 Purpose and Integration Requirements

This section describes the integration features and requirements the mapping Load TRG\_CUSTOMER is expected to meet.

The purpose of the Load TRG\_CUSTOMER mapping is to load the data from the SRC\_CUSTOMER table in the *Orders Application* model into the TRG\_CUSTOMER target table in the *Sales Administration* model.

However, the SRC\_CUSTOMER table does not contain all of the data that is required for this operation. The following information has to be added to the target table:

- The age range (AGE\_RANGE) that is defined in the SRC\_AGE\_GROUP flat file in the *Parameters* model corresponds to the AGE attribute in the source table.
- The last and first names of the customer sales rep. (LAST\_NAME and FIRST\_NAME) that is defined in the SRC\_SALES\_PERSON file in the *Parameters* model correspond to the sales representative ID (SALES\_PERS\_ID) in the source table.

- The transformed value of the numeric data (0, 1, 2) from the DEAR column in the source table into an standard salutation text string in the target (Mr, Mrs, or Ms).
- The concatenated first and last names of the source customers.

The source data is not always consistent with the integrity rules implemented in the target environment. For this mapping, the data has to be cleansed by verifying that all constraints are satisfied and by storing invalid rows in an error table rather than in our target database. In this example, two important integrity rules must be satisfied:

- Customers must be older than 21 (condition AGE > 21)
- The customers must be associated with a city (CITY\_ID) that exists in the TRG\_ CITY table (reference FK\_CUST\_CITY)

The functional details for these rules and the procedure to follow are given in Section 5.1.3, "Creating the Mapping".

## 4.1.2 Mapping Definition

This section describes the mapping Load TRG\_CUSTOMER that will be created in this example. See Section 4.1.3, "Creating the Mapping" for more information.

The Load TRG\_CUSTOMER mapping uses the following data and transformations:

- One target datastore. Table 4–1 lists the details of the target datastore.

*Table 4–1 Target Datastore Details of Load TRG\_CUSTOMER*

Model	Datastore	Description	Type
Sales Administration	TRG_CUSTOMER		Oracle Table

- Three source datastores. Table 5–2 lists the details of the source datastores.

*Table 4–2 Source Datastore Details of Load TRG\_CUSTOMER*

Model	Datastore	Description	Type
Orders	SRC_CUSTOMER	Customers in the source	Oracle table
Application		system	
Parameters	SRC_AGE_GROUP	Age bracket file	File delimited by semicolons
Parameters	SRC_SALES_PERSON	Salesperson file	File of fixed-size records

- One **Join**. Table 4–3 lists the details of the join.

**Table 4–3 Joins used in Load TRG\_CUSTOMER**

Join	Description	SQL Rule
Sales Representatives and Customers	Join SRC_SALES_PERSON and SRC_CUSTOMER	SRC_CUSTOMER.SALES_PERS_ID = SRC_SALES_PERSON.SALES_PERS_ID

- One **Lookup** table. Table 4–4 lists the details of the lookup table.

**Table 4–4 Lookups used in Load TRG\_CUSTOMER**

Lookup	Description	SQL Rule
Customers and age range	The customer's age must be between the minimum and maximum ages in the file	SRC_CUSTOMER.AGE between SRC_AGE_GROUP.AGE_MIN and SRC_AGE_GROUP.AGE_MAX

- Several transformation rules. Table 4–5 lists the details of the transformation rules.

**Table 4–5 Transformation Rules used in Load TRG\_CUSTOMER**

Target Column	Origin	SQL Rule(Expression)
CUST_ID	SRC_CUSTOMER.CUSTID	SRC_CUSTOMER.CUSTID
DEAR	If SRC_CUSTOMER.DEAR = 0 then 'MR' If SRC_CUSTOMER.DEAR = 1 then 'MRS' else 'MS'	CASE WHEN CUSTOMER.DEAR=0 THEN 'Mr' WHEN CUSTOMER.DEAR=1 THEN 'Mrs' ELSE 'Ms' END
CUST_NAME	Concatenation of SRC_CUSTOMER.FIRST_NAME and SRC_CUSTOMER.LAST_NAME in upper case	SRC_CUSTOMER.FIRST_NAME    ' '    UPPER(SRC_CUSTOMER.LAST_NAME)
ADDRESS	SRC_CUSTOMER.ADDRESS	SRC_CUSTOMER.ADDRESS
CITY_ID	SRC_CUSTOMER.CITY_ID	SRC_CUSTOMER.CITY_ID
PHONE	SRC_CUSTOMER.PHONE	SRC_CUSTOMER.PHONE
AGE	SRC_CUSTOMER.AGE	SRC_CUSTOMER.AGE
AGE_RANGE	SRC_AGE_GROUP.AGE_RANGE	SRC_AGE_GROUP.AGE_RANGE

SALES_PERS	Concatenation of SRC_SALES_PERSON.FIRST_NAME and SRC_SALES_PERSON.LAST_NAME in uppercase	SRC_SALES_PERSON.FIRST_NAME    ' '    UPPER(SRC_SALES_PERSON.LAST_NAME)
CRE_DATE	Today's date	SYSDATE
UPD_DATE	Today's date	SYSDATE

---

### 4.1.3 Creating the Mapping

This section describes how to create the Load TRG\_CUSTOMER mapping. To create the Load TRG\_CUSTOMER mapping perform the following procedure:

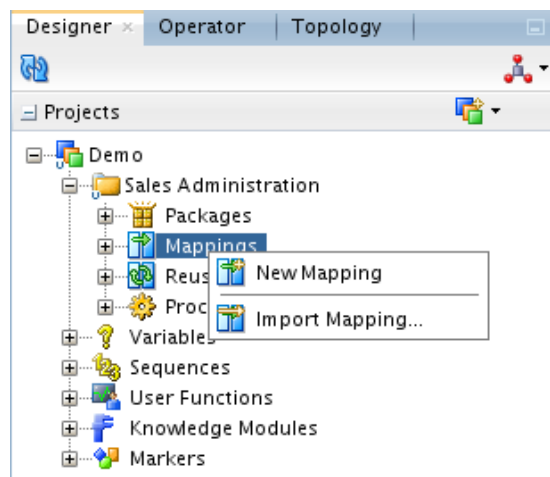
1. Insert a New Mapping
2. Define the Target Datastore
3. Define the Source Datastores
4. Define the Lookup Table
5. Define the Join between the Source Datastores
6. Define the Mappings
7. Define the Data Loading Strategies (LKM)
8. Define the Data Integration Strategies (IKM)
9. Define the Data Control Strategy

#### 4.1.3.1. Insert a New Mapping

To create a new mapping:

1. In Designer Navigator, expand the Demo project node in the Projects accordion.
2. Expand the Sales Administration node.
3. In the Sales Administration folder, right-click the Mapping node and select **New Mapping** as shown in Figure 4–1.

*Figure 4–1 Insert New Mapping*

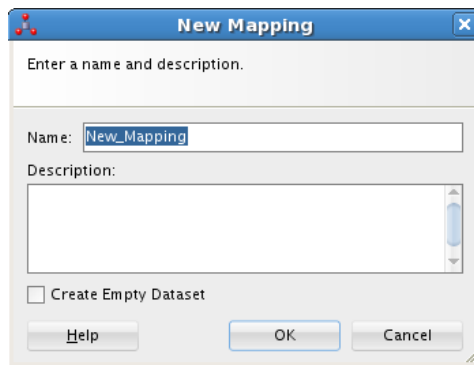


The Mapping Editor is displayed.

4. Enter the name of your mapping (Load TRG\_CUSTOMER) in the Name field as shown in Figure 4-2.



Figure 4-2 Mapping Editor



Ensure the 'Create Empty Dataset' option is not selected.

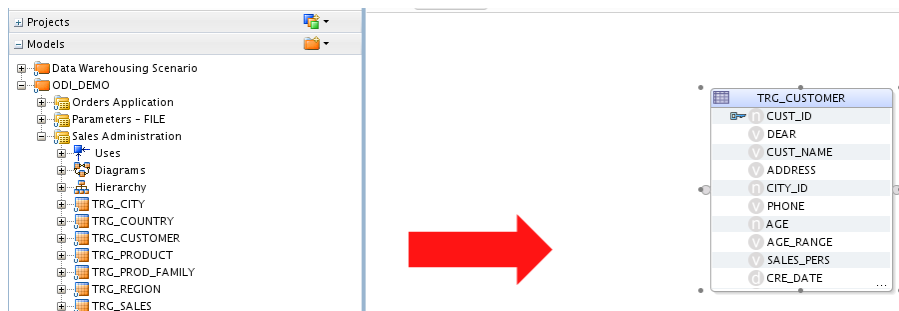
#### 4.1.3.2. Define the Target

The target is the element that will be loaded by the mapping.

**To insert the target in the Load TRG\_CUSTOMER mapping:**

1. Verify you are in the Logical tab of the Mapping Editor.
2. In the Designer Navigator, expand the Models accordion and the *Sales Administration* model.
3. Select TRG\_CUSTOMER datastore under the *Sales Administration* model and drag it into the mapping editor as shown in Figure 4-3.

Figure 5-3 Selecting the Target



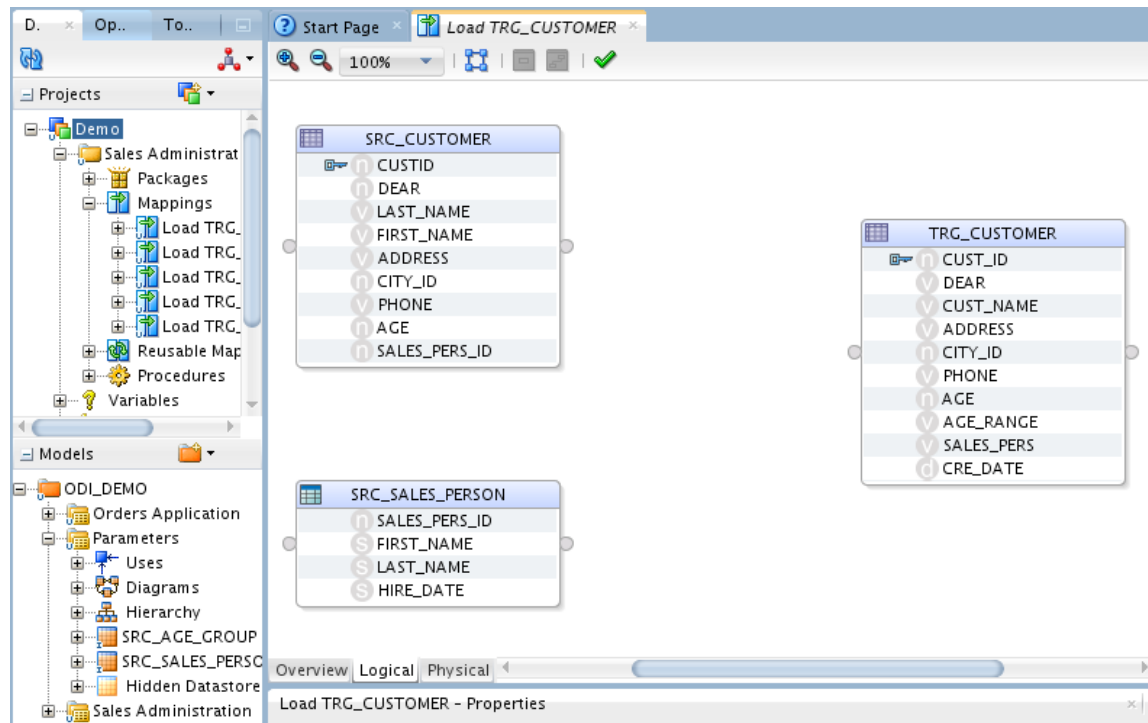
#### 4.1.3.3. Define the Source

The source datastores contain data used to load the target datastore. Two types of datastores can be used as a mapping source: datastores from the models and reusable mappings. This example uses datastores from the *Orders Application* and *Parameters* models.

**To add source datastores to the Load TRG\_CUSTOMER mapping:**

1. Under models, drag the following source datastores into the Source Diagram:
  - SRC\_CUSTOMER from the *Orders Application* model
  - SRC\_SALES\_PERSON from the *Parameters* model
2. The Mapping should look like shown in Figure 4–4.

**Figure 4–4 Adding Data Stores to a Mapping**



#### 4.1.3.4. Define the Lookup

This section describes how to create a lookup that defines that the customer's age must be between the minimum and maximum ages in the file.

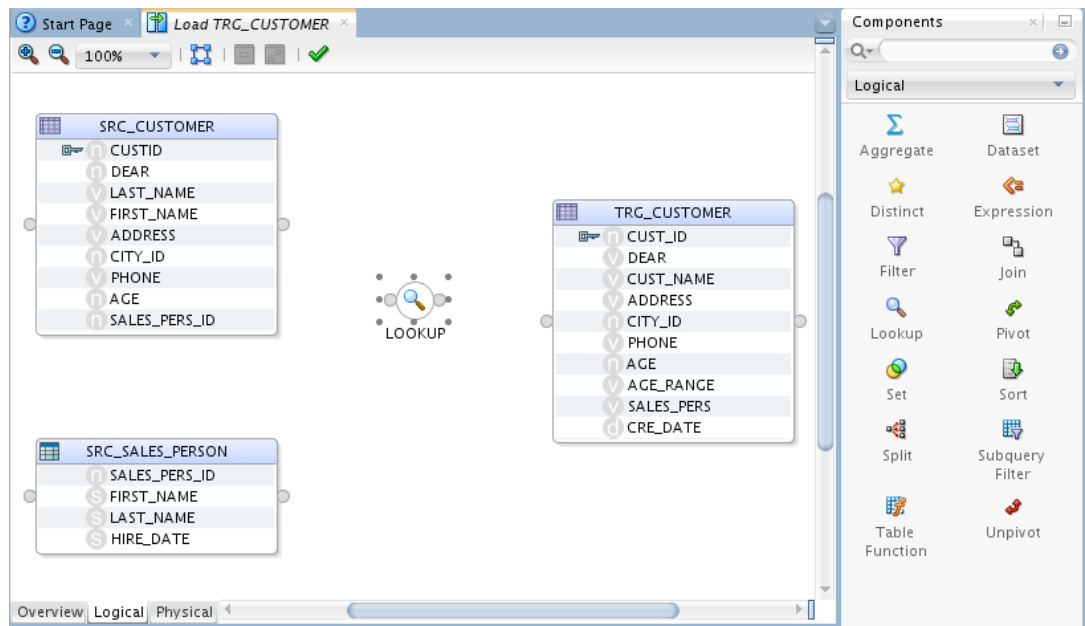
A lookup is a datastore (from a model or the target datastore of a map) - called the *lookup table* - associated to a source datastore - the *driving table* - via a join expression and from which data can be fetched and used in mappings.

Lookup tables are added with the Lookup Component.

**To create a lookup in the Load TRG\_CUSTOMER mapping:**

1. From the Components panel, drag **Lookup** into the mapping as shown in Figure 4–5 below.

Figure 4-5 Insert a Lookup

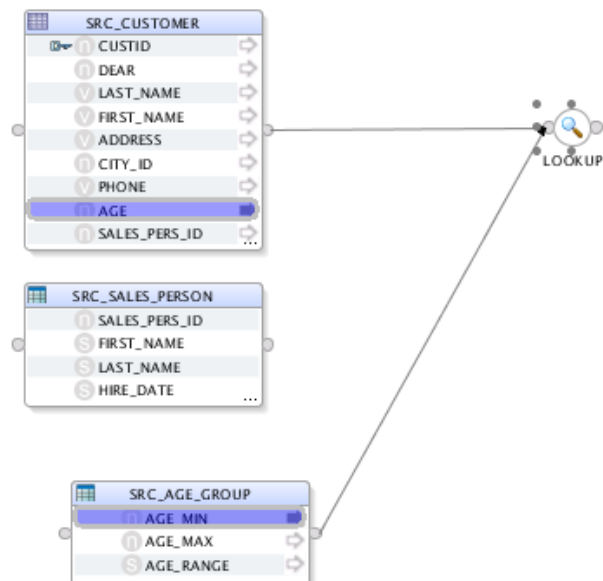


2. From the Parameters model, drag the SRC\_AGE\_GROUP datastore into the Mapping. The SRC\_AGE\_GROUP datastore will be used as a lookup table.

3. Drag the following source columns into the Lookup:

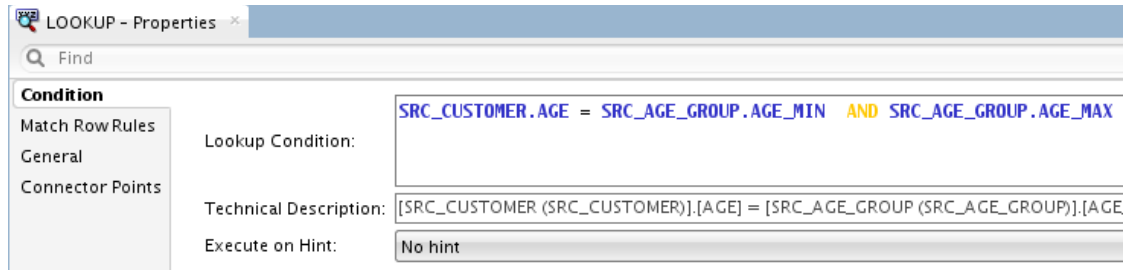
- Age from the SRC\_CUSTOMER source datastore
- AGE\_MIN from the SRC\_AGE\_GROUP datastore
- AGE\_MAX from the SRC\_AGE\_GROUP datastore

Figure 4-6 Select the lookup sources



4. Select the LOOKUP, click **Condition** in the LOOKUP – Properties as in Figure 4-7.

**Figure 4-7 Lookup Condition**

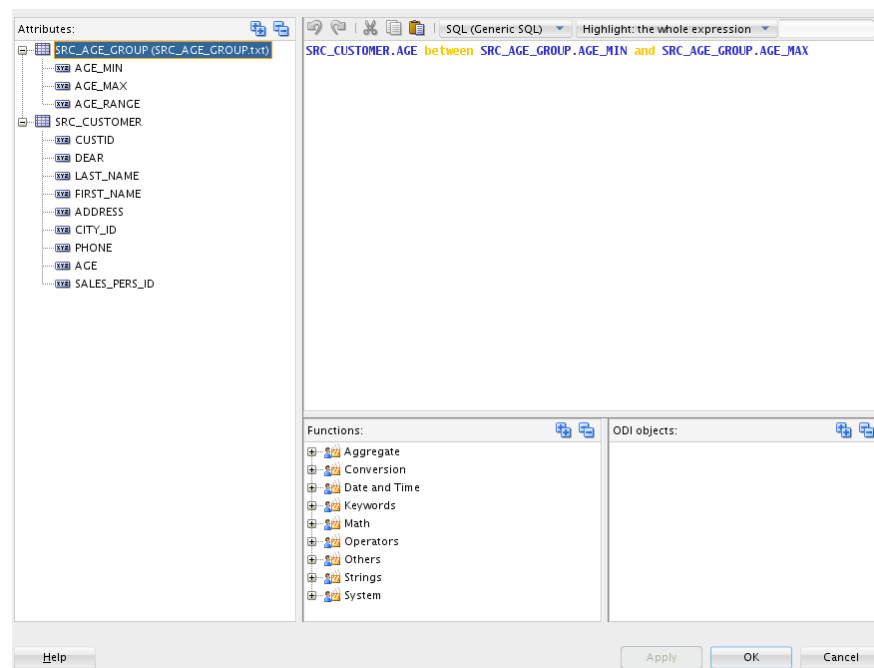


5. Modify the Condition by replacing the '=' with the string 'between'. You should have the following in the Lookup Condition

`SRC_CUSTOMER.AGE between SRC_AGE_GROUP.AGE_MIN and SRC_AGE_GROUP.AGE_MAX`

This corresponds to a join between the SRC\_CUSTOMER and the SRC\_AGE\_GROUP datastore and defines that the customer's age must between the minimum and maximum ages in the file.

**Figure 4-8 Expression Editor with modified lookup condition**



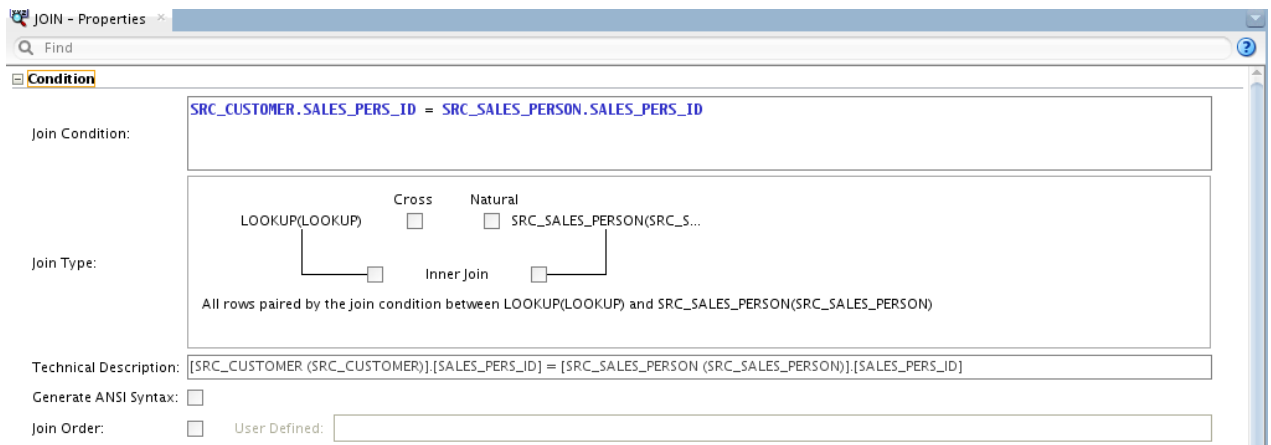
6. Click **Save**.

#### 4.1.3.5. Define the Join between the Source Datastores

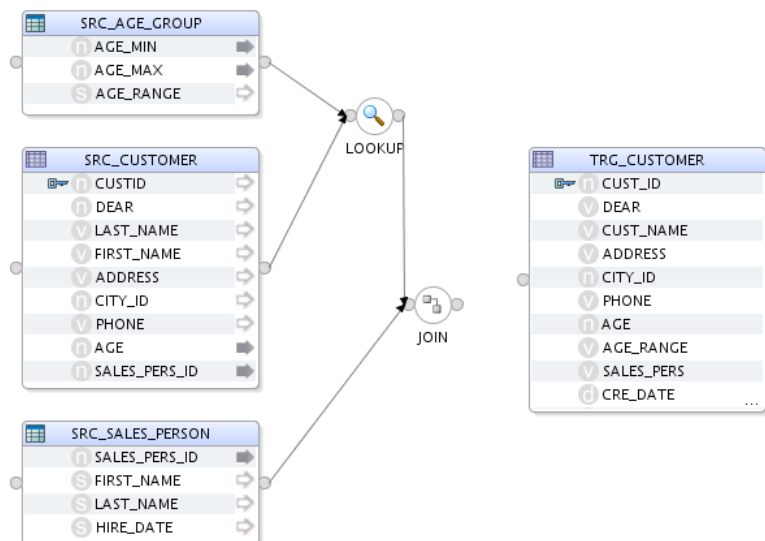
This section describes how to define a join between the source datastores. To create the join defined in Table 4–9:

1. Drag the JOIN component into the mapping.
2. In the mapping, drag the SALES\_PERS\_ID column from the SRC\_CUSTOMER datastore into the JOIN.
3. In the mapping, drag the SALES\_PERS\_ID column from the SRC\_SALES\_PERSON datastore into the join.

