Oracle® Retail Data Extractor for Merchandising

Installation Guide
Release 16.0.2
F11610-01

November 2018

Oracle® Retail Data Extractor for Merchandising Installation Guide, Release 16.0.2

Value-Added Reseller (VAR) Language

Oracle Retail VAR Applications

The following restrictions and provisions only apply to the programs referred to in this section and licensed to you. You acknowledge that the programs may contain third party software (VAR applications) licensed to Oracle. Depending upon your product and its version number, the VAR applications may include:

(i) the **MicroStrategy** Components developed and licensed by MicroStrategy Services Corporation (MicroStrategy) of McLean, Virginia to Oracle and imbedded in the MicroStrategy for Oracle Retail Data Warehouse and MicroStrategy for Oracle Retail Planning & Optimization applications.

(ii) the **Wavelink** component developed and licensed by Wavelink Corporation (Wavelink) of Kirkland, Washington, to Oracle and imbedded in Oracle Retail Mobile Store Inventory Management.

(iii) the software component known as **Access Via™** licensed by Access Via of Seattle, Washington, and imbedded in Oracle Retail Signs and Oracle Retail Labels and Tags.

(iv) the software component known as **Adobe Flex™** licensed by Adobe Systems Incorporated of San Jose, California, and imbedded in Oracle Retail Promotion Planning & Optimization application.

You acknowledge and confirm that Oracle grants you use of only the object code of the VAR Applications. Oracle will not deliver source code to the VAR Applications to you. Notwithstanding any other term or condition of the agreement and this ordering document, you shall not cause or permit alteration of any VAR Applications. For purposes of this section, "alteration" refers to all alterations, translations, upgrades, enhancements, customizations or modifications of all or any portion of the VAR Applications including all reconfigurations, reassembly or reverse assembly, re-engineering or reverse engineering and recompilations or reverse compilations of the VAR Applications or any derivatives of the VAR Applications. You acknowledge that it shall be a breach of the agreement to utilize the relationship, and/or confidential information of the VAR Applications for purposes of competitive discovery.

The VAR Applications contain trade secrets of Oracle and Oracle's licensors and Customer shall not attempt, cause, or permit the alteration, decompilation, reverse engineering, disassembly or other reduction of the VAR Applications to a human perceivable form. Oracle reserves the right to replace, with functional equivalent software, any of the VAR Applications in future releases of the applicable program.

Contents

Send Us Your Comments

Oracle Retail Data Extractor for Merchandising, Installation Guide, Release 16.0.2

Oracle welcomes customers' comments and suggestions on the quality and usefulness of this document.

Your feedback is important, and helps us to best meet your needs as a user of our products. For example:

- Are the implementation steps correct and complete?
- Did you understand the context of the procedures?
- Did you find any errors in the information?
- Does the structure of the information help you with your tasks?
- Do you need different information or graphics? If so, where, and in what format?
- Are the examples correct? Do you need more examples?

If you find any errors or have any other suggestions for improvement, then please tell us your name, the name of the company who has licensed our products, the title and part number of the documentation and the chapter, section, and page number (if available).

> **Note:** Before sending us your comments, you might like to check that you have the latest version of the document and if any concerns are already addressed. To do this, access the Online Documentation available on the Oracle Technology Network Web site. It contains the most current Documentation Library plus all documents revised or released recently.

Send your comments to us using the electronic mail address: retail-doc_us@oracle.com

Please give your name, address, electronic mail address, and telephone number (optional).

If you need assistance with Oracle software, then please contact your support representative or Oracle Support Services.

If you require training or instruction in using Oracle software, then please contact your Oracle local office and inquire about our Oracle University offerings. A list of Oracle offices is available on our Web site at www.oracle.com.

Preface

Oracle Retail Installation Guides contain the requirements and procedures that are necessary for the retailer to install Oracle Retail products.

Audience

This Installation Guide is written for the following audiences:

- Database administrators (DBA)
- System analysts and designers
- Integrators and implementation staff

Customer Support

To contact Oracle Customer Support, access My Oracle Support at the following URL:

https://support.oracle.com

When contacting Customer Support, please provide the following:

- Product version and program/module name
- Functional and technical description of the problem (include business impact)
- Detailed step-by-step instructions to re-create
- Exact error message received
- Screen shots of each step you take

Review Patch Documentation

When you install the application for the first time, you install either a base release (for example, 16.0) or a later patch release (for example, 16.0.2). If you are installing the base release or additional patch releases, read the documentation for all releases that have occurred since the base release before you begin installation. Documentation for patch releases can contain critical information related to the base release, as well as information about code changes since the base release.

Improved Process for Oracle Retail Documentation Corrections

To more quickly address critical corrections to Oracle Retail documentation content, Oracle Retail documentation may be republished whenever a critical correction is needed. For critical corrections, the republication of an Oracle Retail document may at times **not** be attached to a numbered software release; instead, the Oracle Retail document will simply be replaced on the Oracle Technology Network Web site, or, in the case of Data Models, to the applicable My Oracle Support Documentation container where they reside.

This process will prevent delays in making critical corrections available to customers. For the customer, it means that before you begin installation, you must verify that you have the most recent version of the Oracle Retail documentation set. Oracle Retail documentation is available on the Oracle Technology Network at the following URL:

http://www.oracle.com/technetwork/documentation/oracle-retail-100266.html

An updated version of the applicable Oracle Retail document is indicated by Oracle part number, as well as print date (month and year). An updated version uses the same part number, with a higher-numbered suffix. For example, part number E123456-**02** is an updated version of a document with part number E123456-**01**.

If a more recent version of a document is available, that version supersedes all previous versions.

Oracle Retail Documentation on the Oracle Technology Network

Oracle Retail product documentation is available on the following web site:

http://www.oracle.com/technetwork/documentation/oracle-retail-100266.html

(Data Model documents are not available through Oracle Technology Network. You can obtain them through My Oracle Support.)

Conventions

Navigate: This is a navigate statement. It tells you how to get to the start of the procedure and ends with a screen shot of the starting point and the statement "the Window Name window opens."

```
This is a code sample
    It is used to display examples of code
```

1

Preinstallation Tasks

This release of Retail Data Extractor incorporates optional interfaces with these Oracle Retail products which can be sources for the data warehouse: Oracle Retail Merchandising System (RMS), Oracle Retail Invoice Matching (ReIM), and Oracle Retail Price Management (RPM). Additionally, the data warehouse can also operate as a standalone product and be fed from other legacy systems. If Oracle Retail applications are used as the source systems, follow the requirements in the installation guides for each of these applications.

> **Note:** Oracle Retail assumes that the retailer has applied all required fixes for supported compatible technologies.

Roadmap for Installing RDE

This guide explains how you can install and set up the RDE applications, along with the required and optional software.

The instructions in this guide assume knowledge of application servers, databases, and pplication installation or administration, and are intended for system administrators and experienced IT personnel. Before carrying out any of these activities, ensure that you understand UNIX commands (including shell configuration and scripting), directory operations, and subversion links.

In order to implement RDE for production, you must perform the following installation tasks in a sequence:

Preinstallation Tasks for full install

- Plan your environment for implementation capacity planning
- Check Supported Configurations and Requirements
- Create Unix user accounts to install software
- Create staging directory and extract installer files
- Setting up database
- Partitioning strategy
- Oracle Data Integrator Configuration Tasks
- Setting up Password stores
- Setting up the installation Properties File

Installation

- Overview of the Installation Process

Post Installation Configuration

- ODI Configuration
- Batch configuration

Implementation Capacity Planning

There is significant complexity involved in the deployment of Oracle Retail applications, and capacity planning is site specific. Oracle Retail strongly suggests that before installation or implementation you engage your integrator (such as the Oracle Retail Consulting team) and hardware vendor to request a disk sizing and capacity planning effort.

Sizing estimates are based on a number of factors, including the following:

- Workload and peak concurrent users and batch transactions
- Hardware configuration and parameters
- Amount of data
- Application features utilized
- Length of time history is retained

Additional considerations during this process include your high availability needs as well as your backup and recovery methods.

Check Supported Database Server Requirements

General requirements for a database server running Retail Data Extractor include:

Supported on	Versions Supported
Database Server OS	OS certified with Oracle Database 12cR1 (12.1) Enterprise Edition. Options are: - Oracle Linux 6 and 7 for x86-64 (actual hardware or Oracle virtual machine). - Red Hat Enterprise Linux 6 and 7 for x86-64 (actual hardware or Oracle virtual machine). - AIX 7.1 (actual hardware or LPARs) - Solaris 11.x Sparc (actual hardware or logical domains)

Supported on	Versions Supported
Database Server 12cR1	Oracle Database Enterprise Edition 12cR1 (12.1.0.2) with the following specifications: **Components:** ▪ Enterprise Edition ▪ Example CD **Oneoffs:** ▪ 20846438: ORA-600 [KKPAPXFORMFKK2KEY_1] WITH LIST PARTITION ▪ 19623450: MISSING JAVA CLASSES AFTER UPGRADE TO JDK 7 ▪ 20406840: PROC 12.1.0.2 THROWS ORA-600 [17998] WHEN PRECOMPILING BY 'OTHER' USER ▪ 20925154: ORA-39126: WORKER UNEXPECTED FATAL ERROR IN KUPW$WORKER GATHER_PARSE_ITEMS JAVA ▪ 19672263: Patch 19672263: GTT SESSION LEVEL STATISTICS RETURNS ORA-20006 **RAC only:** ▪ 21260431: APPSST 12C : GETTING ORA-4031 AFTER 12C UPGRADE ▪ 21373473: INSTANCE TERMINATED AS LMD0 AND LMD2 HUNG FOR MORE THAN 70 SECS **Other components:** ▪ Perl interpreter 5.0 or later ▪ X-Windows interface ▪ JDK 1.7
Oracle Data Integrator	ODI 12C

> **Note:** By default, JDK is at 1.6. After installing the 12.1.0.2 binary, apply patch 19623450. Then follow the instructions on Oracle Database Java Developer's Guide 12c Release 1 to upgrade JDK to 1.7. The Guide is available here:
>
> **http://docs.oracle.com/database/121/JJDEV/chone.htm#JJDEV01000**

Check Supported ODI Requirements

Variable	Description
Server OS	Operating systems certified include: ▪ Oracle Linux 6 and 7for x86-64 (Actual hardware or Oracle virtual machine). ▪ Red Hat Enterprise Linux 6 and 7 for x86-64 (Actual hardware or Oracle virtual machine). ▪ AIX 7.1 (actual hardware or LPARs) ▪ Solaris 11.x Sparc (actual hardware or logical domains)
ODI Studio UI	ODI Studio (UI) is NOT supported on Solaris and AIX Operating Systems. Please refer to the "Installing ODI Files for Retail Data Extractor on Windows section" in the "Known Issues/Limitations" chapter in case you are planning to install ODI Studio (UI) on either of these two OS. For Windows and Linux OS, ODI Studio (UI) is supported with 1.7 + .
Oracle Data Integrator 11g	Oracle Data Integrator 12C **Components:** ▪ Oracle Data Integrator 12.2.1.3.0 **Options:** ▪ Complete

Supported Oracle Retail Products

Requirement	Version
Oracle Retail Merchandising System (RMS)/Oracle Retail Oracle Retail Sales Audit (ReSA)	16.0.2
Oracle Retail Invoice Matching (ReIM)	16.0.2
Oracle Retail Price Management (RPM)	16.0.2
Oracle Retail Insights	16.0.2

Create a UNIX User Account to Install the Software

It is possible that ODI has been installed on different host. For installing Retail Data Extractor files for ODI, you must be on the same host where the ODI product has been installed. Any user can install Retail Data Extractor files for ODI.

> **Note**: If ODI is installed on Windows, you cannot use the installer to copy ODI -related files. You must copy files manually according to the instruction given. You can also install MMHOME files on Windows. Please refer to Known Issues/Limitations chapter for more details

Create a Staging Directory for Retail Data Extractor Database Files

You may install all components of Retail Data Extractor on one host, or you may install components across multiple hosts. The files to be installed are copied locally only, so you must be logged into the target host to do the installation.

1. Log in to the server from which you want to install one or more components of Retail Data Extractor.

2. Create a staging directory for the Retail Data Extractor installation software. There should be a minimum of 800 MB disk space available in this location.

3. Copy the orde16application.zip file from the Retail Data Extractor 16.0.2 release to the staging directory. This is referred to as STAGING_DIR when installing Retail Data Extractor.

4. Change directories to STAGING_DIR and extract the zip file. This creates an orde/installer subdirectory under STAGING_DIR

2

Preinstallation Tasks - Full

Database Installation Tasks

This section describes the pre-requisites tasks required for a full database installation.

Retail Data Extractor Full Release

Retail Data Extractor 16.0.2 is a full baseline installation only.

It is assumed that Oracle Database 12cR1 (12.1.0.2), with appropriate patches, has already been installed. If not, refer to "Check Supported Database Server Requirements" in Chapter 1 before proceeding.

> **Note:** Become familiar with the Retail Data Extractor application in a development environment before setting up a production system. The following instructions are recommended for development and test environments only. When implementing Retail Data Extractor for a production environment, refer to capacity planning information to determine size requirements for table spaces, tables, and indexes. The installation scripts provided must be modified accordingly.

If you chose to install the Extractor in a separate database, follow the example in Appendix B: CreateDatabase Instance Using an Oracle GenericTemplate to create a new database to create a dedicated database.

If a database has already been created, it is necessary to review the contents of this section to determine if all database components have been installed and configured properly.

> **Note:** Review the "Patching Procedures" chapter to understand the Retail Patching Strategy and the ORPatch concepts before moving on to the next Topic.

Create Staging Directory for Retail Data Extractor Installer

To create the staging directory for Retail Data Extractor installer, complete the following steps.

> **Note:** The same installer can be used to install multiple Retail Data Extractor components. If you are installing any of the Retail Data Extractor components (Database, ODI) on the same server, they can use the same installer and this step does not need to be repeated.

> **Note:** The installation of Retail Data Extractor is carried out from application server. Ensure that the 12cR1 client is installed in application server in order to connect to the database.

1. Log into the application server as a user that can connect to the Retail Data Extractor database.

2. Create a staging directory for the Retail Data Extractor installation software.

3. Copy the orde16application.zip file from the RDE 16.0.2 release to the staging directory. This is referred to as STAGING_DIR when installing Retail Data Extractor database software.

4. Change directories to STAGING_DIR and extract the orde16application.zip file. This creates an orde/installer/ subdirectory under STAGING_DIR.

Create the Retail Data Extractor Database

It is assumed that Oracle Database 12cR1, with appropriate patches, has already been installed. If not, refer to "Check Supported Database Server Requirements" in Chapter 1 before proceeding. Additionally, *STAGING_DIR* in this section refers to the directory created in the section, "Create a Staging Directory for Retail Data Extractor Database Files," in Chapter 1.

If a database has already been created, it is necessary to review the contents of this section to determine if all database components have been installed and configured properly.

If a database instance has not been created, create one following the process in "Appendix: Create the Database Instance Using an Oracle Generic Template."

Create Retail Data Extractor Tablespaces

Complete the following steps.

1. Change directories to <STAGING_DIR>/orde/installer/create_db

2. Log in to SQL*Plus as SYSDBA and execute:

> **Note:** In the create_rde_tablespaces.sql script, replace <data_file_path> with the actual physical path before executing this script.

> **Note:** If you receive the message, "ORA-01543: table space table space name already exists," the tablespace is already in the database. You can ignore it.

```
SQL>@create_rde_tablespaces.sql
```

3. Review create_rde_tablespaces.log for errors and correct as needed.

Create Retail Data Extractor Schema Owners

Before running the installer, run orde/installer/create_db/pre-install-schema-setup.sh to create these required schemas, db link, and external directory:

What	Description	Default Name
RDE RMS	Schema (in the RMS database) with read-only access to the RMS Master Schema (used as RDE Batch User. RDE_BE batch user decommissioned).	RDE_RMS01
Data Mart	Schema (in the RDE database) with read-only access to the Batch schema.	RDE_DM01
ODI MREP	This ODI Master Repository user includes ODI job definitions and topology information.(These will be created using RCU utility)	ODI_RDE_MREP.
ODI WREP	This ODI Work Repository user includes ODI job status information and in ODI 12.2.1.3.0 it is self-contained in Master repository schema.(These will be created using RCU utility)	ODE_RDE_WREP
BATCH_LINK_TO_RMS_DB	Connects Batch schema in the RDE database to the RDE RMS schema in the RMS database	BATCH_LINK_TO_RMS_${RMS_DATABASE}
External directory	Directory defined in the RDE RMS schema and used to access the 'data' directory.	XTERN_DATA_DIR

> **Note:** RMS and RDE databases and hosts may be the same or different. You may run the pre-install-schema-setup.sh script from a completely different machine as long as you have sqlplus installed.

The pre-install-schema-setup.sh script presents text menus to enter your database host or hosts, your RMS and RDE databases, and the password or passwords for the 'sys as sysdba' user which creates the schemas. If you accept the default names and your RMS and RMS databases and hosts are the same then you only need to set three values before creating the schemas.

The following image shows how the schemas are related:
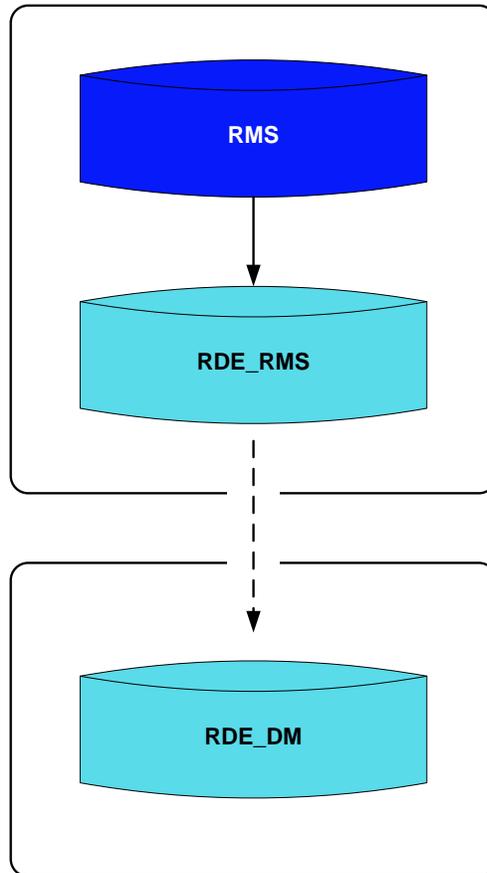
Instructions:

1. Change your working directory to orde/installer/create_db/.

   `chmod +x pre-install-schema-setup.sh.`

2. Run ./pre-install-schema-setup.sh.

   First, you will be shown a list of 'Values' followed by a menu with numbers. The minimum values which need to be set are #1, #3, and #5; the script initially will default the values for #2, #4, and #6, after which you may change the default values.

   To redisplay the menu at any time press enter. Here is what the initial menu looks like:

```
create_db$ ./pre-install-schema-setup.sh
##########################################################################
#### Values
RMS_DATABASE_HOST
RDE_DATABASE_HOST
RMS_DATABASE              ?
RDE_DATABASE              ?
RMS_SYS_PASSWORD          ?
RDE_SYS_PASSWORD          ?
DM_USERNAME               RDE_DM01
DM_PASSWORD               retail
DM_TMPTBLSP               temp
ODI_MREP_USERNAME         ODI_RDE_MREP
ODI_MREP_PASSWORD         retail
ODI_MREP_TMPTBLSP         temp
```

```
ODI_WREP_USERNAME        ODI_RDE_WREP
ODI_WREP_PASSWORD        retail
ODI_WREP_TMPTBLSP        temp
RDE_RMS_USERNAME         RDE_RMS01
RDE_RMS_PASSWORD         retail
RDE_RMS_DEFTBLSP         retail_data
RDE_RMS_TMPTBLSP         temp
RDE_RMS_IDXTBLSP         retail_index
BATCH_TMPTBLSP           temp
BATCH_LINK_TO_RMS_DB     BATCH_LINK_TO_RMS_?
ODI_HOME                 /u01/product/fmw/wls_odi/odi/agent
###############################################################################
1) RMS_DATABASE_HOST                         14) ODI_WREP_PASSWORD
2) RDE_DATABASE_HOST                         15) ODI_WREP_TMPTBLSP
3) RMS_DATABASE                              16) RDE_RMS_USERNAME
4) RDE_DATABASE                              17) RDE_RMS_PASSWORD
5) RMS_SYS_PASSWORD                          18) RDE_RMS_DEFTBLSP
6) RDE_SYS_PASSWORD                          19) RDE_RMS_TMPTBLSP
7) DM_USERNAME                               20) RDE_RMS_IDXTBLSP
8) DM_PASSWORD
9) DM_TMPTBLSP
10) ODI_MREP_USERNAME                        23) BATCH_TMPTBLSP
11) ODI_MREP_PASSWORD                        24) BATCH_LINK_TO_RMS_DB
12) ODI_MREP_TMPTBLSP                        25) ODI_HOME
13) ODI_WREP_USERNAME                        26) Input-Done
#?
```

3. Enter the number of the value you wish to change and press enter. Then enter the value and press enter. When you have finished setting the values enter the number of the 'Input-Done' menu item to advance to the action menu.

The action menu looks similar to the following:

```
#### PRESS ENTER TO REDISPLAY MENU ####
 1) Reset_Values_to_Defaults
 2) Display_Values
 3) Display_Schemas
 4) Display_Some_Properties
 5) Set_Individual_Values
 6) Create_RDE_Roles_and_Grant_Access
 7) Create_RDE_DM_User
 8) Create_RDE_ODI_MREP_User
 9) Create_RDE_ODI_WREP_User

11) Create_RDE_RMS_User
12) Create_RDE_DB_Link_to_RMS_DB
13) Create_External_Dir_on_RMS_DB
14) Quit
#?
```

The eight actions beginning with the word 'Create' are the only ones which actively change your databases. If you wish to reset your values to defaults, choose #1; to set or reset individual values as you did when you first started the program, choose #5. The options beginning with the word 'Display' may help you document your settings.

The script is written to minimize reduce double-entry, to cascade higher-level values to lower-levels, and to aid consistency. For example, when you enter the RMS_DATABASE name the default name of BATCH_LINK_TO_RMS_DB is changed to include that same value for clarity.

DBA's may decide to extract, edit, and run the SQL code themselves in which case it will be found between 'here-doc' EOF markers in the script.

Most users should choose to enter their values for the database hosts, databases, and sys passwords. They will then run all of the 'Create' steps in order by pressing the menu number and Enter.

Do not forget to collect your documentation by running the 'Display' menus.

> **Note:** The script may be run, without altering the databases, before, during, or after the install merely to set values and print out the settings for documentation using the 'Display' choices. As long as you do not run the 'Create' steps the databases are not touched.

While running, the script stores its values in a directory of '.pre-install-schema-setup'. As soon as the script begins to exit it removes password entries from that directory and prints a summary '.pre-install-schema-setup.txt' file; i.e. your data-at-rest has no passwords in it. If you wish to obtain a copy of your settings with the passwords then use the menus with 'Display' in the names and copy them from the display.

> **Note:**
>
> RDE-16.0.2-is a full-installer release. As part of 16.0.2 full-release, RDE_BE user is decommissioned and made RDE_RMSxx schema is a bridging schema, with SELECT grants on RMS(as like earlier 16.0 base) , but will also act as RDE_BE (RDE_BATCH) user for RDE_DM schema. i.e RDE_RMS acts as a BATCH user for all data access from SOURCE system (RMS) as well from Data-Mart (RDE_DM).
>
> Since the installer not supporting upgrade, on-premises customers not going to do full-installation for DB side can manually make RDE_RMSxx existing schema as BATCH bridging schema by providing the RA_BATCH role to the RDE_RMSxx schema and run the **RABATCH_SYNONYM.sql** utility file located under "rde_db/radm/utility_files" directory. The input to the script is: USER: **RDE_DMxx**(DataMart schema) and RABATCHUSER=**RDE_RMSxx**.
>
> User should run as SYSDBA to make these changes in data-mart schema.
>
> Example : Providing quota & grants to existing RDE_RMSxx user.
>
> alter user &RDE_RMSxx quota unlimited on DM_DIM_DATA;
>
> alter user &RDE_RMSxx quota unlimited on DM_DIM_INDEX;
>
> alter user &RDE_RMSxx quota unlimited on DM_FACT_DATA;
>
> alter user &RDE_RMSxx quota unlimited on DM_FACT_INDEX;

```
alter user &RDE_RMSxx quota unlimited on
RETAIL_DATA;

alter user &RDE_RMSxx quota unlimited on
RETAIL_INDEX;

alter user &RDE_RMSxx quota unlimited on USERS;

grant resource, alter session, create library, create procedure,
create sequence, create session, create synonym,   create
table, create trigger, create view, create type, create job to
&RDE_RMSxx;

grant ra_batch to &RDE_RMSxx;

grant select any dictionary to &RDE_RMxx;

grant execute on dbms_rls to &RDE_RMSxx;

grant execute on sys.dbms_system to &RDE_RMSxx;

grant create any index      to &RDE_RMSxx;

grant create any synonym    to &RDE_RMSxx;

grant execute any procedure to &RDE_RMSxx;

grant create any view       to &RDE_RMSxx;
```

Application Installation Tasks

Creating ODI Master and Work Repository

Blank Master and Work repositories must be created via RCU as a pre-requisite before
starting the Retail Data Extractor install. Master and Work Repositories should be created
in a single database schema via the RCU as per the ODI installation (ODI 12.2.1.3.0). This
is updated under ODI Configuration Tasks.

> **Note:** Please refer Oracle® Fusion Middleware Installing
> and Configuring Oracle Data Integrator chapter for creating
> the Master and Work Repository manually.

1. Change directory to $ORACLE_HOME/oracle_common/bin

   ```
   cd $ORACLE_HOME/oracle_common/bin
   ```

2. Run the command ./rcu

3. From the Welcome screen, click **Next**.
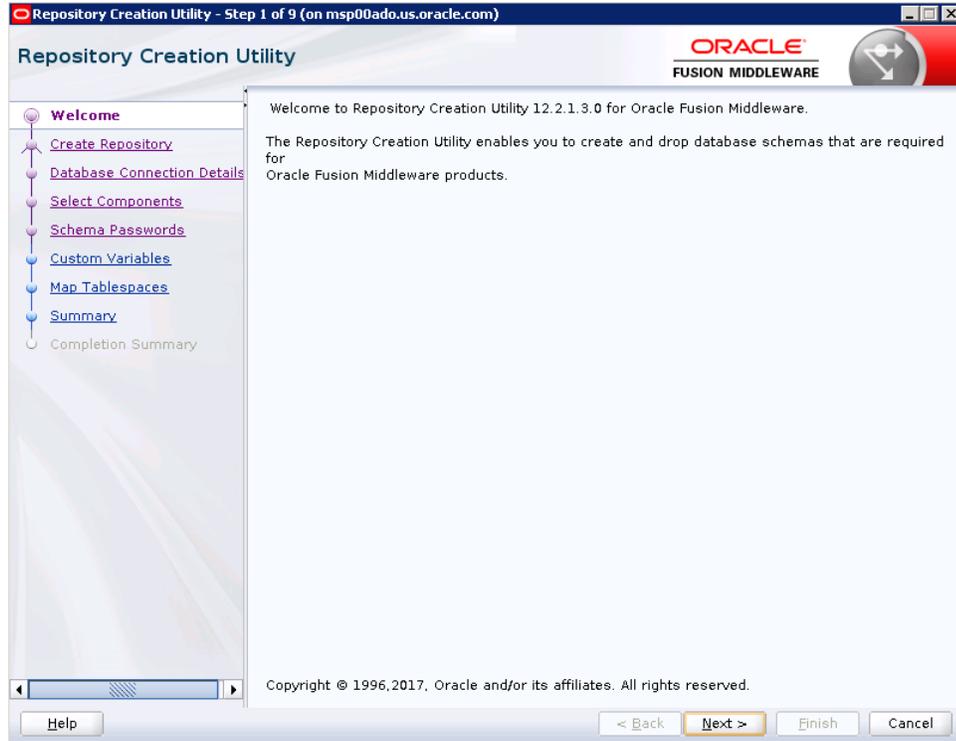
4. Select Create Repository and System Load and Product Load. Click **Next**.
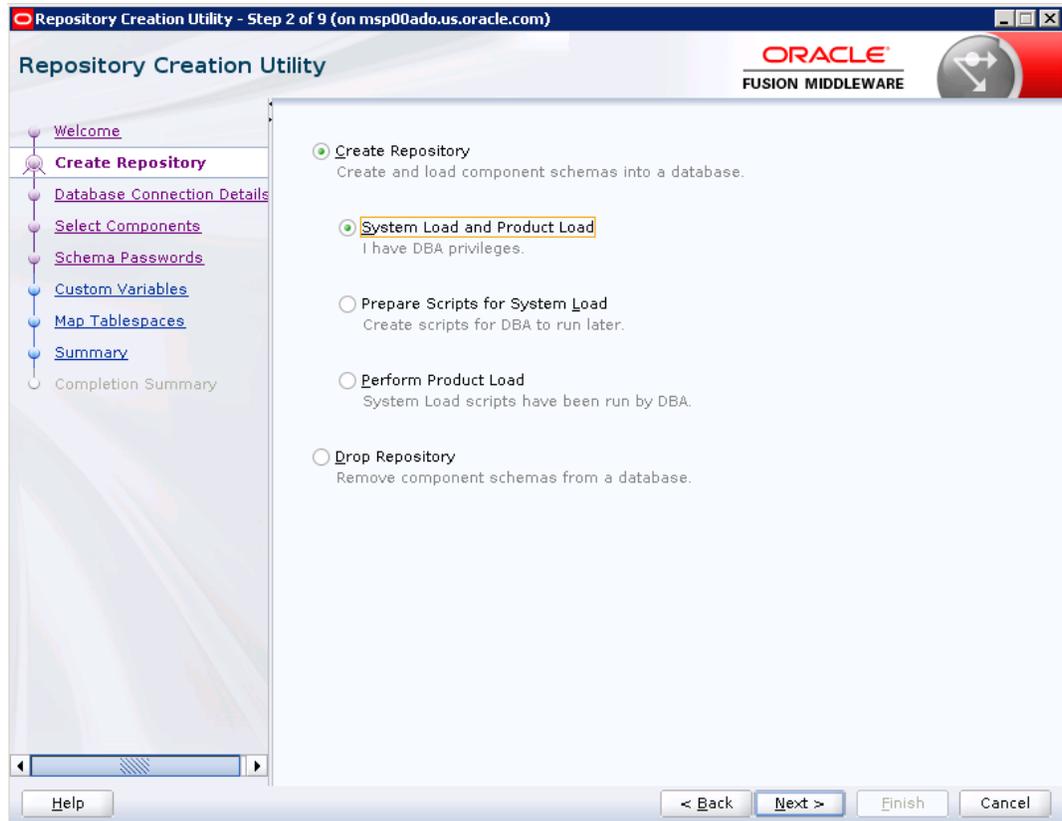
5. Update the Database details and click **Next**.
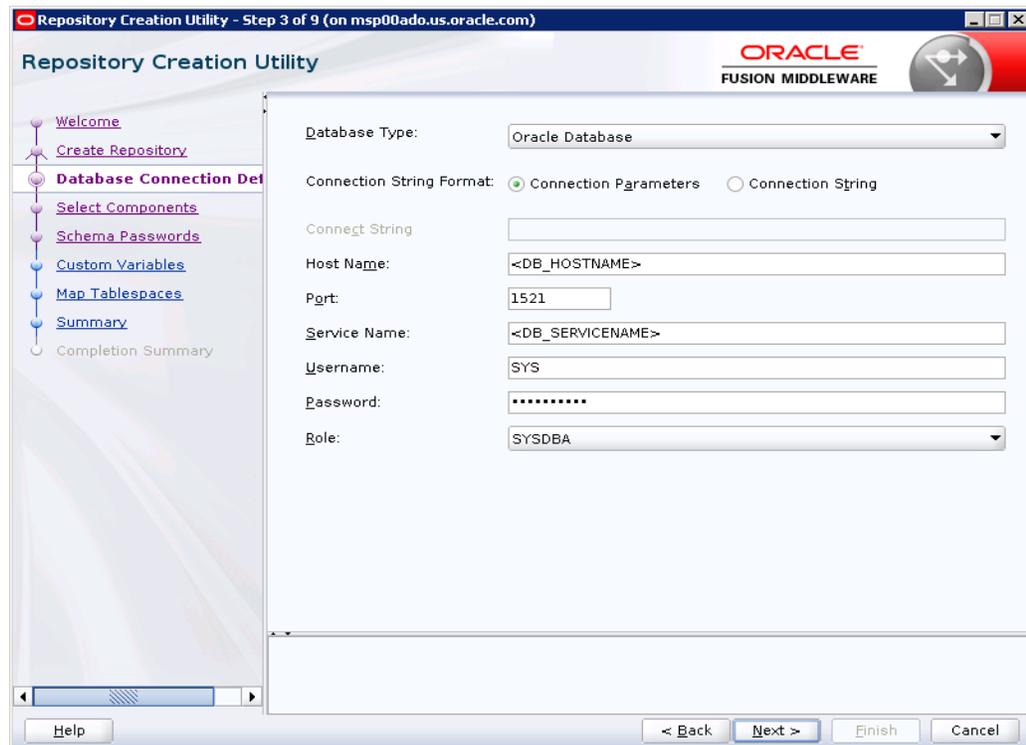
6. Update the new prefix and select AS Common Schemas and ODI Master and Work Repository. Create Prefix as per your choice. Click **Next**.

Example: RDE16
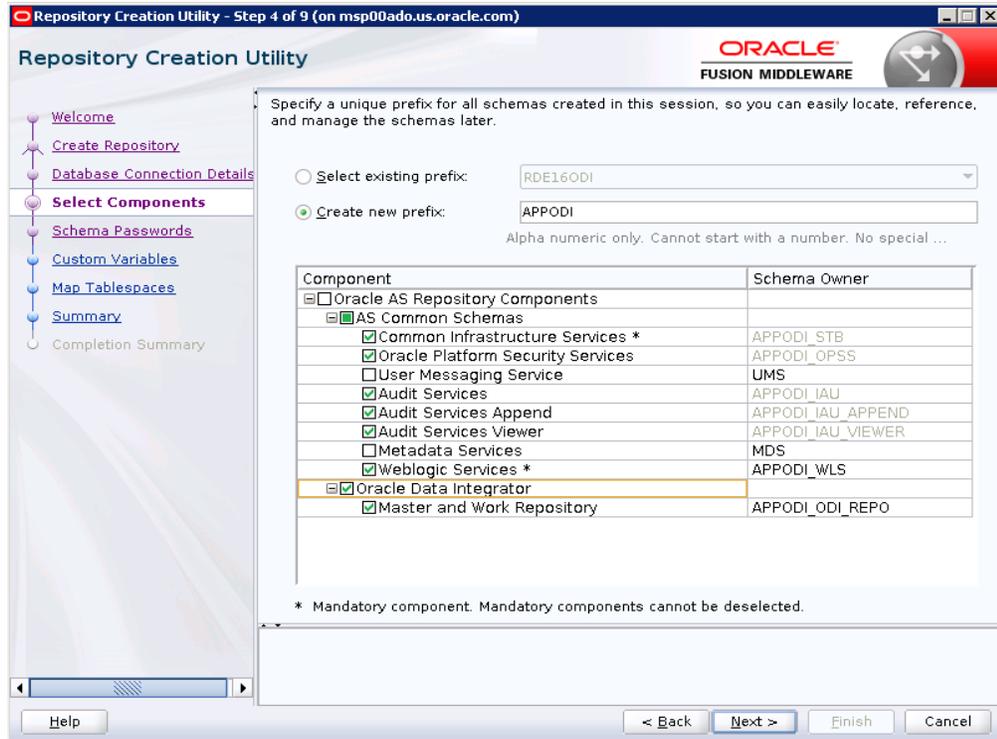
7. Click **OK**.
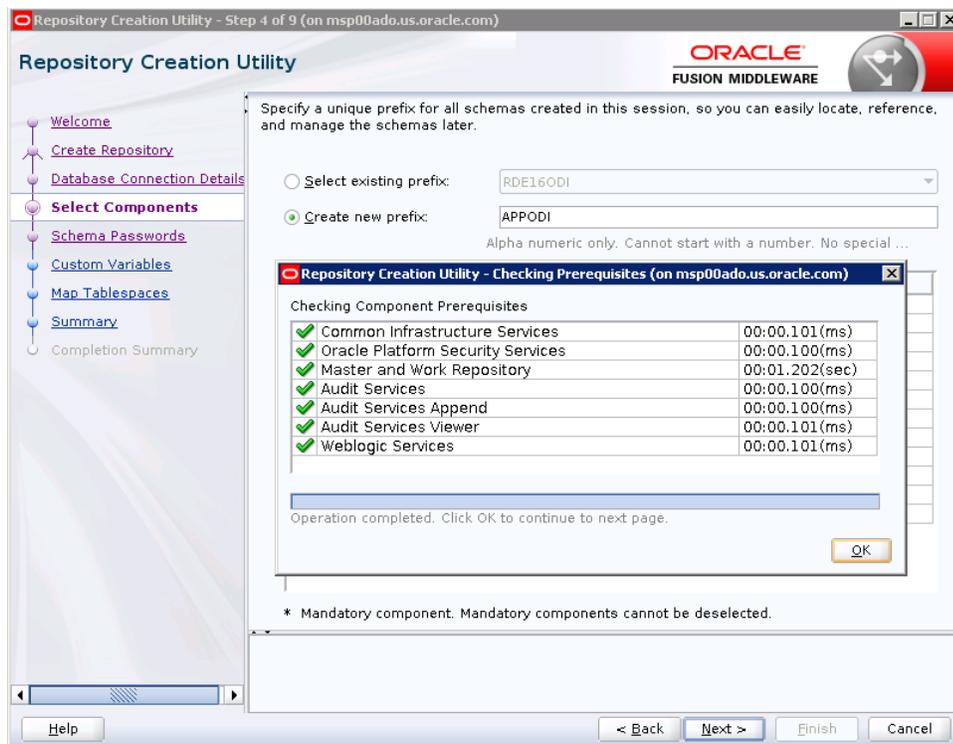
8. Update the schema password and which is required while configuring ODI Domain. Click **Next**.
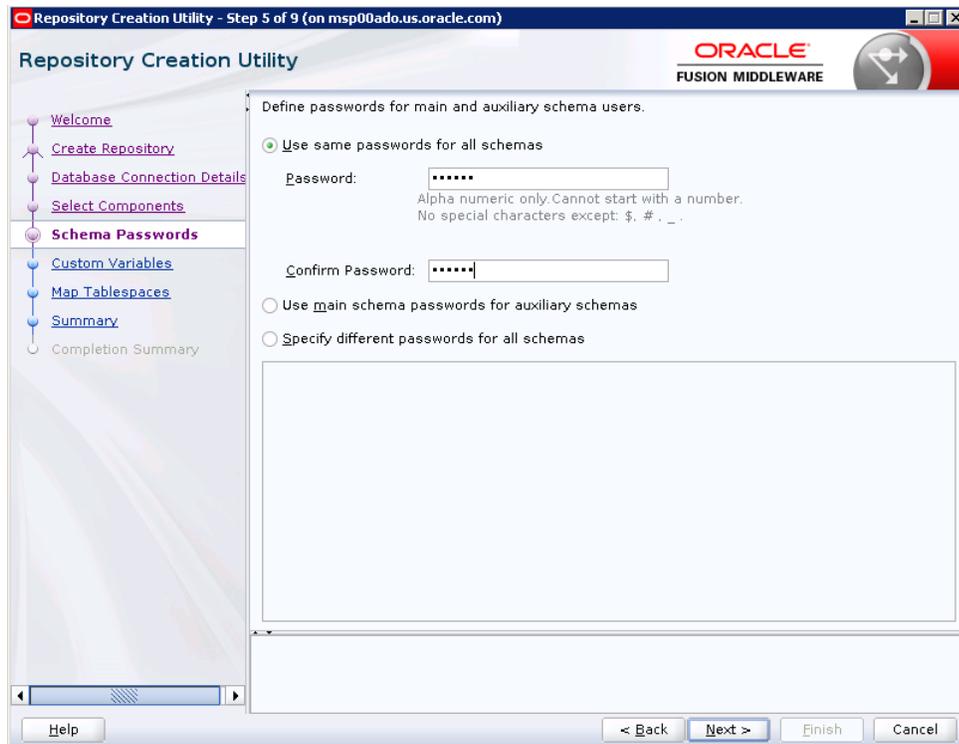
Example:

Provide passwords for Supervisor and Work Repository

WORKREP name – RDE16_WREP

Pwd: Schema password from previous screen.
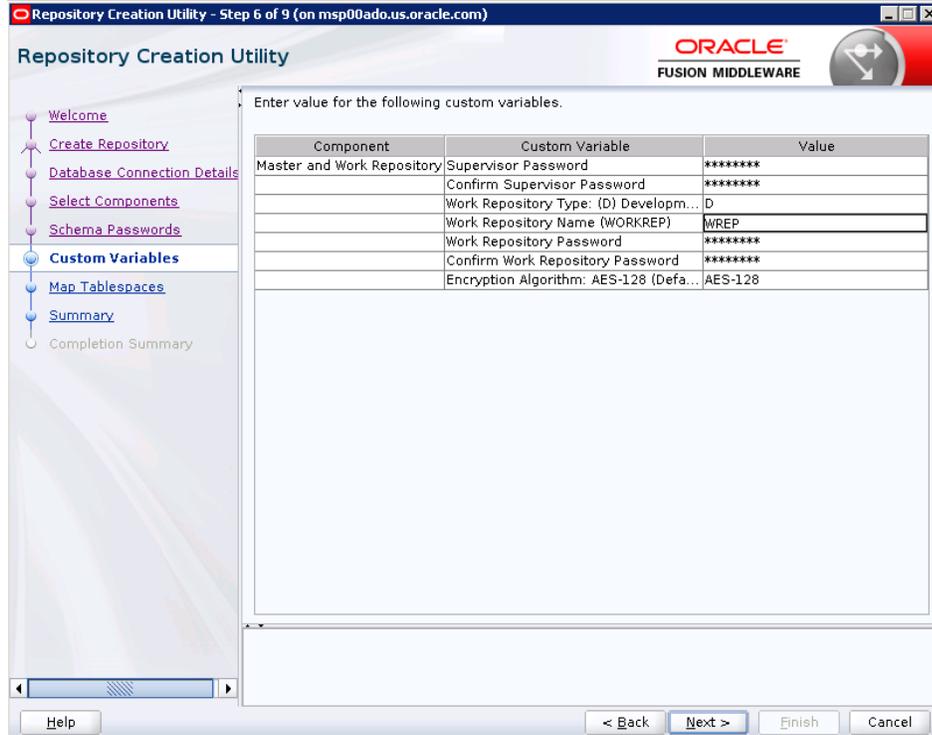
9. Creating new table spaces. Click **Next**.
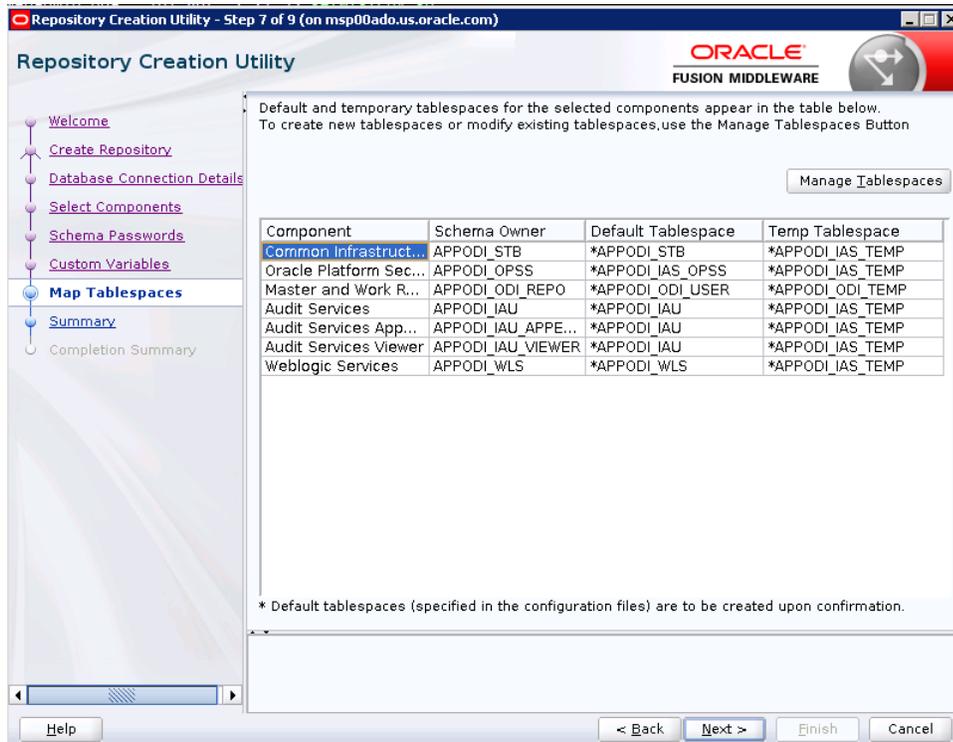
10. Schema's are created and can be used while domain setup. Click **OK**.
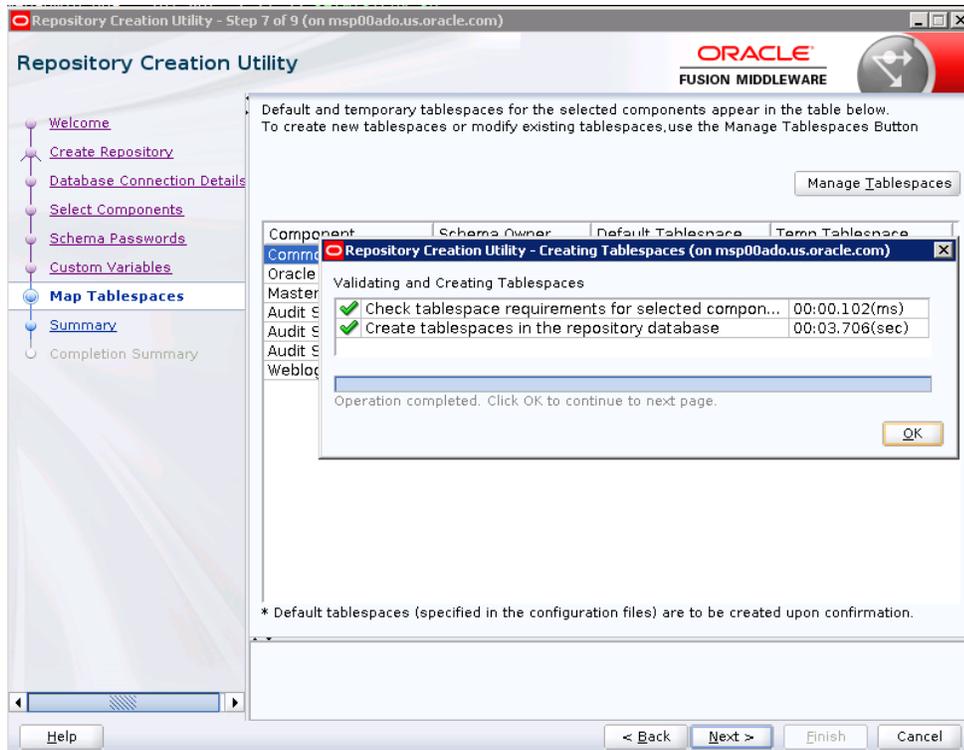
11. Click **Create**.

ODI Domain Configuration

Setup Domain for/with Collocated Agent:

1. Change directory to ORACLE_HOME/oracle_common/common/bin

```
Run ./config.sh and browse the ODIDomain path
```

2. Click **Next**.

3. Select ODI Component, Oracle JRF, Enterprise manager and rest other components will be selected. Click **Next**.
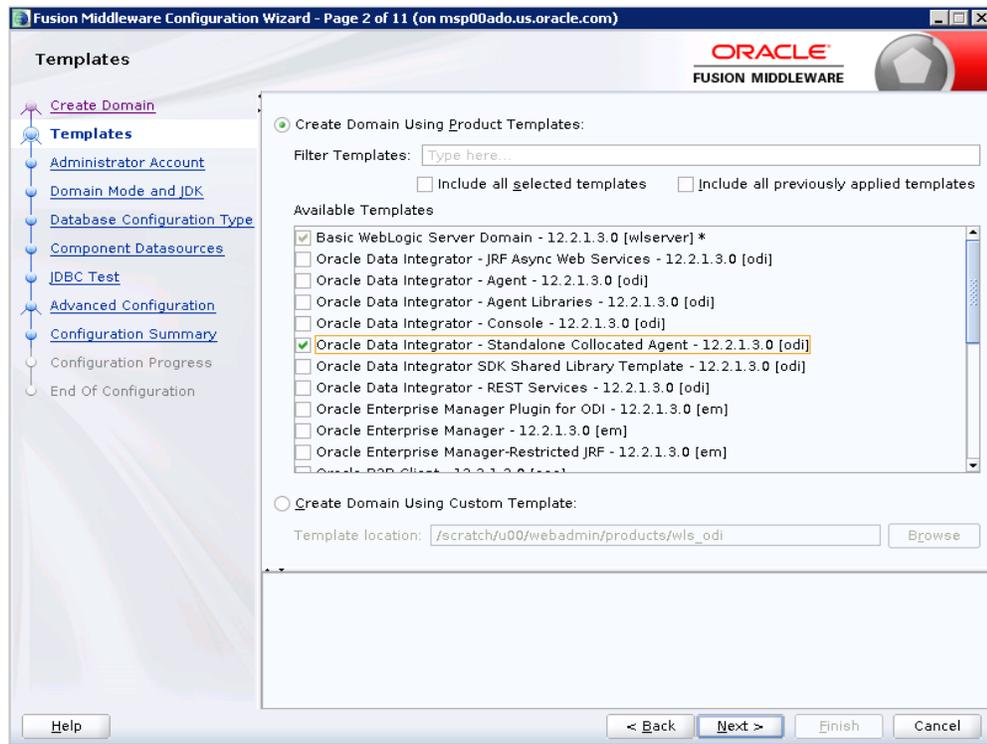
4. Update Domain username and password. Click **Next**.

5. Select the Domain mode and JDK. Click **Next**.
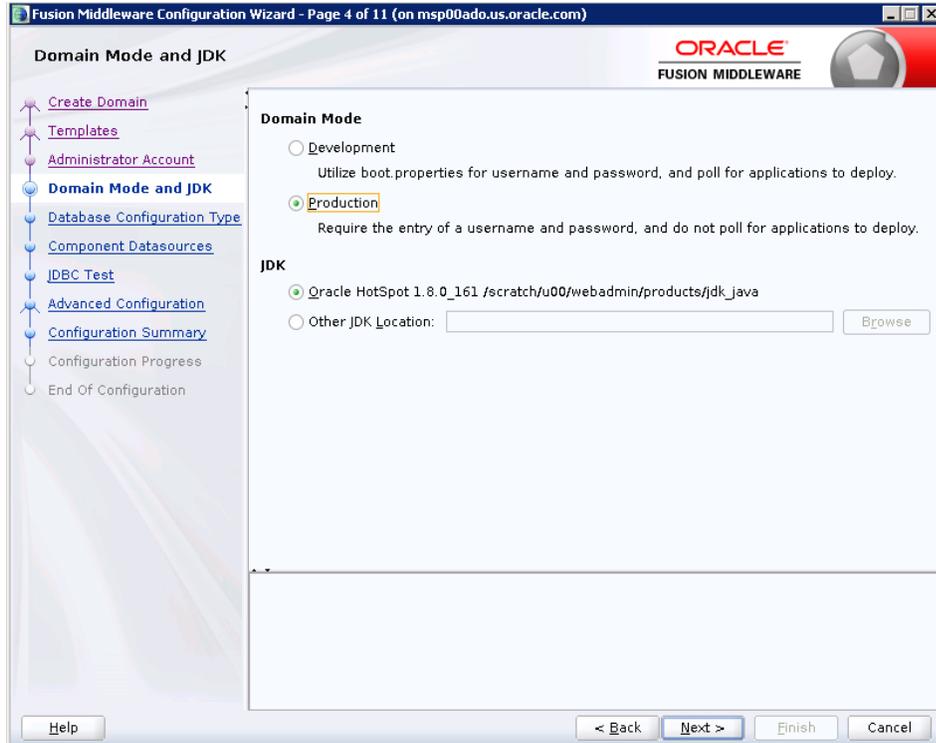
6. Update STB database details and click on Get RCU Configuration. Use RDE Schema.
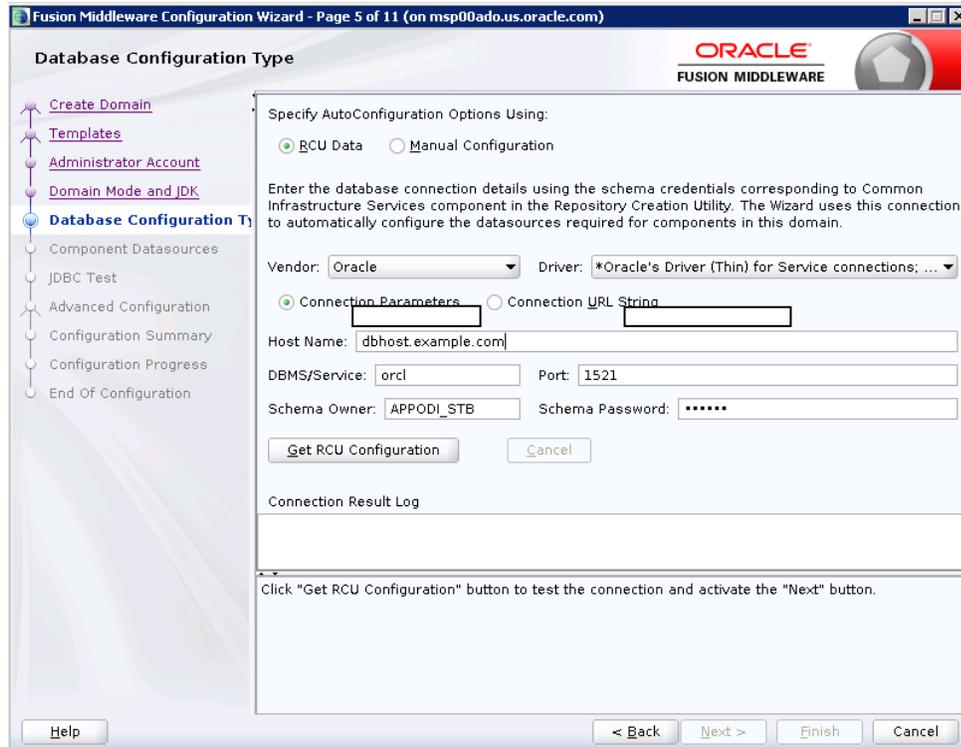
7. Click **Next**.