*Figure 4-9 JOIN Properties showing the Join Condition and Execute*



*Figure 4–10 Source Diagram of the Load TRG\_CUSTOMER Mapping with a Lookup and a Join*



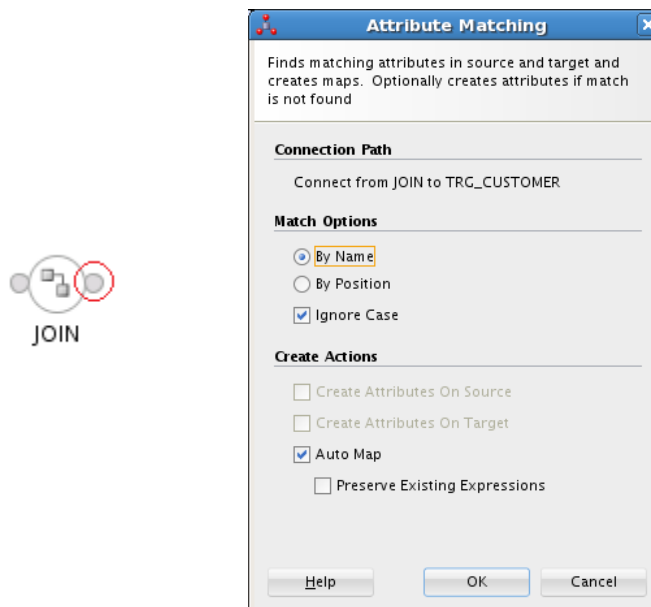
#### 4.1.3.6. Define the Target Expressions

The following columns are mapped in this section: CUST\_ID, DEAR, CUST\_NAME, AGE\_RANGE, SALES\_PERS, CRE\_DATE and UPD\_DATE.

To Auto Map from the sources to the target, the connector points need to be dragged and dropped between components.

1. From the JOIN component, drag the connector point, holding and dragging to the target input connector point. An Attribute matching dialog is displayed, keep the defaults and click OK.

Figure 4–11 Attribute Matching



The transformation rules, defined as *expressions*, are listed on the target column.

Following are the steps to complete the custom mappings.

Click on the TRG\_CUSTOMER datastore in the mapping to display the properties.

Figure 4–12 TRG\_CUSTOMER Properties

TRG_CUSTOMER - Properties				
Find				
<b>Attributes</b>	Attributes:			
General	Name	Data type	Expression	Execute...
Target	CUST_ID	NUMBER		No hint
Journalizing	DEAR	VARCH...	SRC_CUSTOMER.DEAR	No hint
Constraints	CUST_...	VARCH...		No hint
Connector Points	ADDRESS	VARCH...	SRC_CUSTOMER.ADDRESS	No hint
	CITY_ID	NUMBER	SRC_CUSTOMER.CITY_ID	No hint
	PHONE	VARCH...	SRC_CUSTOMER.PHONE	No hint
	AGE	NUMBER	SRC_CUSTOMER.AGE	No hint
	AGE_R...	VARCH...	SRC_AGE_GROUP.AGE_RANGE	No hint
	SALES_...	VARCH...		No hint
	CRE_D...	DATE		No hint

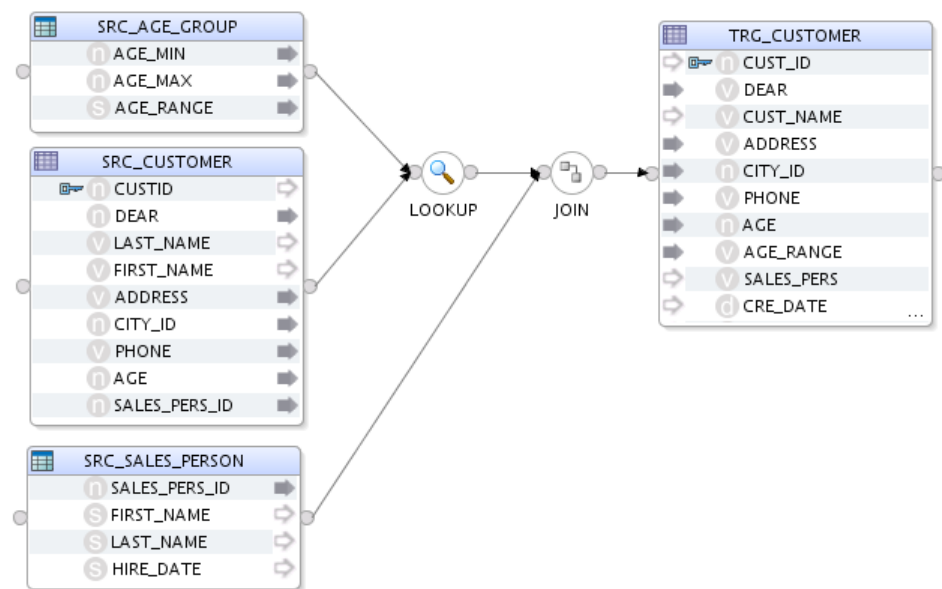
### CUST\_ID Mapping Expression

The CUST\_ID mapping expression maps the SRC\_CUSTOMER.CUSTID source column to the TRG\_CUSTOMER.CUST\_ID target column. Note that these 2 columns have not been automatically mapped, since their names are slightly different.

To define the expression for the CUST\_ID target column:

1. In the SRC\_CUSTOMER data source, select the CUSTID column.
2. Drag it into the CUST\_ID field in the Target Datastore as shown in Figure 4–13.

Figure 4–13 CUST\_ID Mapping Expression



3. Select the mapped field, CUST\_ID in the Target Datastore to display its properties in the Property Inspector.

### DEAR Mapping Expression

This transformation rule maps the source datastore's DEAR column (numeric) as a string expression (0 -->'MR', 1 -->'MRS', 2 -->'MS').

To define the expression for the DEAR target column:

1. In the Target Datastore, select the DEAR target column to display the mapping properties in the Property Inspector.
2. In the Expression field, enter the following mapping expression:

```
CASE
WHEN SRC_CUSTOMER.DEAR = 0 THEN 'Mr'
WHEN SRC_CUSTOMER.DEAR = 1 THEN 'Mrs'
ELSE 'Ms'
END
```

**Tip:** You can drag source columns, for example the SRC\_CUSTOMER.DEAR column, into the Expression field. You can also use the Expression Editor.

### CUST\_NAME Mapping Expression

This transformation rule maps the concatenated value of the first name and uppercase last name of each customer.

To define the expression for the CUST\_NAME target column:

1. In the Target Datastore, select CUST\_NAME to display the expression properties in the Property Inspector.
2. In the Expression field, enter the following mapping expression:

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

**Tip:** Use the Expression Editor to create this rule. By using the Expression Editor, you can avoid most common syntax errors.

### AGE\_RANGE Mapping Expression

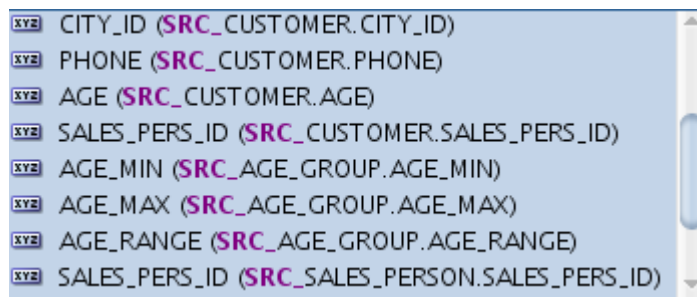
This mapping expression maps the SRC\_AGE\_GROUP.AGE\_RANGE to the TRG\_CUSTOMER.AGE\_RANGE and is already defined.

1. In the Target Datastore, select AGE\_RANGE to display the mapping properties in the Property Inspector.
2. In the Expression field, the following mapping expression should appear:

```
SRC_AGE_GROUP.AGE_RANGE
```

**Tip:** Auto-completion functions are available in ODI Studio. In the Expression, type SRC\_ and then press <CTRL-SPACE>, a pop-up window displays available fields as shown in Figure 4-15.

*Figure 4-15 Auto-completion*



You can also drag and drop the AGE\_RANGE column from SRC\_AGE\_GROUP into AGE\_RANGE in TRG\_CUSTOMER.

### SALES\_PERS Mapping Expression

This will map the concatenated value of the first name and uppercase last name of each salesperson.

To define the mapping expression for the SALES\_PERS target column:



1. In the Target Datastore, select SALES\_PERS to display the expression properties in the Property Inspector.

2. In the Expression field, enter the following mapping expression:

```
SRC_SALES_PERSON.FIRST_NAME || ' ' ||  
UPPER (SRC_SALES_PERSON.LAST_NAME)
```

### CRE\_DATE Mapping Expression

To define the mapping expression for the CRE\_DATE target column:

1. In the Target Datastore, select CRE\_DATE to display the mapping properties in the Property Inspector.

2. In the Expression field, enter the following mapping expression: SYSDATE

3. Verify that **Active** is selected.

4. Unselect **Update**. The mapping will be performed only on Insert.

5. The Property Inspector of the CRE\_DATE attribute appears as shown in Figure 4–16.

*Figure 4–16 Property Inspector of the CRE\_DATE Mapping*

CRE\_DATE - Properties

Find

**Target**

General

Format

Expression: SYSDATE

Execute on Hint: Target

Fixed Execution Location:

Technical Description:

Active: ☒

Key: ☐

Update: ☐

UD 1: ☐

Insert: ☒

Check Not Null (Flow control only): ☐

UD 2: ☐

### UPD\_DATE Mapping Expression

To define the mapping expression for the UPD\_DATE target column:

1. In the Target Datastore, select UPD\_DATE to display the attribute properties in the Property Inspector.

2. In the Expression field, enter the following mapping expression: SYSDATE

3. Verify that **Active Mapping** is selected.

4. Unselect **Insert**. The mapping expression will be performed only on Update.

### Notes on the Expression Editor

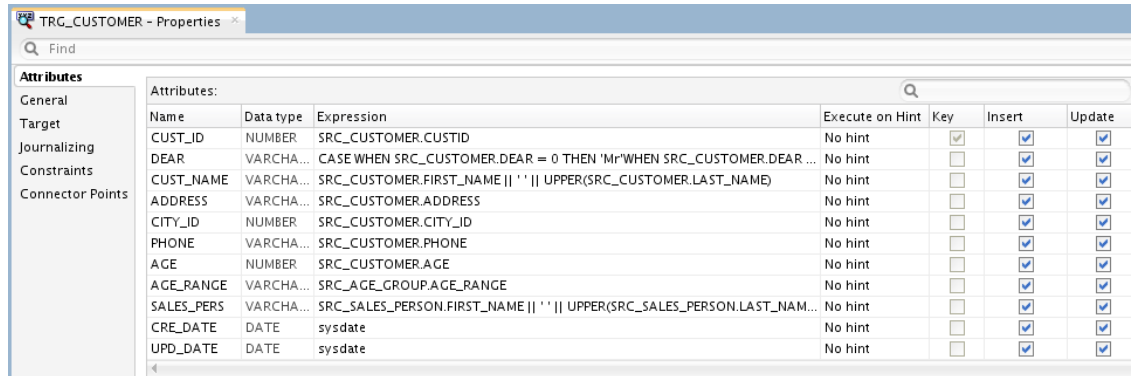
- The Expression Editor that is used to build the Expressions does not contain all the functions specific to a technology. It contains only functions that are common to a large number of technologies. The fact that a function does not appear in the Expression Editor does not prevent it from being entered manually and used in an Expression.

- If you were to execute this mapping on the target using the Execute on Hint field, the Expression Editor would give you the syntax for your target system.

### The Target Datastore Panel

Your transformation rules appear in the Target Datastore Attributes panel as shown in Figure 4–17.

*Figure 4–17 Target Datastore Mappings*



Name	Data type	Expression	Execute on Hint	Key	Insert	Update
CUST_ID	NUMBER	SRC_CUSTOMER.CUSTID	No hint	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
DEAR	VARCHA...	CASE WHEN SRC_CUSTOMER.DEAR = 0 THEN 'Mr' WHEN SRC_CUSTOMER.DEAR ...	No hint	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
CUST_NAME	VARCHA...	SRC_CUSTOMER.FIRST_NAME    ' '    UPPER(SRC_CUSTOMER.LAST_NAME)	No hint	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
ADDRESS	VARCHA...	SRC_CUSTOMER.ADDRESS	No hint	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
CITY_ID	NUMBER	SRC_CUSTOMER.CITY_ID	No hint	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
PHONE	VARCHA...	SRC_CUSTOMER.PHONE	No hint	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
AGE	NUMBER	SRC_CUSTOMER.AGE	No hint	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
AGE_RANGE	VARCHA...	SRC_AGE_GROUP.AGE_RANGE	No hint	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
SALES_PERS	VARCHA...	SRC_SALES_PERSON.FIRST_NAME    ' '    UPPER(SRC_SALES_PERSON.LAST_NAM...	No hint	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
CRE_DATE	DATE	sysdate	No hint	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
UPD_DATE	DATE	sysdate	No hint	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

### Set the Integration Type

Finally click on TRG\_CUSTOMER datastore in the Mapping and in the Properties panel under Target set the Integration Type to Incremental Update.

#### 4.1.3.7. Define the Data Loading Strategies (LKM)

The data loading strategies are defined in the Physical tab of the Mapping Editor. Oracle Data Integrator automatically computes the flow depending on the configuration in the mapping's diagram. It proposes default KMs for the data flow. The Physical tab enables you to view the data flow and select the KMs used to load and integrate data.

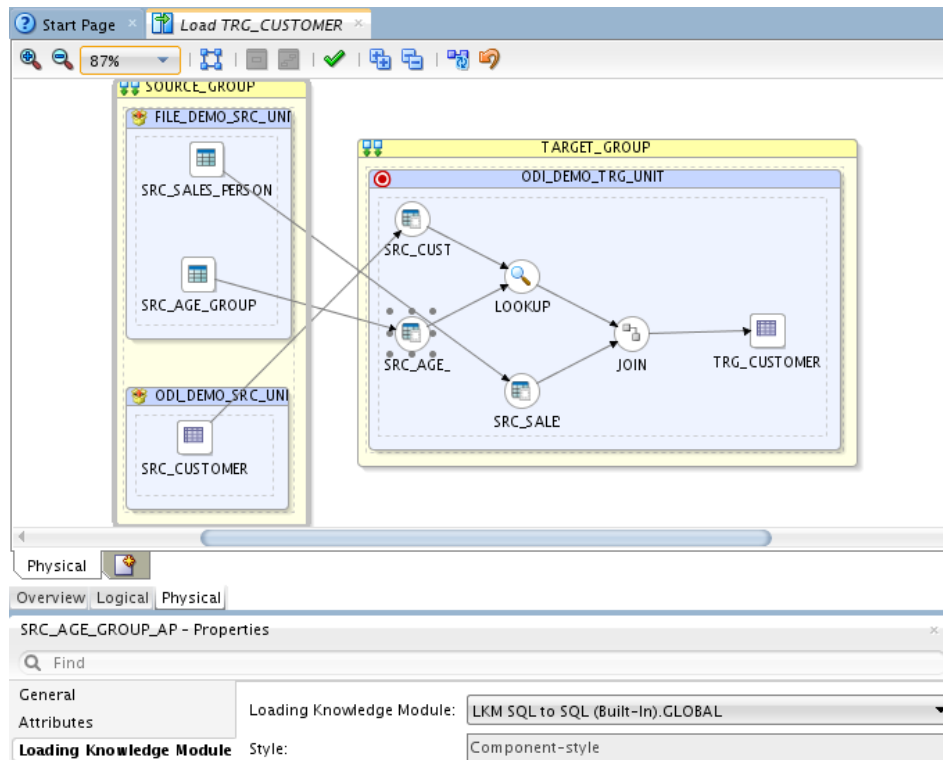
Loading Knowledge Modules (LKM) are used for loading strategies and Integration Knowledge Modules (IKM) are used for integration strategies.

You have to define the way to retrieve the data from the SRC\_AGE\_GROUP, SRC\_SALES\_PERSON files and from the SRC\_CUSTOMER table in your source environment.

To define the loading strategies:

1. In the Physical tab of the Mapping Editor, select the access point that corresponds to the loading of the SRC\_AGE\_GROUP, SRC\_SALES\_PERSON files. In this example, this is the SRC\_AGE\_GROUP\_AP and SRC\_SALES\_PERSON\_AP. The Property Inspector should display the properties of the access points.
2. In the Property Inspector, verify that the **LKM SQL to SQL (Built-In)** is selected in the Loading Knowledge Module Selector list as shown in Figure 4–18.

Figure 4–18 Physical tab of the Load TRG\_CUSTOMER Mapping Editor



#### 4.1.3.8. Define the Data Integration Strategies (IKM)

After defining the loading phase, you need to define the strategy to adopt for the integration of the data into the target table.

To define the integration strategies:

1. In the Physical tab of the Mapping Editor, select TRG\_CUSTOMER in the TARGET\_GROUP object. The Property Inspector will display the properties of the target.
2. In the Property Inspector, set the IKM to **IKM Oracle Incremental Update** in the *Integration Knowledge Module* Selector list. If this IKM is not in the list, make sure you have correctly set the Target Integration Type to Incremental Update in the Logical panel.
3. In the knowledge module options, leave the default values. The Property Inspector appears as shown in Figure 4–19.

Figure 4–19 Property Inspector for Target Area of Load TRG\_CUSTOMER

**Integration Knowledge Module**

Integration Knowledge Module: IKM Oracle Incremental Update

Style: Template-style

Options Description

Options:

Name	Value	Use Default
INSERT	True	<input checked="" type="checkbox"/>
UPDATE	True	<input checked="" type="checkbox"/>
COMMIT	True	<input checked="" type="checkbox"/>
SYNC_JRN_DELETE	True	<input checked="" type="checkbox"/>
FLOW_CONTROL	True	<input checked="" type="checkbox"/>
RECYCLE_ERRORS	False	<input checked="" type="checkbox"/>
STATIC_CONTROL	False	<input checked="" type="checkbox"/>
TRUNCATE	False	<input checked="" type="checkbox"/>
DELETE_ALL	False	<input checked="" type="checkbox"/>
CREATE_TARG_TABLE	False	<input checked="" type="checkbox"/>
DELETE_TEMPORARY_OBJECTS	True	<input checked="" type="checkbox"/>

**Note:** Only the built-in Knowledge Modules or the ones you imported to your Project appear in the KM Selector lists. The demonstration environment already includes the Knowledge Modules required for the getting started examples. You do not need to import KMs into the demonstration Project.

For more information on importing KMs into your Projects, see "Importing a KM" in the *Oracle Fusion Middleware Developer's Guide for Oracle Data Integrator*.

#### 4.1.3.9. Define the Data Control Strategy

In Section 5.1.3.7, "Define the Data Loading Strategies (LKM)" and Section 5.1.3.8, "Define the Data Integration Strategies (IKM)" you have specified the data flow from the source to the target. You must now define how to check your data (CKM) and the constraints and rules that must be satisfied before integrating the data.

To define the data control strategy:

1. In the Mapping Physical tab under the TRG\_CUSTOMER Properties, select Check Knowledge Module, verify that the **CKM Oracle** is selected for Check Knowledge Module.
2. In the Logical view, select the target datastore TRG\_CUSTOMER and verify the Constraints panel. Set the constraints that you wish to verify to `true`.

- PK\_TRG\_CUSTOMER
- FK\_CUST\_CITY

The Constraints tab appears as shown in Figure 4–20

*Figure 4–20 Constraints of TRG\_CUSTOMER*

Constraints:		
Constraint Name	Constraint Type	Value
PK_TRG_CUSTOMER	Primary Key	<default>: true
FK_CUST_CITY	Reference	<default>: true

3. From **File** main menu, select **Save**.

The Load TRG\_CUSTOMER mapping is now ready to be run.

## 4.2 Load TRG\_SALES Mapping Example

This section contains the following topics:

- Purpose and Integration Requirements
- Mapping Definition
- Creating the Mapping

### 4.2.1 Purpose and Integration Requirements

This section describes the integration features and requirements the mapping Load TRG\_SALES is expected to meet.

The purpose of this mapping is to load the SRC\_ORDERS table of orders and the SRC\_ORDER\_LINES table of order lines from the *Orders Application* model into the TRG\_SALES target table in the *Sales Administration* model. The data must be aggregated before it is integrated into the target table. Only orders whose status is CLO are to be used.

However, the source data is not always consistent with the integrity rules present in the target environment. For this transformation, we want to cleanse the data by verifying that all of the constraints are satisfied. We want to place any invalid rows into an error table rather than into our target database. In our case, two important integrity rules must be satisfied:

- The sales must be associated with a product (PRODUCT\_ID) that exists in the TRG\_PRODUCT table (reference FK\_SALES\_PROD)
- The sales must be associated with a customer (CUST\_ID) that exists in the TRG\_CUSTOMER table (reference FK\_SALES\_CUST)

The functional details for these rules and the procedure to follow are given in Section 5.2.3, "Creating the Mapping".

## 4.2.2 Mapping Definition

This section describes the mapping Load TRG\_SALES that will be created in this example. See Section 4.2.3, "Creating the Mapping" for more information.

The Load TRG\_SALES mapping uses the following data and transformations:

- One target datastore. Table 4–7 lists the details of the target datastore.

*Table 4–7 Target Datastore Details of Load TRG\_SALES*

Model	Datastore	Description	Type
Sales Administration	TRG_SALES	Target table in the Sales Administration System	Oracle table

- Two source datastores. Table 4–8 lists the details of the source datastores.

*Table 4–8 Source Datastore Details of Load TRG\_SALES*

Model	Datastore	Description	Type
Orders Application	SRC_ORDERS	Orders table in the source systems	Oracle table
Orders Application	SRC_ORDER_LINES	Order lines table in the source system	

- One **Join**. Table 4–9 lists the details of the join.

*Table 4–9 Joins used in Load TRG\_SALES*

Join	Description	SQL Rule
Commands and Order lines	Join SRC_ORDERS and SRC_ORDER_LINES	SRC_ORDERS.ORDER_ID = SRC_ORDER_LINES.ORDER_ID

- One **Filter**. Table 4–10 lists the details of the filter.

*Table 4–10 Filters used in Load TRG\_SALES*

Description	SQL Rule
Only retrieve completed orders (CLOSED)	SRC_ORDERS.STATUS = 'CLO'
Orders Application	Order lines table in the source system

- Several transformation rules. Table 4–11 lists the details of the transformation rules.