This screen tests the JDBC Component.
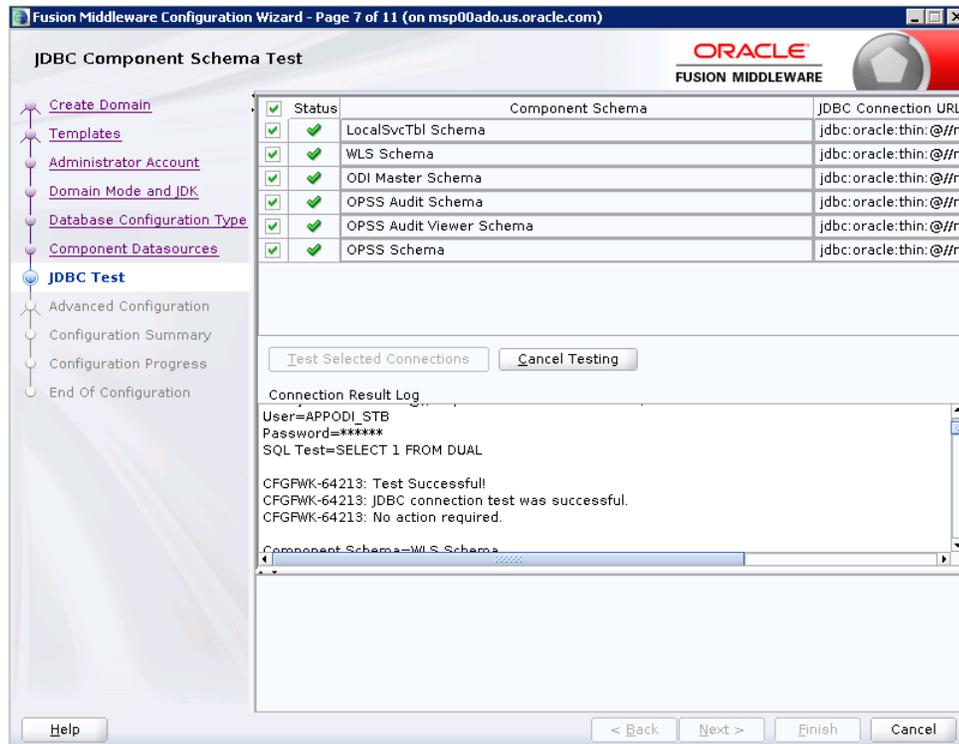
8. Click **Next** and test the Datasource.

9. Select below under advanced configuration screen. Click **Next**.

10. Update Admin server name, Listen Address and port. Click **Next**.
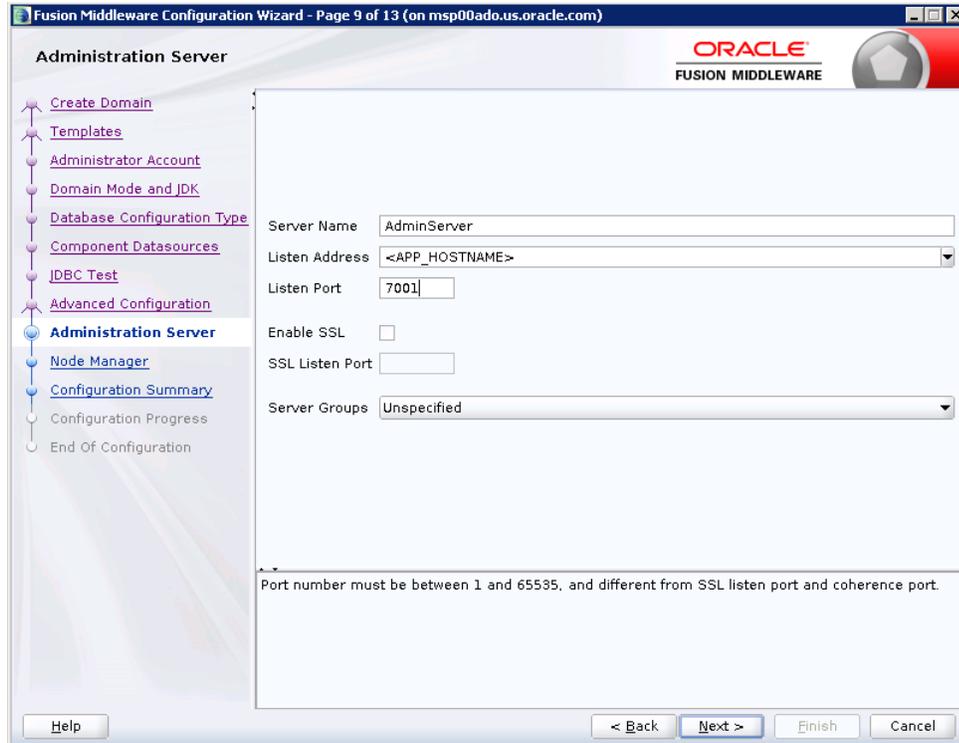
11. Update NodeManager details. Click **Next**.

12. Update system component and make sure you remember the ODI Agent name. Click **Next**.

13. Update ODI Server Configuration and make sure to use the port 20910. Click **Next**.

14. Update Machine details and update Nodemanger Listen address and click **Next**.
Select LocalODIMachine and delete it if ODI installation is on Unix/Linux machine.
Click on Unix Machine and add details as shown below:
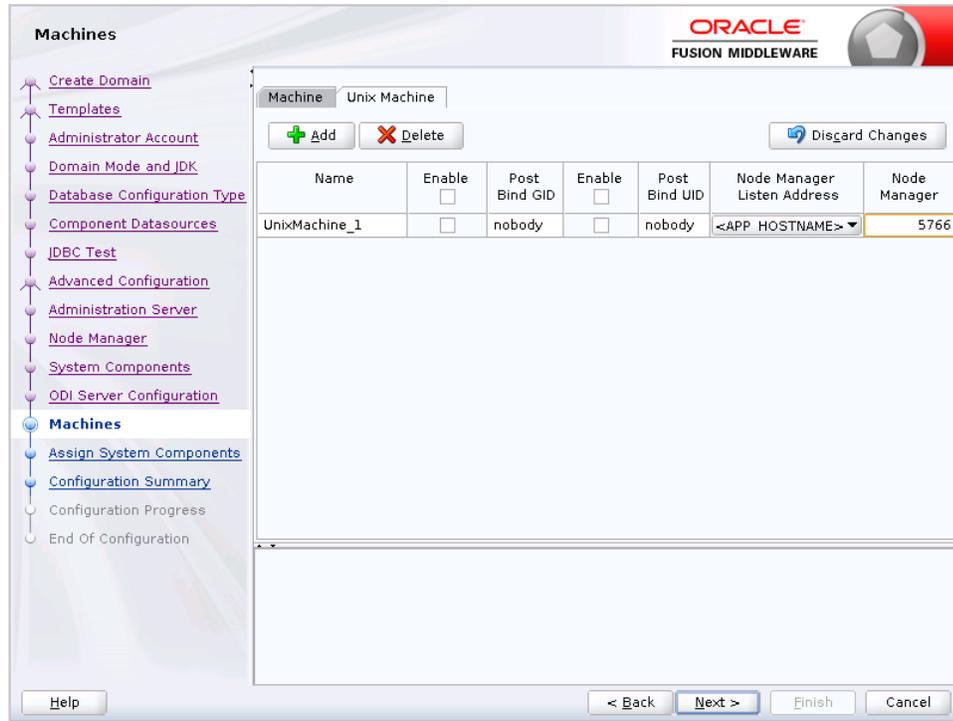
15. Assign System component to Machines and click **Next**.
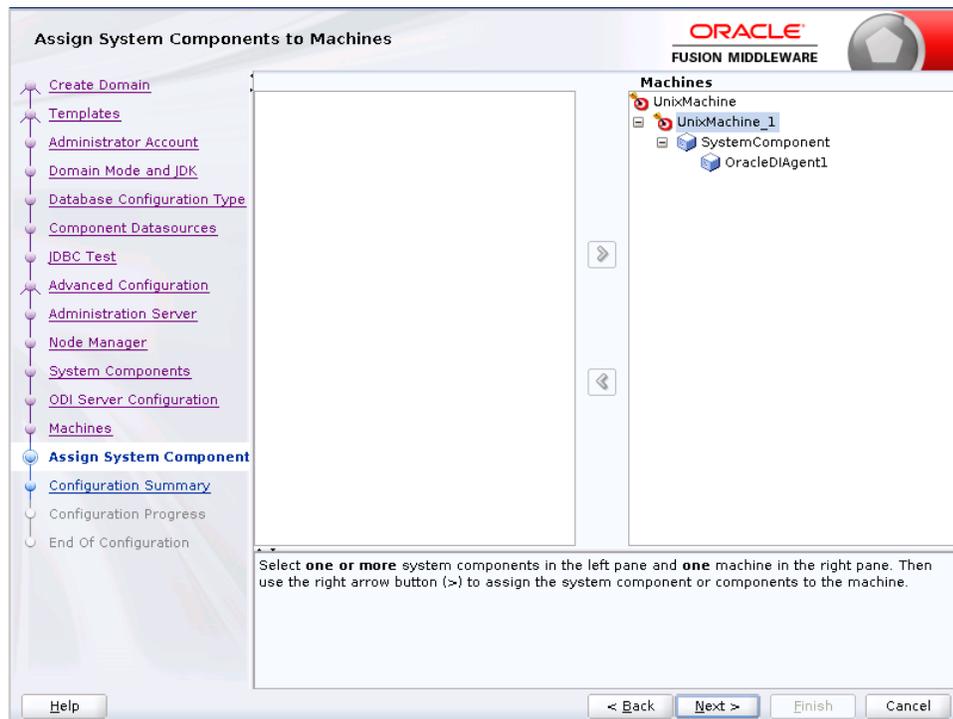
16. Click **Create**.
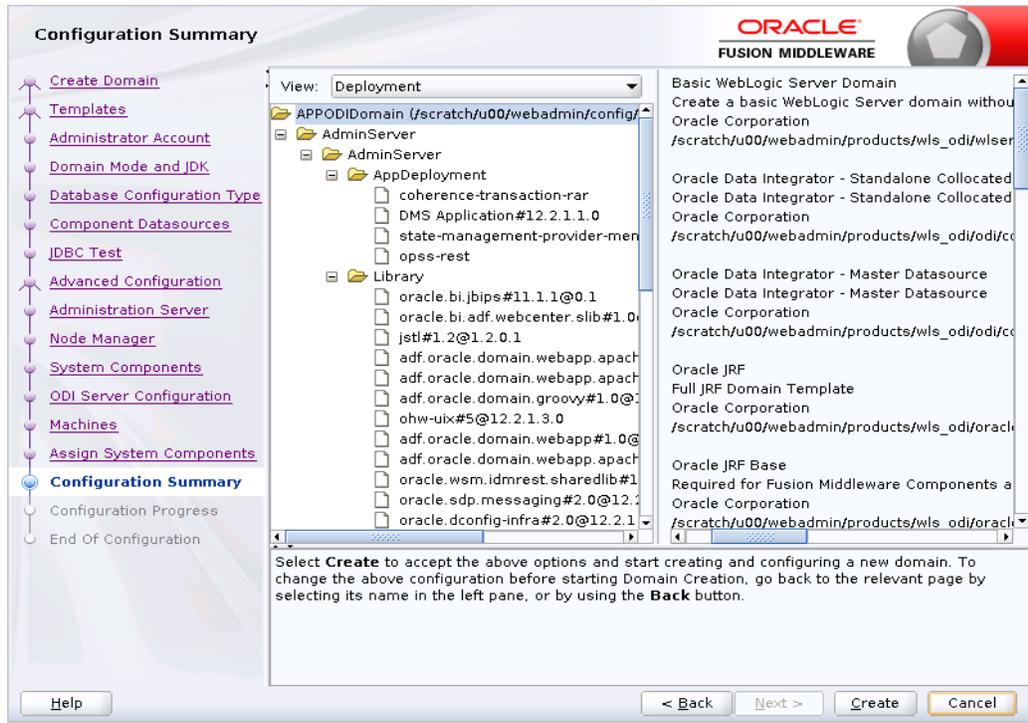
17. Click **Next**.

18. Click **Finish**.

Once Installation is completed, start the Node Manager Domain and Managed server.

19. Start Nodemanager: Go to ODIDOMAIN home and then bin.

```
cd ODIDOMAIN_HOME/bin
./startNodemanger.sh
```

20. Start the ODI agent:

Change directory to ODIDOMAIN_HOME and bin and run below command.

```
./startComponent.sh OracleDIAgent1 (which is already created while setting up domain)
```

21. Stop the ODI agent

Change directory to ODIDOMAIN_HOME and bin and run below command.

```
./stopComponent.sh OracleDIAgent1 (which is already created while setting up domain)
```

Set Environment Variable

Retail Data Extractor release 16.0.2 supports 12cR1. Also, only 64-bit platforms (as noted in hardware specifications) are supported. Therefore, only the 64-bit version of the client libraries should be used.

1. Set and export the following environment variables.

Variable	Description	Example
ORACLE_HOME	Oracle server home. Only applicable when you are installing the database schema objects.	ORACLE_HOME=full_path_of_ 12.1.0.2 client_home export ORACLE_HOME
LD_LIBRARY_PATH	LD Library Path should contain the Oracle database libraries you want to use.	LD_LIBRARY_PATH=$ORACLE_ HOME/lib (64-bit) export LD_LIBRARY_PATH
JAVA_HOME	Java home. Ensure the version of Java is 64-bit.	JAVA_HOME= /opt/app/jdk/jdk1.8 export JAVA_HOME Refer to the ODI product installation guide for the compatible version of JDK.
PATH	PATH should contain directories for Oracle and Java executables	PATH=$JAVA_HOME/bin:$ORACLE_ HOME/bin:$PATH export PATH
TNS_ADMIN	Only applicable if tnsnames.ora is not located under $ORACLE_HOME/network/a dmin. Set TNS_ADMIN to point to a directory where tnsnames.ora is found. If tnsnames.ora is located under $ORACLE_HOME/network/a dmin, (which is true in most cases), do not set this variable.	TNS_ADMIN=/home/user/misc export TNS_ADMIN
NLS_LANG	Locale setting for Oracle database client.	NLS_LANG=AMERICAN_AMERICA. AL32UTF8 export NLS_LANG
ORACLE_SID	The TNS alias mentioned in $ORACLE_HOME/network/a dmin/tnsnames.ora file for the Pluggable Database on which we are going to do the installation	Export ORACLE_SID=<PDB TNS ALIAS>
ODI_JAVA_HOME	ODI JAVA HOME	ODI_JAVA_HOME should be jdk 1.8 Example: export ODI_JAVA_HOME=/u01/java/jdk1.8. 0_75
DISPLAY	Address and port of X server on desktop system of user running install. Optional for dbschema installer.	DISPLAY=<IP address>:0.0 export DISPLAY
ODI_HOME	This directory contains the ODI installation files where ODI server is installed.	Provide the path of ODI_HOME where the ODI server is installed For example: `export ODI_HOME=/u01/product/fmw/wls_odi /odi/agent`

2. If you are going to run the installer in GUI mode using an X server, you need to have the XTEST extension enabled. This setting is not always enabled by default in your X server. See "Appendix: Common Installation Errors" for more details.

Setup Password Stores for Silent Install Mode

Before starting the installer in silent mode, you must setup the user credentials for the application, database user accounts. The installer includes a Credential Storage Manager that you can use to setup a password store for the application installation.. This is needed only when the installer is run in silent mode. Skip this section when running the installer in GUI/text mode.

To setup Credential Storage Manger Password Store:

1. JAVA_HOME environment variable must be set

   ```
   cd <STAGING_DIR>/orde/installer/orde/Build/orpatch/deploy/retail-public-
   security-api/bin
   ```

2. Add execute permission to scripts.

   ```
   chmod +x save_credential.sh dump_credentials.sh
   ./save_credential.sh –a <alias_name> -u <username> -l </location/wallet/dir>
   ```

   Example:

   ```
   $ ./save_credential.sh -a rde_dmAlias -u rde_dmuser -l
   /u00/webadmin/orde/wallet


   ============================================
   Retail Public Security API Utility
   ============================================
   Credential Store type is FILE_STORE
   Credential Store location is file:///u00/webadmin/orde/wallet


   Enter password:


   Verify password:
   Password for name(rde_dmAlias) saved in credentials store.
   ```

   Aliases must be unique and alias names are case sensitive.

3. Repeat step 2 to store all necessary credentials needed for installation, like datasource credentials, WebLogic credentials and any other credentials applicable.

4. Verify the alias names stored in the wallet

   ```
   ./dump_credentials.sh  </location/wallet/dir>
   ```

   Example:

   ```
   $ ./dump_credentials.sh /u00/webadmin/orde/wallet
   ============================================
   Retail Public Security API Utility
   ============================================
   ===============================================================================
   ====================
   Below are the credentials found in the wallet at the
   location:/u00/webadmin/orde/wallet
   ===============================================================================
   ====================
   Application level key partition name:DEFAULT_KEY_PARTITION_NAME
   User Name Alias:rde_dmAlias User Name:rde_dmuser
   User Name Alias:dsKalias User Name:null
   User Name Alias:dsTalias User Name:null
   User Name Alias:dsAlias User Name:rms01
   ```

5. The following credentials must be stored in the wallet for installation to complete successfully
   ▪ ODI supervisor credentials
   ▪ ODI MREP Database credentials
   ▪ ODI WREP Database credentials
   ▪ RADM Database user credentials
   ▪ RDE_RMS Database credentials
   ▪ Orpatch wallet credentials

Alias	Property name in ant.install.property
ODI Supervisror credentials	input.odi.supervisor.alias =SUPERVISOR
ODI MREP Database credentials	input.odi.mrep.alias = mrep-alias
ODI WREP Database credentials	Input.odi.wrep.alias = wrep-alias
RDE_DM Database credentials	input.radm.db.alias = rde_dm-alias
RDE_RMS Database credentials	input.rabe.db.alias = rde_rms-alias
Orpatch wallet credentials	input.wallet.alias = null.value

6. The alias names in ant.install.properties file must match the alias names stored in the wallet created during pre-install setup.

Setting Up Installation Properties File

In order to install the Retail Data Extractor, need to specify the properties to use during the installation process. The ant.install.properties.sample file in the <staging_dir>/rde/installer

Ant installer parameters reference

Parameter	Description
Ant.install.config.version=[version number]	The version of RDE 16.0.2.0 installer configuration Default = 1.1
input..retail.home = [full path to RETAIL_HOME directory]	Directory where the Retail Data Extractor files are installed. Default = <Empty>
input.fresh.install	True if full install, false if patch install
input..retrieve.credentials =[true or false]	Default = false Needed for silent installer to retrieve credentials from wallet
input.wallet.dir = [/path/of/wallet/dir]	Wallet location Default = <Empty> Needed for silent installer to retrieve credentials from wallet

Parameter	Description
input.do.install.mmhome.files = [true or false]	True if you wish to install mmhome files else set it to false
input.do.install.rms.db = [false/true]	False/true depending if RDE needs to be integrated with RMS
input.do.install.ra.db = [true, false]	True if RA database component is to be installed else set it to false
input.do.install.odi.files = [true, false]	True, if odi files are needed to be installed else set this property to false
input.odi.home.dir = [Path_To_ODI_home]	If baove property is set to true, ODI HOME is the installation directory of ODI 12C
input.odi.mrep.create = [true, false]	True if mrep repository is supposed to be created else false.
input.odi.wrep.create = [true, false]	True if wrep repository is supposed to be created else false.
input.sys.db.username = SYS	Default to SYS. Any DB user that has SYSDBA perms
input.odi.supervisor.alias = [SUPERVISOR_USERNAME]_ALIAS]	Default value = SUPERVISOR
input.odi.repo.db.host = [HostName]	The hostname on which odi 12c is installed
input.odi.repo.db.sid = [SERVICE_NAME]	SID to connect to db
input.odi.repo.jdbc.driver = oracle.jdbc.OracleDriver	JDBC driver to be used to connect to odi repo
input.odi.port = 1521	Defaults to 1521. Set it to port to connect to odi Repository
input.odi.mrep.db.alias = [MREP_USERNAME_ALIAS]	By default mrep –alias. Can be changed to any value
input.odi.wrep.db.alias = [WREP_USERNAME_ALIAS]	Defaults to wrep-alias
input.radm.db.sid = [SERVICE_NAME]	SID to connect to db
input.radm.datasource.connectstring = jdbc:oracle:oci:@[SERVICE_NAME]	Connect string to connect to RADM datasource
input.radm.db.alias = radm-alias	Defaults to radm-alias
input.rabe.db.alias = rde_rms-alias	Defaults to rde_rms-alias
input.wallet.alias = null.value	Defaults to null.value.

3

Installation Tasks - Full

Run the Retail Data Extrator Database Schema Installer from ODI Host.

Perform the following procedure to install Retail Data Extractor:

> **Note:** See "Appendix: Retail Data Extractor Application Installer Screens" for details on every screen and field in the database schema installer.

> **Note:** The installer does not run on Windows. If ODI is installed on Windows, you cannot use the installer to copy ODI files for Retail Data Extractor. You must follow the manual installation process detailed in the "Known Issues/Limitations" chapter.

1. Change directories to <STAGING_DIR>/orde/installer
2. Run the install.sh script to start the installer.

> **Note:** The following are the usage details for install.sh. The typical usage for GUI mode is no arguments.
>
> ```
> ksh install.sh [text | silent]
> ```

> **Note:** Usually, if you have multiple instances, the RDE RMS user schema would be on one instance and all Retail Data Extractor schemas (RDE data mart) are on the other instance.

Depending on system resources, a typical installation takes about 30 to 60 minutes.

> **Note:** Ignore the following warning and press enter to continue
>
> WARNING: If you intend to install/upgrade ODI files/repositories,
> you must have jps-api.jar under /u00/webadmin/products/wls_odi/odi/../oracle_common/modules/
> This jar is not found.
> Press enter to continue.

3. Verify that the installer reports "SUCCESS" for the Preinstall Check. If it reports "FAILED," check for errors in the output under the "Checking environment for installation" section, and verify that your environment variables are set properly.
4. ORDE 16.0.2 is a full baseline installation.
5. Check the Install appropriate checkbox and continue with installer. Depending on system resources, a typical installation can take 30 minutes to 1 hour.

6. The new ORDE Application RETAIL_HOME parameter has been introduced with the change to the installer utilizing the ORPatch toolset. The installer requires specifying the RETAIL_HOME path. This path will be the destination for the entire mmhome directory, ORPatch files and other required installer artifacts. The MMHOME path will still be required in the ra.env configuration file and is referenced throughout this document. The default MMHOME path will be RETAIL_HOME/mmhome.

7. The RDE Installer provides the option of creating the ODI Repository, this option should be unchecked and the ODI Repository should be created using the wizard in ODI Studio. It is recommended to select the option "Install ODI files" for copying the ODI files and scripts in addition to the Data Extractor objects.

8. After the installer is complete, you can check its log file: orde-install.<timestamp>.log in the path <STAGING_DIR>/orde/installer/logs

9. The installer leaves behind the ant.install.properties file for future reference and repeat installations. This file contains inputs you provided. As a security precaution, make sure that the file has restrictive permissions.

> **Example:** `chmod 600 ant.install.properties`

Resolving Errors Encountered During Database Schema Installation

If the database schema installer encounters any fatal errors, it halts execution immediately. To resolve this issue, you can choose one of the following options.

Restart with a Clean Set of Schemas

To restart with a clean set of schemas, complete the following steps.

1. Clean up all database schema objects created by the installer for the Retail Data Extractor data mart schema, Retail Data Extractor batch schema i.e. RDE_RMS user.

   You may even drop these schemas and recreate them by logging into SQL*Plus as SYSDBA.

2. Remove the RETAIL_HOME provided for the installation and recreate the directory.

3. Rerun the installer. If this message is displayed ("A previous installation attempt was detected. Do you want to resume the previous installation?"), enter no<ENTER>

4. The installation runs as if run for the first time on clean schemas.

Resume from the Previous Point of Failure

To resume from the previous point of failure, complete the following steps.

1. If a SQL file failed to complete successfully, the installation log indicates the name of the SQL file that failed and points to the directory ($RETAIL_HOME/orpatch/logs) where you can go to look at the exact errors. Additional error information for invalid objects can be found in $RETAIL_HOME/orpatch/logs/detail_logs/dbsql_{schema}/invalids. The {schema} refers to rde_dm,rde_rms

   If the fatal installation error happened while importing data using the import utility, you must resolve the error also.

2. Re-run the installer. If this message is displayed ("A previous installation attempt was detected. Do you want to resume the previous installation?"), enter **yes<ENTER>**.

3. Subsequent executions of the installer skip the SQL scripts which have already been executed in previous installer runs. This is possible because the installer maintains

entries in a table called DBMANIFEST of the scripts that have been run. It also maintains an orpatch_restart.state file when the install restarts.

> **Note:** If you are resuming the previously-failed database object installation, you must take care to restore the database schema to the resumable state. For example, say XYZ.sql failed to complete successfully, which ended up creating several database objects but not all. Since you will be resuming with this file, you must remove any objects that were created with this file by examining the file and the database.

Post Installation Tasks – Full

Oracle Data Integrator Configuration Tasks

It is assumed that Oracle Data Integrator 12C (12.2.1.3) software has already been installed. If you need more information in addition to the following tasks, refer to ODI installation documentation.

Please follow ODI documentation for any ODI administration activities like setting up agents or ODI code migration across different environments using the ODI export/import process.

Terminology

This section provides definitions for applicable terminology.

ODI Home Directory

This directory contains the ODI installation files. For this section the ODI home is set under the following path:

<ODI 12.2.1.3.0 Installation directory>/odi

ODI Domain is setup under the ODI_DOMAIN_HOME directory.

ODI User and Password

The default ODI user = SUPERVISOR

The default ODI password = The ODI password is set at the time of Pre-Installation Tasks while setting up repositories via RCU.

JDBC Connectivity

By default, the JDBC connection is specified as jdbc:oracle:thin:@<host>:<port/ServiceName>. During configuration, customers must replace this with the actual credentials for the environment.

> **Note:** OCI connectivity can also be used instead of JDBC. Contact your database administrator.

Database Links

Public Database Links need to be created between RMS USER (example RMS01) and Retail Data Extractor batch user (RDE_RMS01) and the DB-Links need to be explicitly mentioned in the Data Servers, configured in the ODI Topology.

In ODI, Data Server for RDE_DM01 (used in this guide): ORACLE_BI_APPLICATIONS_RA_INSTALL

Data Server for RDE_RMS01 (used in this guide): ORACLE_BI_APPLICATIONS

Data Server for RMS01/RDE_RMS01 (used in this guide): ORACLE_RETAIL_SOURCE

MMHOME UNIX Scripts Setup

To complete the MMHOME setup after the installer run, set the ra.env file as per the details below:

1. From the UNIX command prompt, navigate to $MMHOME/etc directory and open file ra.env

2. Validate the file and update the parameters with their values as per the environment setup.

ra.env configuration:

Variable	Description	Example
MMHOME	This is the mmhome directory present under RETAIL HOME directory	RETAIL_HOME/mmhome
VERSION	This is the scenario version for execution set to 001	001
RETAIL_HOME	This is the RETAIL HOME set at the time of installation	
CONTEXT_ENV	This is the context set for execution. Ideally set to DEVELOPMENT. If the instance and Agent name is different then what is set, please modify the same	DEVELOPMENT -INSTANCE=OracleDIAgent1 - NAME=OracleDIAgent
CONTEXT_INSTALL	This is the context install set for execution. Ideally set to DEVELOPMENT_INSTALL. If the instance and Agent name is different then what is set, please modify the same	DEVELOPMENT_INSTALL -INSTANCE=OracleDIAgent1 - NAME=OracleDIAgent
ODI_WORKREP	Name of the ODI WORKREP	
JAVA_HOME	Path to JAVA Home	
ODI_JAVA_HOME	Path to JAVA Home	
ODI_HOME	ODI installation directory. This directory should not include the agent path	
ODI_DOMAIN_HOME	This is the directory where ODI Domain is configured.	
ORACLE_HOME	Oracle Home directory. The Oracle database client is installed here	
AGENT_NAME	Name of the ODI agent configured	
AGENT_URL	This is the agent url	Example = - AGENT_URL=http://@hostname@:20910/${@MyAgent@}

3. Save the file and export the parameters in UNIX environment.

4. Navigate to the src directory $MMHOME/src and convert all the ksh scripts to unix format as there can be some hidden windows characters.

```
bash-4.2$
bash-4.2$ dos2unix *
```

Windows-based text editors put special characters at the end of lines to denote a line return or newline. Normally harmless, some applications on a Linux server cannot understand these characters and can cause the service to not respond correctly.

Connecting to the Retail Data Extractor ODI Repository

To connect to the Retail Data Extractor ODI repository, create a new Retail Data Extractor ODI Login as follows.

Pre-requisite steps to be performed before importing Master and Work repository code in ODI 12.2.1.3.0

The heap space issue like java.lang.OutOfMemoryError exception for GC Overhead limit exceeded might occur while ODI code import. In order to bypass this known error please follow the below steps in Application server where the ODI_HOME exists.

Update the odi.conf file by setting the MaxMetaspaceSize.

Example:

```
/2048M
```

Location of odi.conf file : < odi_home>/odi/studio/bin/odi.conf

We can even set the initial and maximum size of the memory allocation pool by adding the Xms and Xmx VM Options to product.conf file as shown below

Example:

```
AddVMOption -Xms2048m
AddVMOption -Xmx2048m
```

> **Note:** The above settings can be modified or increased in order to resolve the memory issues.

Sample location of product.conf file: ~/.odi/12.2.1.2.0/product.conf

1. Depending on the operating system you are using to launch ODI follow the procedure for either UNIX or Windows.

   UNIX

   From the UNIX environment navigate to the path mentioned below set up DISPLAY and execute the following command to launch ODI Studio

   ```
   Path: cd <$ODIHOME>/studio/
   ```

   ```
   Command to Execute: ./odi.sh
   ```

   Windows

   For Windows, do the following:

   From the Programs menu, select Oracle -> ODI Data Integrator.

> **Note:** Please refer to the Known Issues/Limitations chapter for the compatible JDK version to be used to launch ODI Studio.

2. Complete the below steps to login to create a ODI connection.
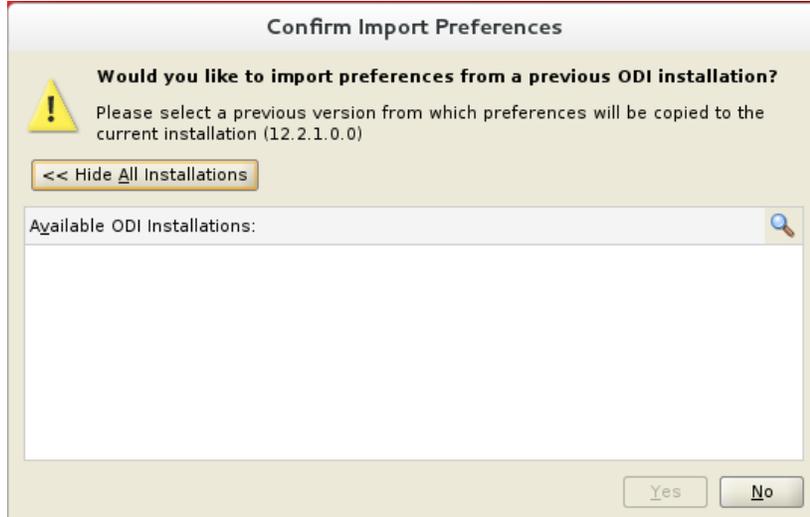
3. Click Yes.
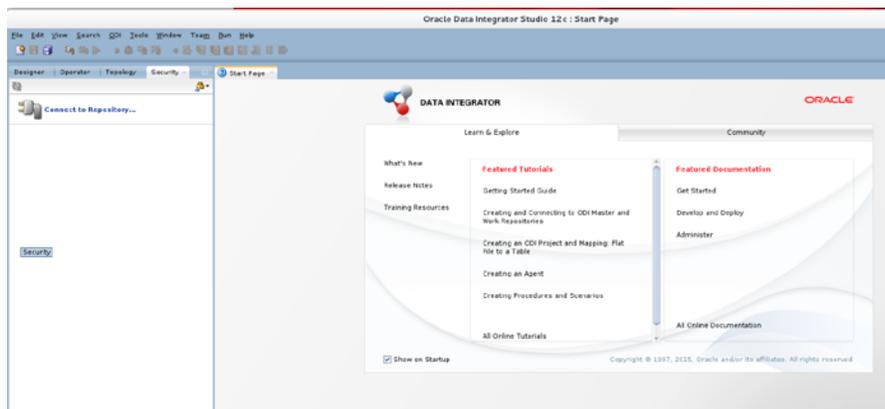
4. Navigate to File > New > In the New Gallery, in the Categories tree, select ODI.

5. From the Items list, select Create a New ODI Repository Login.
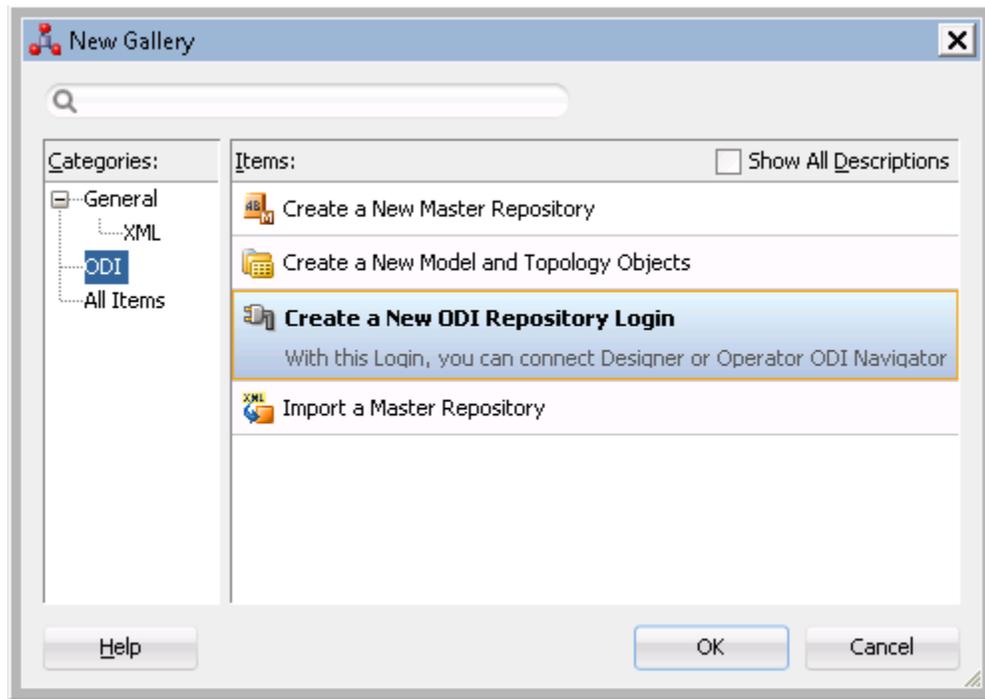
6. Configure Repository Connections with the parameters as shown in the sample screenshot below. To enter the JDBC URL, click the button next to JDBC URL field and select jdbc:oracle:thin:@<host>:<port>/<ServiceName> and update the host, port and Service name information. Click Test to verify the successful Master Repository connection.

7. Click on the Work Repository radio button and Select the work repository name from the Work Repositories List and click OK.
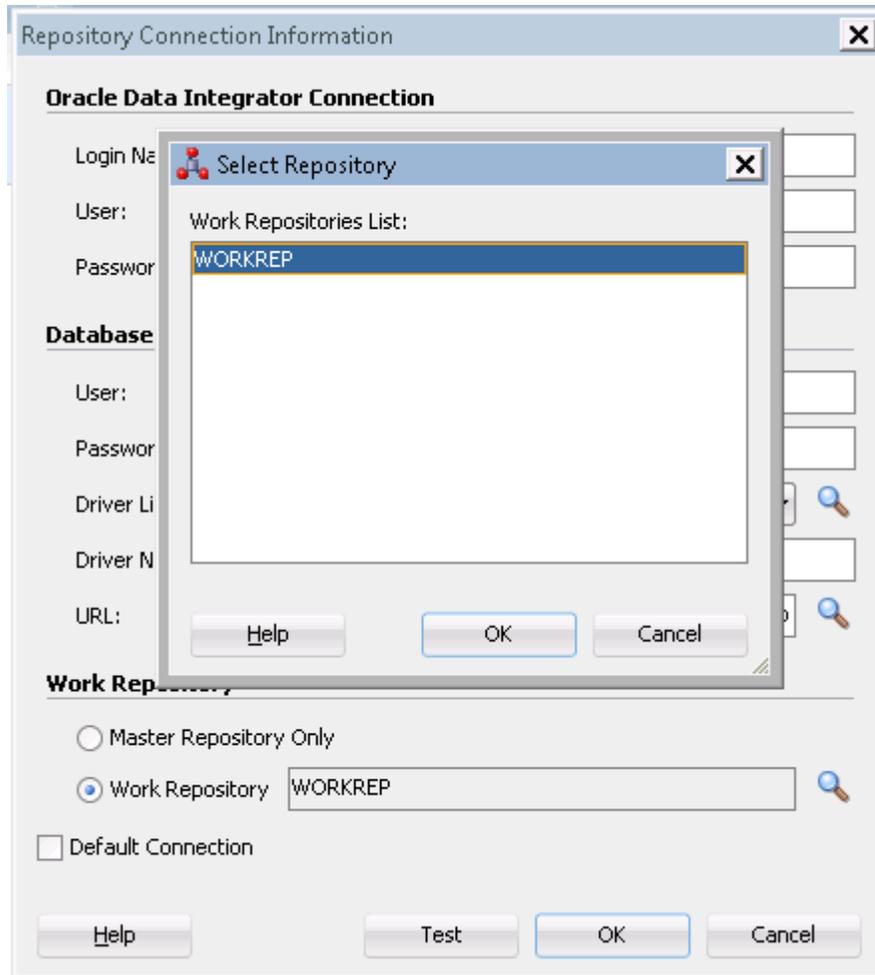
8. Click Test to verify a successful connection. Click OK to save the connection.
9. Login using above credentials and also it can be stored in wallet.

Importing the Master Repository Zip Files

1. Select the Topology tab and click the Topology Menu icon from the upper right side as shown in the screen.
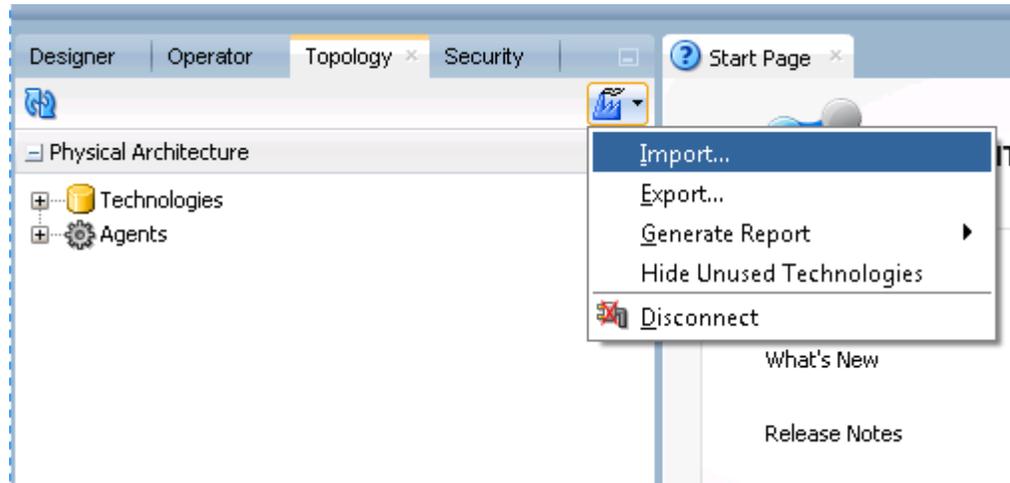
2. Select **Import** from the dropdown list. The Import Selection screen is launched.
3. Select **Import the Master Repository** option and click **OK**.
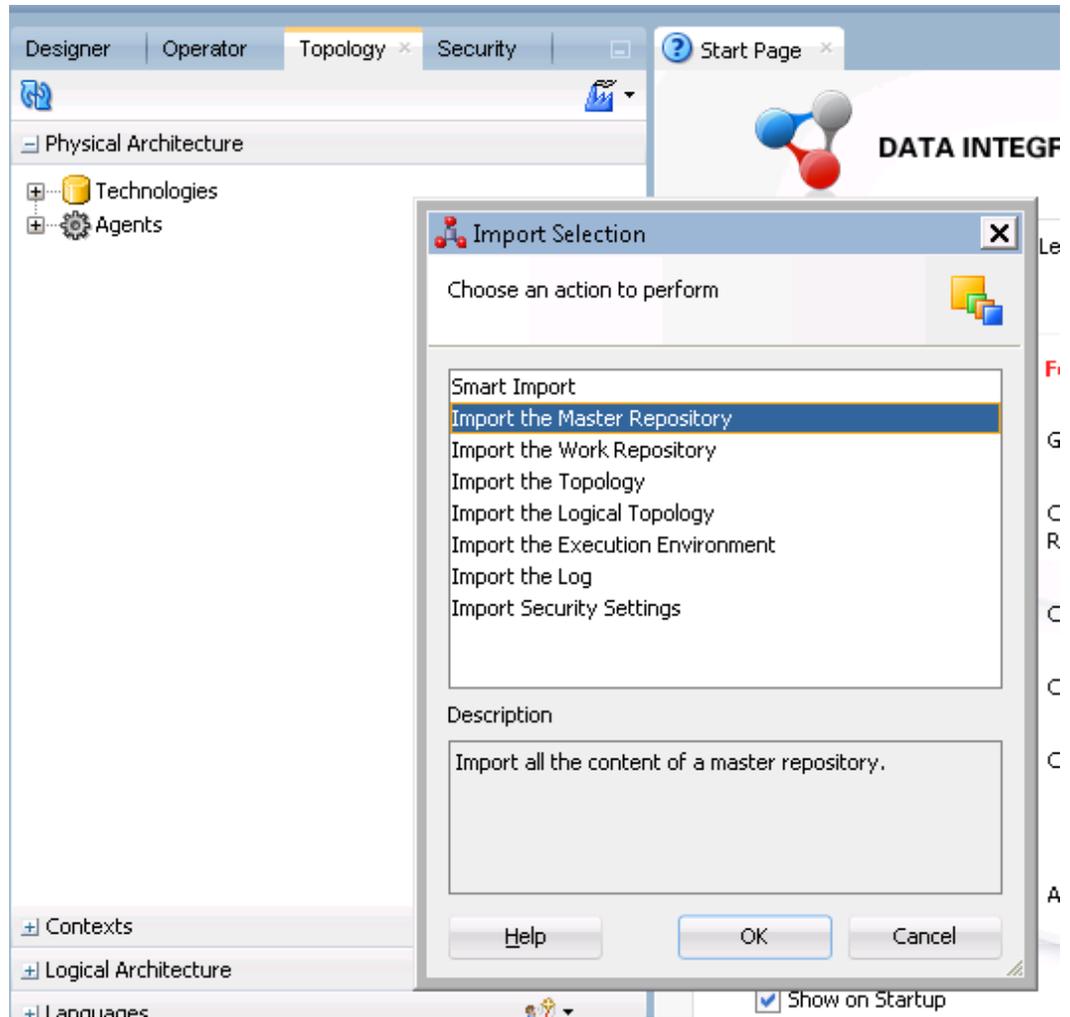
4. Select Synonym Mode Insert from Import Mode dropdown. .Select the Import from a zip file radio button.

   ▪ Browse to the location of the zip files, and select the zip file to import from <STAGING_DIR>\
     orde\installer\orde\mmhome\full\ra_odi_source_code\RDE_MREP.zip
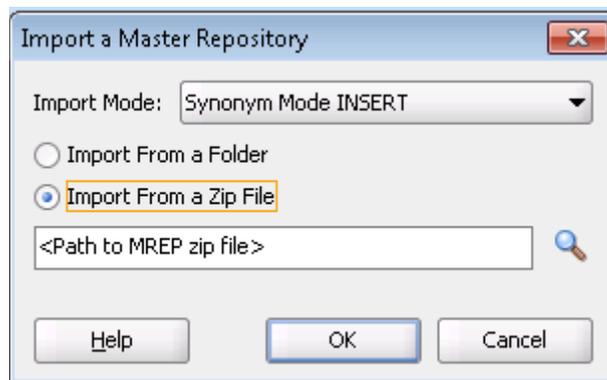
5. Click **OK**.

Importing the Work Repository Zip Files

1. From the Topology tab, click the drop down list on the upper right side of the panel and select **Import**. The Import Selection screen is displayed. Select **Import the Work Repository** option and click **OK**.
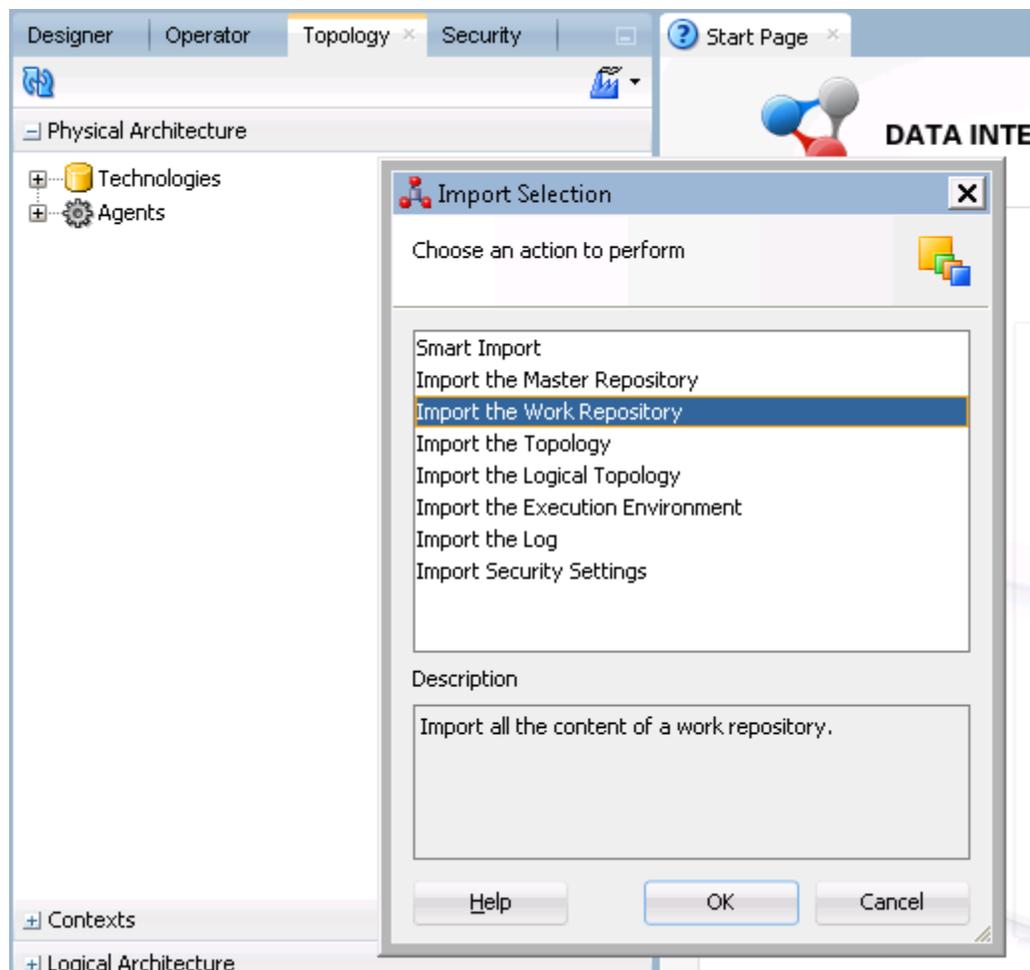
2. In the Import Mode field, select Synonym Mode Insert and select Import from the zip file.

3. Browse to the location of the zip files, select the zip file to import.

4. Select from location: <STAGING_DIR>\
    orde\installer\orde\mmhome\full\ra_odi_source_code\RDE_WREP.zip
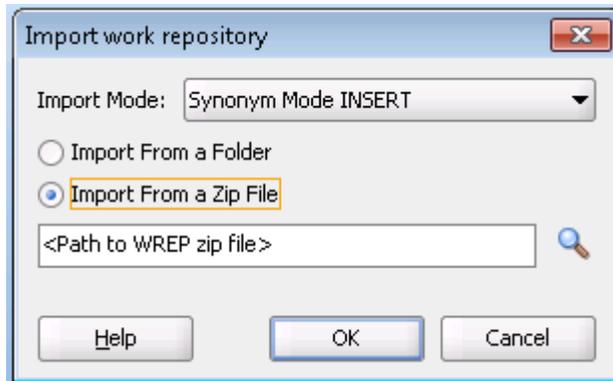
5. From the Import work repository screen, click **OK**.

6. Click **Yes** if any warnings show up.

> **Important:** Importing the Work Repository can take as long as an hour. Do not close the Designer window until the import is complete.

7. After the .zip file is imported, the Projects screen appears.
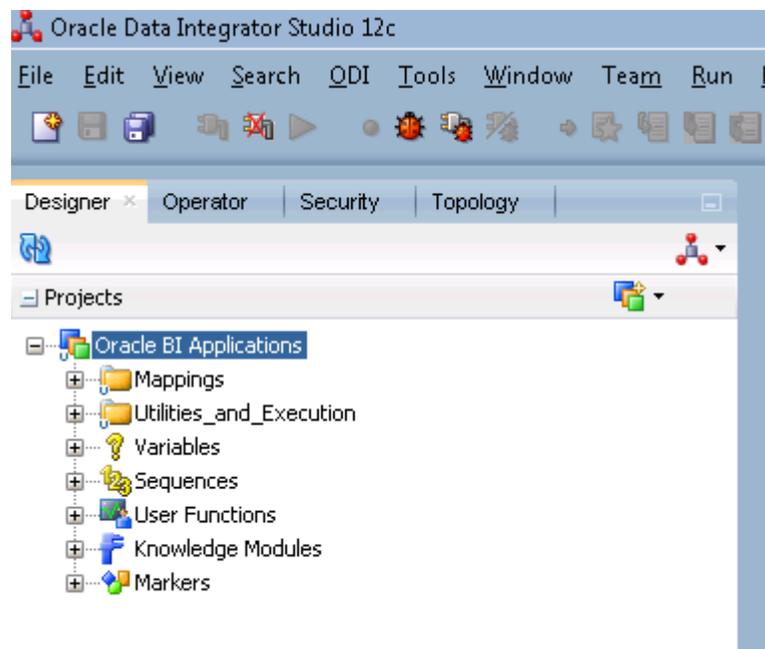
Configuration of ODI Agent:

1. From the Topology tab, expand Physical Architecture and select Agents.
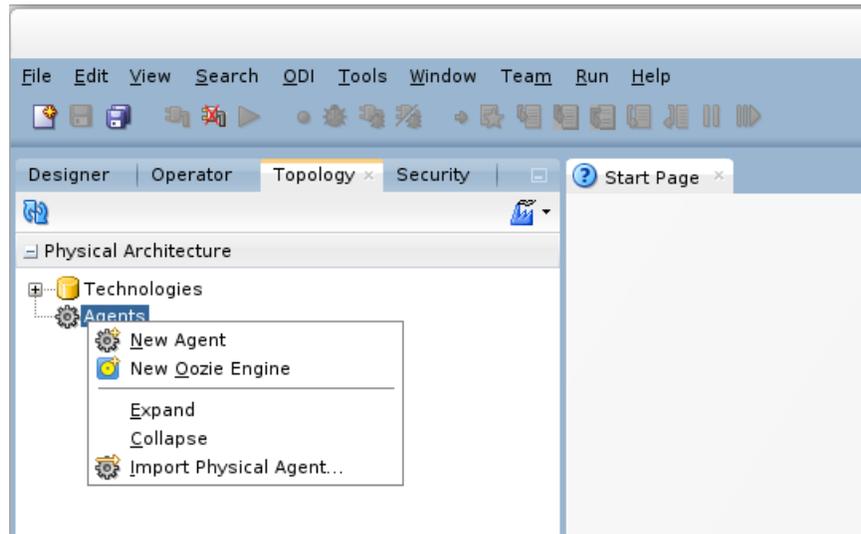
2. Right-click NEW agent and create the ODI agent based on the Agent created on the WebLogic server in pre-requisite.
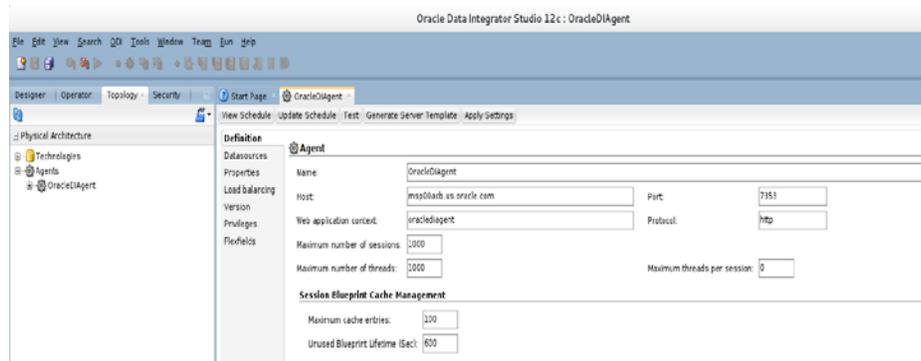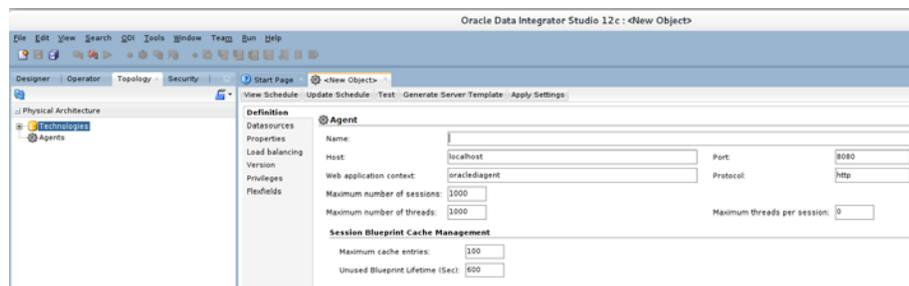
3. From the Topology tab, expand Logical Architecture and select Agents.
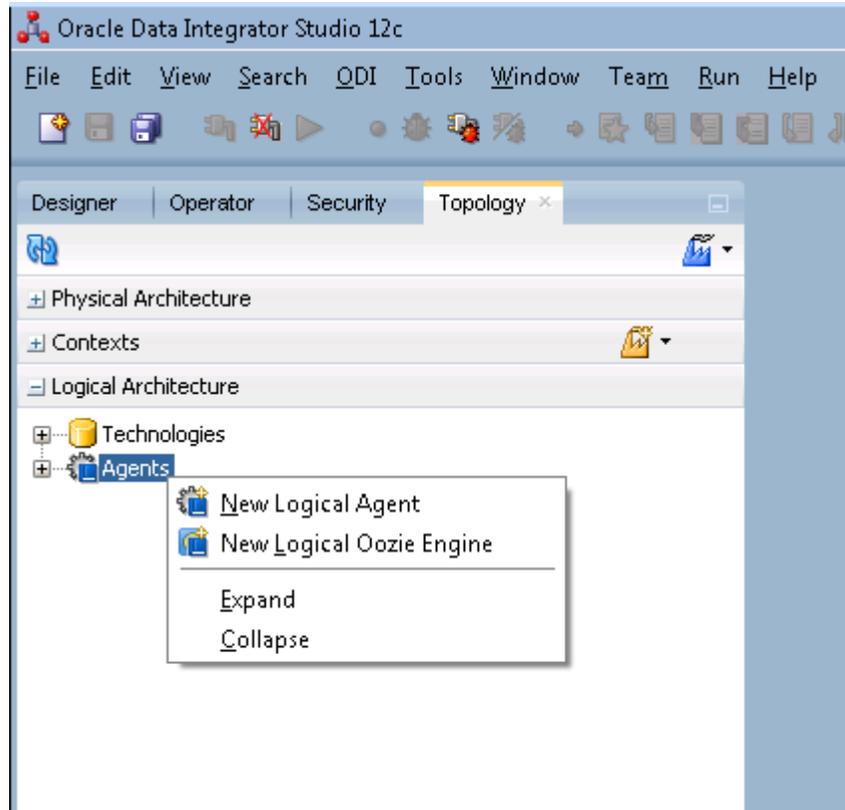
4. Right-click Agent and select New Logical Agent

5. Configure the physical Agent for the new logical agent (OracleDIAgent1)
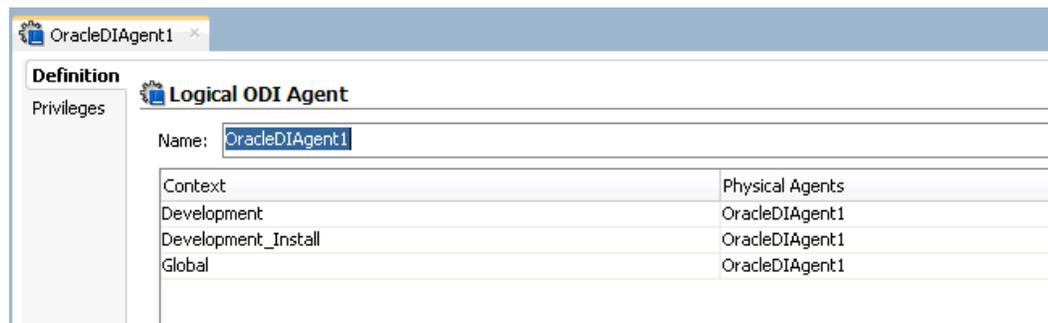
Topology Configuration for Physical and Logical Schemas

To configure schemas, do the following.