**Table 4–11 Transformation Rules used in Load TRG\_SALES**

Target Column	Origin	SQL Rule(Expression)
CUST_ID	CUST_ID from SRC_ ORDERS	SRC_ORDERS.CUST_ ID
PRODUCT_ID	PRODUCT_ID from SRC_ORDER_LINES	SRC_ORDER_LINES.PRODUCT_ID
FIRST_ORD_ID	Smallest value of ORDER_ID	MIN(SRC_ ORDERS.ORDER_ID)
FIRST_ORD_DATE	Smallest value of the ORDER_DATE from SRC_ORDERS	MIN(SRC_ ORDERS.ORDER_ DATE)
LAST_ORD_ID	Largest value of ORDER_ID	MAX(SRC_ ORDERS.ORDER_ID)
LAST_ORD_DATE	Largest value of the ORDER_DATE from SRC_ORDERS	MAX(SRC_ ORDERS.ORDER_ DATE)
QTY	Sum of the QTY quantities from the order lines	SUM(SRC_ORDER_ LINES.QTY)
AMOUNT	Sum of the amounts from the order lines	SUM(SRC_ORDER_ LINES.AMOUNT)
PROD_AVG_PRICE	Average amount from the order lines	AVG(SRC_ORDER_ LINES.AMOUNT)

### 4.2.3 Creating the Mapping

This section describes how to create the Load TRG\_SALES mapping. To create the Load TRG\_SALES mapping perform the following procedure:

1. Insert a Mapping
2. Define the Target Datastore
3. Define the Source Datastores
4. Define Joins between the Source Datastores
5. Define the Order Filter
6. Define the Transformation Rules
7. Define the Data Loading Strategies (LKM)
8. Define the Data Integration Strategies (IKM)
9. Define the Data Control Strategy

#### 4.2.3.1. Insert a New Mapping

To create a new mapping:

1. In Designer Navigator, expand the Demo project node in the Projects accordion.
2. Expand the Sales Administration node.
3. In the Sales Administration folder, right-click the Mappings node and select **New Mapping**.
4. Enter the name of your mapping (Load TRG\_SALES) in the Name field. Create Empty Dataset should be unchecked.

### 4.2.3.2. Define the Target Datastore

To insert the target datastore in the Load TRG\_SALES mapping:

1. Go to the Logical tab of the Mapping Editor.
2. In the Designer Navigator, expand the Models accordion and the *Sales Administration* model.
3. Select the TRG\_SALES datastore under the *Sales Administration model* and drag it into the mapping.

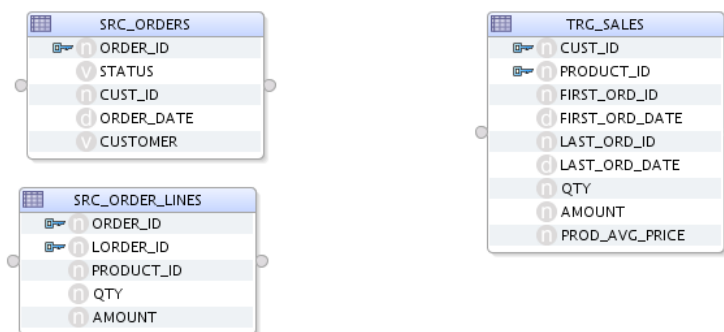
### 4.2.3.3. Define the Source Datastores

The Load TRG\_SALES mapping example uses datastores from the *Orders Application* model.

To add source datastores to the Load TRG\_SALES mapping:

1. In the Mapping tab, drag the following source datastores into the Source Diagram:
  - SRC\_ORDERS from the *Orders Application* model
  - SRC\_ORDER\_LINES from the *Orders Application* model

*Figure 4-21 Load TRG\_SALES Mapping*



### 4.2.3.4. Define the Order Filter

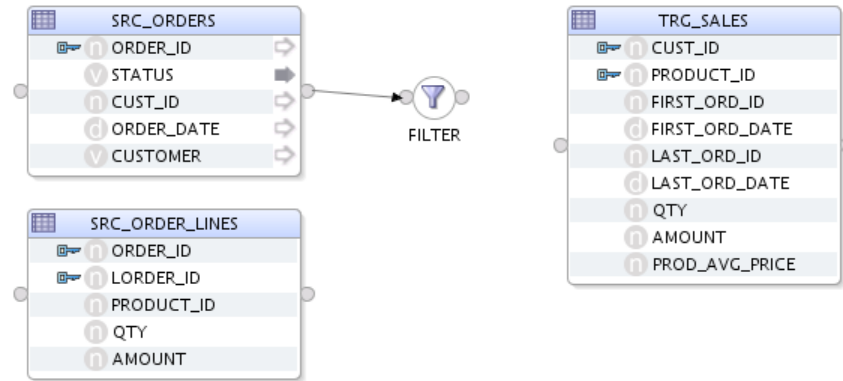
In this example, only completed orders should be retrieved. A filter needs to be defined on the SRC\_ORDERS datastore.

**To define the filter:**

1. In the mapping, select the STATUS column of the SRC\_ORDERS datastore and drag it onto the Mapping Diagram.
2. The filter appears as shown in Figure 4-22.



*Figure 4-22 Filter on SRC\_ORDERS*



3. Select the filter in the Source Diagram to display the filter properties in the Property Inspector.
4. In the Condition tab of the Property Inspector, modify the filter rule by typing:  
`SRC_ORDERS.STATUS = 'CLO'`

#### 4.2.3.5. Define Joins between the Source Datastores

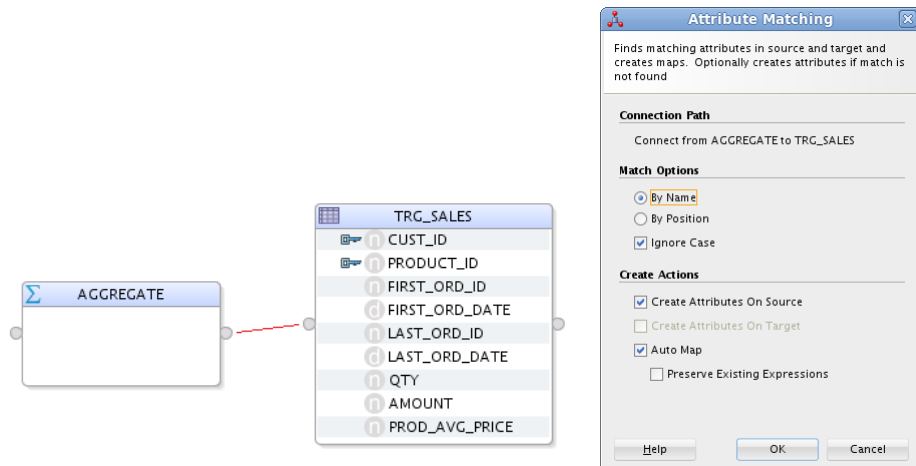
This section describes how to define joins between the source datastores. To create the join defined in Table 4-9:

1. Drag the JOIN component into the mapping from the Components palette
  2. Drag the ORDER\_ID column of the SRC\_ORDERS datastore into the JOIN.
  3. Drag the ORDER\_ID column of the SRC\_ORDER\_LINES datastore into the JOIN.
- A join linking the two datastores appears. This is the join on the order number. The join has the following expression:  
`SRC_ORDERS.ORDER_ID=SRC_ORDER_LINES.ORDER_ID`

#### 4.2.3.6. Define the Transformation Rules

Many of the transformations used for this mapping will use an aggregate function. These functions are implemented using the AGGREGATE Component.

1. From the Components palette, drag the AGGREGATE component into the mapping.
2. Drag the AGGREGATE output connector point to the TRG\_SALES input connector point. This action will start an Automap, selecting OK will backfill the AGGREGATE from the Target attributes.



3. Define the following transformations rules.

Define the following transformation rules in the Aggregate component:

- **CUST\_ID:** Drag the SRC\_ORDERS.CUST\_ID column into the CUST\_ID column in the Aggregate Component. This transformation rule maps the CUST\_ID column in your SRC\_ORDERS table to the CUST\_ID column in your target table.
- **PRODUCT\_ID:** Drag the SRC\_ORDER\_LINES.PRODUCT\_ID column into the PRODUCT\_ID column in the Aggregate Component. This transformation rule maps the PRODUCT\_ID column in your SRC\_ORDER\_LINES table to the PRODUCT\_ID column in your target table.
- **FIRST\_ORD\_ID:** Drag the SRC\_ORDERS.ORDER\_ID column into the Expression field. Enter the following text in the Expression field:

```
MIN (SRC_ORDERS . ORDER_ID)
```

This transformation rule maps the minimum value of the ORDER\_ID column in your SRC\_ORDERS table to the FIRST\_ORD\_ID column in your target table.

- **FIRST\_ORD\_DATE:** Drag the SRC\_ORDERS.ORDER\_DATE column into the Implementation field. Enter the following text in the Expression field:

```
MIN (SRC_ORDERS . ORDER_DATE)
```

This transformation rule maps the minimum value of the ORDER\_DATE column in your SRC\_ORDERS table to the FIRST\_ORD\_DATE column in your target table.

- **LAST\_ORD\_ID:** Drag-and-drop the SRC\_ORDERS.ORDER\_ID column into the Expression field. Enter the following text in the Expression field:

```
MAX (SRC_ORDERS . ORDER_ID)
```

This transformation rule maps the maximum value of the ORDER\_ID column in your SRC\_ORDERS table to the LAST\_ORD\_ID column in your target table.

- **LAST\_ORD\_DATE:** Drag the SRC\_ORDERS.ORDER\_DATE column into the Expression field. Enter the following text in the Expression field:

```
MAX (SRC_ORDERS.ORDER_DATE)
```

This transformation rule maps the maximum value of the ORDER\_DATE column in your SRC\_ORDERS table to the LAST\_ORD\_DATE column in your target table.

- **QTY:** Enter the following text in the Expression field:

```
SUM (SRC_ORDER_LINES.QTY)
```

This transformation rule maps the sum of the product quantities to the QTY column in your target table.

- **AMOUNT:** Enter the following text in the Expression field:

```
SUM (SRC_ORDER_LINES.AMOUNT)
```

This transformation rule maps the sum of the product prices to the AMOUNT column in your target table.

- **PROD\_AVG\_PRICE:** Drag the SRC\_ORDERLINES.AMOUNT column into the Expression field. Enter the following text in the Expression field:

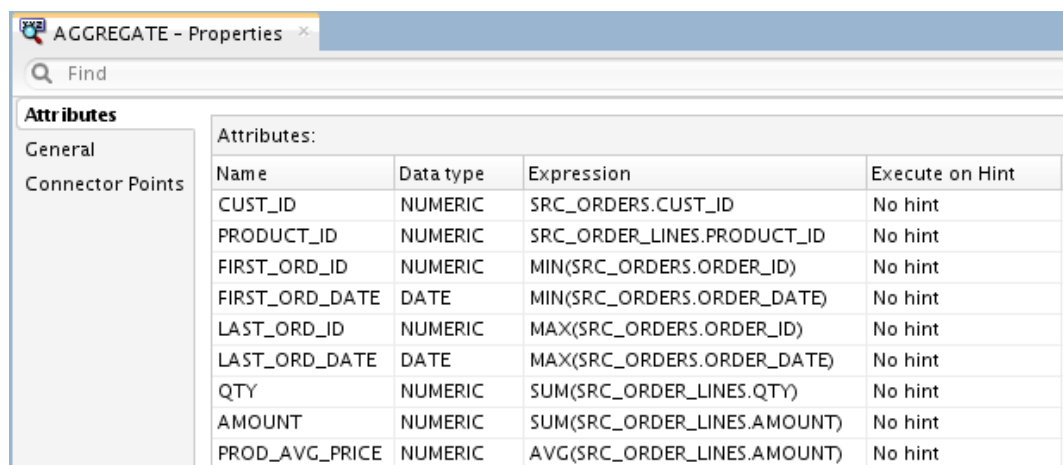
```
AVG (SRC_ORDER_LINES.AMOUNT)
```

This transformation rule maps the average of the product prices to the PROD\_AVG\_PRICE column in your target table.

Review carefully your Aggregate rules and make sure that you have defined the rules as shown in Figure 4–23 below.

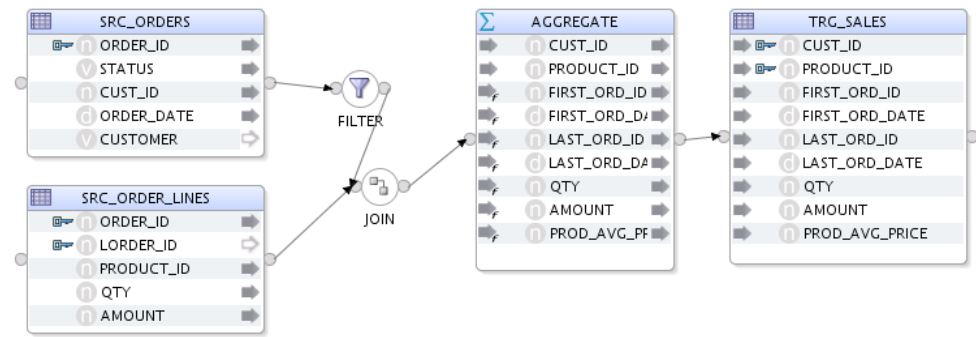
Note that even though this example uses aggregation functions, you do not have to specify the group by rules: Oracle Data Integrator will infer that from the mappings, applying SQL standard coding practices.

*Figure 4–23 Aggregate Properties*



AGGREGATE - Properties				
Find				
Attributes				
General				
Connector Points				
Attributes:				
Name	Data type	Expression	Execute on Hint	
CUST_ID	NUMERIC	SRC_ORDERS.CUST_ID	No hint	
PRODUCT_ID	NUMERIC	SRC_ORDER_LINES.PRODUCT_ID	No hint	
FIRST_ORD_ID	NUMERIC	MIN(SRC_ORDERS.ORDER_ID)	No hint	
FIRST_ORD_DATE	DATE	MIN(SRC_ORDERS.ORDER_DATE)	No hint	
LAST_ORD_ID	NUMERIC	MAX(SRC_ORDERS.ORDER_ID)	No hint	
LAST_ORD_DATE	DATE	MAX(SRC_ORDERS.ORDER_DATE)	No hint	
QTY	NUMERIC	SUM(SRC_ORDER_LINES.QTY)	No hint	
AMOUNT	NUMERIC	SUM(SRC_ORDER_LINES.AMOUNT)	No hint	
PROD_AVG_PRICE	NUMERIC	AVG(SRC_ORDER_LINES.AMOUNT)	No hint	

Figure 4-24 Mapping logical view



#### Setting the Integration Type:

Click on the TRG\_SALES datastore in the mapping, in the Properties panel under Target set the Integration Type to Incremental Update.

#### 4.2.3.7. Define the Data Loading Strategies (LKM)

In the Physical tab, Oracle Data Integrator indicates the various steps that are performed when the map is executed.

In the Physical tab you define how to load the result of the orders and order line aggregates into your target environment with a Loading Knowledge Module (LKM).

To define the loading strategies:

1. In the Physical tab of the Mapping Editor, select the source set that corresponds to the loading of the order line's filtered aggregate results. In this example, this is the AGGREGATE\_AP access point in the ODI\_DEMO\_TRG\_UNIT.
2. In the Property Inspector, set the LKM to **LKM SQL to SQL (Built-In).GLOBAL** using the LKM Selector list as shown in Figure 4-26.

Figure 4-25 Physical tab of Load TRG\_SALES Mapping

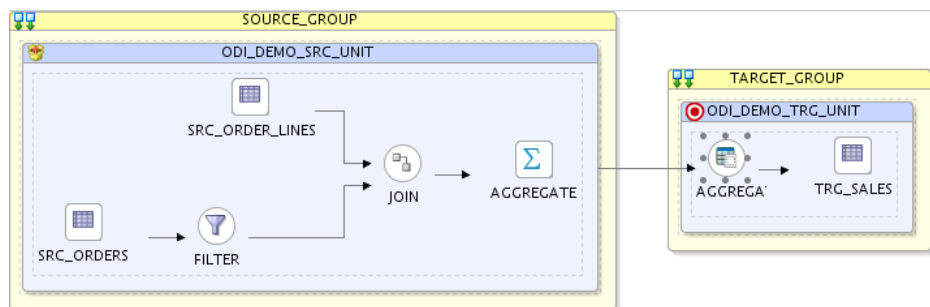


Figure 4-26 AGGREGATE\_AP Properties, Loading Knowledge Module Selection

AGGREGATE\_AP - Properties

Find

**General**

**Attributes**

**Loading Knowledge Module**

Loading Knowledge Module: LKM SQL to SQL (Built-In).GLOBAL

Style: Component-style

#### 4.2.3.8. Define the Data Integration Strategies (IKM)

After defining the loading phase, you need to define the strategy to adopt for the integration of the data into the target table.

To define the integration strategies:

1. In the Physical tab of the Mapping Editor, select the Target object (TRG\_SALES). The Property Inspector should display the properties of the target.
2. In the Property Inspector, set the IKM to **IKM Oracle Incremental Update** using the IKM Selector list. If this IKM is not in the list, make sure you have correctly set the Target Integration Type to Incremental Update in the Logical panel.
3. In the knowledge module options, leave the default values.

#### 4.2.3.9. Define the Data Control Strategy

In Section 4.2.3.7, "Define the Data Loading Strategies (LKM)" and Section 4.2.3.8, "Define the Data Integration Strategies (IKM)" you have specified the data flow from the source to the target. You must now define how to check your data (CKM) and the constraints and rules that must be satisfied before integrating the data.

To define the data control strategy:

1. In the Physical tab of the Mapping Editor for the Target, verify that the **CKM Oracle** is selected.

**Figure 4-27 Load TRG\_SALES Mapping**

Pop.TRG\_SALES\_DS

Overview Logical Physical

TRG\_SALES - Properties

Find

**General**

**Attributes**

**Integration Knowledge Module**

**Check Knowledge Module**

Check Knowledge Module: CKM Oracle

Options Description

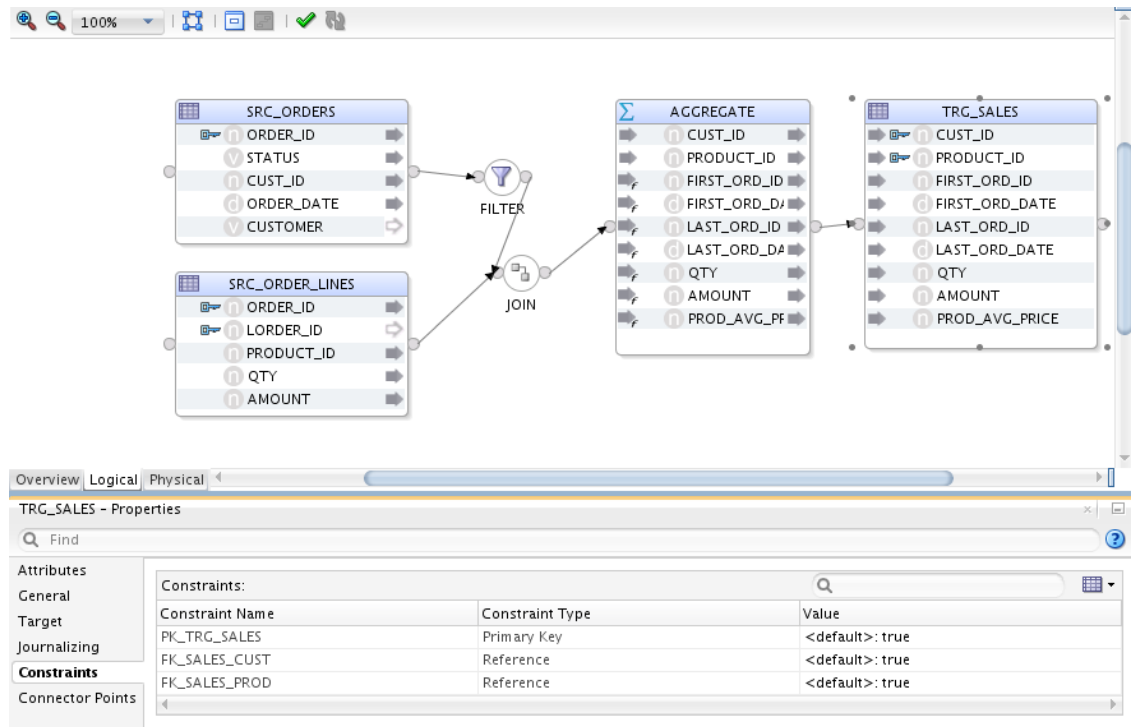
Options:

Name	Value	Use Default
DROP_ERROR_TABLE	False	<input checked="" type="checkbox"/>
DROP_CHECK_TABLE	False	<input checked="" type="checkbox"/>
CREATE_ERROR_INDEX	True	<input checked="" type="checkbox"/>
COMPATIBLE	9	<input checked="" type="checkbox"/>
VALIDATE	False	<input checked="" type="checkbox"/>

2. In the Logical tab of TRG\_SALES, select Constraints. Set the constraints that you wish to verify to true:

- PK\_TRG\_SALES
- FK\_SALES\_CUST
- FK\_SALES\_PROD

*Figure 4-28 Constraint Definition for TRG\_SALES*



3. From File main menu, select Save.

The Load TRG\_SALES mapping is now ready to be executed.

---

## 5 Implementing Data Quality Control

This chapter describes how to implement data quality control. An introduction to data integrity control is provided.

This chapter includes the following sections:

- Section 5.1, "Introduction to Data Integrity Control"
- Section 5.2, "SRC\_CUSTOMER Control Example"

### 5.1 Introduction to Data Integrity Control

Data integrity control is essential in ensuring the overall consistency of the data in your information systems applications.

Application data is not always valid for the constraints and declarative rules imposed by the information system. You may, for instance, find orders with no customer, or order lines with no product, and so forth.

Oracle Data Integrator provides a working environment to detect these constraint violations and to store them for recycling or reporting purposes.

There are two different types of controls: *Static Control* and *Flow Control*. We will examine the differences between the two.

#### Static Control

Static Control implies the existence of rules that are used to verify the integrity of your application data. Some of these rules (referred to as constraints) may already be implemented in your data servers (using primary keys, reference constraints, etc.)

With Oracle Data Integrator, you can enhance the quality of your data by defining and checking additional constraints, without declaring them directly in your servers. This procedure is called **Static Control** since it allows you to perform checks directly on existing - or static - data.

#### Flow Control

The information systems targeted by transformation and integration processes often implement their own declarative rules. The **Flow Control** function is used to verify an application's incoming data according to these constraints before loading the data into these targets. The flow control procedure is detailed in the "Mappings" chapter.

### Benefits

The main advantages of performing data integrity checks are the following:

- *Increased productivity* by using the target database for its entire life cycle. Business rule violations in the data slow down application programming throughout the target database's life-cycle. Cleaning the transferred data can therefore reduce application programming time.
- *Validation of the target database's model*. The rule violations detected do not always imply insufficient source data integrity. They may reveal a degree of incompleteness in the target model. Migrating the data before an application is rewritten makes it possible to validate a new data model while providing a test database in line with reality.
- *Improved quality of service* for the end-users. Ensuring data integrity is not always a simple task. Indeed, it requires that any data violating declarative rules must be isolated and recycled. This implies the development of complex programming, in particular when the target database incorporates a mechanism for verifying integrity constraints. In terms of operational constraints, it is most efficient to implement a method for correcting erroneous data (on the source, target, or recycled flows) and then to reuse this method throughout the enterprise.

## 5.2 SRC\_CUSTOMER Control Example

This example guides you through the data integrity audit process (Static Control).

The *Orders Application* contains data that does not satisfy business rule constraints on a number of different levels. The objective is to determine which data in this application does not satisfy the constraints imposed by the information system.

This section includes the following topics:

- Objective
- Interpreting the Problem
- Creating Constraints
- Run the Static Control
- Follow the Execution of the Control in Operator Navigator
- Interpreting the Results in Operator Navigator

### 5.2.1 Objective

Some data in our source may be inconsistent. There may be constraints in the target table that are not implemented in the source table or there may be supplementary rules that you wish to add. In our case we have two constraints that we want to enforce on the SRC\_CUSTOMER table:

- **Customers must be over 21 years of age.** However there could be some records corresponding to younger customers in the input table.
- **The CITY\_ID column must refer to an entry in the SRC\_CITY table.** However there could be some values that do not exist in the city table.

We want to determine which rows do not satisfy these two constraints and automatically copy the corresponding invalid records into an error table for analysis.



## 5.2.2 Interpreting the Problem

Enforcing these types of rules requires the use of a *check constraint* (also referred to as a *condition*), as well as a *reference constraint* between the SRC\_CITY and SRC\_CUSTOMER tables.

## 5.2.3 Creating Constraints

This section describes how to create the following constraints:

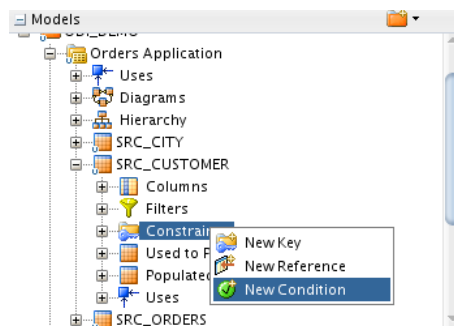
- Age Constraint
- Reference Constraint

### 5.2.3.1. Age Constraint

Creating an age constraints consists in adding a data validity condition on a column. To create the age constraint:

1. In the Models accordion in Designer Navigator, expand the *Orders Application* model.
2. Expand the SRC\_CUSTOMER datastore.
3. Right-click the Constraints node and select **New Condition** as shown in Figure 5–1.

*Figure 5–1 Insert New Condition*



4. In the Definition tab of the Condition Editor:
  - In the Name field, enter the name of your condition. For example: AGE > 21.
  - From the Type list, select **Oracle Data Integrator Condition**.
  - In the Where clause field, enter the following SQL code:  
`SRC_CUSTOMER.AGE > 21`

---

#### Note:

- You can enter this text directly in the Where clause field or you can use the Expression Editor. To open the Expression Editor click **Launch the expression editor** in the Where clause toolbar menu.
  - The constraints created by Oracle Data Integrator are not actually created on the database. The constraints are stored in the Repository.
-

- In the Message field, specify the error message as it will appear in your error table:  
Customer age is not over 21!

Figure 5–2 shows the Condition Editor.

**Figure 5–2 Condition Editor**

Definition	
Control	Condition [Model: Orders Application ▶ Datastore: SRC_CUSTOMER]
Markers	Name: AGE > 21
Memo	Type: Oracle Data Integrator Condition
Version	Where (Use the table alias: SRC_CUSTOMER):
Privileges	SRC_CUSTOMER.AGE > 21
Flexfields	
	Message:
	Customer age is not over 21!

5. From the File main menu, select **Save** to save the condition.

### 5.2.3.2. Reference Constraint

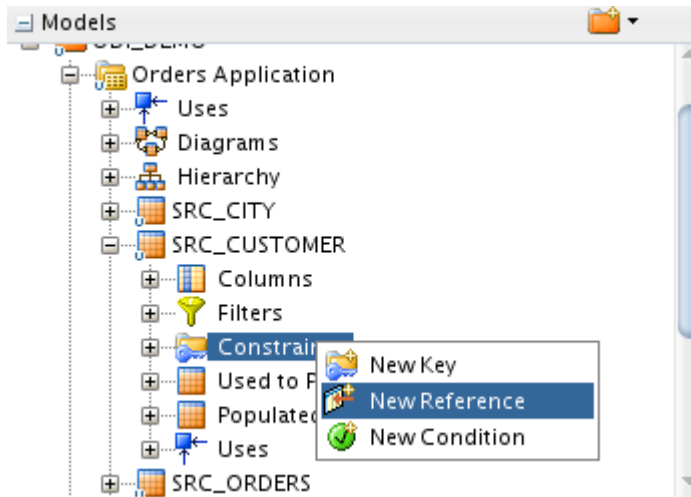
This section describes how to create a reference constraint based on the CITY\_ID column between the SRC\_CUSTOMER table and the SRC\_CITY table.

This constraint allows checking that customers are located in a city that exists in the SRC\_CITY table.

To create the reference constraint:

1. In the Models accordion in Designer Navigator, expand the *Orders Application* model.
2. Expand the SRC\_CUSTOMER datastore.
3. Right-click the Constraints node and select **New Reference** as shown in Figure 5–3.

Figure 5-3 Insert New Reference

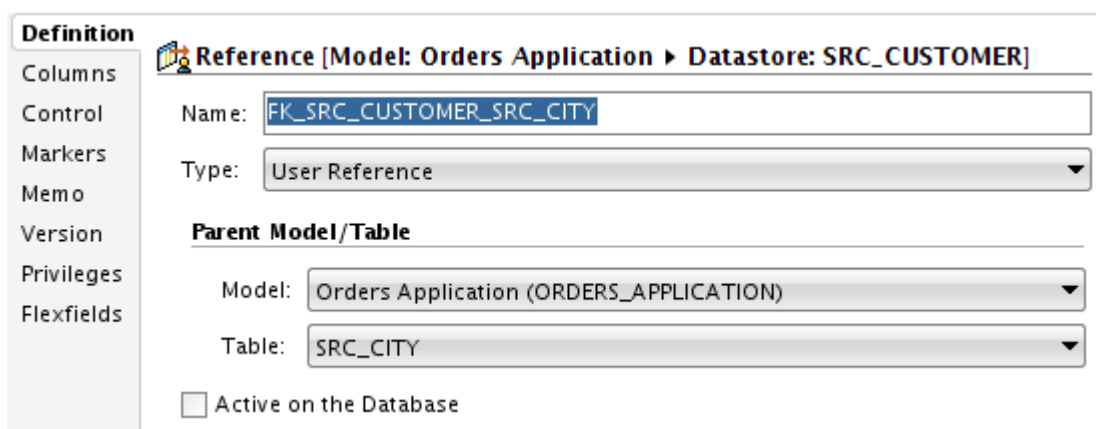


4. In the Definition tab of the Reference Editor:

- From the Type list, select **User Reference**.
- From the Model list in the Parent Model/Table section, select **Orders Application**. This is the data model containing the table you want to link to.
- From the Table list, select **SRC\_CITY**. This is the table you want to link to.

Figure 5-4 shows the Reference Editor.

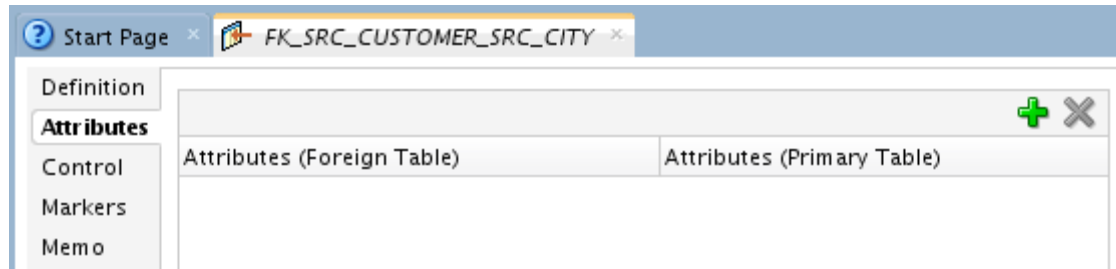
Figure 5-4 Reference Editor



5. In the Reference Editor, go to the Attributes tab.

6. On the Columns tab, click **Add** as shown in Figure 5-5.

**Figure 5–5 Columns tab of the Reference Editor**

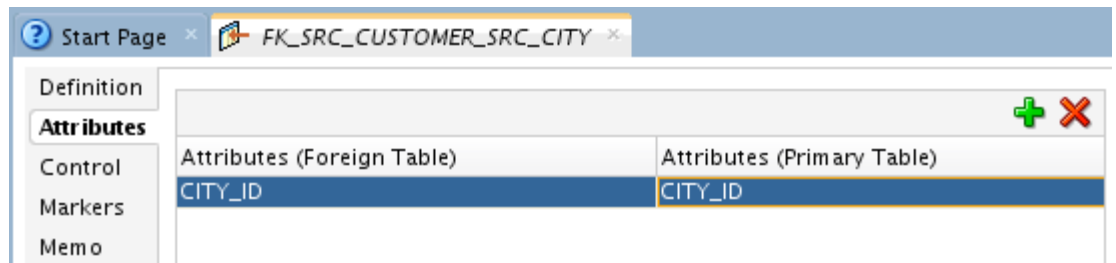


A new row is inserted in the columns table.

7. In this step you define the matching columns:
  - Click on the row that appears. This will bring up a drop-down list containing all of the columns in the appropriate table.
  - From the Columns (Foreign Table) list, select **CITY\_ID**.
  - From the Columns (Primary Table) list, select **CITY\_ID**.

Figure 5–6 shows the Columns tab of the Reference Editor with the selected matching columns.

**Figure 5–6 Columns tab of the Reference Editor with matching columns**



Note that in this example the Foreign Table is SRC\_CUSTOMER and the Primary Table is SRC\_CITY. Note also that it is not required for foreign keys that the column names of the Foreign Table and the Primary Table match. It just happens that they do in this example.

8. Select **File > Save** to save this reference.

**Tip:** You can alternately use the [CTRL - S] shortcut to save the current Editor.

## 5.2.4 Run the Static Control

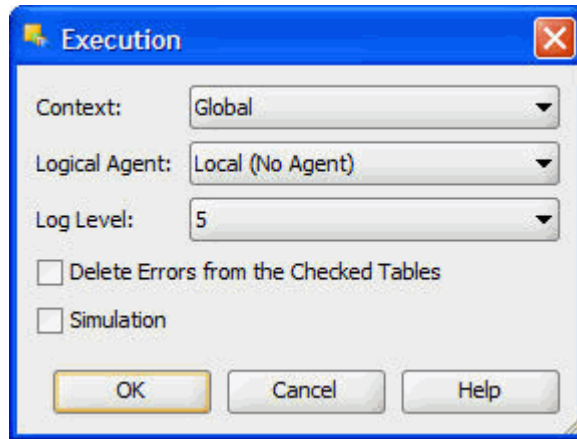
Running the static control verifies the constraints defined on a datastore. You can now verify the data in the SRC\_CUSTOMER datastore against the constraints defined in Section 5.2.3, "Creating Constraints".

To run the static control:

1. In the Models accordion in Designer Navigator, right-click the SRC\_CUSTOMER datastore.

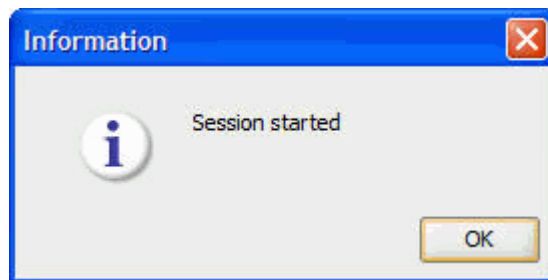
2. Select **Control > Check**.
3. The Execution dialog is displayed as shown in Figure 5–7.

*Figure 5–7 Execution Dialog*



4. Click **OK** in the Execution dialog.
5. The Information Dialog is displayed as shown in Figure 5–8.

*Figure 5–8 Information Dialog*



6. Click **OK** in the Information Dialog.

Oracle Data Integrator automatically generates all of the code required to check your data and start an execution session.

### 5.2.5 Follow the Execution of the Control in Operator Navigator

Through Operator Navigator, you can view your execution results and manage your development executions in the sessions.

To view the execution results of your control:

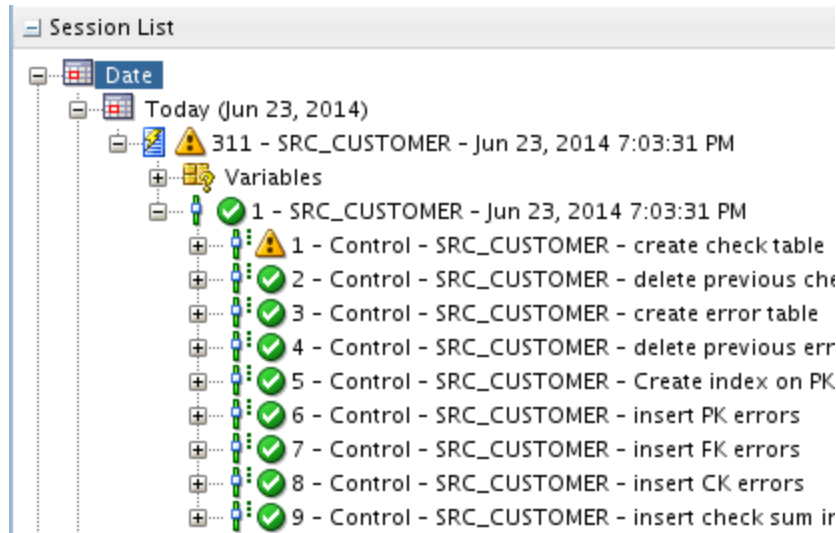
1. In the Session List accordion in Operator Navigator, expand the All Executions node.

The Session List displays all sessions organized per date, physical agent, status, keywords, and so forth.

2. Refresh the displayed information clicking **Refresh** in the Operator Navigator toolbar.

The log for one execution session appears as shown in Figure 5–9.

*Figure 5–9 Session List in Operator Navigator*



The log comprises 3 levels:

- Session (corresponds to an execution of a scenario, a mapping, a package or a procedure undertaken by an execution agent)
- Step (corresponds to a checked datastore, a mapping, a procedure or a step in a package or in a scenario)
- Task (corresponds to an elementary task of the mapping, process or check)

## 5.2.6 Interpreting the Results in Operator Navigator

This section describes how to determine the invalid records. These are the records that do not satisfy the constraints and has been rejected by the static control.

This section includes the following topics:

- Determining the Number of Invalid Records
- Reviewing the Invalid Records

### 5.2.6.1. Determining the Number of Invalid Records

To determine the number of invalid records:

1. In the Session List accordion in Operator Navigator, expand the All Executions node and the SRC\_CUSTOMER session.
2. Double-click the SRC\_CUSTOMER step to open the Session Step Editor.

3. The Record Statistics section details the changes performed during the static control. These changes include the number of inserts, updates, deletes, errors, and the total number of rows handled during this step.

Figure 5–10 shows the Session Step Editor of the SRC\_CUSTOMER step.

*Figure 5–10 SRC\_CUSTOMER Session Step Editor*

The screenshot displays the 'Session Step SRC\_CUSTOMER' editor. It features a sidebar with 'Definition' and 'Privileges' tabs. The main area is titled 'Session Step' and contains several input fields: 'Step Name' (SRC\_CUSTOMER), 'Step Type' (Datastore Check), 'Order Number' (1), 'Status' (Done), 'Context' (Global), and 'No. of Executions' (1). Below these are two expandable sections: 'Record Statistics' and 'Execution Statistics'. The 'Record Statistics' section includes fields for 'No. of Inserts' (0), 'No. of Updates' (0), 'No. of Deletes' (0), 'No. of Rows' (8), 'No. of Errors' (6, highlighted with a red box), and 'Maximum errors allowed' (with a percentage checkbox). The 'Execution Statistics' section includes 'Start' (Jun 23, 2014 7:03:31 PM), 'End' (Jun 23, 2014 7:03:33 PM), 'Duration (seconds)' (2), and 'Return Code' (0).

The number of invalid records is listed in the No. of Errors field. Note that the static control of the SRC\_CUSTOMER table has revealed 6 invalid records. These records have been isolated in an error table. See Section 5.2.6.2, "Reviewing the Invalid Records" for more information.

### 5.2.6.2. Reviewing the Invalid Records

You can access the invalid records by right-clicking on the table in your model and selecting **Control > Errors...**

To review the error table of the static control on the SRC\_CUSTOMER table:

1. In Designer Navigator, expand the *Orders Application* model.
2. Right-click the SRC\_CUSTOMER datastore.
3. Select **Control > Errors...**
4. The Error Table Editor is displayed as shown in Figure 5–11.

*Figure 5-11 Error Table of SRC\_CUSTOMER Table*

Start Page   Session Step SRC_CUSTOMER   Errors: SRC_CUSTOMER						
ODI_ROW_ID	...	ODI_ERR_MESS	ODI_CHECK_DATE	CUSTID	...	LAST_NAME
1	AAAVO1AABAA,S	ODI-15065: Join error (FK_SRC_CUSTOMER_SRC_CITY) between the	-08-19 08:28:50.0	203	0	Robert
2	AAAVO1AABAA,S	Customer age is not over 21!	-08-19 08:28:50.0	101	0	Brendt
3	AAAVO1AABAA,S	Customer age is not over 21!	-08-19 08:28:50.0	201	0	Sartois
4	AAAVO1AABAA,S	Customer age is not over 21!	-08-19 08:28:50.0	301	1	Edwards
5	AAAVO1AABAA,S	Customer age is not over 21!	-08-19 08:28:50.0	401	2	Diemers
6	AAAVO1AABAA,S	Customer age is not over 21!	-08-19 08:28:50.0	501	0	Arai

The records that were rejected by the check process are the following:

- 5 records in violation of the AGE > 21 constraint (the actual age of the customer is 21 or younger, see the AGE column for details).
- 1 record in violation of the FK\_CITY\_CUSTOMER constraint (The CITY\_ID value does not exist in the SRC\_CITY table).

You can view the entire record in this Editor. This means that you can instantly see which values are incorrect, for example the invalid CITY\_ID value in the top record.

Note that the error message that is displayed is the one that you have defined when setting up the AGE > 21 constraint in Section 6.2.3.1, "Age Constraint".

Now that the static controls have been run on the source data, you are ready to move on to the implementation of mappings.



---

## 6 Working with Packages

This chapter describes how to work with Packages in Oracle Data Integrator. The *Load Sales Administration* package is used as an example. An introduction to Packages and automating data integration between applications is provided.

This chapter includes the following sections:

- Section 6.1, "Introduction"
- Section 6.2, "Load Sales Administration Package Example"

### 6.1 Introduction

This section provides an introduction to automating data integration using packages in Oracle Data Integrator.

#### 6.1.1 Automating Data Integration Flows

The automation of the data integration is achieved by sequencing the execution of the different steps (mappings, procedures, and so forth) in a package and by producing a production scenario containing the ready-to-use code for each of these steps.

This chapter describes how to sequence the execution of the different steps. How to produce the production scenario is covered in Chapter 8, "Deploying Integrated Applications".

#### 6.1.2 Packages

A *Package* is made up of a sequence of steps organized into an execution diagram. Packages are the main objects used to generate scenarios for production. They represent the data integration workflow and can perform, for example, the following jobs:

- Start a reverse-engineering process on a datastore or a model
- Send an email to an administrator
- Download a file and unzip it
- Define the order in which mappings must be executed
- Define loops to iterate over execution commands with changing parameters

In this Getting Started exercise, you will load your *Sales Administration* application using a sequence of mappings. Since referential constraints exist between tables of this application, you must load target tables in a predefined order. For example, you cannot load the TRG\_CUSTOMER table if the TRG\_CITY table has not been loaded first.

In the Section 6.2, "Load Sales Administration Package Example", you will create and run a package that includes mappings that are included in the Demo project and mappings that you've created in Chapter 5, "Working with Mappings".

### 6.1.2.1 Scenarios

A *scenario* is designed to put source components (mapping, package, procedure, variable) into production. A scenario results from the generation of code (SQL, shell, and so forth) for this component.

Once generated, the code of the source component is frozen and the scenario is stored inside the Work repository. A scenario can be exported and then imported into different production environments.

---

**Note:** Once generated, the scenario's code is frozen, and all subsequent modifications of the package and/or data models which contributed to its creation will not affect it. If you want to update a scenario - for example because one of its mappings has been changed - then you must generate a new version of the scenario from the package or regenerate the existing scenario.

---

See "Working with Scenarios" in the *Oracle Fusion Middleware Developer's Guide for Oracle Data Integrator* for more information.

In Chapter 8, "Deploying Integrated Applications", you will generate the *LOAD\_SALES\_ADMINISTRATION* scenario from a package and run this scenario from Oracle Data Integrator Studio.

## 6.2 Load Sales Administration Package Example

This section contains the following topics:

- Purpose
- Developments Provided with Oracle Data Integrator
- Problem Analysis
- Creating the Package

### 6.2.1 Purpose

The purpose of the Load Sales Administration package is to define the complete workflow for the loading of the Sales Administration application and to set the execution sequence.

### 6.2.2 Mappings Provided with Oracle Data Integrator

The demo repository is delivered with a number of Mappings. The Demo project now contains the following objects as shown in Figure 6–1:

**Seven Mappings:**

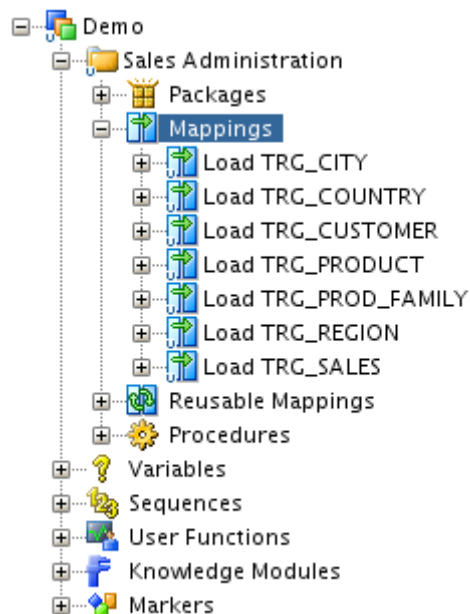
- **Load TRG\_CITY:** a mapping that populates the TRG\_CITY table. This mapping is delivered with the demo repository.
- **Load TRG\_COUNTRY:** a mapping that populates the TRG\_COUNTRY table. This mapping is delivered with the demo repository.
- **Load TRG\_CUSTOMER:** a mapping that populates the TRG\_CUSTOMER table. This mapping is created in Section 5.1, "Load TRG\_CUSTOMER Mapping Example".

- **Load TRG\_PRODUCT:** a mapping populates the TRG\_PRODUCT table. This mapping is delivered with the demo repository.
- **Load TRG\_PROD\_FAMILY:** a mapping that populates the TRG\_PROD\_FAMILY table. This mapping is delivered with the demo repository.
- **Load TRG\_REGION:** a mapping that populates the TRG\_REGION table. This mapping is delivered with the demo repository.
- **Load TRG\_SALES:** a mapping that populates the TRG\_SALES table. This mapping is created in Section 5.2, "Load TRG\_SALES Mapping Example".

**One procedure:**

The **Delete Targets** procedure empties all of the tables in the *Sales Administration* application. This operation is performed by using a *Delete* statement on each table.

*Figure 6–1 Demo Project*



### 6.2.3 Problem Analysis

In order to load the *Sales Administration* application correctly (in accordance with the referential integrity constraints), the tasks must be executed in the following order:

1. Empty the Sales Administration tables with the Delete Targets procedure
2. Load the TRG\_COUNTRY table with the Load TRG\_COUNTRY mapping
3. Load the TRG\_REGION table with the Load TRG\_REGION mapping
4. Load the TRG\_CITY table with the Load TRG\_CITY mapping
5. Load the TRG\_PROD\_FAMILY table with the Load TRG\_PROD\_FAMILY mapping
6. Load the TRG\_PRODUCT table with the Load TRG\_PRODUCT mapping
7. Load the TRG\_CUSTOMER table with the Load TRG\_CUSTOMER mapping

8. Load the TRG\_SALES table with the Load TRG\_SALES mapping

Such an integration process is built in Oracle Data Integrator in the form of a Package.