Modify the topology configurations by logging into ODI Studio as follows.

1. From a UNIX prompt, execute:

    <$ODIHOME>/studio/odi.sh

2. Connect to the repository and navigate to Topology.

    > **Note:** From the Programs menu, to launch in Windows,
    > select Oracle -> ODI Studio -> Topology.

3. Click the Physical Architecture tab to configure the Physical Schemas.

4. Select Technologies → Oracle.

5. Select the data server ORACLE_BI_APPLICATIONS to Modify the connection details based on your database connectivity credentials.
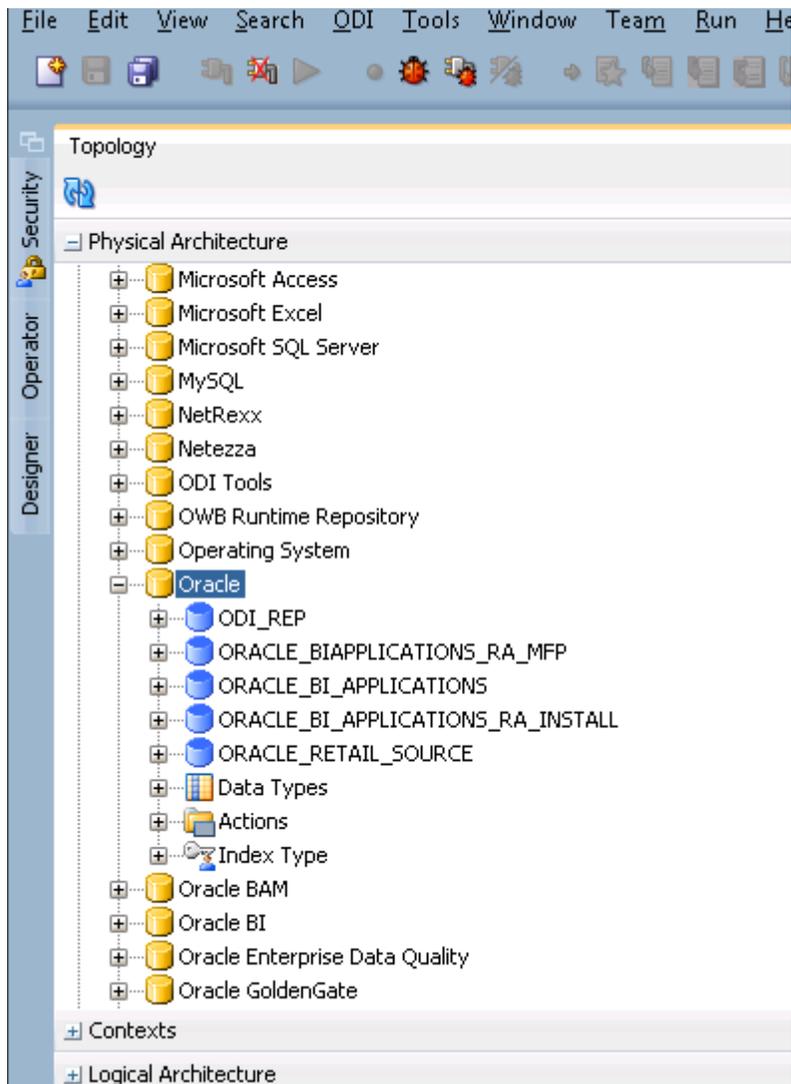
6. Modify the connection setup by clicking **Test Connection**.
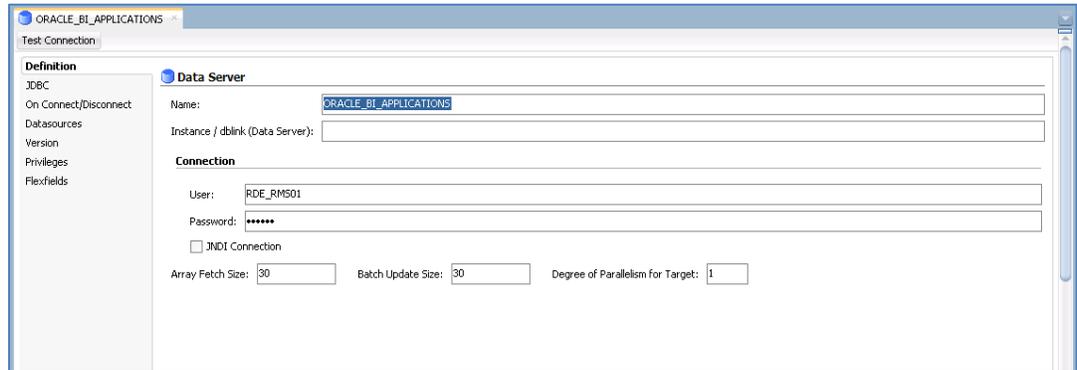
7. Modify the JDBC connection details:

8. Select Yes to save and click **Test Connection**.

9. Select an Agent (Local is acceptable) to test the connection and click **Test**.
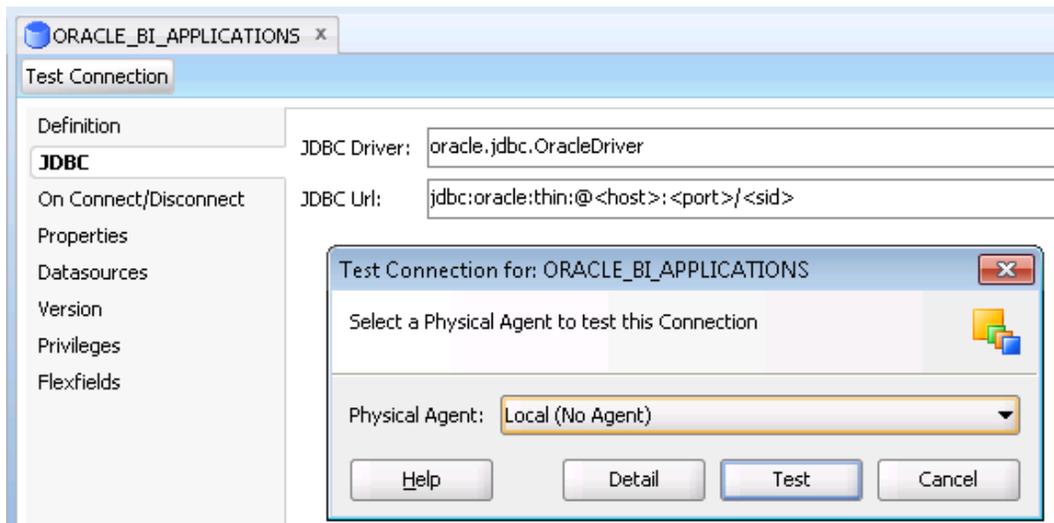
10. When the test is complete, click **OK**. The Data Server is now successfully verified/ modified.
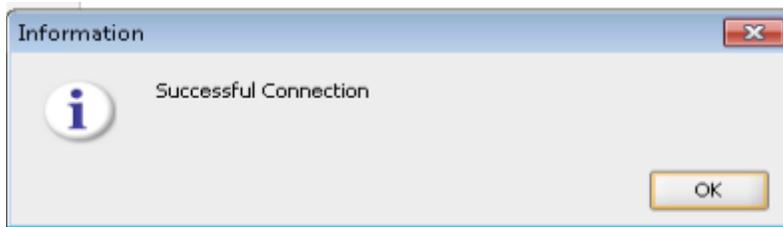
11. Select the data server ORACLE_RETAIL_SOURCE to Modify the connection details based on your database connectivity credentials.
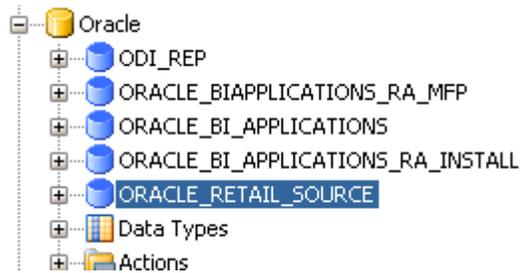
12. Modify the database connection details for the data server.
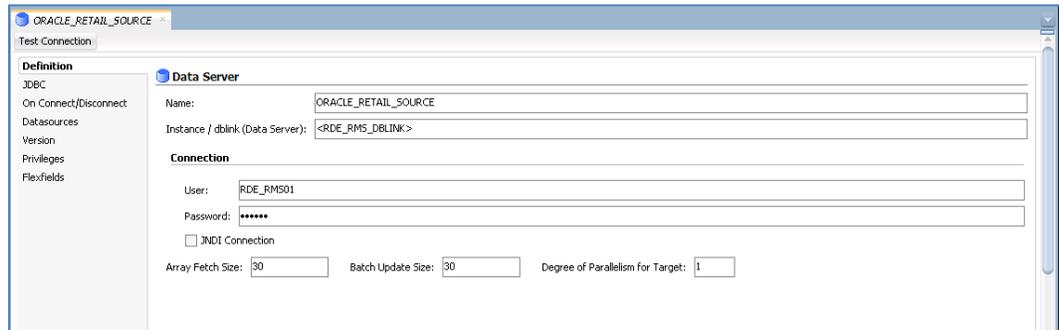
13. Modify the JDBC connection details:

14. Select Yes to save and click **Test Connection**.

15. Select an Agent (Local is acceptable) to test the connection and click **Test**.
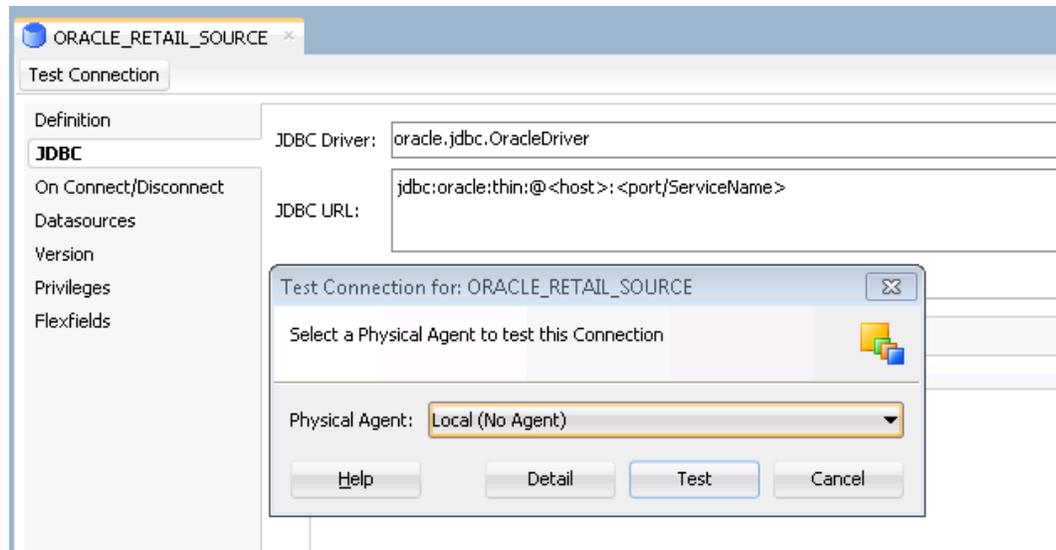
16. When the test is complete, click OK. The Data Server is now successfully modified.

17. Select the data server ORACLE_BI_APPLICATIONS_RA_INSTALL to Modify the connection details based on your database connectivity credentials.
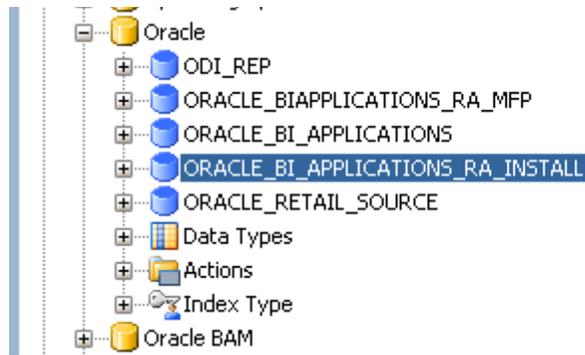
18. Modify the database connection details for the data server.
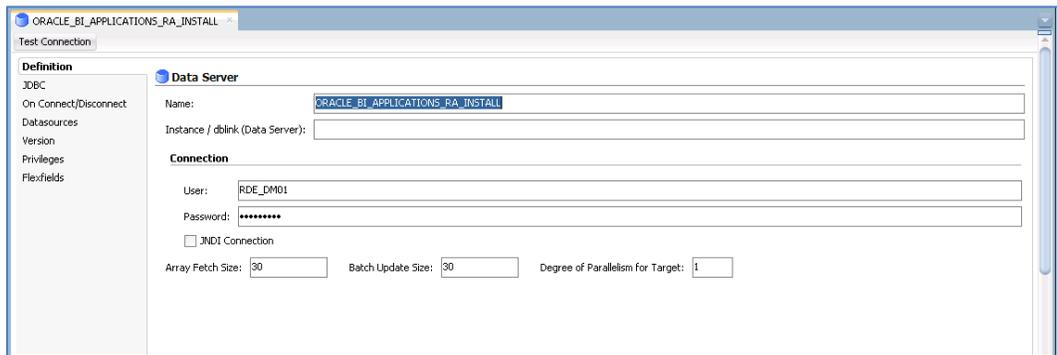
19. Modify the JDBC connection details:
20. Select Yes to save and click **Test Connection**.
21. Select an Agent (Local is acceptable) to test the connection and click **Test**.
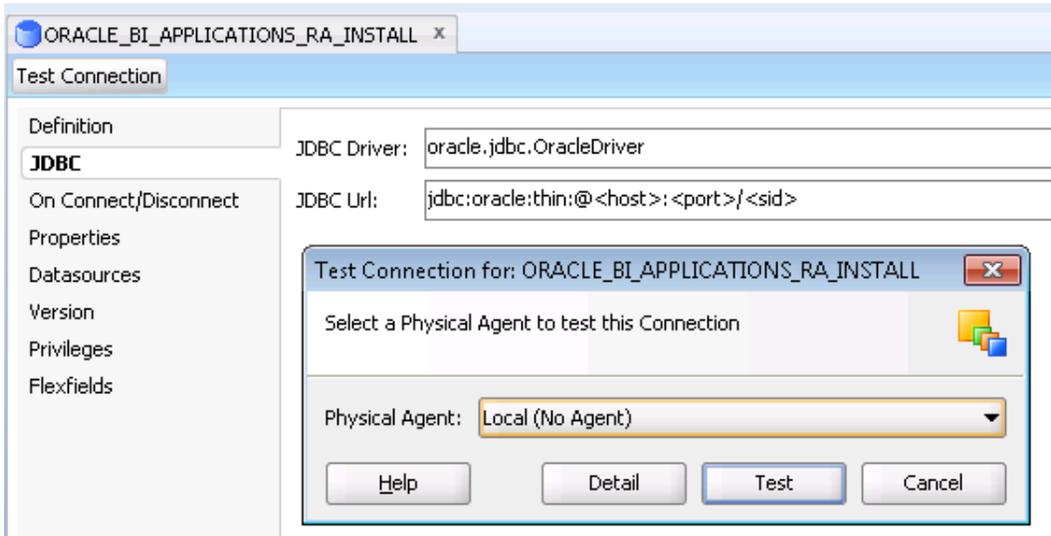
22. When the test is complete, click **OK**. The Data Server is now successfully modified.
23. Select the data server ODI_REP to modify the connection details based on your database connectivity credentials.
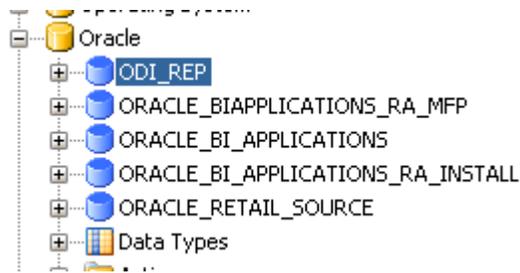
24. Modify the database connection details for the data server.
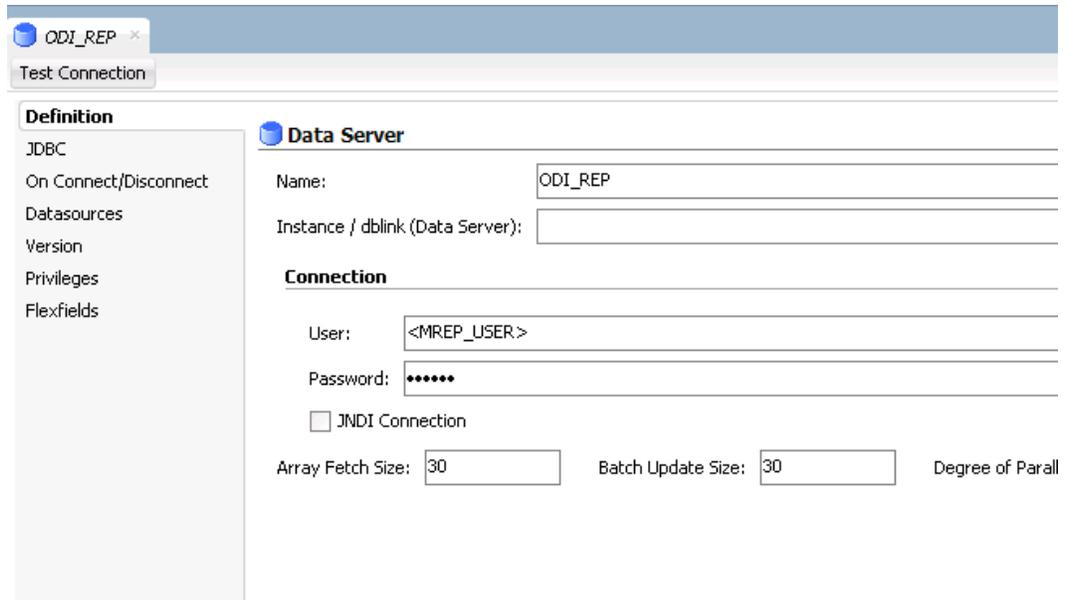
25. Modify the JDBC connection details:

26. Select Yes to save and click **Test Connection**.

27. Select an Agent (Local is acceptable) to test the connection and click **Test**.
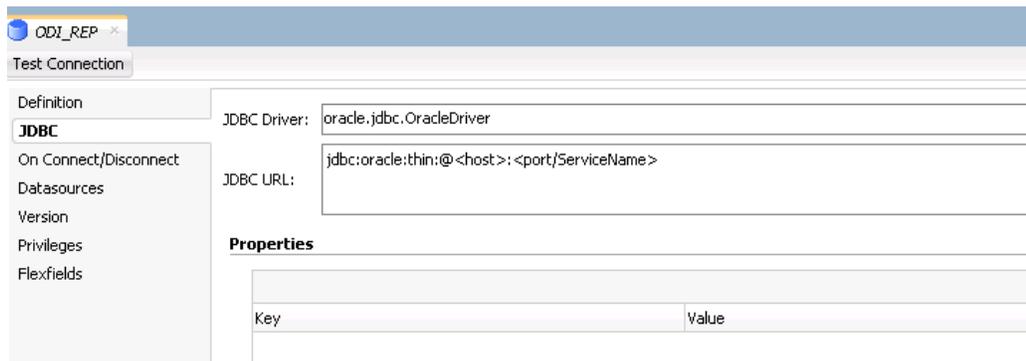
28. When the test is complete, click OK. The Data Server is now successfully modified.

Modify Physical Schemas

The physical schemas inside all data servers are already modified

For example, under ORACLE_BI_APPLICATIONS data server, verify physical schemas for Retail Data Extractor Batch User (for example, RDE_RMS01), Retail Data Extractor Data Mart User (RDE_DM01), and Retail Analytics Data Mining User (RASE) as explained in the following steps.

1. Double-click and select the physical Schema under the data server ORACLE_BI_APPLICATIONS.

2. Modify the database details such as Master Schema and Work Schema as follows :

   a. Modify the schemas from the drop down list (highlighted).
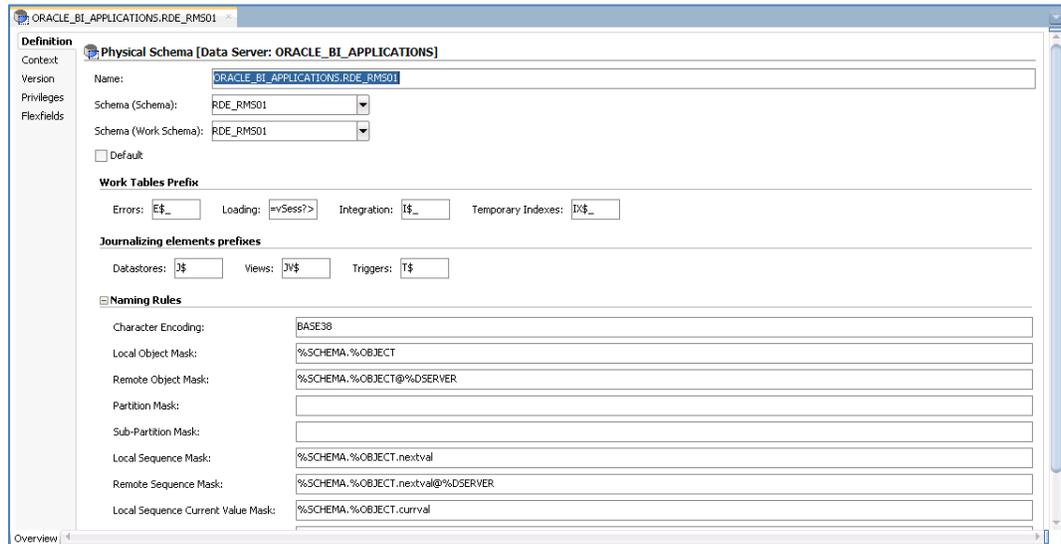
   b. RA Data mart schema should be the default schema set, if not, Choose RDE_DM01 schema as default, to make it a default schema.

   c. Setup the WORK_SCHEMA for Retail Data Extractor Data Mart User (RDE_DM01) is setup with Retail Data Extractor Batch User (for example, RDE_RMS01).
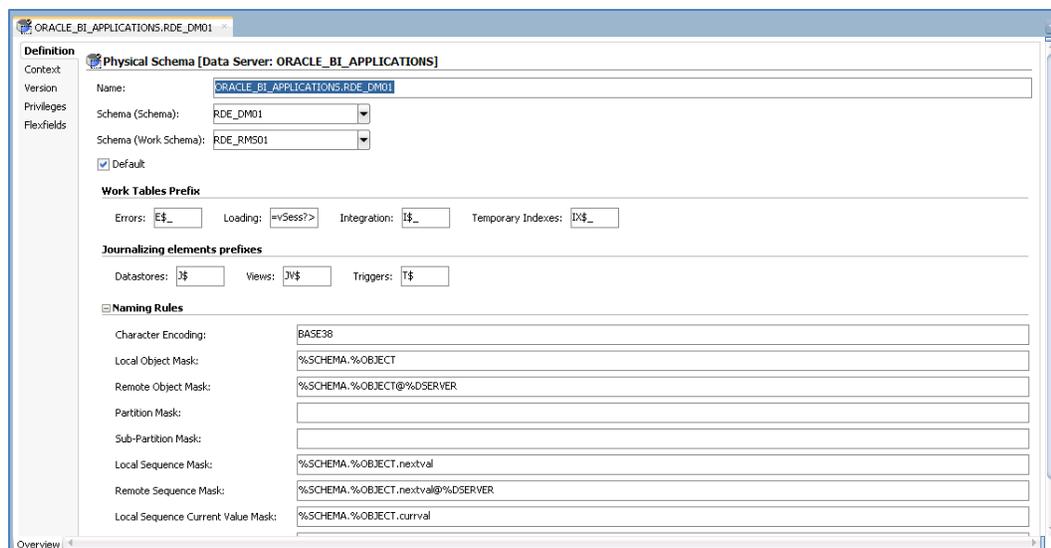
    d. Retain the values for E$, Loading, I$, Data stores, Views, and Triggers options as they are.