## 6.2.4 Creating the Package

This section describes how to create the Load Sales Administration Package. To create the Load Sales Administration Package perform the following steps:

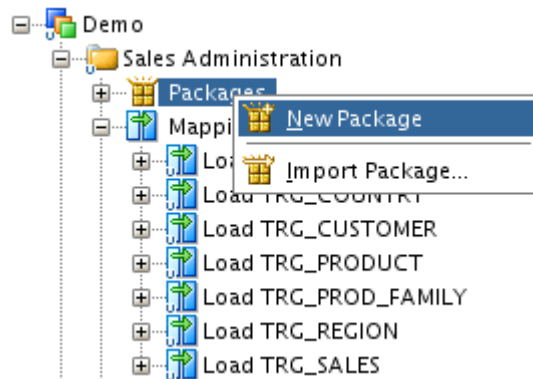
1. Create a New Package
2. Insert the Steps in the Package
3. Define the Sequence of Steps in the Package

### 6.2.4.1 Create a New Package

To create a new Package:

1. In Designer Navigator, expand the Demo project node in the Projects accordion.
2. Expand the Sales Administration node.
3. In the Sales Administration folder, right-click the Packages node and select **New Package** as shown in Figure 6–2.

*Figure 6–2 Insert New Package*



The Package Editor is started.

4. Enter the name of your Package 'Load Sales Administration' in the Name field.

### 6.2.4.2 Insert the Steps in the Package

To insert the steps in the Load Sales Administration Package:

1. Select the following components one by one from the Projects accordion and drag-and-drop them into the diagram:
  - Delete Targets (Procedure)
  - Load TRG\_COUNTRY
  - Load TRG\_REGION
  - Load TRG\_CITY
  - Load TRG\_CUSTOMER

- Load TRG\_PROD\_FAMILY
- Load TRG\_PRODUCT
- Load TRG\_SALES

These components are inserted in the Package and appear as steps in the diagram. Note that the steps are not sequenced yet.

### 6.2.4.3 Define the Sequence of Steps in the Package

Once the steps are created, you must reorder them into a data processing chain. This chain has the following rules:

- It starts with a unique step defined as the *First Step*.
- Each step has two termination states: Success or Failure.
- A step in failure or success can be followed by another step, or by the end of the Package.
- In case of failure, it is possible to define a number of retries.

A Package has one entry point, the First Step, but several possible termination steps. The Load Sales Administration Package contains only steps on Success.

#### Defining the First Step

To define the first step in the Load Sales Administration Package:

---

**Note:** If you have dragged and dropped the Package components in the order defined in Section 6.2.4.2, "Insert the Steps in the Package", the Delete Target procedure is already identified as the first step and the first step symbol is displayed on the step's icon. If this is the case, define the next steps on success.

---

1. Select and right-click the *Delete Target* procedure step.
2. Select **First Step** from the contextual menu. A small green arrow appears on this step.

#### Defining the Next Steps on Success

To define the next steps on success:

1. In the Package toolbar tab, select **Next Step on Success**.



2. Select the Delete Targets step.
3. Keep the mouse button pressed and move the cursor to the icon of the step that must follow in case of a success (here the Load TRG\_COUNTRY step) and release the mouse button.

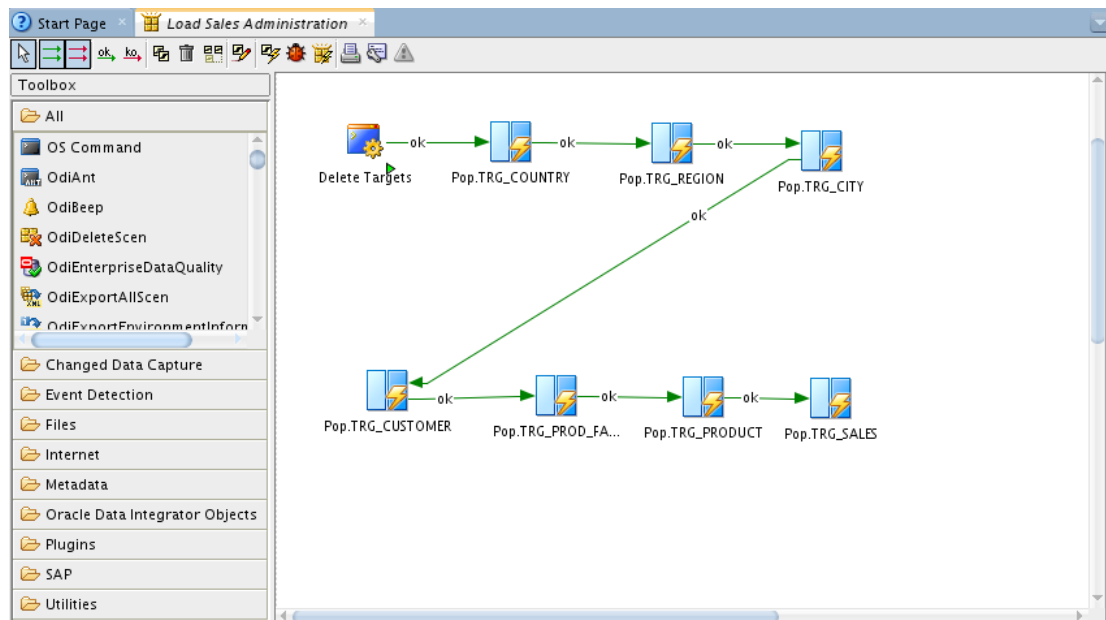
A green arrow representing the success path between the steps, with an ok label on it appears.

4. Repeat this operation to link all your steps in a success path sequence. This sequence should be:

- Delete Targets (First Step)
- Load TRG\_COUNTRY
- Load TRG\_REGION
- Load TRG\_CITY
- Load TRG\_CUSTOMER
- Load TRG\_PROD\_FAMILY
- Load TRG\_PRODUCT
- Load TRG\_SALES

The resulting sequence appears in the Package diagram as shown in Figure 6–3.

*Figure 6–3 Load Sales Administration Package Diagram*



5. From the File main menu, select **Save**. The package is now ready to be executed.

---

## 7 Executing Your Developments and Reviewing the Results

This chapter describes how to execute the Load Sales Administration Package you have created in Chapter 6, "Working with Packages" and the mappings Load TRG\_CUSTOMER and Load TRG\_SALES you have created in Chapter 4, "Working with Mappings". This chapter also describes how to follow the execution and how to interpret the execution results.

This chapter includes the following sections:

- Section 7.1, "Executing the Load Sales Administration Package"
- Section 7.2, "Executing the Load TRG\_SALES Mapping"

### 7.1 Executing the Load Sales Administration Package

This section contains the following topics:

- Run the Package
- Follow the Execution of the Package in Operator Navigator
- Interpreting the Results of the Load TRG\_CUSTOMER Session Step

#### 7.1.1 Run the Package

To run the Load Sales Administration Package:

1. In Designer Navigator, expand the Packages node under the Sales Administration node.
2. Select the Load Sales Administration Package.
3. Right-click and select **Run**.
4. In the Run Dialog, leave the default settings and click **OK**.
5. The Session Started Information Dialog is displayed. Click **OK**.

Oracle Data Integrator now starts an execution session.

#### 7.1.2 Follow the Execution of the Package in Operator Navigator

Through Operator Navigator, you can view your execution results and manage your development executions in the sessions.

To view the execution results of the Load Sales Administration Package:

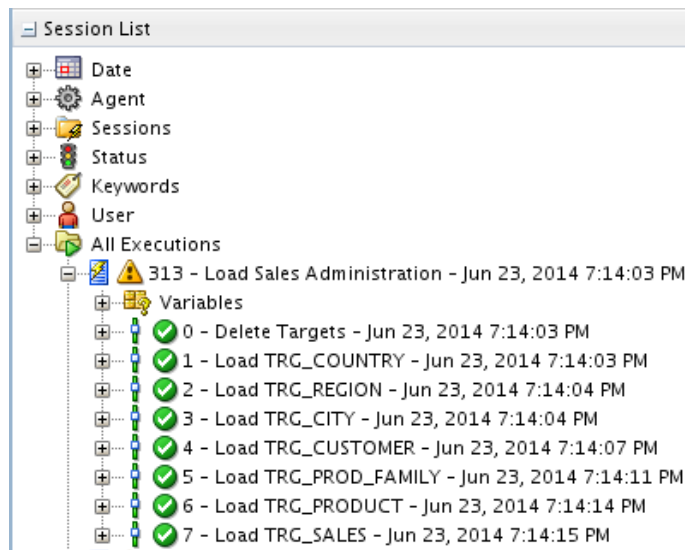
1. In the Session List accordion in Operator Navigator, expand the All Executions node.
2. Refresh the displayed information by clicking **Refresh** in the Operator Navigator toolbar.

The Refresh button is:



3. The log for the execution session of the Load Sales Administration Package appears as shown in Figure 7–1.

**Figure 7–1 Load Sales Administration Package Session Log**



### 7.1.3 Interpreting the Results of the Load TRG\_CUSTOMER Session Step

This section describes how to determine the invalid records detected by the Load TRG\_CUSTOMER mapping. These are the records that do not satisfy the constraints and have been rejected by the flow control of the Load TRG\_CUSTOMER mapping.

This section includes the following topics:

- Determining the Number of Processed Records
- Viewing the Resulting Data
- Reviewing the Invalid Records and Incorrect Data
- Correcting Invalid Data
- Review the Processed Records

#### 7.1.3.1 Determining the Number of Processed Records

To determine the number of records that have been processed by the Load TRG\_CUSTOMER mapping (this is the number of inserts, updates, deletes, and errors):

1. In the Session List accordion in Operator Navigator, expand the All Executions node.
2. Refresh the displayed information clicking **Refresh** in the Operator Navigator toolbar menu.



3. Expand the Load Sales Administration Package Session and open the Session Step Editor for the Load TRG\_CUSTOMER step. This is step 4.

4. On the Definition tab of the Session Step Editor, you can see in the Record Statistics section that the loading of the TRG\_CUSTOMER table produced 31 inserts and isolated 2 errors in an error table.

**Note:** Your individual results may vary. This is fine as long as the overall execution is successful.

Figure 7–2 shows the Record Statistics section of the Session Step Editor:

Figure 7–2 Record Statistics in the Session Step Editor

Record Statistics

No. of Inserts: 31

No. of Updates: 0

No. of Deletes: 0

No. of Errors: 2

No. of Rows: 91

Maximum errors allowed:

### 7.1.3.2 Viewing the Resulting Data

In this example, the resulting data are the 31 rows that have been inserted in the TRG\_CUSTOMER table during the mapping run.

To view the data resulting of your mapping run:

1. In Designer Navigator, expand the Models accordion and the *Sales Administration* model.
2. Select the TRG\_CUSTOMER datastore.
3. Right-click and select **View Data** to view the data in the target table.

Note that you can also select **Data...** to view and edit the data of the target table. The View Data Editor is displayed as shown in Figure 7–3.

Figure 7–3 View Data Editor

	CUST_ID	DEAR	CUST_NAME	ADDRESS	CITY_ID	PHONE	AGE	AGE_RANGE	SALES_PERS
1	102		Robin MCCARTHY	27 Pasadena Drive	11	(214) 555 3075	29	20-29 years	Andrew ANDERSEN
2	103		Peter TRAVIS	7835 Hartford Drive	12	(510) 555 4448	34	30-39 years	John GALAGERS
3	104		Joe LARSON	87 Carmel Blvd.	13	(213) 555 5095	45	40-49 years	Jeffrey JEFERSON
4	105		Tony GOLDSCHMIDT	91 Torre drive	14	(619) 555 6529	55	50-59 years	Jennie DAUMESNIL
5	106		William BAKER	2890 Grant Avenue	15	(312) 555 7040	64	60 years or more	Steve BARROT
6	107		Jack SWENSON	64 Imagination Drive	19	(202) 555 8125	74	60 years or more	Mary CARLIN
7	202		Philippe MICHAUD	197 impasse Renoir	23	78 21 86 20	22	20-29 years	Paul EDWOOD
8	204		Christine MARTIN	12 allee Victor Hugo	24	25 26 46 26	42	40-49 years	Rodolph BAUMAN
9	205		Luc PIAGET	38 allee des Saules	29	53 42 24 28	56	50-59 years	Stanley FISCHER
10	206		Michele GENTIL	17montee des Chenes	25	65 62 26 13	67	60 years or more	Andrew ANDERSEN

### 7.1.3.3 Reviewing the Invalid Records and Incorrect Data

You can access the invalid records by right-clicking on the datastore in your model and selecting **Control > Errors...**

To review the error table of the TRG\_CUSTOMER datastore:

1. In Designer Navigator, expand the *Sales Administration* model.
2. Select the TRG\_CUSTOMER datastore.
3. Right-click and select **Control > Errors...**
4. The Error Table Editor is displayed as shown in Figure 7–4.

*Figure 7–4 Error Table of TRG\_CUSTOMER*

	ODI_ROW_ID	...	ODI_ERR_MESS	ODI_CHECK_DATE	CUST_ID	DEAR	CUST_NAME
1	AAAHwmAABAA	F	ODI-15065: Join error (FK_CUST_CITY) between the table TRG_CUS	-03-10 17:34:32.0	207	Mrs	Marie-Chantale DUPONT
2	AAAHwmAABAA	F	ODI-15065: Join error (FK_CUST_CITY) between the table TRG_CUS	-03-10 17:34:32.0	203	Mr	Christian ROBERT

The mapping that you have executed has identified and isolated **2** invalid records in an error table that was automatically created for you.

In this error table, you can see that the mapping rejected:

- Records that did not satisfy the FK\_CUST\_CITY constraint (for example, the CITY\_ID value does not exist in the table of cities TRG\_CITY table).

You can use the ODI\_CHECK\_DATE field to identify the records rejected for your latest execution.

The invalid records were saved into an error table and were not integrated into the target table.

---

## 8 Deploying Integrated Applications

This chapter describes how to run the Load Sales Administration Package in a production environment.

This chapter includes the following sections:

- Section 8.1, "Introduction"
- Section 8.2, "Scenario Creation"
- Section 8.3, "Run the Scenario"
- Section 8.4, "Follow the Execution of the Scenario"

### 8.1 Introduction

The automation of the data integration flows is achieved by sequencing the execution of the different steps (mappings, procedures, and so forth) in a package and by producing a production scenario containing the ready-to-use code for each of these steps.

Chapter 6, "Working with Packages" describes the first part of the automation process: sequencing the execution of the different processes in a Package.

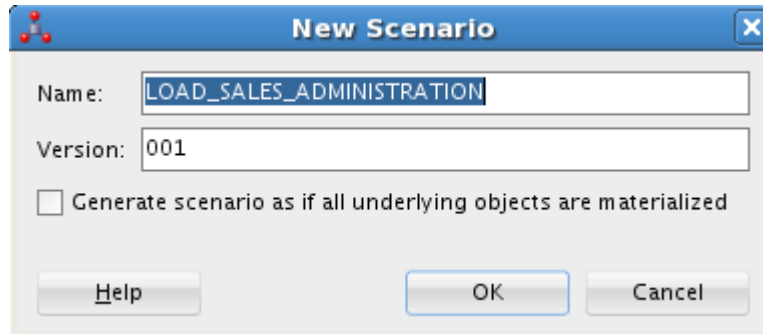
This chapter describes the second part: how to produce a scenario that runs automatically the Load Sales Administration Package in a production environment.

### 8.2 Scenario Creation

To generate the `LOAD_SALES_ADMINISTRATION` scenario that executes the Load Sales Administration Package:

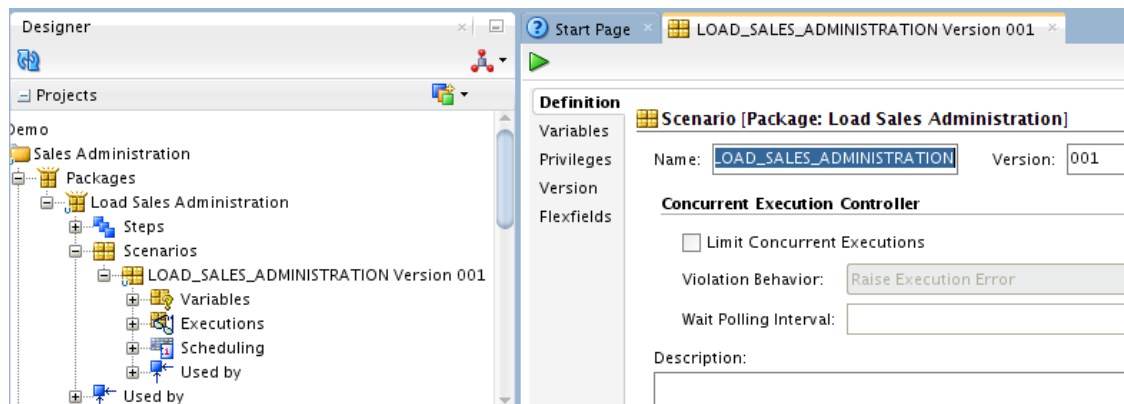
1. In the Project accordion, expand Sales Administration and then Packages.
2. Right click on Load Sales Administration and select **Generate Scenario...** The New Scenario dialog appears as shown in Figure 8–1.

Figure 8–1 New Scenario Dialog



3. The Name and Version fields of the Scenario are preset. Leave these values and click **OK**.
4. Oracle Data Integrator processes and generates the scenario. The new scenario appears on the Scenarios tab of the Package Editor and in the Demo Project as shown in Figure 8–2.

Figure 8–2 *LOAD\_SALES\_ADMINISTRATION* Scenario



## 8.3 Run the Scenario

Scenarios can be executed in several ways:

- Executing a Scenario from ODI Studio
- Executing a Scenario from a Command Line
- Executing a Scenario from a Web Service.

This Getting Started describes how to execute a scenario from ODI Studio. See "Executing a Scenario" in the *Oracle Fusion Middleware Developer's Guide for Oracle Data Integrator* for more information about how to execute a scenario from a command line and a web service.

### 8.3.1 Executing a Scenario from ODI Studio

You can start a scenario from Oracle Data Integrator Studio from Designer or Operator Navigator.

To start the LOAD\_SALES\_ADMINISTRATION scenario from Oracle Data Integrator Studio:

1. Select the LOAD\_SALES\_ADMINISTRATION scenario in the Projects accordion (in Designer Navigator) or the Load Plans and Scenarios accordion (in Designer and Operator Navigator).
2. Right-click, then select **Run**.
3. In the Execution Dialog, leave the default settings and click **OK**.
4. The Session Started Information Dialog is displayed. Click **OK**. The scenario is executed.

## 8.4 Follow the Execution of the Scenario

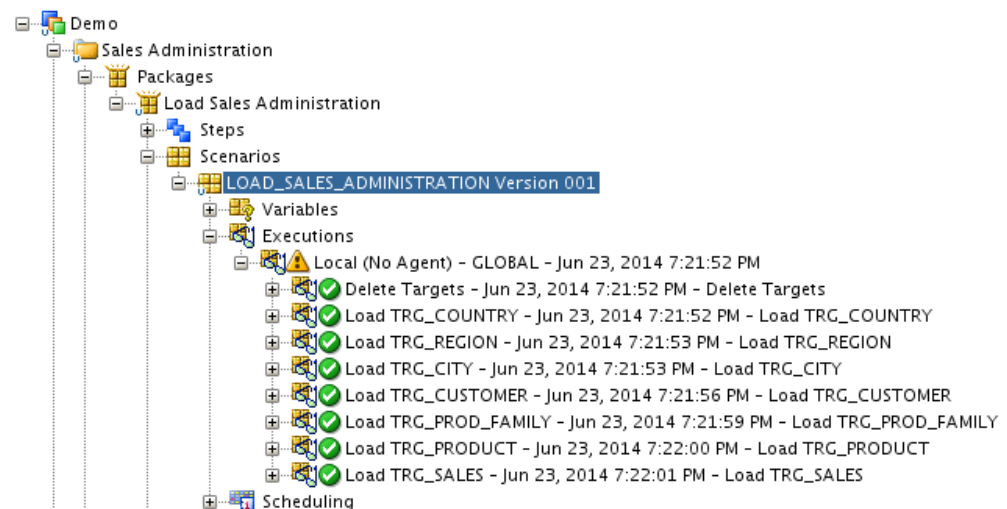
You can review the scenario execution in Operator Navigator, and find the same results as those obtained when the package was executed as described in Section 7.1.1, "Run the Package".

It is also possible to review the scenario execution report in Designer Navigator.

To view the execution results of the LOAD\_SALES\_ADMINISTRATION scenario in Designer Navigator:

1. In the Projects accordion in Designer Navigator, expand the Scenarios node under the Load Sales Administration package.
2. Refresh the displayed information by clicking **Refresh** in the Designer Navigator toolbar menu.
3. The log for the execution session of the LOAD\_SALES\_ADMINISTRATION scenario appears as shown in Figure 8–3.

**Figure 8–3** LOAD\_SALES\_ADMINISTRATION Scenario Session Log



---

## 9 Using Oracle Data Integrator with Oracle GoldenGate

This chapter describes how to configure and use Changed Data Capture (CDC) with Oracle GoldenGate and Oracle Data Integrator.

This tutorial is only included in the ODI Getting Started VirtualBox image available at: <http://www.oracle.com/technetwork/middleware/data-integrator/overview/index.html>

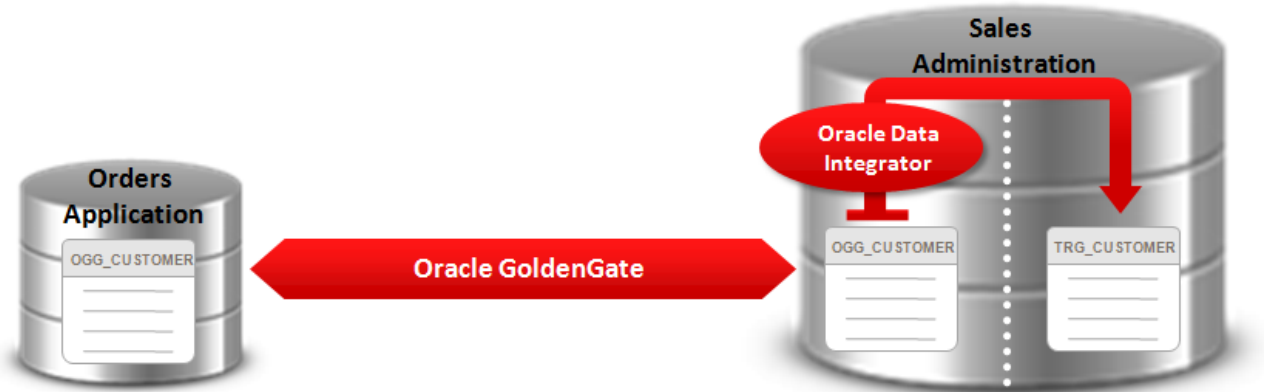
This chapter includes the following sections:

- Section 9.1, "Introduction"
- Section 9.2, "Reviewing the Oracle GoldenGate JAgent configuration in ODI Studio"
- Section 9.3, "Initial load"
- Section 9.4, "Setting up Changed Data Capture"
- Section 9.5, "Synchronizing the changed data"

### 9.1 Introduction

The demo environment used for this Getting Started tutorial also includes two installations of Oracle GoldenGate 12c. A source installation is used to capture data from the *Orders Application* schema while a target installation delivers the captured data into the *Sales Administration* schema. Once the data has been replicated by GoldenGate ODI will detect the changed records and perform some transformations before inserting the data into the final target table.

*Figure 9-1 Real-time data integration architecture with GoldenGate and Data Integrator*



In this chapter we will configure Oracle GoldenGate through ODI Studio using the following Journalization Knowledge Module: JKM Oracle to Oracle Consistent (OGG Online).

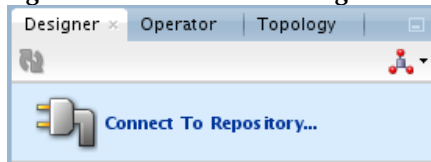
Please refer to the *Connectivity and Knowledge Modules Guide for Oracle Data Integrator* for more information about the GoldenGate Knowledge Modules and how the two products integrate.

The *Using Journalizing* chapter of the *Developer's Guide for Oracle Data Integrator* contains valuable information about the Changed Data Capture framework in ODI.

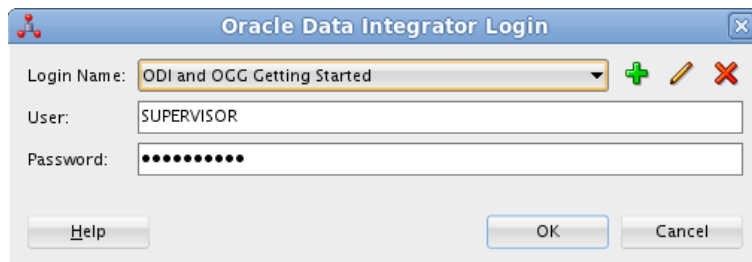
### 9.1.1 Connect to the ODI Work Repository

1. Open up ODI Studio.
2. If you were already connected to a different repository, use *Disconnect* in the *ODI* menu to close the current connection.
3. Click on *Connect to Repository...* then set the Login Name to *ODI and OGG Getting Started* and click OK.

**Figure 9-2 Oracle Data Integrator Studio 12c**

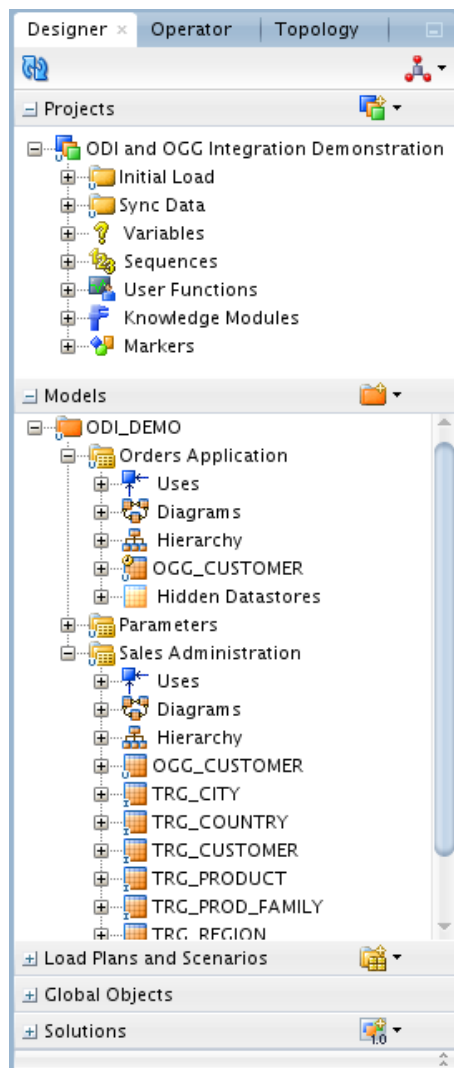


When prompted for a wallet password enter *welcome1*



The Designer Navigator appears as shown in Figure 9–3.

Figure 9–3 Designer Navigator



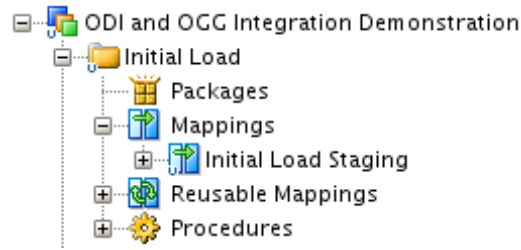
The demonstration environment provides the objects you will need in order to use Oracle Data Integrator and Oracle GoldenGate together:

- In the Models accordion, you will find the data models corresponding to the *Orders Application*, *Parameters*, and *Sales Administration* applications:
  - The *Orders Application* model contains a single datastore called *OGG\_CUSTOMER* which contains the data that will be captured and replicated by Oracle GoldenGate.
  - The *Sales Administration* model contains several datastores including *OGG\_CUSTOMER* which is a copy of the *OGG\_CUSTOMER* datastore seen in the *Orders Applications* model. This table is the target of the Oracle GoldenGate processes and the changed records will be replicated into it from the *OGG\_CUSTOMER* table contained in the *Orders Applications* model. This model also contains the *TRG\_CUSTOMER* datastore which is the final target table and will be populated by ODI using the changed data replicated by GoldenGate.



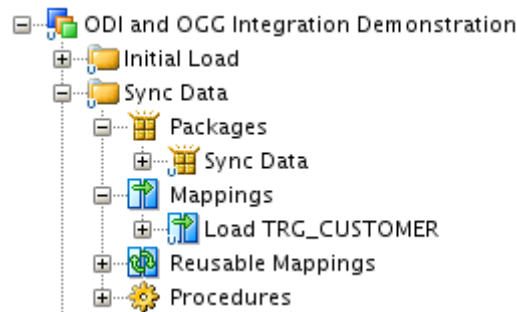
- In the Projects accordion, you will find the *ODI and OGG Integration Demonstration* project and two folders *Initial Load* and *Sync Data* which already contain several mappings you will be using in this chapter.
  - The *Initial Load* folder contains a Mapping named *Initial Load Staging* which is used to perform the initial load of the *OGG\_CUSTOMER* table in the *Sales Administration* model

**Figure 9– 4 Initial Load folder**



- The *Sync Data* folder contains a Mapping named *Load TRG\_CUSTOMER* which is used to perform both an initial load of *TRG\_CUSTOMER* using only ODI as well as an incremental load of *TRG\_CUSTOMER* using ODI and GoldenGate. This folder also contains a Package called *Sync Data* which will be used to orchestrate the Data Integrator and GoldenGate processes.

**Figure 9–5 Sync Data folder**



The necessary Knowledge Modules (KM) are already available in the *ODI and OGG Integration Demonstration* project:

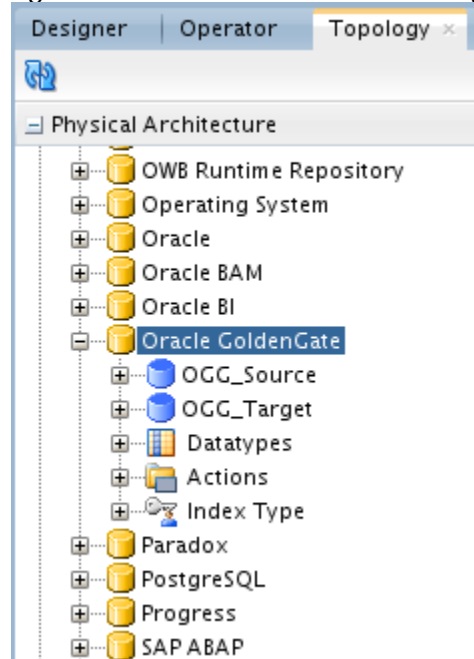
- IKM Oracle Incremental Update
- IKM Oracle Insert
- IKM SQL Control Append
- LKM Oracle to Oracle Pull (DB Link)
- LKM SQL to SQL (Built-In)
- CKM Oracle
- JKM Oracle to Oracle Consistent (OGG Online)

## 9.2 Reviewing the Oracle GoldenGate JAgent configuration in ODI Studio

Oracle Data Integrator 12c can integrate with Oracle GoldenGate through the GoldenGate JAgents. The JAgents are defined in ODI as Data Servers in Topology. In this getting started environment we have already pre-configured the connections to the source GoldenGate installation JAgent (*OGG\_Source*) and the target GoldenGate installation JAgent (*OGG\_Target*). We will now review their configuration:

1. Open up the Topology Navigator and expand the *Physical Architecture* and *Technologies* nodes.
2. Scroll down in the *Technologies* list and expand the *Oracle GoldenGate* node.

**Figure 9-6 Oracle GoldenGate technology**



3. Double-click on *OGG\_Source* to review its configuration settings. In Figure 9-7 we can see the various configuration parameters stored in ODI such as the JAgent host and port as well as the GoldenGate installation directory.

This information allows ODI to communicate with the JAgent and remotely deploy configuration settings into a GoldenGate installation.

Figure 9-7 Oracle GoldenGate JAgent configuration

The JAgent processes are not yet running we will start them once we will have finished the GoldenGate configuration in Designer and we will come back to Topology to test the connections.

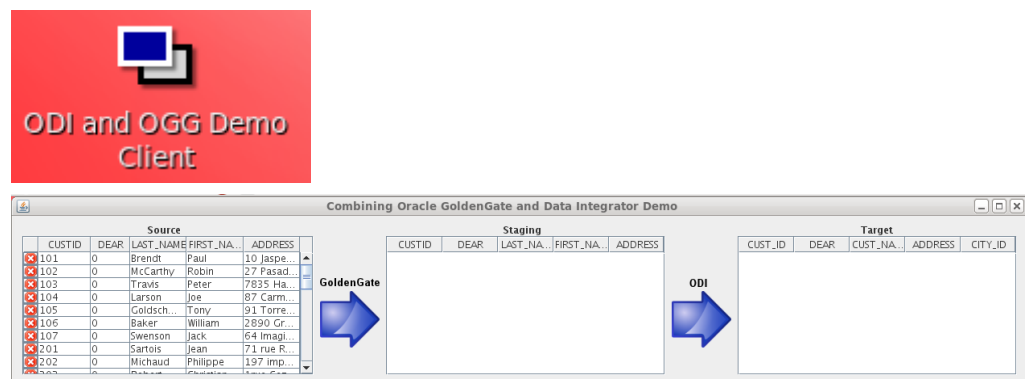
## 9.3 Initial load

Before we load the data and start the changed data capture processes we will look at the data contained in our source (OGG\_CUSTOMER in Orders Application model), staging (OGG\_CUSTOMER in Sales Administration model) and target (TRG\_CUSTOMER in Orders Application model) tables.

### 9.3.1 Starting the ODI and OGG Demo Client

1. Go back to the VirtualBox desktop
2. Double-click on the *ODI and OGG Demo Client* shortcut on the Linux desktop.

Figure 9-8 Starting the ODI and OGG Demo Client



This client is not part of ODI or GoldenGate and was created specifically to showcase the integration between the two products. Once the client is opened it will show the Staging and Target tables as empty. We will first do their initial load and then stream and transform the data into the target table.

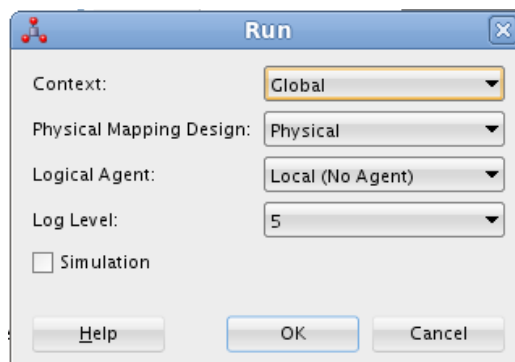
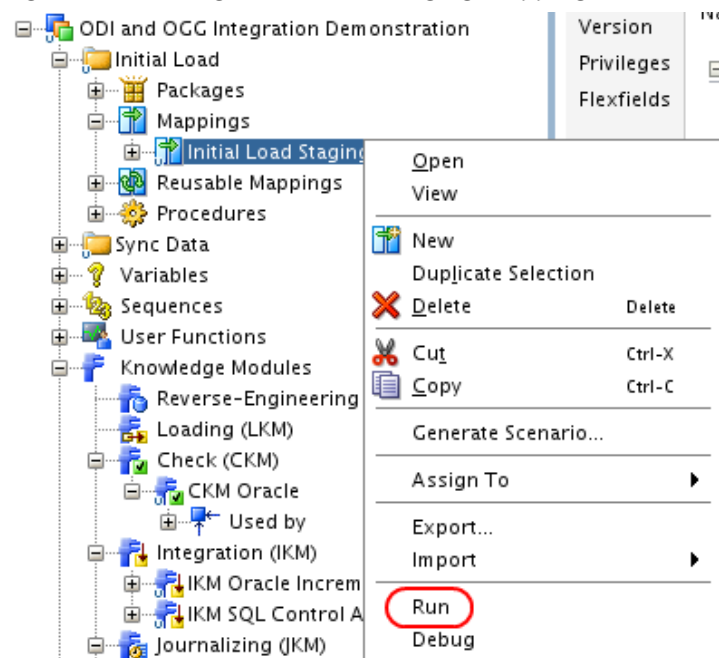
**Note:** If the Staging and Target tables are not empty you must truncate them using the *Clean Up Target Tables Procedure* available in the *Initial Load* folder in ODI Studio. Simply run this procedure to clean up the environment.

## 9.3.2 Running the Mappings

We will be using ODI to perform the initial load of the GoldenGate target table (*OGG\_CUSTOMER* in *Sales Administration* model) and the Data Integrator target table (*TRG\_CUSTOMER* in *Sales Administration* model).

1. Go back to Designer, expand the *Initial Load* folder and expand the *Mappings* node. The Mapping called *Initial Load Staging* will be doing a bulk load from the source table *OGG\_CUSTOMER* in the *Orders Application* model into the *OGG\_CUSTOMER* table in the *Sales Administration* model
2. Right-click on *Initial Load Staging* and select *Run*

**Figure 9-9 Running Initial Load Staging Mapping**



Leave the default settings and click *OK* in the *Run* window and then click *OK* again to close the *Information* window.

3. Go back to the *ODI and OGG Demo Client* window and you will now see that the Staging table has been populated as shown below

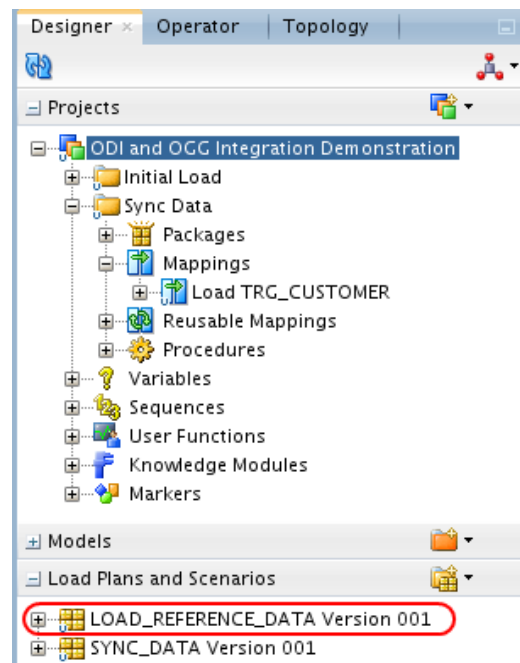
**Figure 9-10 ODI and OGG Demo Client**

Source					Staging					Target				
CUSTID	DEAR	LAST_NAME	FIRST_NAME	ADDRESS	CUSTID	DEAR	LAST_NAME	FIRST_NAME	ADDRESS	CUST_ID	DEAR	CUST_NAME	ADDRESS	CITY_ID
101	O	Brenda	Paul	10 Jasper...	101	O	Brenda	Paul	10 Jasper...					
102	O	McCarthy	Robin	27 Pasade...	102	O	McCarthy	Robin	27 Pasade...					
103	O	Travis	Peter	7835 Hartf...	103	O	Travis	Peter	7835 Hartf...					
104	O	Larson	Joe	87 Carmel...	104	O	Larson	Joe	87 Carmel...					
105	O	Goldschmidt	Tony	91 Torre d...	105	O	Goldschmidt	Tony	91 Torre d...					
106	O	Baker	William	2890 Gran...	106	O	Baker	William	2890 Gran...					
107	O	Swenson	Jack	64 Imagina...	107	O	Swenson	Jack	64 Imagina...					
201	O	Santos	Jean	71 rue Ro...	201	O	Santos	Jean	71 rue Ro...					
202	O	Michaud	Philippe	197 Impas...	202	O	Michaud	Philippe	197 Impas...					

We will now perform the initial load of *TRG\_CUSTOMER* in the *Sales Administration* model.

1. Go back to Designer
2. Loading data into *TRG\_CUSTOMER* requires other tables like *TRG\_CITY* or *TRG\_REGION* to be populated. If you didn't previously run the *Load Sales Administration* package from Chapter 6 "Working with Packages" then you will need to first execute the following scenario: *LOAD\_REFERENCE\_DATA*. You can find this scenario in the *Load Plans and Scenarios* accordion as shown below:

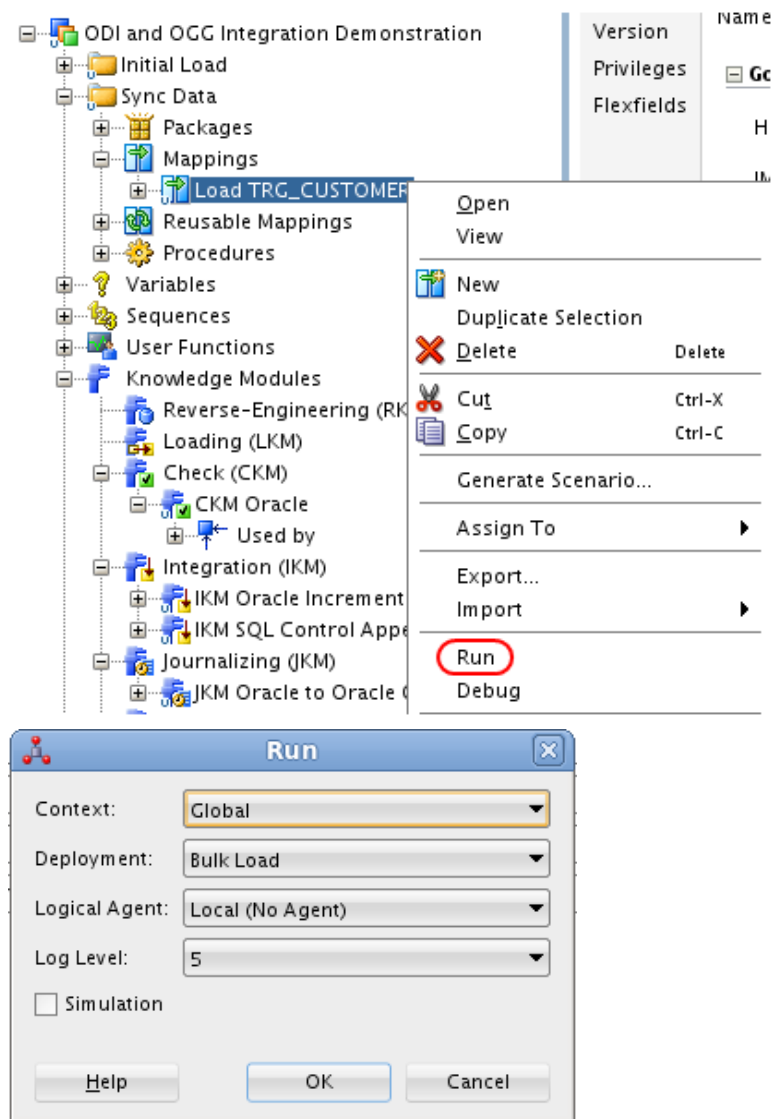
Figure 9-11 LOAD\_REFERENCE\_DATA Scenario



Right-click on it and select *Run* then leave the default settings in the *Run* window and click *OK* to close the *Information* window.

3. Now expand the *Sync Data* folder and expand the *Mappings* node. The Mapping called *Load TRG\_CUSTOMER* will be doing a bulk load from the staging table *OGG\_CUSTOMER* in the *Sales Administration* model into the *TRG\_CUSTOMER* table in the same model
4. Right-click on *Load TRG\_CUSTOMER* and select *Run*

Figure 9-12 Running Load TRG\_CUSTOMER Mapping



Leave the default settings, make sure you are using the *Bulk Load* Deployment Specification and click *OK* in the *Run* window and then click *OK* again to close the *Information* window.

- Go back to the ODI and OGG demo client window and you will now see that the Target table has been populated as shown below

**Figure 9-13 ODI and OGG Demo Client**

Combining Oracle GoldenGate and Data Integrator Demo																
Source						Staging						Target				
CUSTID	DEAR	LAST_NAME	FIRST_NAME	ADDRESS		CUSTID	DEAR	LAST_NAME	FIRST_NAME	ADDRESS		CUST_ID	DEAR	CUST_NAME	ADDRESS	CITY_ID
101	0	Brendt	Paul	10 Jasper		101	0	Brendt	Paul	10 Jasper		101	0	Paul Brendt	10 Jasper	107
102	0	McCarthy	Robin	27 Pasade		102	0	McCarthy	Robin	27 Pasade		102	0	Robin Mcc	27 Pasade	11
103	0	Travis	Peter	7835 Hart		103	0	Travis	Peter	7835 Hart		103	0	Peter Travis	7835 Hart	12
104	0	Larson	Joe	87 Carmel		104	0	Larson	Joe	87 Carmel		104	0	Joe Larson	87 Carmel	13
105	0	Goldschmidt	Tony	91 Terre d		105	0	Goldschmidt	Tony	91 Terre d		105	0	Tony Gold	91 Terre	14
106	0	Baker	William	2890 Gra		106	0	Baker	William	2890 Gra		106	0	William Ba	2890 Gra	15
107	0	Swenson	Jack	64 Imagin		107	0	Swenson	Jack	64 Imagin		107	0	Jack Swen	64 Imagin	19
201	0	Sartois	Jean	71 rue Ro		201	0	Sartois	Jean	71 rue Ro		201	0	Jean Sartois	71 rue Ro	25
202	0	Michaud	Philippe	197 Impas		202	0	Michaud	Philippe	197 Impas		202	0	Philippe M	197 Impa	23

**Note:** If TRG\_CUSTOMER is not getting populated it is most likely due to the missing reference data. Please run the scenario *LOAD\_REFERENCE\_DATA* as mentioned in Step #2 above and then run *Load TRG\_CUSTOMER* again.

Now that we have data in all our tables we can start the GoldenGate processes, initialize the ODI CDC infrastructure and propagate changed records from the source table *OGG\_CUSTOMER* in the *Orders Application* model into the target table *TRG\_CUSTOMER* in the *Sales Administration* model.

## 9.4 Setting up Changed Data Capture

The ODI CDC infrastructure and Oracle GoldenGate Capture and Delivery processes configuration is done at the Models level in Designer.

- In ODI Studio go to Designer and expand the *Models* node.
- Double-click on the *Orders Application* model to open it and click on the *Journalizing* panel.