3. Double click and select the physical Schema under the data server ORACLE_BI_APPLICATIONS_RA_INSTALL.
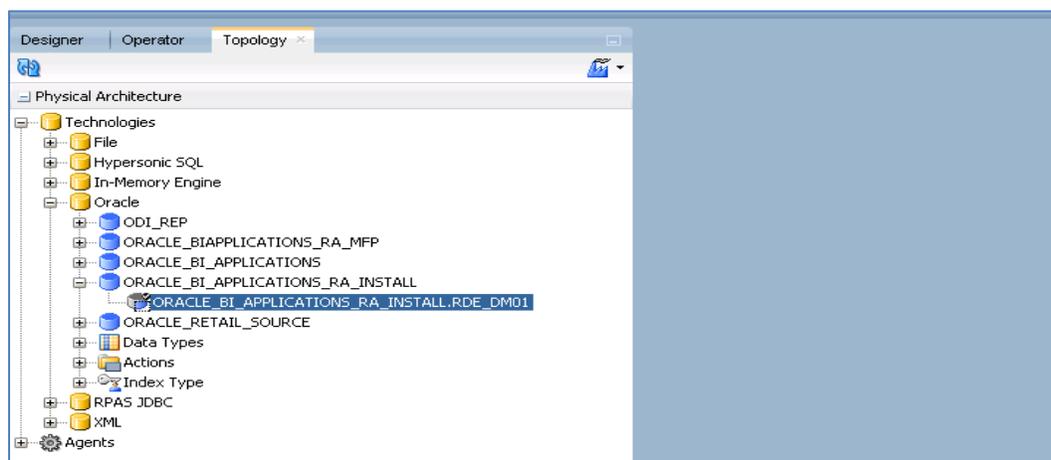
4. Modify the database details such as Master Schema and Work Schema as follows :

    a. Modify the schemas from the drop down list (highlighted).
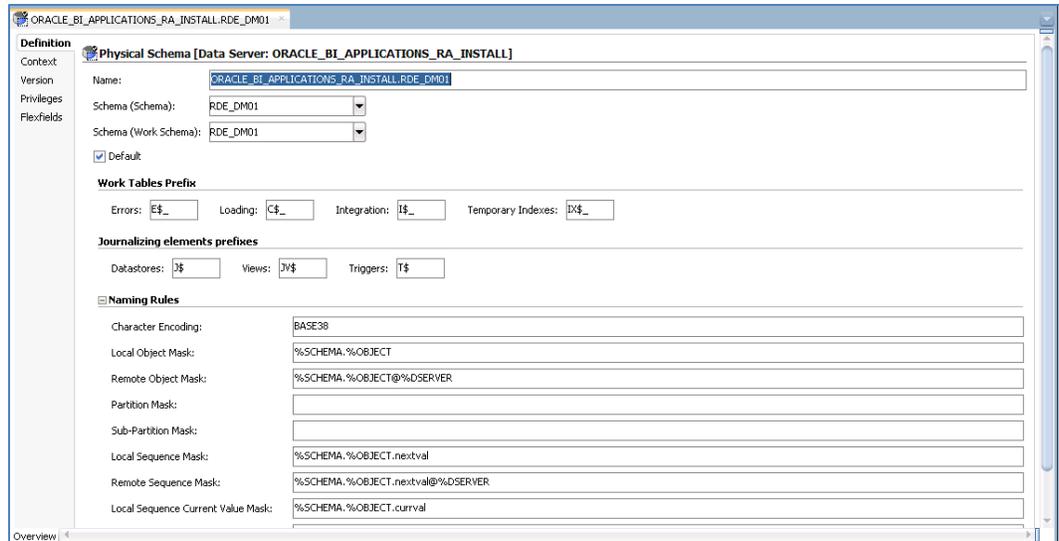
b. RA Data mart schema should be the default schema set, if not, Choose RDE_DM01 schema as default, to make it a default schema.

c. Retain the values for E$, Loading, I$, Data stores, Views, and Triggers options as they are.

5. Double click and select the physical Schema under the data server ORACLE_RETAIL_SOURCE.
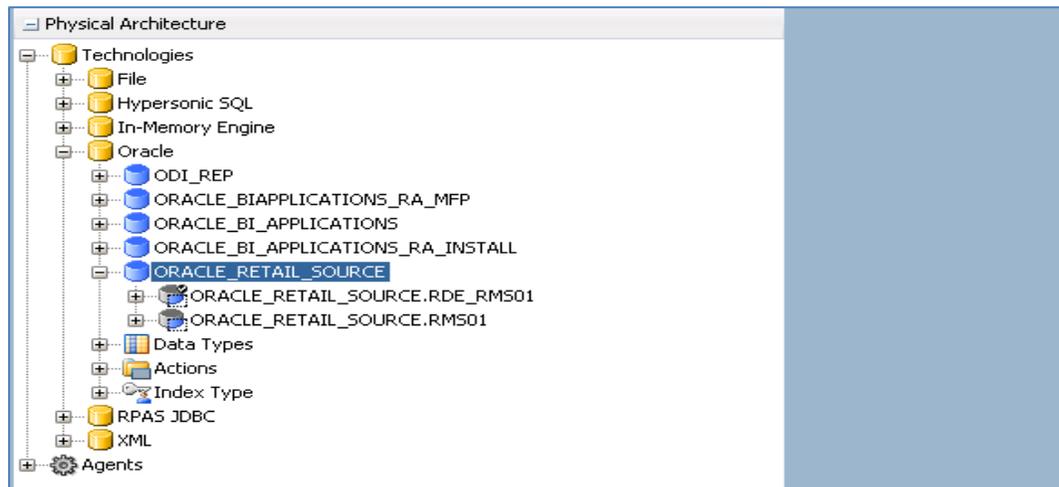
6. Modify the database details such as Master Schema and Work Schema as follows:

a. Modify the schemas from the drop down list (highlighted).
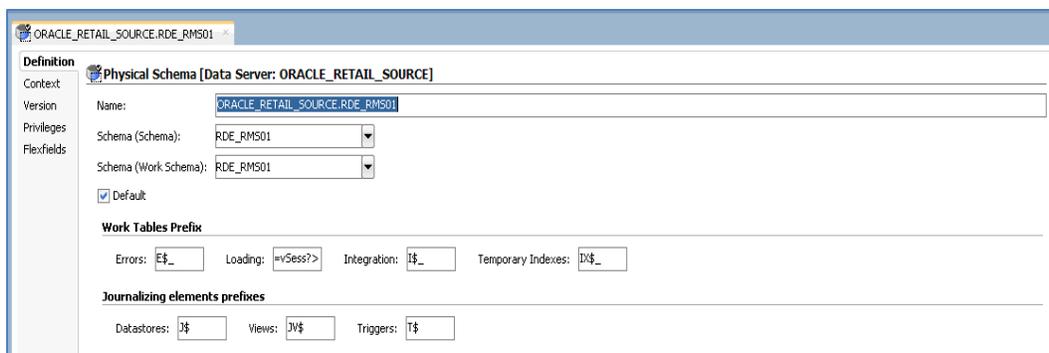
    b. RA RMS schema should be the default schema set, if not, Choose RDE_RMS01 schema as default, to make it a default schema.

    c. Setup the WORK_SCHEMA for Retail Data Extractor RDE_RMS01 schema (RDE_RMS01) is setup with Retail Data Extractor RMS User (for example, RDE_RMS01).
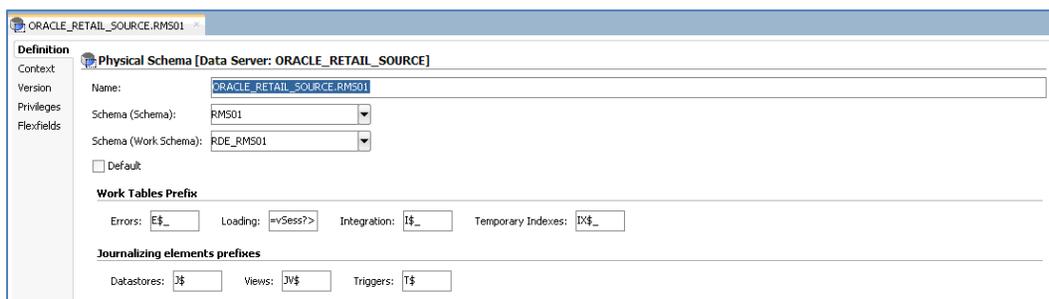
    d. Retain the values for E$, Loading, I$, Data stores, Views, and Triggers options as they are.

7. Double-click and select the physical Schema under the data server ODI_REP.
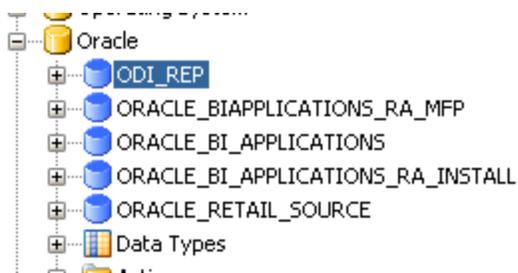
8. Modify the database details such as Master Schema and Work Schema as follows :

    a. Modify the schemas from the drop down list (highlighted).
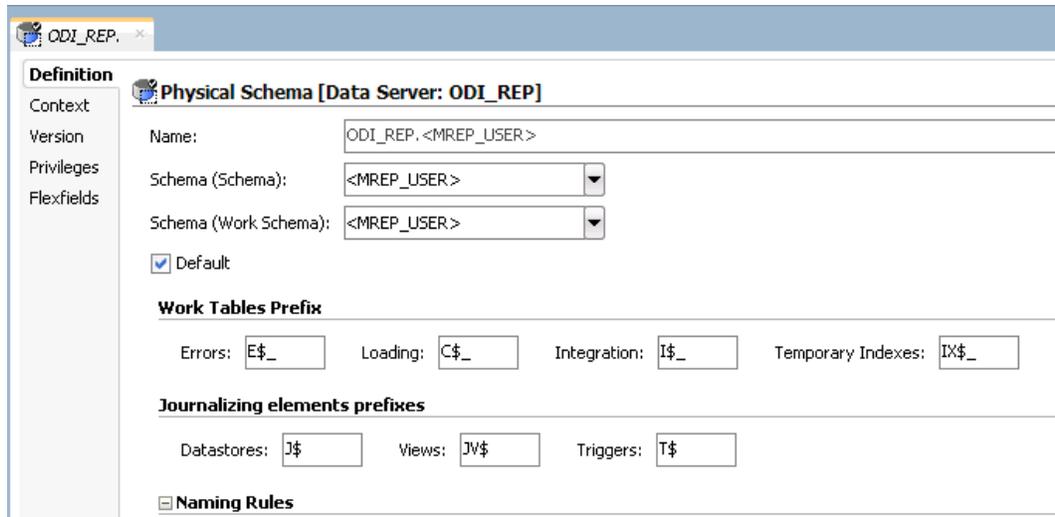
     b. RA Data mart schema should be the default schema set, if not, Choose RDE_DM01 schema as default, to make it a default schema.

     c. Retain the values for E$, Loading, I$, Data stores, Views, and Triggers options as they are.

9. Refer to Contexts, which are already created with the import of the master rep zip file.
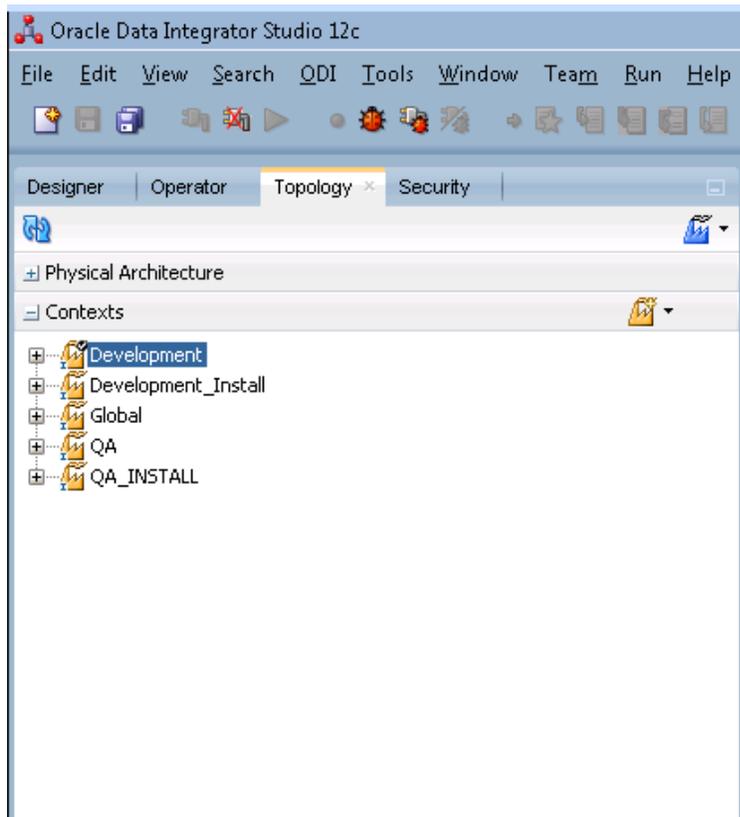
10. Verify the Context Details.

Development ×

Definition
Agents
Schemas
Version
Privileges
Flexfields

Context

Name: Development

Code: DEVELOPMENT

Password:

☑ Default

The following steps must be performed to ensure correct mapping of the Logical schema with the Physical schemas by means of a Context.

11. Edit "Development" Context by going to Topology → Contexts → double click Development → Select Schemas.

Development ×

Definition
Agents
Schemas
Version
Privileges
Flexfields

Logical Schema	Physical Schemas
MEMORY_ENGINE	<Undefined>
SUNOPSIS_MEMORY_ENGINE	SUNOPSIS_MEMORY_ENGINE_Default
BIAPPS_11g_RA	ORACLE_BI_APPLICATIONS.RDE_DM01
BIAPPS_11g_RATEMP	ORACLE_BI_APPLICATIONS.RDE_RMS01
BIAPPS_11g_RA_RSE	ORACLE_BI_APPLICATIONS.<RSEUSER>
FILE_DEMO_SRC	<Undefined>
FILE_PARAMS	FILE_GENERIC.#GLOBAL.GET_ODI_HOME/data/lkpfiles
FILE_RA_INSTALL	FILE_GENERIC.#GLOBAL.GET_ODI_HOME/file/ra/install
FILE_RA_SOURCE_INSTALL	FILE_GENERIC.#GLOBAL.GET_ODI_HOME/file/ra_source/install
FILE_SOURCE	FILE_GENERIC.#GLOBAL.GET_ODI_HOME/data/srcfiles
File Server for SAP ABAP	<Undefined>
HSQL_DEMO_SRC	<Undefined>
HSQL_DEMO_TARG	<Undefined>
ODI_REP	ODI_REP.RDE16ODI_MREP_DEV
RETAIL_SOURCE	ORACLE_RETAIL_SOURCE.RDE_RMS01
RETAIL_SOURCETEMP	ORACLE_RETAIL_SOURCE.RDE_RMS01
RPAS JDBC MFPCost	RPAS JDBC_MFPCST.domain
RPAS JDBC MFPRetail	RPAS JDBC_MFPRTL.domain
XML_DEMO_GEO	XML_GEO_DIM.GEO

12. For the BIAPPS_11g_RATEMP logical schema, map it to a physical schemas value of ORACLE_BI_APPLICATIONS.RDE_RMS01 from the drop down as shown below if not done already.

13. For the BIAPPS_11g_RA logical schema, map it to a physical schemas value of ORACLE_BI_APPLICATIONS.RDE_DM01 from the drop down as shown below if not done already.

14. For the RETAIL_SOURCE logical schema, map it to a physical schemas value of ORACLE_RETAIL_SOURCE. RDE_RMS01 SCHEMA from the drop down.

15. For RETAIL_SOURCETEMP logical schema, map it to a physical schema value of ORACLE_RETAIL_SOURCE.RDE_RMS01 SCHEMA.

16. Click **Save** when prompted to save your changes.

17. Edit Development_Install context. Go to Topology → Contexts and double click Development_Install → Select Schemas.
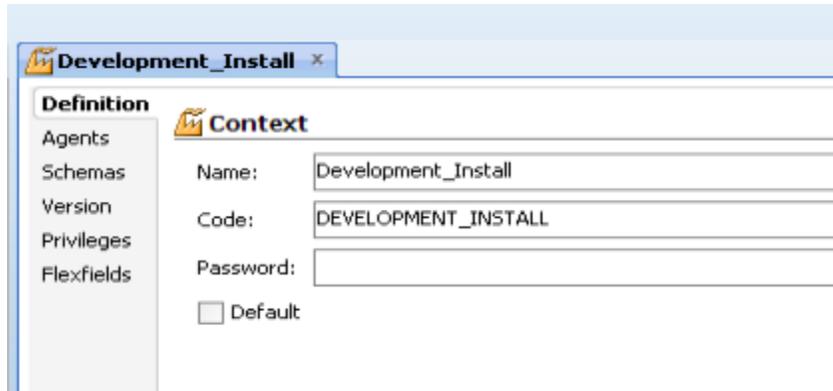
18. For the BIAPPS_11g_RATEMP logical schema, map it to a physical schema value of ORACLE_BI_APPLICATIONS.RDE_DM01 from the drop down as shown below if not done already.

19. For the BIAPPS_11g_RA logical schema, map it to a physical schemas value of ORACLE_BI_APPLICATIONS_RA_INSTALL.RDE_DM01 from the drop down as shown below if not done already.

20. For the RETAIL_SOURCE logical schema, map it to a physical schemas value of ORACLE_RETAIL_SOURCE.RMS MASTER SCHEMA from the drop down.

21. For RETAIL_SOURCETEMP logical schema, map it to a physical schema value of ORACLE_RETAIL_SOURCE.RDE_RMS01 SCHEMA.
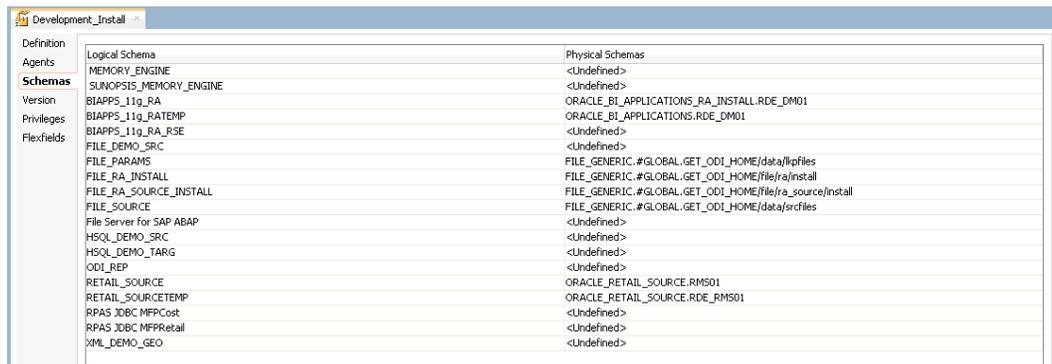
22. Click **Save** when prompted to save your changes.

23. Repeat the above step for GLOBAL context.

File Configuration in Topology Manager

1. Open ODI Topology.

   - From a UNIX prompt, execute

     ./ <$ODIHOME>/studio/odi.sh
     Navigate to Topology

   - In Windows, from the Programs menu, select Oracle -> ODI Studio -> Topology.

2. From the Topology manager, select Physical Architecture and choose Technologies=File.

3. Check for file data server that is already imported with the import of the master rep zip file.
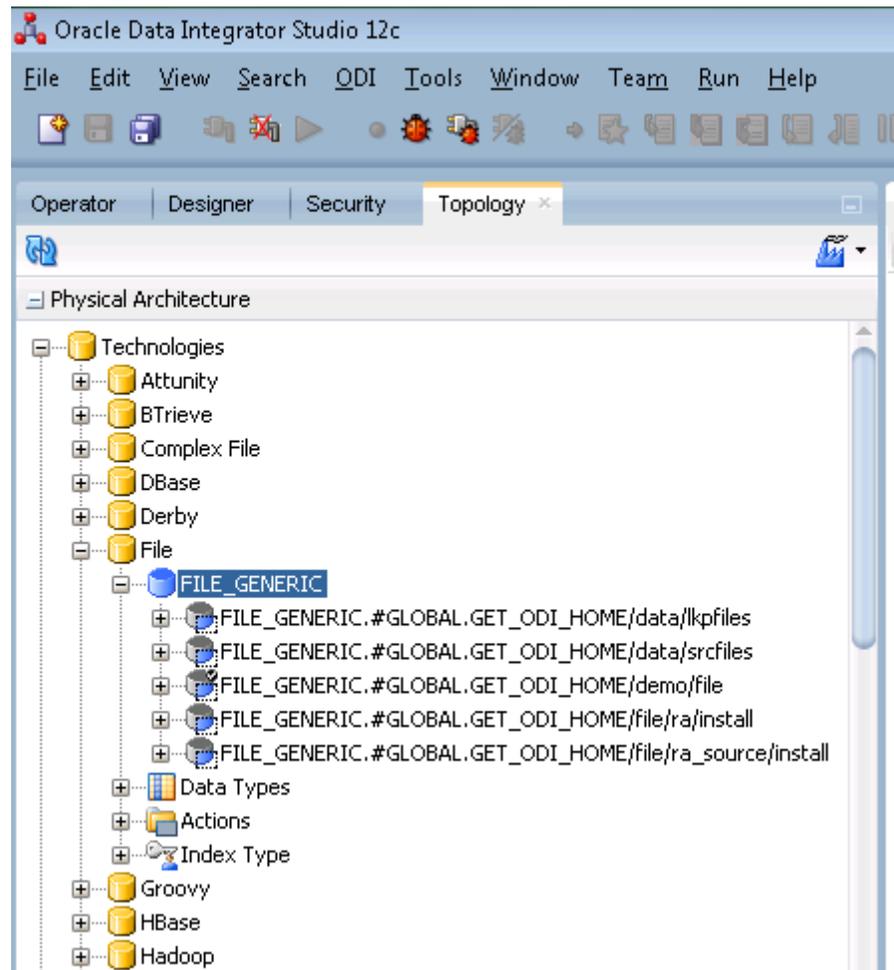
4. Modify the Data Server details, as necessary.

> **Note:** For File Technology you may not need to enter the user/password.
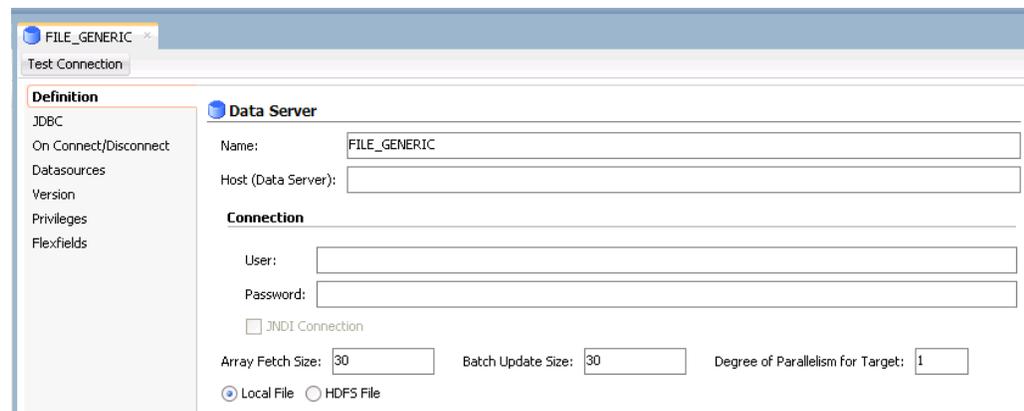
5. Modify the JDBC information in the JDBC tab.
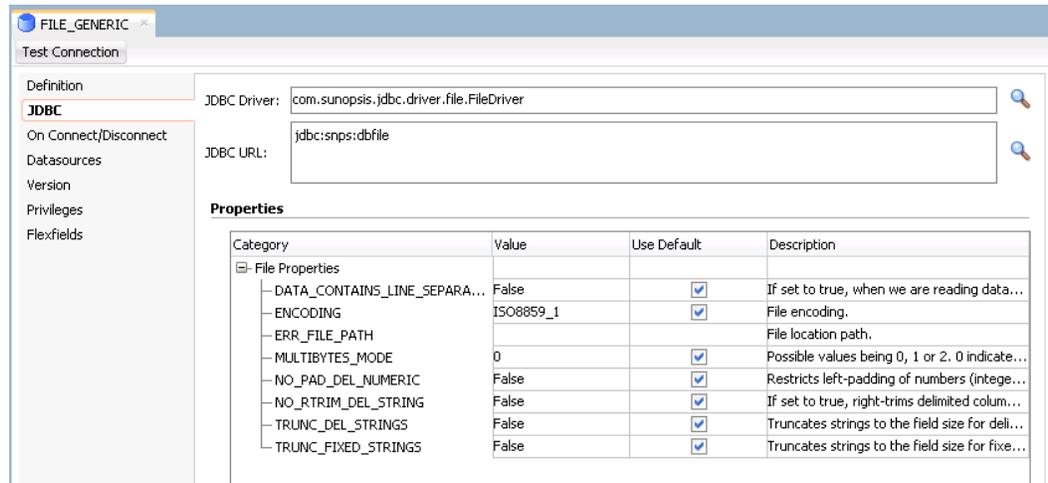
6. Select an Agent to test the connection and click **Test**.
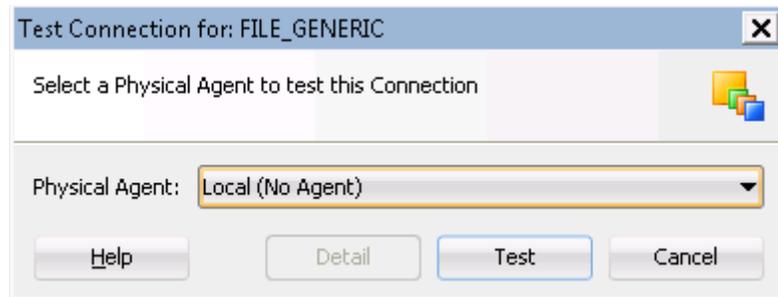
7. When the test is complete, click **OK**.
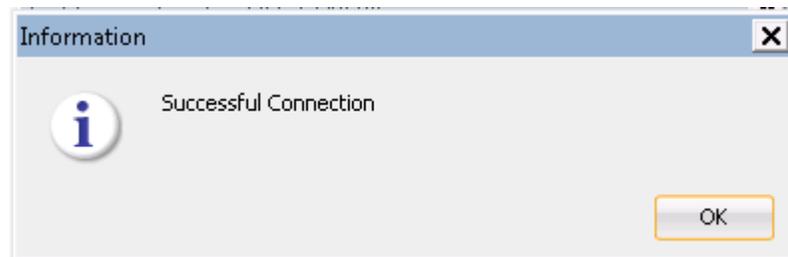
8. Click **OK** to save.

9. Check for the file physical schema under the Data Server:

10. Enter the details of the file physical schema if they need to be modified or updated.

11. Specify the directory for parent and work schemas where the file is located. Keep the default values for E$, loading and I$ fields as defined by ODI.

> **Note:** File name:
>
> FILE_GENERIC is the name of the data server.
>
> GET_ODI_HOME function extracts the path of $MMHOME. The path /data/lkpfiles is the directory under MMHOME with the respective source file.

12. Close this window to complete the setup.
13. To check the logical schemas, go to the Logical Architecture tab.
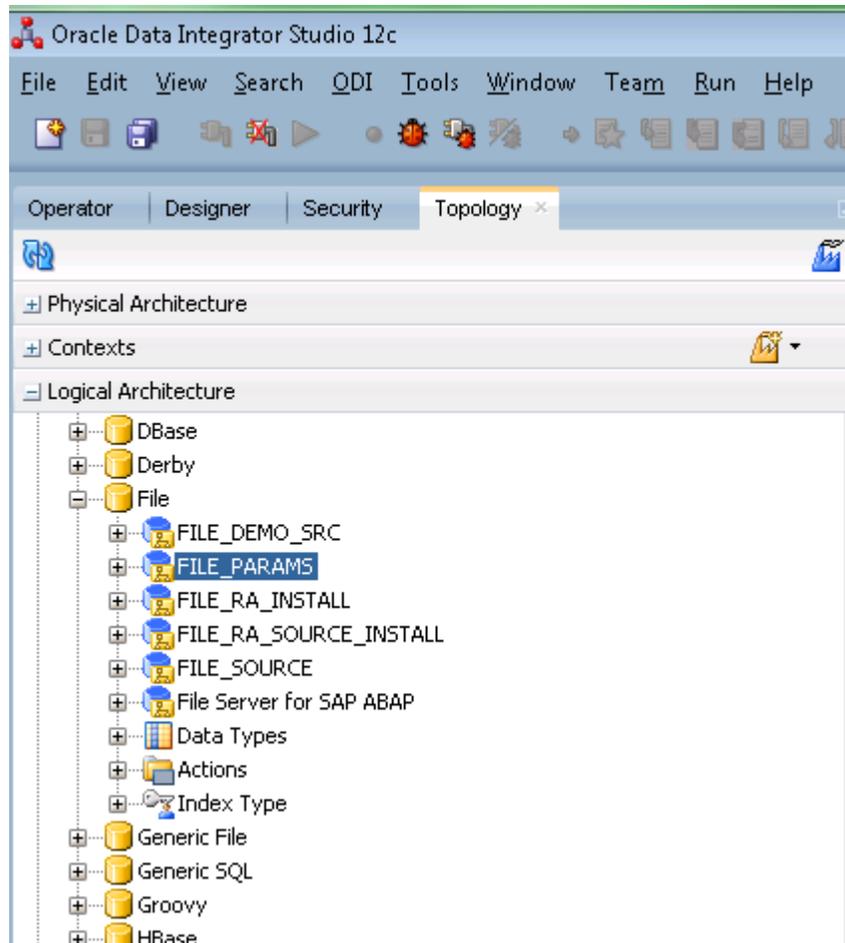
14. Validate the logical schema details.
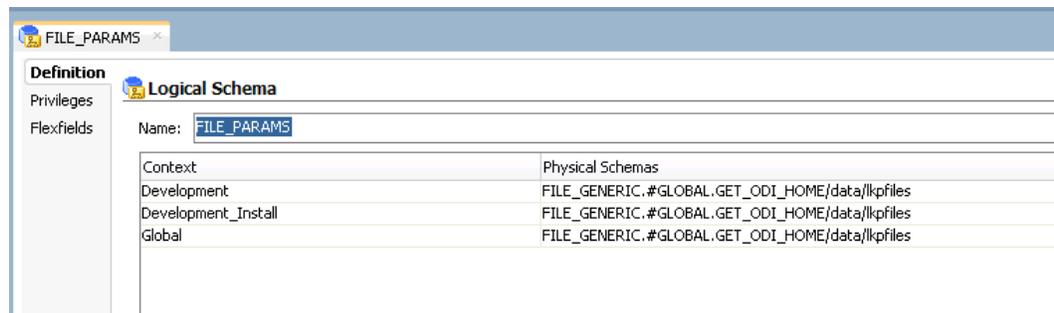
15. Click **OK** to save.

The FILE topology configuration is now complete.

> **Note:** Because the CONTEXT was already created, use the
> same context throughout the development/QA/production
> life cycle.

Configure ODI Designer

Perform the following procedure to configure the ODI Designer:

1. Click the ODI Designer tab.
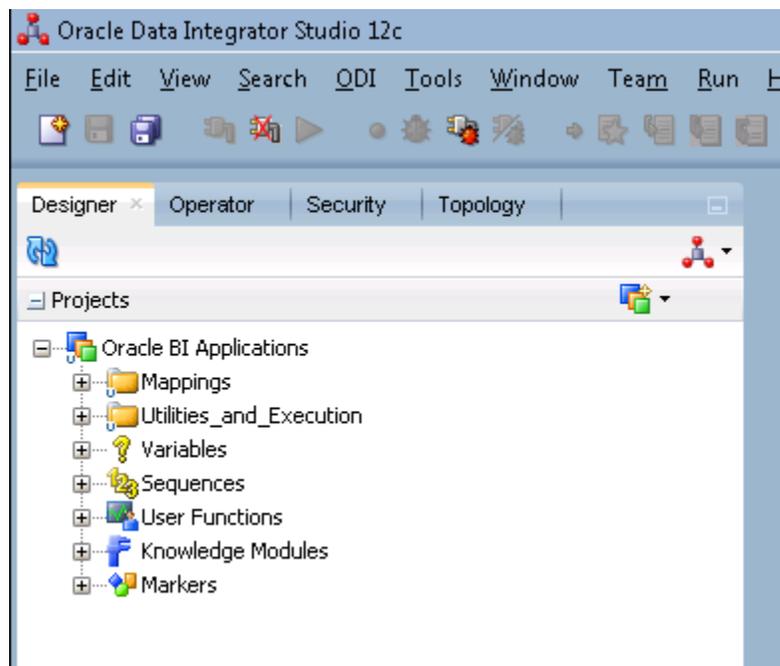
2. Regenerate the Scenarios as follows with the following Unix command:

3. Execute the following command: <ODI_DOMAIN_HOME>/bin/startcmd.sh -INSTANCE=<odi agent name> OdiGenerateAllScen -PROJECT=RA_BI -MODE=REPLACE

Example:

/u01/domains/wls_odi/ODIDomain/bin/startcmd.sh -INSTANCE=OracleDIAgent1 OdiGenerateAllScen -PROJECT=RA_BI -MODE=REPLACE

> **Note:** Ensure the ODI_REP is setup with MREP connection details in topologies before regenerating the scenarios.

Retail Data Extractor Seed Data Setup

This section describes the steps required for Retail Data Extractor seed data setup.

> **Note:** To configure the environment file ra.env (MMHHOME/etc/ra.env) with the correct parameters, contact the administrator for the ODI Home and Oracle Home parameters. ODI Home can be the same as Java Home.

> **Note:** All the seed and general scripts must be executed under $MMHOME/src folder in UNIX.

By default the following permissions are given to users to access files packaged with Retail Data Extractor once installation is complete.

- All Retail Data Extractor scripts should at least have 750 permission
- All configuration files should at least have 660 permission
- All static data (csv files) should at least have 640 permission

From the permission above besides owner (the installer user), the group member can also view and execute scripts, he can also read and modify the configuration files. The group member can also read the static file. A user out of group cannot do anything to RI files and explicit permission needs to be given by the Admin to users outside of group.

Seed Control Load for Source Dependent Extracts (SDE)

Complete the following steps.

1. Verify the data for PARAM_NAME as necessary in <$ODIHOME> /file/ra_source/install/C_ODI_PARAM.csv.

   GLOBAL~001~SRC_BASE_HOME,< MMHOME>

2. Changing ORG_ID and CALENDAR_ID values is not required; you can use the default C_ODI_PARAM.csv.

3. Modify PARAM_VALUE data for the following PARAM_NAME as necessary in <$ODIHOME>/ file/ra_source/install/RA_SRC_CURR_PARAM_G.csv where

   - VDATE is the current business date
   - NEXT_VDATE is the next date after the current business date
   - LAST_EOM_DATE is the last date of current month
   - CURR_BOM_DATE is the beginning date of the current month

4. Run SDE_RetailLoadControlSeedData.ksh from UNIX to load seed data into the Retail Data Extractor source table C_ODI_PARAM and RA_SRC_CURR_PARAM_G.

5. Validate the table RA_TRUNCATE_TBL for the OWNER information to truncate RADM tables from RA Batch user. By default RA_TRUNCATE_TBL.OWNER_NAME is set to RADM01 as the DM user name; if this user name is different please update OWNER_NAME in the table with DM user name, before starting any batch executions.

Preload Retail Data Extractor Business Calendar

This section describes the loading of time into Retail Data Extractor. The time dimension can be loaded with a 454 calendar, 13 period time calendar or a 454 with Gregorian calendar. Populate these tables according to business requirements. If RMS is implemented, the time dimension with 454-calendar time or 454 with Gregorian calendar the calendar information can be extracted from this system.

Extract the 4-5-4 or 4-5-4 with Gregorian Time Calendar

If RMS is being used as the source of the time calendar, run Retail Data Extractor time calendar SDE program mcalperiodsde.ksh under $MMHOME/src directory. Refer to the *Oracle Retail Data Extractor Operations Guide* on how to execute this program.

If RMS is not being used as Retail Data Extractor source system, users have to manually populate Retail Data Extractor business calendar staging table W_MCAL_PERIOD_DS. Refer to the *Oracle Retail Data Extractor Operations Guide* for API of this staging table.

Retail Data Extractor Configuration

Operating System

Server OS Configuration

ODI

ODI admin OS user can be different from a Retail Data Extractor batch OS user, however, this admin user should belong to the same user group as Retail Data Extractor batch OS user. The access permission to those files installed through ODI installation should be modified to group read and execution. World readable files should not be allowed in product environment. For example, the permission of odiparams.sh should be modified to 750, so it will not have world read permission or execute permission.

Seed Data Setup

See the Retail Data Extractor Seed Data Setup section for how to seed Retail Data Extractor initial data.

Retail Data Extractor initial loading data files (csv files) that are located under $MMHOME should be owned by Retail Data Extractor batch user and installed with750 permission. No world readable files should be allowed.

Retail Data Extractor shell scripts, error files, and log files that are located under $MMHOME should be owned by Retail Data Extractor batch user and installed with 750 permission. No world readable files should be allowed.

File (ODI_Post_Install.txt) which contain password to protect Retail Data Extractor ODI code and should be owned by Retail Data Extractor batch user and installed with 600 permission. Password in ODI should be changed after installation is done.

Oracle BI EE Server

Infrastructure/Middleware

ODI

JDBC connection

Oracle Database access through the ODI tool can be secured by using a JDBC connection. The JDBC connection can be set up in both the ODI client and in the ODI executable script.

In the ODI client under Topology Oracle Technology, select JDBC jdbc driver and jdbc url for all Data Servers used by Retail Data Extractor. Login credentials will also be provided here with an encrypted password. The database and ODI credential will be stored securely through ODI.

The JDBC setup in the ODI executable file is through the ODI file odiparams.sh which is under the $ODI_HOME/bin directory. ODI and Oracle database login credential will be provided in this file with encrypted password. The password encryption and login credential are stored securely through ODI.

See the *Oracle Retail Merchandising Security Guide* Release 16.0.2, section "Configuring SSL on the Oracle Data Integrator (ODI)" for more information on configuring SSL for an ODI Agent.

See the "Oracle Data Integrator Configuration Tasks" section for details on how to configure the ODI database connection.

Database

Out of box, Retail Data Extractor does not use any default username or password for Oracle database login. No default account is used. All Oracle Retail Data Extractor database users and users credential are created during the installation.

For details on each database user access permission, see the "Create Retail Data Extractor Schema Owners" section in chapter 2.

RGBU Application Configuration

Technology Considerations

The Retail Data Extractor application requires OS, database, application server, report/analytical engine, and ETL component. Oracle ODI is used as ETL tool to load and transform data. Oracle Database is used as database to store Retail Data Extractor data. For OS, refer to the "Preinstallation Tasks" section. During the installation, users should use the recommended secure configurations from these products for application deployment such as SSO and SSL.

Application Specific Configurations

Retail Data Extractor Non Configurable Parameter List

The Retail Data Extractor Non Configurable Parameter List includes parameters used by the Retail Data Extractor programs. They are not configurable to batch users.

All these parameters are stored in the C_ODI_PARAM table either in the Retail Data Extractor data schema.

Scenario Name	Param Name	Schema	Comment
GLOBAL	ORG_ID	RDE	This is the organization ID for the retailer. It will be used for calendar and other dimension lookup. It is set to '1' for current release.
GLOBAL	CALENDAR_ID	RDE	This is the ID of the calendar that retailer is going to use. It is set to 'Retail Calendar~41' for current release.
GLOBAL	RA_STORE_COMP_NUM_OF_OPEN_DAYS	RDE	This is the number of open days for the store, set to 365 for this release
GLOBAL	RA_SRC_CATEGORY	RDE	It is set to 'RETAIL' for the current release.
GLOBAL	RA_SRC_SUPPLIER_ACTIVE_DESC	RDE	It is set to 'Active' for current release.
GLOBAL	RA_SRC_SUPPLIER_INACTIVE_DESC	RDE	It is set to 'Inactive' for current release.

Scenario Name	Param Name	Schema	Comment
GLOBAL	RA_SRC_STORE_TYPE_FRANCHISE	RDE	It is set to 'Franchise' for current release.
GLOBAL	RA_SRC_STORE_TYPE_WHOLESALE	RDE	It is set to 'Wholesale' for current release.
GLOBAL	RA_SRC_STORE_TYPE_COMPANY	RDE	It is set to 'Company' for current release.
GLOBAL	RA_SRC_LOCATION_TYPE_NAME_STORE	RDE	It is set to 'STORE' for current release.
GLOBAL	RA_SRC_LOCATION_TYPE_NAME_WH	RDE	It is set to 'WAREHOUSE' for current release.
GLOBAL	RA_SRC_LOCATION_TYPE_NAME_PARTNER	RDE	It is set to 'PARTNER' for current release.
GLOBAL	WHOLESALE_CHANNEL	RDE	It is set to 'Y' for current release.
GLOBAL	CHANGE_ON_DT	RDE	
GLOBAL	INVADJ_REASON_CAT_CODE	RDE	This is the string code for reason category used for RA inventory adjustment. It is set to 'RTL_INVADJ_REASON'
GLOBAL	RTVR_REASON_CAT_CODE	RDE	This is the string code for Return to Vendor reason category used for RA RTVR. It is set to 'RTVR'
GLOBAL	RTL_STATUS_CLASS	RDE	This is the string code for status class used for RA. It is set to 'RTL_INV_STATUS'.
GLOBAL	REASON_CLASS	RDE	This is the string code for reason class used for RA reason code. It is set to 'RTL_REASON'
GLOBAL	INVADJ_REASON_CAT_NAME	RDE	This is the string code for reason category name used for RA inventory adjustment. It is set to 'Inventory Adjustment Reasons'
SDE_RETAILCOLORDIMENSION	COLOR	RDE	It is set to 'C' for current release.
SDE_RETAILDOMAINMEMBERLKUP	COLOR	RDE	It is set to 'C' for current release.
SDE_RETAILDOMAINMEMBERLKUP	FABRIC	RDE	It is set to 'FAB' for current release.
SDE_RETAILDOMAINMEMBERLKUP	SCENT	RDE	It is set to 'SC' for current release.
SDE_RETAILDOMAINMEMBERLKUP	STYLE	RDE	It is set to 'STY' for current release.
SDE_RETAILDOMAINMEMBERLKUP	SIZE	RDE	It is set to 'S' for current release.

Scenario Name	Param Name	Schema	Comment
SDE_RETAILITEMATTRDIMENSION	FLAVOR	RDE	It is set to 'F' for current release.
SDE_RETAILITEMATTRDIMENSION	FABRIC	RDE	It is set to 'FAB' for current release.
SDE_RETAILITEMATTRDIMENSION	SCENT	RDE	It is set to 'SC' for current release.
SDE_RETAILITEMATTRDIMENSION	STYLE	RDE	It is set to 'STY' for current release.
SDE_RETAILITEMATTRDIMENSION	SIZE	RDE	It is set to 'S' for current release.
SDE_RETAILITEMDIFFDIMENSION	BRAND	RDE	It is set to 'B' for current release.
SDE_RETAILITEMDIFFDIMENSION	COLOR	RDE	It is set to 'C' for current release.
SDE_RETAILITEMDIFFDIMENSION	FLAVOR	RDE	It is set to 'F' for current release.
SDE_RETAILITEMDIFFDIMENSION	FABRIC	RDE	It is set to 'FAB' for current release.
SDE_RETAILITEMDIFFDIMENSION	SCENT	RDE	It is set to 'SC' for current release.
SDE_RETAILITEMDIFFDIMENSION	STYLE	RDE	It is set to 'STY' for current release.
SDE_RETAILITEMDIFFDIMENSION	SIZE	RDE	It is set to 'S' for current release.
SDE Scenarios	TARGET_TABLE_NAME	RDE	This is to indicate the target table that the scenario will populate. It is used by the Retail Data Extractor programs and should not be modified by a batch user.
SDE Scenarios	EXECUTION_ID	RDE	It is set to '1' and is not used in the current release.

Retail Data Extractor Configurable Parameter List

The Retail Data Extractor configurable parameter includes a list of parameters that are used by either one program or a group of programs if the Scenario Name is 'GLOBAL'. The value of each 'GLOBAL' parameter will be the same even it can be used by different programs.

All these parameters are stored in the C_ODI_PARAM table either in the Retail Data Extractor data schema.

Scenario Name	Param Name	Schema Used	Comment
SDE_RETAILITEMDIMENSION	IS_INCREMENTAL	RDE	This is to indicate if the incremental approach or the snapshot approach is going to be used for item dimension during daily ETL loading. The valid value is 'Y', 'N'.
GLOBAL	SRC_BASE_HOME	RDE	This is the MMHOME where SDE programs are located
SDE_RETAILITEMSUPPLIERDIMENSION	IS_INCREMENTAL	RDE	This is to indicate if the incremental approach or the snapshot approach is going to be used for item dimension during daily ETL loading. The valid value is 'Y', 'N'.
SDE_RETAILITEMLOCATIONRANGEDIMENSION	IS_INCREMENTAL	RDE	This is to indicate if the incremental approach or the snapshot approach is going to be used for item dimension during daily ETL loading. The valid value is 'Y', 'N'.
SDE_RETAILSALESFCDYFACT	LOC_NUM_OF_THREAD	RDE	This is to indicate the number of threads program runs into to setup Multi threading to gain performance.
SDE_RETAILSALESFCWKFACT	LOC_NUM_OF_THREAD	RDE	This is to indicate the number of threads program runs into to setup Multi threading to gain performance.
SDE_RETAILSUPPLIERINVOICEMATCHFACT	LOC_NUM_OF_THREAD	RDE	This is to indicate the number of threads program runs into to setup Multi threading to gain performance.
SDE_RETAIL_SALESMARKDOWNFACT	LOC_NUM_OF_THREAD	RDE	This is to indicate the number of threads program runs into to setup Multi threading to gain performance.
SDE_RETAILINVRECEIPTSFACT	LOC_NUM_OF_THREAD	RDE	This is to indicate the number of threads program runs into to setup Multi threading to gain performance.

Scenario Name	Param Name	Schema Used	Comment
SDE_RETAILINVPOSITIONTRANSACTIONFACT	LOC_NUM_OF_THREAD	RDE	This is to indicate the number of threads program runs into to setup Multi threading to gain performance.
SDE_RETAILBASECOSTFACT	LOC_NUM_OF_THREAD	RDE	This is to indicate the number of threads program runs into to setup Multi threading to gain performance.
SDE_RETAILNETCOSTFACT	LOC_NUM_OF_THREAD	RDE	This is to indicate the number of threads program runs into to setup Multi threading to gain performance.
SDE_RETAILWHOLESALEFRANCHISEFACT	LOC_NUM_OF_THREAD	RDE	This is to indicate the number of threads program runs into to setup Multi threading to gain performance.
SDE_RETAILPRICEFACT	LOC_NUM_OF_THREAD	RDE	This is to indicate the number of threads program runs into to setup Multi threading to gain performance.
SDE_RETAILSUPPLIERCOMPLIANCEUFFACT	LOC_NUM_OF_THREAD	RDE	This is to indicate the number of threads program runs into to setup Multi threading to gain performance.
SDE_RETAILSTOCKLEDGERWEEKFACT	LOC_NUM_OF_THREAD	RDE	This is to indicate the number of threads program runs into to setup Multi threading to gain performance.
SDE_RETAILSUPPLIERCOMPLIANCEFACT	LOC_NUM_OF_THREAD	RDE	This is to indicate the number of threads program runs into to setup Multi threading to gain performance.
SDE_RETAILSTOCKLEDGERMONTHFACT	LOC_NUM_OF_THREAD	RDE	This is to indicate the number of threads program runs into to setup Multi threading to gain performance.
SDE_RETAIL_SALESTRANSACTIONFACT	LOC_NUM_OF_THREAD	RDE	This is to indicate the number of threads program runs into to setup Multi threading to gain performance.
SDE_RETAILINVENTORYTRANSFERFACT	LOC_NUM_OF_THREAD	RDE	This is to indicate the number of threads program runs into to setup Multi threading to gain performance.
SDE_RETAILINVENTORYRETURNTOVENDORFACT	LOC_NUM_OF_THREAD	RDE	This is to indicate the number of threads program runs into to setup Multi threading to gain performance.

Scenario Name	Param Name	Schema Used	Comment
SDE_RETAILINVENTO RYADJUSTMENTFACT	LOC_NUM_OF_THREAD	RDE	This is to indicate the number of threads program runs into to setup Multi threading to gain performance.

Patching Procedures

Oracle Retail Patching Process

The patching process for many Oracle Retail products has been substantially revised from prior releases. Automated tools are available to reduce the amount of manual steps when applying patches. To support and complement this automation, more information about the environment is now tracked and retained between patches. This information is used to allow subsequent patches to identify and skip changes which have already been made to the environment. For example, the patching process uses a database manifest table to skip database change scripts which have already been executed.

The enhanced product patching process incorporates the following:

- Utilities to automate the application of Oracle Retail patches to environments.
- Unified patches so that a single patch can be applied against Database, Forms, Java applications, Batch, etc. installations.
- Database and Environment manifests track versions of files at a module level.
- Centralized configuration distinguishes installation types (Database, Forms, Java, Batch, etc.).
- Patch inventory tracks the patches applied to an environment.

These enhancements make installing and updating Oracle Retail product installations easier and reduce opportunities for mistakes. Some of these changes add additional considerations to patching and maintaining Oracle Retail product environments. Additional details on these considerations are found in later sections.

Supported Products and Technologies

Several products and technologies are supported by the enhanced patching process. The utilities, processes and procedures described here are supported with the following products and listed technologies:

Product	Supported Technology
Oracle Retail Merchandising System (RMS)	- Database scripts - Batch scripts - RETL scripts - Data Conversion Scripts - BI Publisher Reports - Java Application
Oracle Retail Warehouse Management System (RWMS)	- Database scripts - Batch scripts - Forms - BI Publisher Reports

Product	Supported Technology
Oracle Retail Price Management (RPM)	▪ Database scripts (included with RMS) ▪ Java Application ▪ Batch scripts
Oracle Retail Invoice Matching (ReIM)	▪ Database scripts (included with RMS) ▪ Java Application ▪ Batch scripts
Oracle Retail Allocation	▪ Database scripts (included with RMS) ▪ Java Application ▪ Batch scripts
Oracle Retail Sales Audit (ReSA)	▪ Database scripts (included with RMS) ▪ Java Application
Oracle Retail Insights (RI) Previously called Oracle Retail Analytics (RA)	▪ Database scripts
Oracle Retail Advanced Science Engine (ORASE)	▪ Database scripts ▪ Batch scripts
Oracle Retail Data Extractor (RDE)	▪ Database scripts
Oracle Retail Application Admin Console (ORAAC). Previously called Oracle Retail Application Security Role Manager (RASRM)	▪ Java Application

Patch Concepts

During the lifecycle of an Oracle Retail environment, patches are applied to maintain your system. This maintenance may be necessary to resolve a specific issue, add new functionality, update to the latest patch level, add support for new technologies, or other reasons.

A patch refers to a collection of files to apply to an environment. Patches could be cumulative, such as the 16.0 release, or incremental, such as a hot fix for just a few modules. Patches may contain updates for some or all components of a product installation including database, application code, forms, and batch. In a distributed architecture the same patch may need to be applied to multiple systems in order to patch all of the components. For example, if a patch contains both database and application changes, the patch would need to be applied to both the database server and the application server.

The top-level directory for the installation of an Oracle Retail product is referred to as the RETAIL_HOME. Underneath RETAIL_HOME are all of the files related to that product installation, as well as configuration and metadata necessary for the Oracle Retail Patch Assistant to maintain those files. In some cases the runtime application files also exist under RETAIL_HOME. For example, compiled RMS batch files, the compiled RWMS forms, or Java Application batch scripts.

Patching Utility Overview

Patches are applied and tracked using utilities that are specifically designed for this purpose. The primary utility is described briefly below and additional information is available in later sections.

Oracle Retail Patch Assistant (ORPatch)

ORPatch is the utility used to apply patches to an Oracle Retail product installation. It is used in the background by the installer when creating a new installation or applying a cumulative patch. It is used directly to apply an incremental patch to an environment.

Oracle Retail Merge Patch (ORMerge)

ORMerge is a utility to allow multiple patches to be combined into a single patch. Applying patches individually may require some steps to be repeated. Merging multiple patches together allows these steps to be run only once. For example, applying several incremental patches to database packages will recompile invalid objects with each patch. Merging the patches into a single patch before applying them will allow invalid objects to be recompiled only once.

Oracle Retail Compile Patch (ORCompile)

ORCompile is a utility to compile components of Oracle Retail products outside of a patch. It allows RMS Batch, and RWMS Forms to be fully recompiled even if no patch has been applied. It also contains functionality to recompile invalid database objects in product schemas.

Oracle Retail Deploy Patch (ORDeploy)

ORDeploy is a utility to deploy components of Oracle Retail Java products outside of a patch. It allows RPM, ReIM, Allocation and ReSA java applications to be redeployed to WebLogic even if a patch has not been applied. It contains functionality to optionally include or not include Java customizations when redeploying.

Changes with 16.0

Some products and technologies are supported by the enhanced patching process for the first time in 16.0. In those cases all of the content in this chapter is new with 16.0.

New technologies

For the 16.0 release Oracle Retail Merchandising System (RMS) has a new ADF application component that is integrated with Orpatch.

Patching Considerations

Patch Types

Oracle Retail produces two types of patches for their products: cumulative and incremental.

Cumulative Patches

A cumulative patch includes all of the files necessary to patch an environment to a specific level or build a new environment at that level. Examples of cumulative patches would be 16.0, 15.0.2, and so on. Cumulative patches come with a standard Oracle Retail installer and so can be applied to an environment with the installer rather than with ORPatch or other utilities.

Incremental Patches

An incremental patch includes only selected files necessary to address a specific issue or add a feature. Examples of incremental patches would be a hot fix for a specific defect. Incremental patches do not include an installer and must be applied with ORPatch.

Incremental Patch Structure

An Oracle Retail incremental patch generally contains several files and one or more subdirectories. The subdirectories contain the contents of the patch, while the individual files contain information about the patch and metadata necessary for patching utilities to correctly apply the patch. The most important files in the top-level directory are the README.txt, the manifest files.