**Figure 9-14 Orders Application model Journalizing panel**

Start Page | Orders Application

Reverse Engineer | Check Model | Generate and deploy

Definition  
Reverse Engineer  
Selective Reverse-Engineering  
Control  
**Journalizing**  
Journalized Tables  
Services  
Markers  
Memo  
Version  
Privileges  
Flexfields

Journalizing Mode: ☒ Consistent Set ☐ Simple

**Journalizing KM**

Knowledge Module: JKM Oracle to Oracle Consistent (OGG Online).ODI and OGG Integration Demonstration

**GoldenGate Process Selection**

Capture Process: CAP\_LS [Create]

Delivery Process: DEL\_LS [Create]

Initial Load Capture Process: CAP\_IL\_LS [Create]

Initial Load Delivery Process: DEL\_IL\_LS [Create]

Option	Value
ONLINE	true
VALIDATE	true
LOCAL_TEMP_DIR	<Default>: <?= java.lang.System.getProperty("java.io.tmpdir") ?>
NB_APPLY_PROCESS	<Default>:1
SRC_SETUP_OGG_PROCESSES	<Default>:true
STG_SETUP_OGG_PROCESSES	<Default>:true
COMPATIBLE	<Default>:11
CHECKPOINT_TABLE_NAME	<Default>:ODIOGGCKPT

The Journalizing panel is where the mechanisms used for Changed Data Capture in ODI are set up. In this example we use Oracle GoldenGate to detect and replicate changed records, to do so we have picked the *JKM Oracle to Oracle Consistent (OGG*



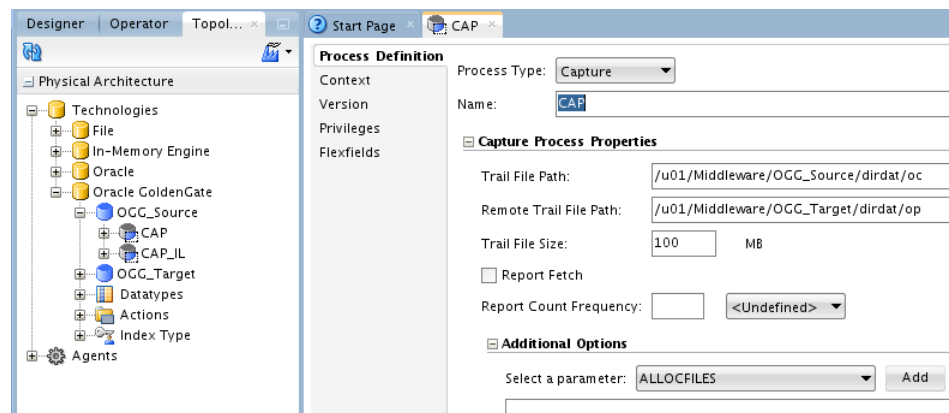
Online) Journalization Knowledge Module.

This JKM will generate the ODI CDC infrastructure as well as remotely configure the source and target GoldenGate instances.

The *GoldenGate Process Selection* section can be used to define new GoldenGate Capture and Delivery processes from ODI Studio or to select existing ones. In this environment we have already pre-configured all the Capture and Delivery processes.

3. You can optionally go back to Topology and under *Physical Architecture*, expand *OGG\_Source* and open up the *CAP* Physical Schema underneath it to review the capture process settings as shown in the figure below.

**Figure 9–15 GoldenGate Capture process configuration**



Now that we have reviewed the GoldenGate settings we can start it in this environment.

For more information about the ODI CDC framework or the GoldenGate parameters please refer to the resources mentioned in section 9.1 *Introduction*.

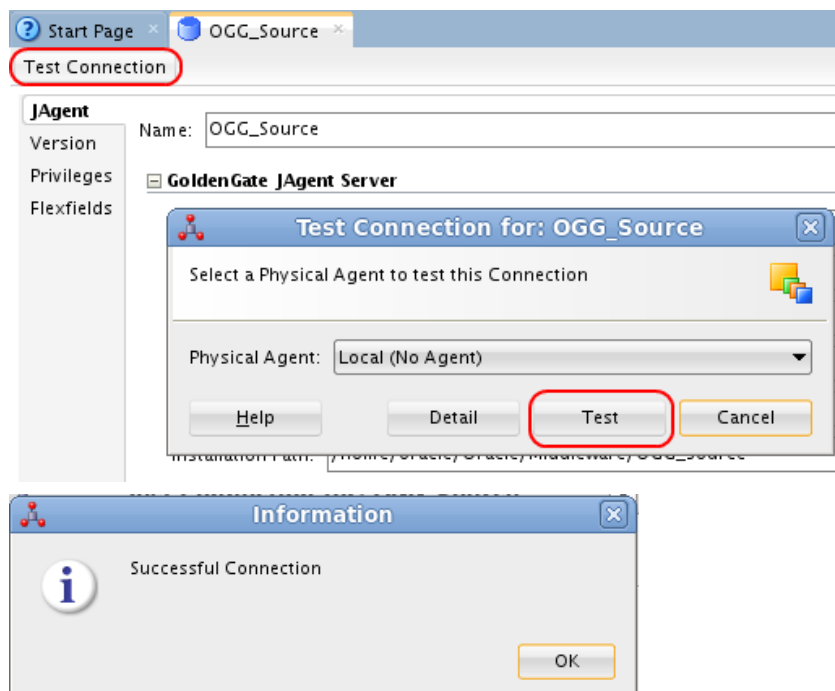
4. Go to the VirtualBox image desktop
5. Start the GoldenGate Capture and Delivery processes using the *StartOGG* command. It will start the Manager and JAgent processes for both the GoldenGate source and target.

**Figure 9-16 Start GoldenGate using Start OGG**



6. Wait a minute so the script can complete and go back to ODI Studio and go to Topology open up *OGG\_Source* then click on *Test Connection* and finally on *Test* as shown below.

**Figure 9-17 Testing GoldenGate JAgent connection**

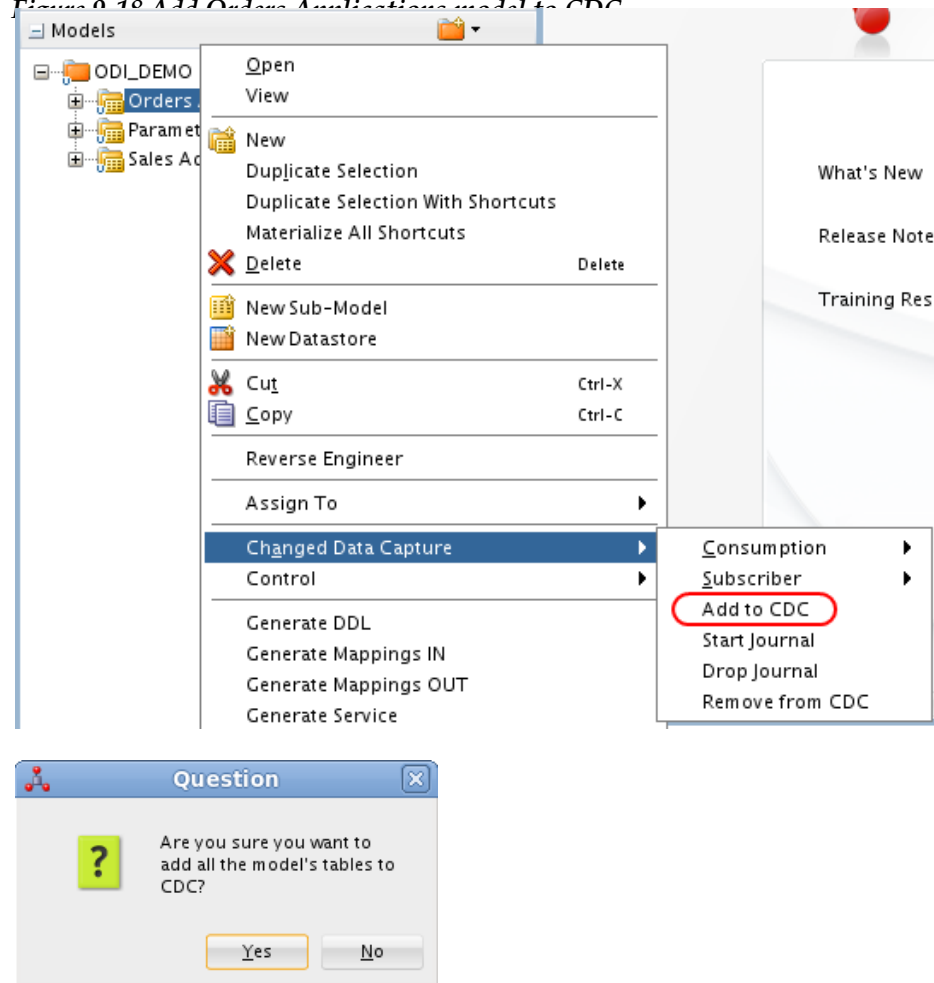


The test results should show a successful connection, you can repeat the same steps for *OGG\_Target*.

The GoldenGate infrastructure is running fine, we can finish the configuration.

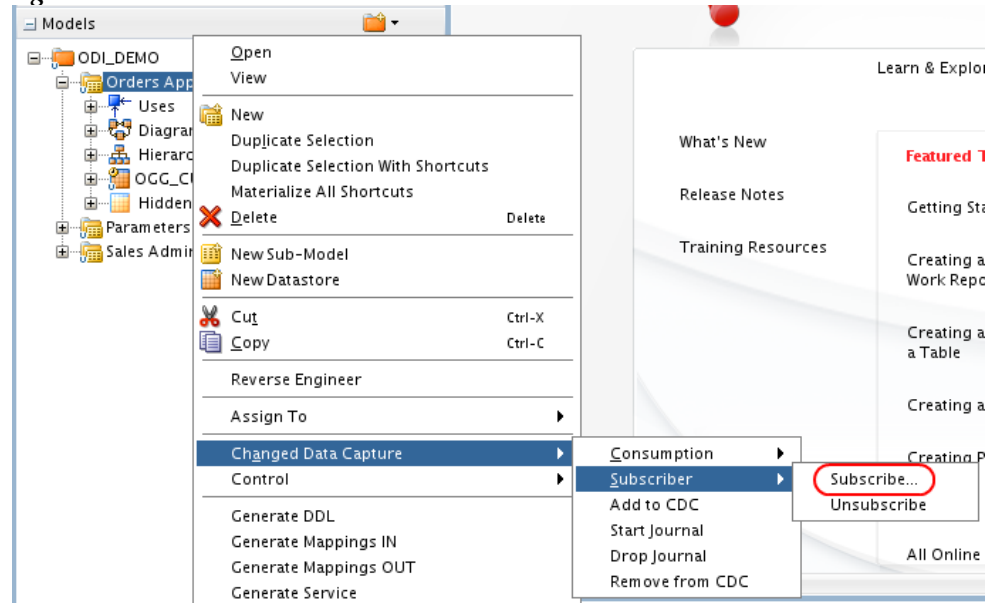
1. Go back to Designer
2. Right-click on the *Orders Application* model and select *Changed Data Capture* then *Add to CDC*. This will register all the datastores contained in that model into the ODI Changed Data Capture framework. Click *Yes* to close the *Question* window.

Figure 9-18 Add Orders Application model to CDC



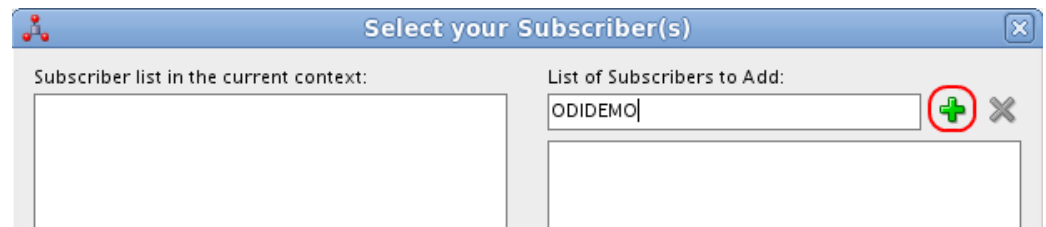
3. Next we will add a Subscriber to the ODI CDC infrastructure. A subscriber is an alias representing a process interested in changed data. Since we can have multiple processes interested in the same changed data we can define multiple subscribers. Each subscriber can move the changed data when needed without impacting the other ones.  
Right-click on the *Orders Application* model and select *Changed Data Capture* then *Subscriber* and finally click on *Subscribe*...

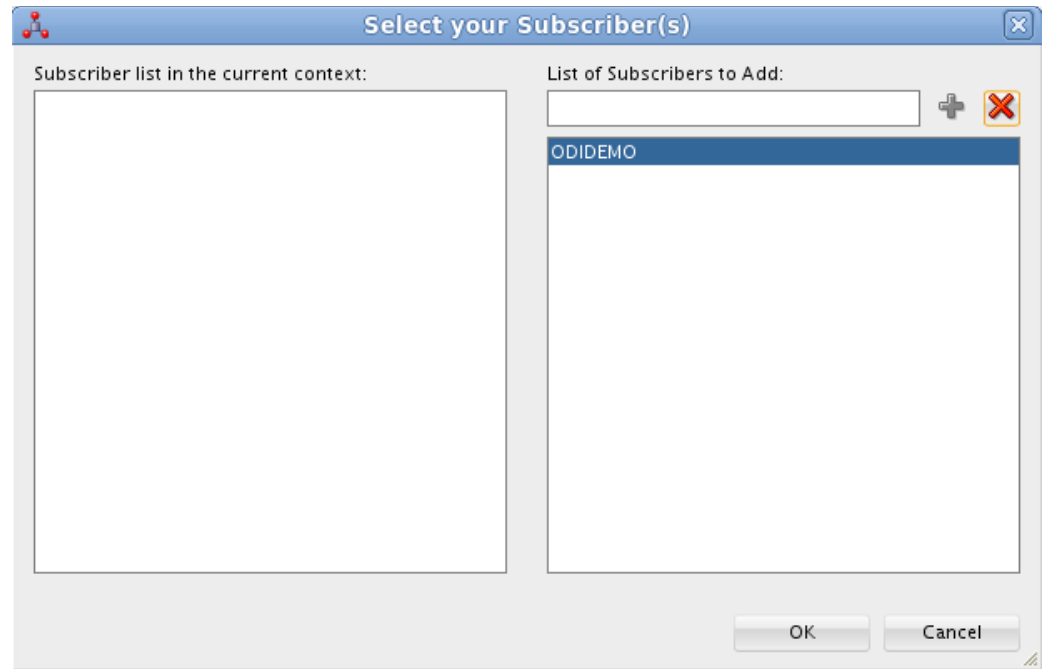
**Figure 9-19 Add a Subscriber**



In the *Select your Subscriber(s)* window enter ODIDEMO in the *List of Subscribers to Add* field then click on the + icon to add ODIDEMO to the list. Click OK when done.

**Figure 9-20 Enter Subscriber name**

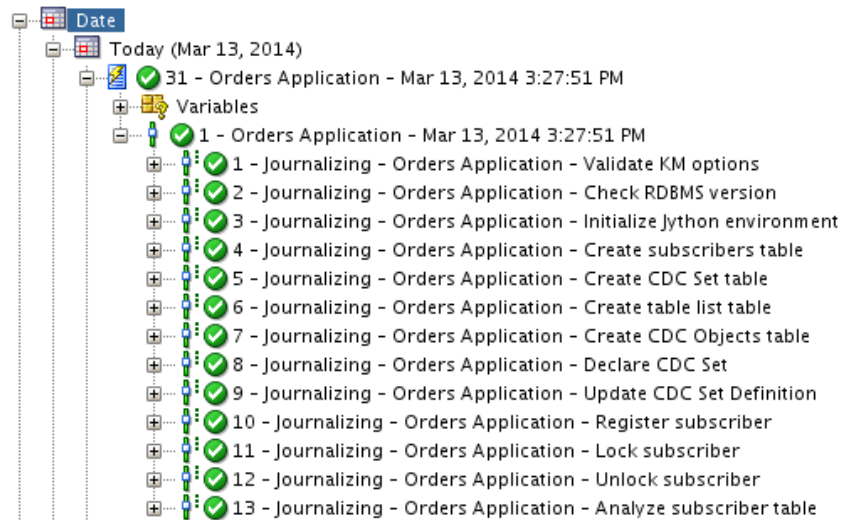




Click OK in the *Run* window and finally click OK to close the *Information* window.

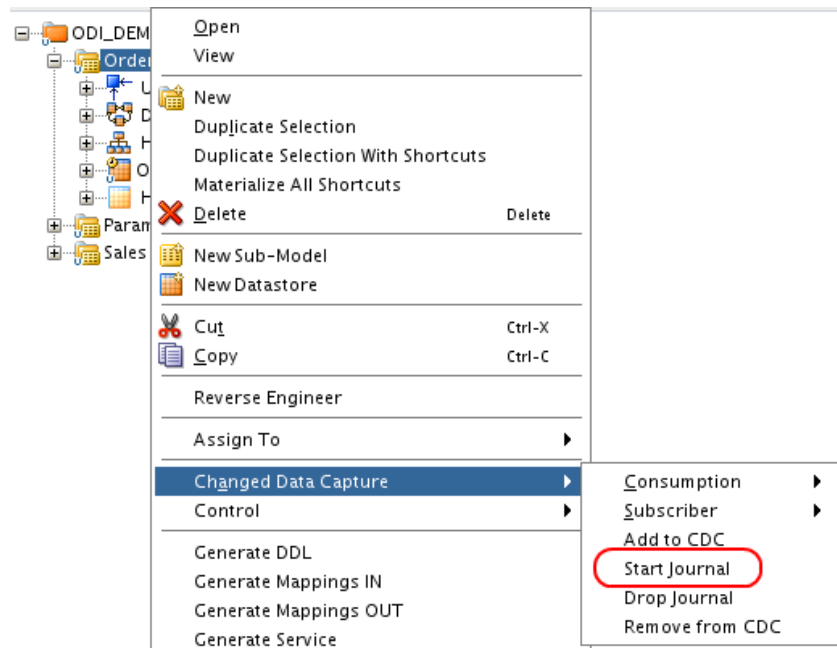
4. A Session is now running, go to Operator to verify that all the steps were executed correctly as shown in Figure 10-22 below. Click on the Refresh button as needed.

**Figure 9-21 Session monitoring in Operator**



5. Go back to Designer and right-click on the *Orders Application* model and select *Changed Data Capture* then *Start Journal*

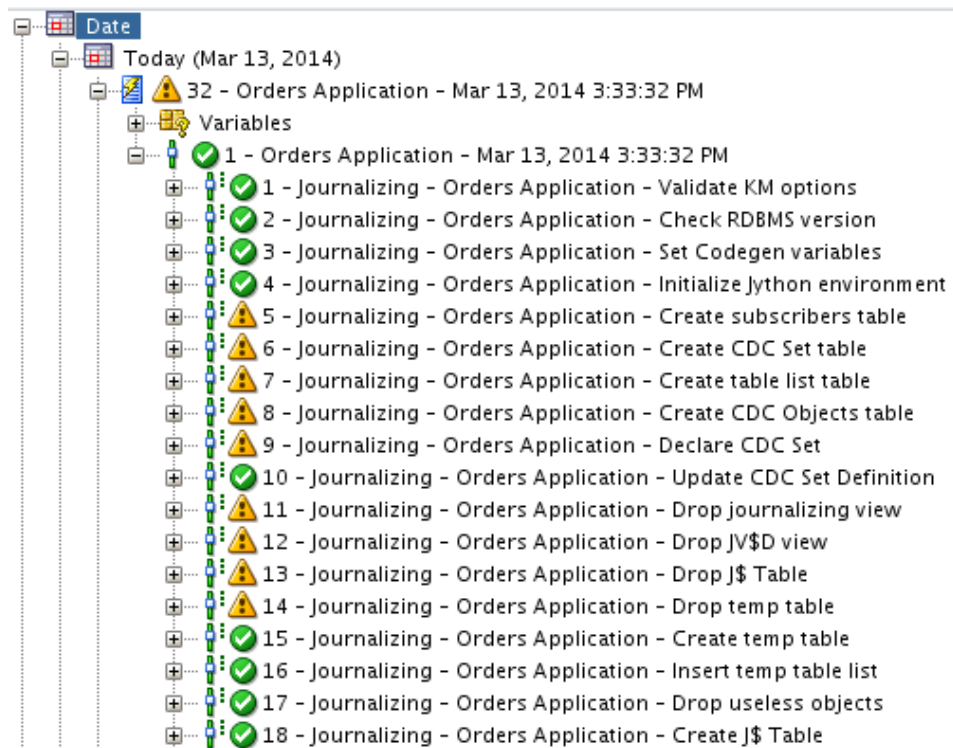
**Figure 9-22 Start Journal**



Leave the default settings and click *OK* in the *Run* window and finally click *OK* to close the *Information* window.

We can monitor the status of the new Session in Operator, go there to verify that all the steps were executed correctly as shown in Figure 9-23 below

Figure 9-23 Session monitoring in Operator



Some Tasks will show a warning status (yellow icon) which is fine as long as Step 1 - *Orders Applications* is successful (green icon).

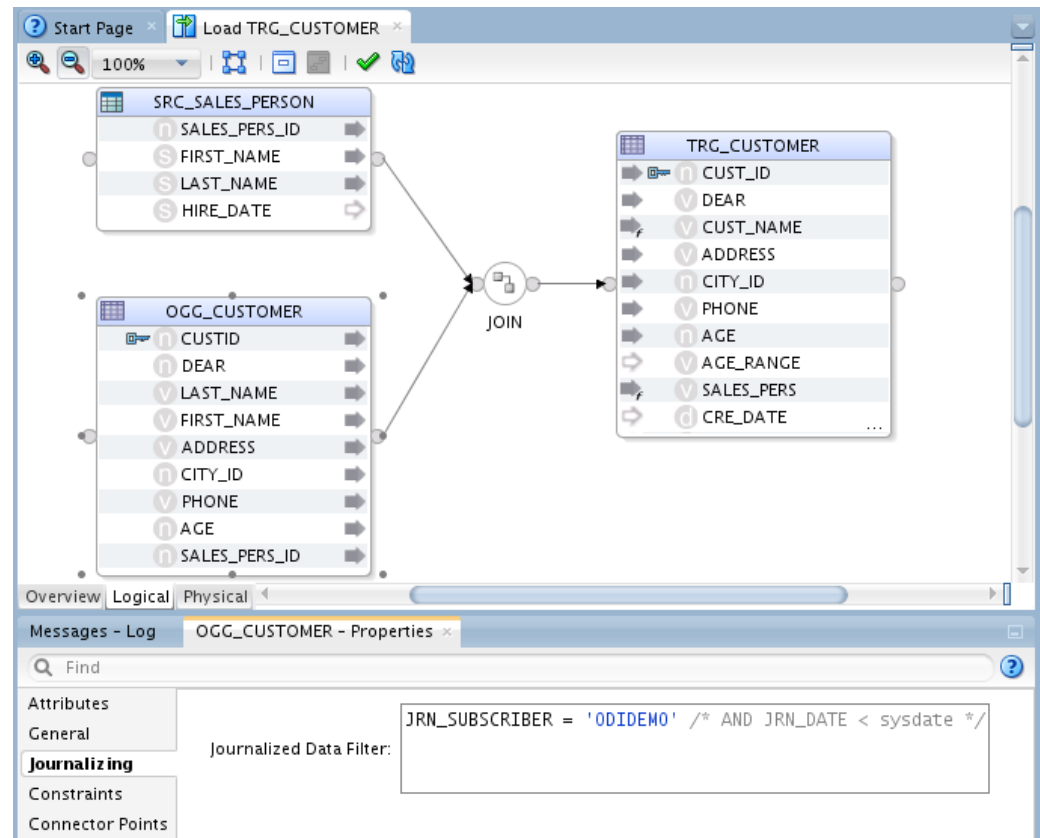
## 9.5 Synchronizing the changed data

Now that the overall configuration of the ODI CDC framework and GoldenGate is done, we can focus our attention on the Mapping that will move the replicated data from GoldenGate into our target table TRG\_CUSTOMER.

### 9.5.1 Load TRG\_CUSTOMER Mapping

1. In Designer, open up the *Projects* accordion
2. Expand the *Sync Data* folder then expand the *Mappings* node
3. Open up the *Load TRG\_CUSTOMER* Mapping, in the Logical tab click on *OGG\_CUSTOMER*. It is worth noting that *OGG\_CUSTOMER* used in this Mapping is coming from the *Orders Applications* source model. It is the actual source table and not the replicated copy used by GoldenGate (located in the *Sales Administration* model). ODI will transparently know which table to use based on the Journalizing settings of the Mapping. In the Properties window, select the Journalizing panel as shown below

Figure 9-24 Load TRG\_CUSTOMER Mapping – Journalizing panel

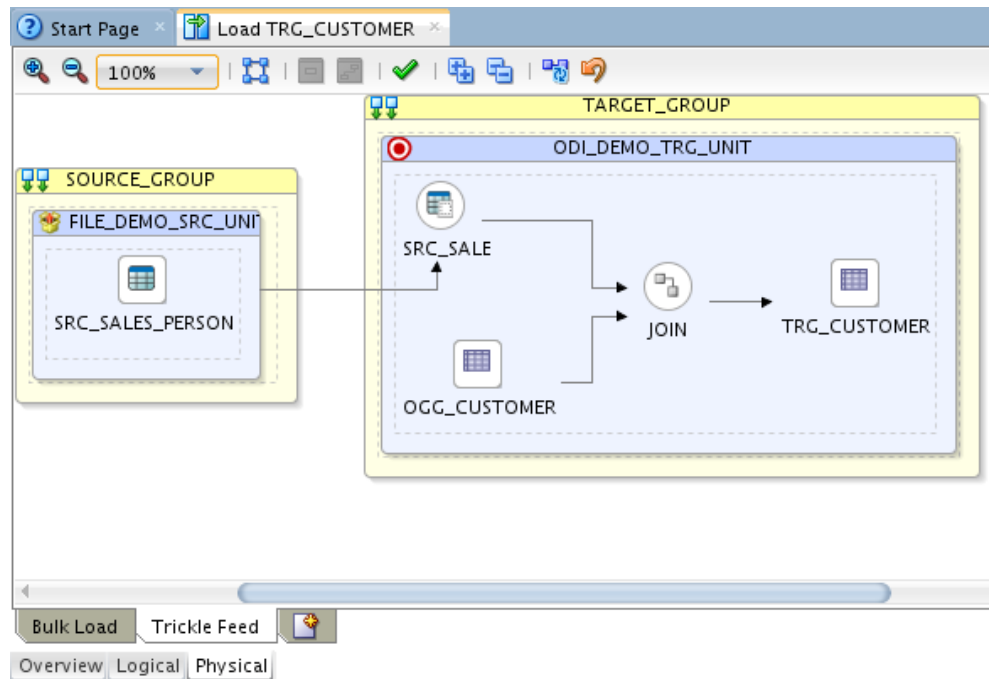


As you can see the *Journalized Data Filter* is set to use the ODIDEMO Subscriber that was previously registered in the ODI CDC framework.

4. Click on the Physical tab to review the Mapping physical designs



Figure 9-25 Load TRG\_CUSTOMER Mapping Physical Tab



5. Two deployment specifications have been created in this getting started environment. *Bulk Load* does a bulk insert into TRG\_CUSTOMER and was previously used to perform an initial load of the target table. *Trickle Feed* performs an incremental load of TRG\_CUSTOMER using the changed data coming from Oracle GoldenGate.
6. Click on *Trickle Feed* to open up the incremental load Physical Design.
7. Click on OGG\_CUSTOMER in ODI\_DEMO\_TRG\_UNIT to open up its Properties. The *Journalized Data Only* parameter has already been checked and ensures ODI connects to its CDC infrastructure to get only the changed records from GoldenGate instead of going directly against the source table.

Figure 9-26 OGG\_CUSTOMER Properties

Messages - Log		OGG_CUSTOMER - Properties
Find		
<b>General</b>		
Attributes	Journalized Data Only:	<input checked="" type="checkbox"/>
Extract Options	Name:	OGG_CUSTOMER
	Description:	

This enables us to create a single logical Mapping design and apply it to multiple physical scenarios:

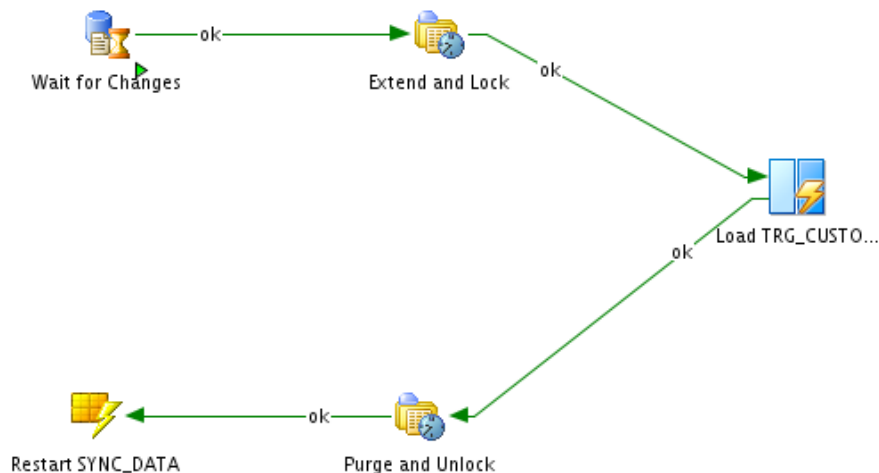
- a. A bulk load use case in which the data is moved from *OGG\_CUSTOMER* in *Orders Applications* to *TRG\_CUSTOMER* using only ODI. The IKM used is the *IKM SQL Control Append* and this is represented by the *Bulk Load* deployment specification
- b. An incremental load use case in which the data is replicated from *OGG\_CUSTOMER* in *Orders Applications* to *OGG\_CUSTOMER* in *Sales Administration* using GoldenGate and then further transformed from *OGG\_CUSTOMER* into *TRG\_CUSTOMER*, both located in *Sales Administration*. The IKM used is the *IKM Oracle Incremental Update* and this is represented by the *Trickle Feed* deployment specification

## 9.5.2 Sync Data Package

As we have seen previously the *Load TRG\_CUSTOMER* Mapping will be moving the changed data into the target table. Before we can run the Mapping we need to review a Package which will be orchestrating the execution of the Mapping along with some additional CDC tasks.

1. Expand the *Packages* node in the *Sync Data* folder
2. Open up the *Sync Data* Package

**Figure 9-27 Sync Data Package Diagram**



This package will wait for new changes to be replicated by Oracle GoldenGate and upon detection it will move and transform the replicated data from the staging table into the target table using Data Integrator.

The ODI Tool step *Wait for Changes* is doing the CDC detection using *OdiWaitForLogData*, it is waiting for some changed records to be published by GoldenGate. Once the detection is successful the next Model step *Extend and Lock* will logically lock the records that will be moved for our ODIDEMO subscriber.

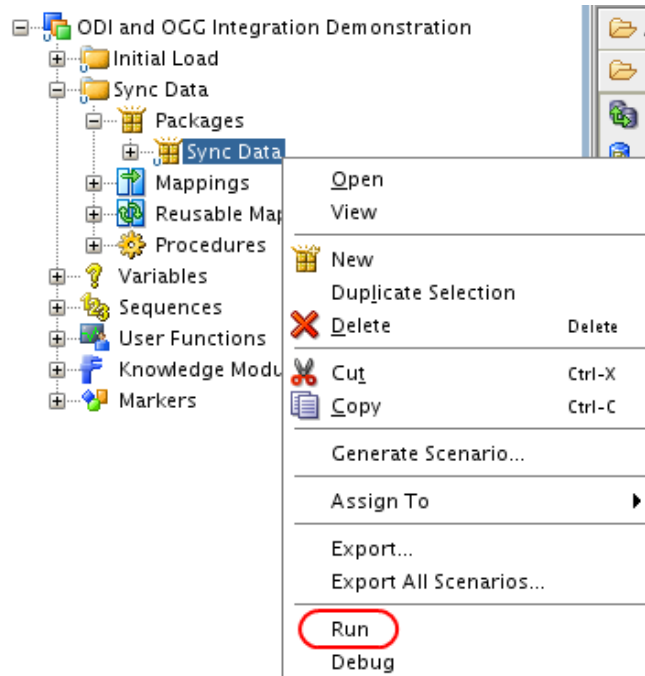
The Mapping *Load TRG\_CUSTOMER* will load the changed data using the *Trickle Feed* physical design and the ODIDEMO Subscriber as we have seen in section 10.3.3.

Once the Mapping is executed the Model step *Purge and Unlock* removes the logical lock on the changed records and purges them for the ODIDEMO Subscriber.

Finally the Package ends with a step executing a Scenario. It allows us to essentially loop through the same Package creating a new Session for every execution. This is a best practice rather than creating the loop directly in the Package as this allows us to better control the overall execution.

3. Right-click on the *Sync Data* Package and select *Run* to start its execution

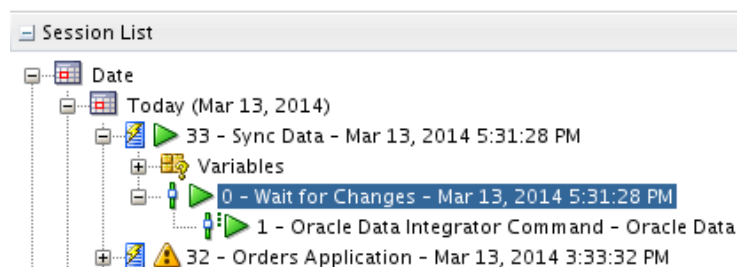
**Figure 9-28 Execute Sync Data Package**



Leave the default settings and click *OK* in the *Run* window and finally click *OK* to close the *Information* window.

4. Go to Operator and review the Session execution. Expand the *Session List* accordion then expand *Date, Today* and the Session named *Sync Data* as shown below

**Figure 9-29 Sync Data Package Execution in Operator**



The Package execution is waiting on the *Wait for Changes* step which is expected. We now have to make some changes in the source data using the *ODI and OGG Demo Client*.

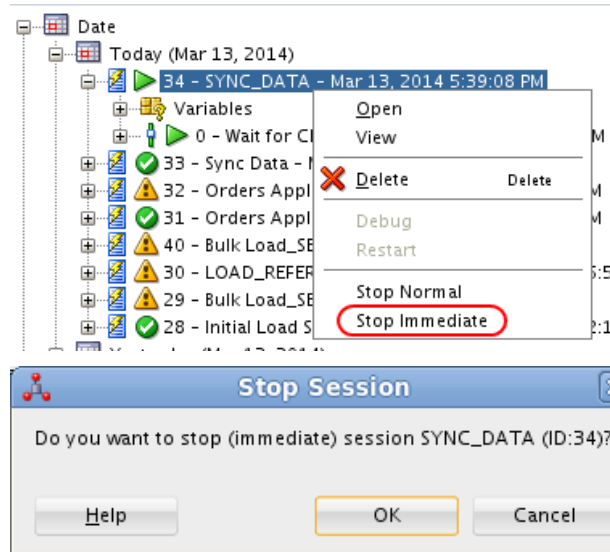
5. Go back to the *ODI and OGG Demo Client* window and make some changes in the source table such as changing the first name of several records. You will then see the changed data being propagated first by GoldenGate from Source to Staging then by Data Integrator from Staging to Target. The changed records are highlighted in yellow as seen in Figure 9-30 below.

**Figure 9-30 ODI and OGG Demo Client**

Source					Staging					Target				
CUSTID	DEAR	LAST_NAME	FIRST_NAME	ADDRESS	CUSTID	DEAR	LAST_NAME	FIRST_NAME	ADDRESS	CUST_ID	DEAR	CUST_NAME	ADDRESS	CITY_ID
101	0	Brendt	Paula	10 Jasper Blvd	101	0	Brendt	Paula	10 Jasper Blvd	101	0	Paula Brendt	10 Jasper Bl	107
102	0	McCarthy	Robin	27 Pasadena	102	0	McCarthy	Robin	27 Pasadena	102	0	Robin Mccar	27 Pasadena	11
103	0	Travis	Peter	7835 Hantor	103	0	Travis	Peter	7835 Hantor	103	0	Peter Travis	7835 Hantor	12
104	0	Larson	Joel	87 Carmel Bl	104	0	Larson	Joel	87 Carmel Blvd	104	0	Joel Larson	87 Carmel Bl	13
105	0	Goldschmidt	Tonio	91 Torre drive	105	0	Goldschmidt	Tonio	91 Torre drive	105	0	Tonio Golds	91 Torre drive	14
106	0	Baker	William	2890 Grant A	106	0	Baker	William	2890 Grant A	106	0	William Baker	2890 Grant	15
107	0	Swenson	Jack	64 Magnatio	107	0	Swenson	Jack	64 Magnatio	107	0	Jack Swenson	64 Magnatio	19
201	0	Sartois	Jeanne	71 rue Rouss	201	0	Sartois	Jeanne	71 rue Rouss	201	0	Jeanne Sartois	71 rue Rouss	25
202	0	Michaud	Philippe	197 impasse	202	0	Michaud	Philippe	197 impasse	202	0	Philippe Mic	197 impasse	23

6. You can go back to Operator to see the new *SYNC\_DATA* Sessions that were triggered by the changed data detection.
7. Optionally, you can stop the Scenario execution from Operator. Right-click on the Scenario currently in Running status (green Play button) and select *Stop Immediate*. Click OK in the *Stop Session* window.

**Figure 9-31 Stop Scenario execution from Operator**



You have successfully moved and transformed changed records using Oracle Data Integrator and Oracle GoldenGate.

---

## 10 Going Further with Oracle Data Integrator

This chapter provides information for going further with Oracle Data Integrator. This chapter includes the following sections:

- Section 10.1, "Summary"
- Section 10.2, "What else can you do with Oracle Data Integrator?"
- Section 10.3, "Learn More"

### 10.1 Summary

Congratulations! You have now completed an ETL project and learned about the fundamentals of Oracle Data Integrator. You have also learned about the ODI Changed Data Capture framework and used it with Oracle GoldenGate.

In this Getting Started guide, you learned how to:

- Create mappings to load the data from the *Orders Application* and *Parameters* applications into the *Sales Administration* data warehouse (Chapter 4, "Working with Mappings")
- Define and implement data integrity rules in the *Orders Application* application (Chapter 5, "Implementing Data Quality Control")
- Sequence your developments (Chapter 6, "Working with Packages")
- Prepare your process for deployment (Chapter 8, "Deploying Integrated Applications")
- Use Oracle Data Integrator and Oracle GoldenGate together (Chapter 9, "Using Oracle Data Integrator with Oracle GoldenGate")

#### 10.1.1 Getting Started Tutorial Solution

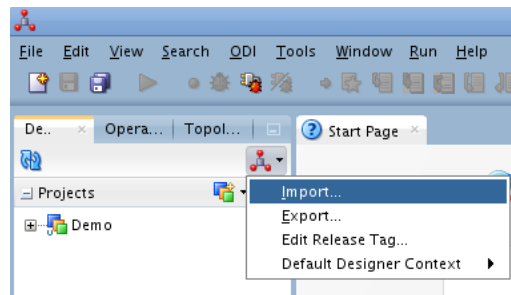
For those whom may wish to review the solution to the ETL Project, an ODI Smart Export is available on OTN. The Smart Export file contains the demonstration mappings, which are created within this document in an XML metadata format.

- The below link needs to be downloaded and extracted into a directory which is accessible from ODI studio.

<http://www.oracle.com/technetwork/middleware/data-integrator/overview/odi12cgettingstartedsolution-2047298.zip>

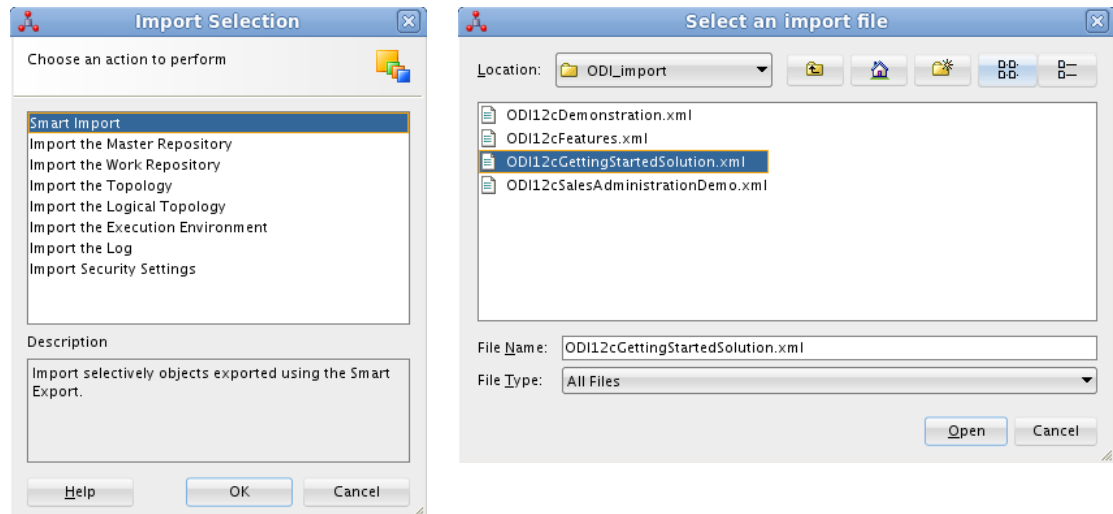
- Connect to the ODI Getting Started login and invoke Smart Import from the Connect Navigator

*Figure 10-1 Starting the ODI Import Wizard*



- Select Smart Import and browse to the Smart Import xml file, **ODI12cGettingStartedSolution.xml**

*Figure 10-2 Starting the ODI Smart Import and Selecting the File Name*



- When prompted enter 'welcome1' as the Export Key
- Continue to click through the import process and the full ODI Getting Started Demonstration is imported.

Please reference the ODI Documentation for further instruction or questions on how to import.

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

## 10.2 What else can you do with Oracle Data Integrator?

You have learned how to use Oracle Data Integrator for a typical Data Warehousing project. But Oracle Data Integrator is capable of addressing any type of data-driven integration, from batch to near-real-time, as for example:

- Data Migration - with or without subsequent replication between the old and the new system
- Point-to-point Data Integration
- Data Replication

Furthermore, in this Getting Started guide you have only seen Oracle Data Integrator connecting to a relational database and files. Oracle Data Integrator can also access and integrate all database systems, Big Data technologies such as Hive, Sqoop or HBase, ERPs and CRMs, mainframes, flat files, LDAP directories, XML data sources, and so forth - all within the same toolset and using the same methodology.

Oracle Data Integrator is the only integration platform that unifies data, event, and service-based integration with a common declarative rules driven approach. It enables the enterprise to present a single view of its Information System, with a single, unified access model.

Some of the benefits that you will find from using Oracle Data Integrator include:

- **Unified integration support:** Oracle Data Integrator is the only integration application software to support data-, event- and service-oriented integration with the same interface. This unique feature allows IT teams to cover all integration needs: batch and real-time, asynchronous and synchronous - regardless of data volumes or latency requirements.
- **Enhanced productivity and a short learning curve:** the declarative rules driven approach is shared throughout Oracle Data Integrator, regardless of the data, event or service orientation of each integration mechanism. With a common use model and shared user interfaces throughout the platform, the learning curve is shortened and productivity is dramatically increased.
- **Shared, reusable metadata:** with a single metadata repository that is fully integrated with all components of Oracle Data Integrator, the consistency of the integration processes is guaranteed. The repository also promotes the reusability of declarative rules for data transformation and data validation across processes.
- **Support for multiple applications:** Oracle Data Integrator is well suited to a broad range of integration projects- ETL, Data Migration, Master data management, Business Activity Monitoring (BAM), Business Process Management (BPM), Business Process Reengineering (BPR), and Web Services integration - implemented using a combination of Data-oriented, Event-oriented, and Service-oriented mechanisms.

## 10.3 Learn More

You can learn more about creating your own integration projects with Oracle Data Integrator in the guides listed in Table 10–1.

**Table 10–1**      **Oracle Data Integrator Documentation**

Document	Description
<i>Oracle Fusion Middleware Installation Guide for Oracle Data Integrator</i>	Provides Oracle Data Integrator installation information including pre-installation requirements and troubleshooting.
<i>Oracle Fusion Middleware Upgrade Guide for Oracle Data Integrator</i>	Provides 12c upgrade information for Oracle Data Integrator.
<i>Oracle Fusion Middleware Developer's Guide for Oracle Data Integrator</i>	Provides guidelines for developers interested in using Oracle Data Integrator for integration projects.
<i>Oracle Fusion Middleware Connectivity and Knowledge Modules Guide for Oracle Data Integrator</i>	Describes Oracle Data Integrator Knowledge Modules and technologies and how to use them in integration projects.
<i>Oracle Fusion Middleware Knowledge Module Developer's Guide for Oracle Data Integrator</i>	Describes how to develop your own Knowledge Modules for Oracle Data Integrator.

If you have any questions, comments or feedback regarding the Getting Started Demonstration and Environment, feel free to discuss on the **ODI OTN Forum**:

[https://forums.oracle.com/community/developer/english/business\\_intelligence/system\\_management\\_and\\_integration/data\\_integrator](https://forums.oracle.com/community/developer/english/business_intelligence/system_management_and_integration/data_integrator)

The Oracle Data Integrator home page on the Oracle Technology Network also provides the following resources to learn more about other features of Oracle Data Integrator:

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

- View the *Oracle by Example Series for ODI*. The Oracle by Example (OBE) series provides step-by-step instructions on how to perform a variety of tasks using Oracle Data Integrator Suite.
- You can find all Oracle Data Integrator **documentation** on the Oracle Data Integrator documentation page on the Oracle Technology Network, at:

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

To learn more about the new features that have been introduced in Oracle Data Integrator 12c, see "What's New in Oracle Data Integrator?" in the *Oracle Fusion Middleware Developer's Guide for Oracle Data Integrator* and the Release Notes.

Thank you for choosing Oracle Data Integrator



**Oracle Corporation, World Headquarters**

500 Oracle Parkway  
Redwood Shores, CA 94065, USA

**Worldwide Inquiries**

Phone: +1.650.506.7000  
Fax: +1.650.506.7200

---

**CONNECT WITH US**

[blogs.oracle.com/oracle](https://blogs.oracle.com/oracle)



[facebook.com/oracle](https://facebook.com/oracle)



[twitter.com/oracle](https://twitter.com/oracle)



[oracle.com](https://oracle.com)

Copyright © 2018, Oracle and/or its affiliates. All rights reserved. This document is provided *for* information purposes only, and the contents hereof are subject to change without notice. This document is not warranted to be error-free, nor subject to any other warranties or conditions, whether expressed orally or implied in law, including implied warranties and conditions of merchantability or fitness for a particular purpose. We specifically disclaim any liability with respect to this document, and no contractual obligations are formed either directly or indirectly by this document. This document may not be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without our prior written permission.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group. 0116

Oracle Data Integrator 12c Getting Started Guide  
December 2018  
Author: Oracle



Oracle is committed to developing practices and products that help protect the environment