README File

The README.txt file contains information about the incremental patch and how to apply it. This may include manual steps that are necessary before, after or while applying the patch. It will also contain instructions on applying the patch with ORPatch.

Manifest Files

Each patch contains manifest files which contain metadata about the contents of a patch and are used by ORPatch to determine the actions necessary to apply a patch. Patches should generally be run against all installations a product in an environment, and ORPatch will only apply the changes from the patch that are relevant to that installation.

> **Note:** Cumulative patches use a different patch structure because they include a full installer which will run ORPatch automatically.

Version Tracking

The patching infrastructure tracks version information for all files involved with a product installation. The RETAIL_HOME contains files which track the revision of all files within the RETAIL_HOME including batch, forms, database, Java archives and other files. In addition, records of database scripts that have been applied to the product database objects are kept within each database schema.

Apply all Patches with Installer or ORPatch

In order to ensure that environment metadata is accurate all patches must be applied to the Oracle Retail product installation using patching utilities. For cumulative patches this is done automatically by the installer. For incremental patches ORPatch must be used directly. This is especially important if database changes are being applied, in order to ensure that the database-related metadata is kept up-to-date.

Environment Configuration

A configuration file in $RETAIL_HOME/orpatch/config/env_info.cfg is used to define the details of a specific Oracle Retail environment. This file defines:

- The location of critical infrastructure components such as the ORACLE_HOME on a database or middleware server.
- The location of Oracle Wallets to support connecting to the database users.
- The type of file processing which is relevant to a particular host. For example, if this is a host where database work should be done, or a host where batch compilation

should be done, a host where Java applications should be deployed, etc. This allows a single database, forms and batch patch to be run against all types of hosts, applying only the relevant pieces on each server.

- Other configuration necessary to determine proper behavior in an environment.

Retained Installation Files

The RETAIL_HOME location of an Oracle Retail product installation contains all of the files associated with that installation. This can include database scripts, Java files, Forms, Batch, RETL and Data Conversion files as with previous versions and also includes all database scripts. This allows objects to be reloaded during patching, including any necessary dependencies.

Reloading Content

In order to ensure that database contents and generated files exactly match patched versions, when applying cumulative patches some content is regenerated even if it does not appear to have changed.

On a cumulative patch this includes:

- All re-runnable database content will be reloaded
  - Packages and Procedures
  - Database Types (excluding RIB objects)
  - Control scripts
  - Triggers
  - WebService jars and packages
  - Form Elements
- All RWMS forms files will be recompiled
- All RMS batch files will be recompiled

When applying incremental patches, only changed files will be reloaded. However this does not apply to RMS batch, which is fully recompiled with any change.

Java Hotfixes and Cumulative Patches

When applying cumulative patches to Java applications components with ORPatch, all hotfixes related to base product ear files included with the patch will be rolled back. This increases the likelihood of a successful deployment because hotfixes may not be compatible with updated product ear files, or may already be included with the ear. Before applying a cumulative patch to Java applications, check the patch documentation to determine which hotfixes are not included in the ear. Then work with Oracle Support to obtain compatible versions of the fixes for the updated ear version. In some cases this may be the same hotfix, in which case it can be re-applied to the environment. In other cases a new hotfix may be required.

Backups

Before applying a patch to an environment, it is extremely important to take a full backup of both the RETAIL_HOME file system and the Oracle Retail database. Although ORPatch makes backups of files modified during patching, any database changes cannot be reversed. If a patch fails which contains database changes, and cannot be completed, the environment must be restored from backup.

Disk Space

When patches are applied to an environment, the old version of files which are updated or deleted are backed up to $RETAIL_HOME/backups/backup-<timestamp>. When applying large patches, ensure there is sufficient disk space on the system where you unzip the patch or the patching process may fail. Up to twice as much disk space as the unzipped patch may be required during patching.

In addition to backups of source files, the existing compiled RWMS Forms and RMS Batch files are saved before recompilation. These backups may be created during patches:

- Batch 'lib' directory in $RETAIL_HOME/oracle/lib/bin-<timestamp>
- Batch 'proc' directory in $RETAIL_HOME/oracle/proc/bin-<timestamp>
- Forms 'toolset' directory in $RETAIL_HOME/base/toolset/bin-<timestamp>
- Forms 'forms' directory in $RETAIL_HOME/base/forms/bin-<timestamp>

Periodically both types of backup files can be removed to preserve disk space.

Patching Operations

Running ORPatch

ORPatch is used to apply patches to an Oracle Retail product installation. When applying a patch which includes an installer, ORPatch does not need to be executed manually as the installer will run it automatically as part of the installation process. When applying a patch that does not include an installer, ORPatch is run directly.

ORPatch performs the tasks necessary to apply the patch:

- Inspects the patch metadata to determine the patch contents and patch type.
- Reads the environment configuration file to determine which product components exist in this installation.
- Assembles a list of patch actions which will be run on this host to process the patch.
- Executes pre-checks to validate that all patch actions have the necessary configuration to proceed.
- Compares version numbers of files from the patch against the files in the environment.
- Backs up files which will be updated.
- Copies updated files into the installation.
- Loads updated files into database schemas, if applicable.
- Recompiles RMS batch, if applicable.
- Recompiles RWMS forms, if applicable.
- Constructs updated Java archives and deploys them to WebLogic, if applicable
- Updates Java batch files and libraries, if applicable
- Records the patch in the patch inventory.

If a patch does not contain updated files for the database or system, no action may be taken. If a previously failed ORPatch session is discovered, it will be restarted.

Preparing for Patching

Before applying a patch to your system, it is important to properly prepare the environment.

Single Patching Session

It is extremely important that only a single ORPatch session is active against a product installation at a time. If multiple patches need to be applied, you can optionally merge them into a single patch and apply one patch to the environment. Never apply multiple patches at the same time.

Shutdown Applications

If a patch updates database objects, it is important that all applications are shutdown to ensure no database objects are locked or in use. This is especially important when applying changes to Oracle Retail Integration Bus (RIB) objects as types in use will not be correctly replaced, leading to "ORA-21700: object does not exist or marked for delete" errors when restarting the RIB.

Backup Environment

Before applying a patch to an environment, it is important to take a full backup of both the RETAIL_HOME file system and the retail database. Although ORPatch makes

backups of files modified during patching, any database changes cannot be reversed. If a patch which contains database changes fails and cannot be completed, the environment must be restored from backup.

Log Files

When applying a patch, ORPatch will create a number of log files which contain important information about the actions taken during a patch and may contain more information in the event of problems. Log files are created in the $RETAIL_HOME/orpatch/logs directory. Logs should always be reviewed after a patch is applied.

After a patch session the log directory will contain at a minimum an ORPatch log file and may also contain other logs depending on the actions taken. The following table describes logs that may exist.

Log File	Used For
orpatch-<date>-<time>.log	Primary ORPatch log file
detail_logs/dbsql_<component>/invalids/*	Details on the errors causing a database object to be invalid
detail_logs/analyze/details	Detail logs of files that will be created/updated/removed when a patch is applied
detail_logs/compare/details	Detail logs of the differences between two sets of environment metadata
orpatch_forms_<pid>_child_<num>.log	Temporary logs from a child process spawned to compile forms in parallel. After the child process completes, the contents are append to the primary orpatch log file
detail_logs/rmsbatch/lib/*	Detail logs of the compilation of RMS Batch libraries
detail_logs/rmsbatch/proc/*	Detail logs of the compilation of RMS Batch programs
detail_logs/dbsql_rms/rms_db_ws_consumer_jars/*	Detail logs of the loadjava command to install RMS WebService Consumer objects
detail_logs/dbsql_rms/rms_db_ws_consumer_libs/*	Detail logs of the loadjava command to install RMS WebService Consumer libraries
detail_logs/forms/rwms_frm_forms/*	Detail logs of the compilation of each RWMS Forms file
detail_logs/dbsql_rwms/rwms_db_sp _jars/*	Detail logs of the loadjava command to install RWMS SP jars
detail_logs/javaapp_<product>/deploy/*	Detail logs of the deploy of a Java product

Unzip Patch Files

Before executing ORPatch, the patch files must be unzipped into a directory. This directory will be passed to ORPatch as the "-s <source directory>" argument on the command-line when applying or analyzing a patch.

Location of ORPatch

The ORPatch script will be located in $RETAIL_HOME/orpatch/bin.

Command Line Arguments

ORPatch behavior is controlled by several command-line arguments. These arguments may be actions or options. Command and option names can be specified in upper or lower case, and will be converted to upper-case automatically. Arguments to options, for example the source directory patch, will not be modified.

ORPatch command-line actions:

Action	Description
apply	Tells ORPatch to apply a patch, requires the –s option Example: orpatch apply –s $RETAIL_HOME/stage/patch123456
analyze	Tells ORPatch to analyze a patch, requires the –s option Example: orpatch analyze –s $RETAIL_HOME/stage/patch123456
lsinventory	Tells ORPatch to list the inventory of patches that have been applied to this installation
exportmetadata	Tells ORPatch to extract all metadata information from the environment and create a $RETAIL_HOME/support directory to contain it. Requires the –expname option.
Diffmetadata	Tells ORPatch to compare all metadata from the current environment with metadata exported from some other environment. Requires the –expname and –srcname options.
Revert	Tells ORPatch to revert the files related to a patch, requires the –s option Example: orpatch revert –s $RETAIL_HOME/backups/backup-09302013-153010

Note: An action is required and only one action can be specified at a time.

ORPatch command-line arguments:

Argument	Valid For Actions	Description
-s <source dir>	apply analyze	Specifies where to find the top-level directory of the patch to apply or analyze. The source directory should contain the manifest.csv and patch_info.cfg files.
-new	apply	Forces ORPatch to not attempt to restart a failed ORPatch session

Argument	Valid For Actions	Description
-expname	exportmetadata diffmetadata lsinventory	Defines the top-level name to be used for the export or comparison of environment metadata. When used with lsinventory, it allows an exported inventory to be printed.
-srcname	diffmetadata	Defines the 'name' to use when referring to the current environment during metadata comparisons.
-dbmodules	diffmetadata	When comparing metadata at a module-level, compare the dbmanifest information rather than the environment manifest. This method of comparing metadata is less accurate as it does not include non-database files.
-jarmodules	analyze diffmetadata	When used with analyze, requests a full comparison of the metadata of Java archives included in the patch versus the metadata of the Java archives in the environment. This behavior is automatically enabled when Java customizations are detected in the environment. Analyzing the contents of Java archives allows for detailed investigation of the potential impacts of installing a new Java ear to an environment with customizations.

When used with diffmetadata, causes metadata to be compared using jarmanifest information rather than the environment manifest. This provides more detailed information on the exact differences of the content of Java archives, but does not include non-Java files. |
| -selfonly | apply analyze | Only apply or analyze changes in a patch that relate to orpatch itself. This is useful for applying updates to orpatch without applying the entire patch to an environment. |
| -s <backup dir> | revert | Specifies the backup from a patch that should be reverted to the environment. This restores only the files modified during the patch, the database must be restored separately or the environment will be out-of-sync and likely unusable. |

Analyzing the Impact of a Patch

In some cases, it may be desirable to see a list of the files that will be updated by a patch, particularly if files in the environment have been customized. ORPatch has an 'analyze' mode that will evaluate all files in the patch against the environment and report on the files that will be updated based on the patch.

To run ORPatch in analyze mode, include 'analyze' on the command line. It performs the following actions:

- Identifies files in the environment which the patch would remove.
- Compares version numbers of files in the patch to version numbers of files in the environment.
- Prints a summary of the number of files which would be created, updated or removed.

- Prints an additional list of any files that would be updated which are registered as being customized.
- Prints an additional list of any files which are in the environment and newer than the files included in the patch. These files are considered possible conflicts as the modules in the patch may not be compatible with the newer versions already installed. If you choose to apply the patch the newer versions of modules in the environment will NOT be overwritten.
- If a Java custom file tree is detected, prints a detailed analysis of the modules within Java ear files that differ from the current ear file on the system.
- Saves details of the files that will be impacted in $RETAIL_HOME/orpatch/logs/detail_logs/analyze/details.

This list of files can then be used to assess the impact of a patch on your environment.

To analyze a patch, perform the following steps:

1. Log in as the UNIX user that owns the product installation.
2. Set the RETAIL_HOME environment variable to the top-level directory of your product installation.

   `Export RETAIL_HOME=/u00/oretail/tst`

3. Set the PATH environment variable to include the orpatch/bin directory

   `export PATH=$RETAIL_HOME/orpatch/bin:$PATH`

4. Set the JAVA_HOME environment variable if the patch contains Java application files.

   `Export JAVA_HOME=/u00/oretail/java_jdk`

   > **Note:** If the JAVA_HOME environment variable is not specified, the value from RETAIL_HOME/orpatch/config/env_info.cfg will be used.

5. Create a staging directory to contain the patch, if it does not already exist.

   `mkdir –p $RETAIL_HOME/stage`

6. Download the patch to the staging directory and unzip it.
7. Execute orpatch to analyze the patch.

   `Orpatch analyze –s $RETAIL_HOME/stage/patch123456`

8. Repeat the patch analysis on all servers with installations for this product environment.
9. Evaluate the list(s) of impacted files.

For more information on registering and analyzing customizations, please see the Customization section later in this document.

Applying a Patch

Once the system is prepared for patching, ORPatch can be executed to apply the patch to the environment. The patch may need to be applied to multiple systems if it updates components that are installed on distributed servers.

To apply a patch, perform the following steps:

1. Log in as the UNIX user that owns the product installation.
2. Set the RETAIL_HOME environment variable to the top-level directory of your product installation.

   `Export RETAIL_HOME=/u00/oretail/tst`

3. Set the PATH environment variable to include the orpatch/bin directory

   `export PATH=$RETAIL_HOME/orpatch/bin:$PATH`

4. Set the DISPLAY environment variable if the patch contains Forms.

   `Export DISPLAY=localhost:10.0`

   > **Note:** If the DISPLAY environment variable is not specified, the value from RETAIL_HOME/orpatch/config/env_info.cfg will be used.

5. Set the JAVA_HOME environment variable if the patch contains Java application files.

   `Export JAVA_HOME=/u00/oretail/java_jdk`

   > **Note:** If the JAVA_HOME environment variable is not specified, the value from RETAIL_HOME/orpatch/config/env_info.cfg will be used.

6. Create a staging directory to contain the patch, if it does not already exist.

   `mkdir -p $RETAIL_HOME/stage`

7. Download the patch to the staging directory and unzip it.

8. Review the README.txt included with the patch. If manual steps are specified in the patch, execute those steps at the appropriate time.

9. Shutdown applications.

10. Execute ORPatch to apply the patch.

    `Orpatch apply -s $RETAIL_HOME/stage/patch123456`

11. After ORPatch completes, review the log files in $RETAIL_HOME/orpatch/logs.

12. Repeat the patch application on all servers with installations for this product environment.

13. Restart applications.

Restarting ORPatch

If ORPatch is interrupted while applying a patch, or exits with an error, it saves a record of completed work in a restart state file in $RETAIL_HOME/orpatch/logs. Investigate and resolve the problem that caused the failure, then restart ORPatch.

By default when ORPatch is started again, it will restart the patch process close to where it left off. If the patch process should **not** be restarted, add '-new' to the command-line of ORPatch.

Please note that starting a new patch session without completing the prior patch may have serious impacts that result in a patch not being applied correctly. For example, if a patch contains database updates and batch file changes and ORPatch is aborted during the load of database objects, abandoning the patch session will leave batch without the latest changes compiled in the installation.

Listing the Patch Inventory

After a patch is successfully applied by ORPatch the patch inventory in $RETAIL_HOME/orpatch/inventory is updated with a record that the patch was applied. This inventory contains a record of the patches applied, the dates they were applied, the patch type and products impacted.

To list the patch inventory, perform the following steps:

1. Log in as the UNIX user that owns the product installation.

2. Set the RETAIL_HOME environment variable to the top-level directory of your product installation.

```
export RETAIL_HOME=/u00/oretail/tst
```

3. Set the PATH environment variable to include the orpatch/bin directory

```
export PATH=$RETAIL_HOME/orpatch/bin:$PATH
```

4. Execute orpatch to list the inventory.

```
Orpatch lsinventory
```

Exporting Environment Metadata

ORPatch functionality is driven based on additional metadata that is stored in the environment to define what version of files are applied to the environment, and which database scripts have been applied to database schemas. This environment metadata is used to analyze the impact of patches to environments and controls what actions are taken during a patch. The metadata is stored in several locations depending on the type of information it tracks and in some cases it may be desirable to extract the metadata for analysis outside of ORPatch. For example, Oracle Support could ask for the metadata to be uploaded to assist them in triaging an application problem.

ORPatch provides a capability to export all of the metadata in an environment into a single directory and to automatically create a zip file of that content for upload or transfer to another system. The exact metadata collected from the environment depends on the products installed in the RETAIL_HOME.

ORPatch metadata exported:

Installed Product Component	Exported Metadata	Description
Any	orpatch/config/env_info.cfg orpatch/config/custom_hooks.cfg ORPatch inventory files	ORPatch configuration and settings
Any	All env_manifest.csv and deleted_env_manifest.csv files	Environment manifest files detailing product files installed, versions, customized flags and which patch provided the file
Database Schemas	DBMANIFEST table contents	Database manifest information detailing which database scripts were run, what version and when they were executed
Java Applications	All files from javaapp_<product>/config except jar files	Environment-specific product configuration files generated during installation
Java Applications	Combined export of all META-INF/env_manifest.csv files from all product ear files	Jar manifest information detailing files, versions, customized flags and which patch provided the file

Installed Product Component	Exported Metadata	Description
Java Applications	orpatch/config/javaapp_<product>/ant.deploy.properties	Environment properties file created during product installation and used during application deployment
Java Applications	<weblogic_home>/server/lib/weblogic.policy	WebLogic server java security manager policy file
RMS Batch	orpatch/config/rmsbatch_profile	Batch compilation shell profile
RWMS Forms	orpatch/cofngi/rwsmforms_profile	Forms compilation shell profile

Exports of environment metadata are always done to the $RETAIL_HOME/support directory. When exporting metadata, you must specify the –expname argument and define the name that should be given to the export. The name is used for the directory within $RETAIL_HOME/support and for the name of the zip file.

To extract an environment's metadata, perform the following steps:

1. Log in as the UNIX user that owns the product installation.
2. Set the RETAIL_HOME environment variable to the top-level directory of your product installation.
   ```
   export RETAIL_HOME=/u00/oretail/tst
   ```
3. Set the PATH environment variable to include the orpatch/bin directory
   ```
   export PATH=$RETAIL_HOME/orpatch/bin:$PATH
   ```
4. Execute orpatch to export the metadata.
   ```
   Orpatch exportmetadata –expname test_env
   ```

This example would export all metadata from the environment to the $RETAIL_HOME/support/test_env directory. A zip file of the metadata would be created in $RETAIL_HOME/support/test_env.zip.

> **Note:** The $RETAIL_HOME/support/<name> directory should be empty or not exist prior to running exportmetadata in order to ensure accurate results.

Comparing Environment Metadata

Once metadata has been exported from an environment, it can be used to compare the environment manifest metadata of two environments. ORPatch provides a capability to compare metadata of the current environment with the exported metadata of another environment. Note that even though there are many types of metadata exported by ORPatch, only environment manifest metadata is evaluated during comparisons. Metadata comparison happens in four phases: product comparison, patch comparison, ORPatch action comparison, and module-level comparison.

Product comparison compares the products installed in one environment with the products installed in another environment. Patch comparison compares the patches applied in one environment with the patches applied in another environment, for

common products. This provides the most summarized view of how environments differ. Patches which only apply to products on one environment are not included in the comparison.

Since each patch may impact many files, the comparison then moves on to more detailed analysis. The third phase of comparison is to compare the enabled ORPatch actions between environments. These actions roughly correspond to the installed 'components' of a product. For example, one environment may have database and forms components installed while another has only forms. Action comparison identifies components that are different between environments. The final phase of comparison is at the module level for actions that are common between environments. Modules which exist only on one environment, or exist on both environments with different revisions, or which are flagged as customized are reported during the comparison.

Differences between environment metadata are reported in a summarized fashion during the ORPatch execution. Details of the comparison results are saved in $RETAIL_HOME/orpatch/logs/detail_logs/compare/details. One CSV file is created for each phase of comparison: product_details.csv, patch_details.csv, action_details.csv and module_details.csv.

In order to be compared by ORPatch, exported metadata must be placed in the $RETAIL_HOME/support directory. The metadata should exist in the same structure that it was originally exported in. For example, if the metadata was exported to $RETAIL_HOME/support/test_env on another system, it should be placed in $RETAIL_HOME/support/test_env on this system.

When reporting differences between two environments, ORPatch uses names to refer to the environments. These names are defined as part of the diffmetadata command. The –expname parameter, which defines the directory containing the metadata, is also used as the name when referring to the exported metadata. The –srcname parameter defines the name to use when referring to the current environment. As an example, if you had exported the 'test' environment's metadata and copied it to the 'dev' environment's $RETAIL_HOME/support/test_env directory, you could run "orpatch diffmetadata –expname test_env –srcname dev_env". The detail and summary output would then refer to things that exist on dev but not test, revisions in the test environment versus revisions in the dev environment, etc.

ORPatch will automatically export the environment's current metadata to $RETAIL_HOME/support/compare prior to starting the metadata comparison.

To compare two environment's metadata, perform the following steps:

1. Export the metadata from another environment using orpatch exportmetadata.

2. Transfer the metadata zip from the other system to $RETAIL_HOME/support.

3. Log in as the UNIX user that owns the product installation.

4. Set the RETAIL_HOME environment variable to the top-level directory of your product installation.

    `export RETAIL_HOME=/u00/oretail/dev`

5. Set the PATH environment variable to include the orpatch/bin directory

    `export PATH=$RETAIL_HOME/orpatch/bin:$PATH`

6. Unzip the metadata zip file.

    `unzip test_env.zip`

7. Execute orpatch to compare the metadata

    `orpatch diffmetadata –expname test_env –srcname dev_env`

This example would compare the current environment against the metadata extracted in $RETAIL_HOME/support/test_env directory.

> **Note:** The $RETAIL_HOME/support/compare directory
> will be automatically removed before environment metadata
> is exported at the start of the comparison.

Reverting a Patch

In general it is best to either completely apply a patch, or restore the entire environment from the backup taken before starting the patch. It is important to test patches in test or staging environments before applying to production. In the event of problems, Oracle Retail recommends restoring the environment from backup if a patch is not successful.

> **Note:** Reverting patches in an integrated environment can be
> extremely complex and there is no fully automated way to
> revert all changes made by a patch. Restoring the
> environment from a backup is the recommended method to
> remove patches.

It is, however, possible to revert small patches using the backups taken by ORPatch during a patch. This will restore only the files modified, and it is still necessary to restore the database if any changes were made to it.

> **Note:** Reverting a patch reverts only the files modified by
> the patch, and does not modify the database, or recompile
> forms or batch files after the change.

When multiple patches have been applied to an environment, reverting any patches other than the most recently applied patch is strongly discouraged as this will lead to incompatible or inconsistent versions of modules applied to the environment. If multiple patches are going to be applied sequentially it is recommended to first merge the patches into a single patch that can be applied or reverted in a single operation.

To revert a patch, perform the following steps:

1. Log in as the UNIX user that owns the product installation.

2. Set the RETAIL_HOME environment variable to the top-level directory of your product installation.

   ```
   export RETAIL_HOME=/u00/oretail/tst
   ```

3. Set the PATH environment variable to include the orpatch/bin directory

   ```
   export PATH=$RETAIL_HOME/orpatch/bin:$PATH
   ```

4. Identify the backup directory in $RETAIL_HOME/backups that contains the backup from the patch you want to restore.

   - The backup directory will contain a patch_info.cfg file which contains the name of the patch the backup is from.

   - It is possible to have two directories for the same patch, if ORPatch was updated during the patch. It is not possible to revert the updates to ORPatch. Select the backup directory that does not contain orpatch files.

   - If it is not clear which backup directory to use, restore the environment from backup

5. Execute orpatch to revert the environment using the contents of the backup directory

   ```
   orpatch revert -s $RETAIL_HOME/backups/backup-11232013-152059
   ```

6. Restore the database from backup if the patch made database changes

7. Use the orcompile script to recompile forms if the patch included RWMS forms files

   ```
   orcompile -a RWMS -t FORMS
   ```

8. Use the orcompile script to recompile batch if the patch included RMS batch files

   ```
   orcompile -a RMS -t BATCH
   ```

9. Use the ordeploy script to redeploy the appropriate Java applications if the patch included Java files

   ```
   ordeploy -a RPM -t JAVA
   ordeploy -a REIM -t JAVA
   ordeploy -a ALLOC -t JAVA
   ordeploy -a RESA -t JAVA
   ordeploy -a RMS -t JAVA
   ```

Merging Patches

When patches are applied individually some ORPatch tasks such as compiling forms and batch files or deploying Java archives are performed separately for each patch. This can be time-consuming. An alternative is to use the ORMerge utility to combine several patches into a single patch, reducing application downtime by eliminating tasks that would otherwise be performed multiple times. Patches merged with ORMerge are applied with ORPatch after the merge patch is created.

Source and Destination Directories

ORMerge uses source and destination areas in order to merge patch files. The source area is a single directory that contains the extracted patches to merge. The destination area is the location where the merged patch will be created. If a file exists in one or more source patches, only the highest revision will be copied to the merged patch.

The source and destination directories should exist under the same parent directory. That is, both the source and destination directories should be subdirectories of a single top-level directory.

The source directory must have all patches to be merged as immediate child directories. For example if three patches need to be merged the directory structure would look like this:

Source and Destination Directory Example
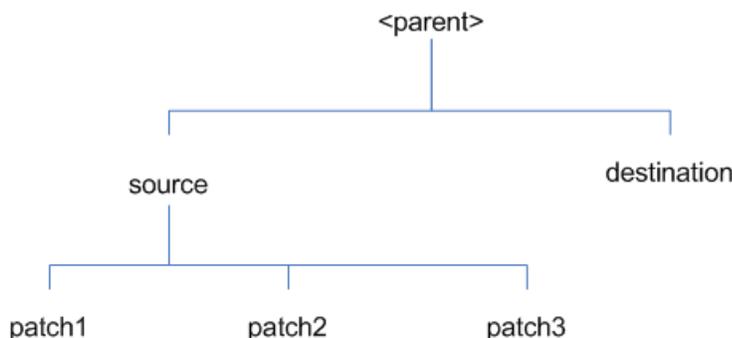
In the example above, the manifest.csv and patch_info.cfg files for each patch to be merged must exist in source/patch1, source/patch2, and source/patch3.

ORMerge Command-line Arguments

Argument	Required	Description
-s	Yes	Path to source directory containing patches to merge

Argument	Required	Description
-d	Yes	Path to destination directory that will contain merged patch
-name	No	The name to give the merged patch. If not specified, a name will be generated. When the merged patch is applied to a system, this name will appear in the Oracle Retail patch inventory.
-inplace	No	Used only when applying a patch to installation files prior to the first installation. See "Patching prior to the first install" in the Troubleshooting section later, for more information.

Running the ORMerge Utility

To merge patches, perform the following steps:

1. Log in as the UNIX user that owns the product installation.

2. Set the RETAIL_HOME environment variable to the top-level directory of your product installation.

   ```
   export RETAIL_HOME=/u00/oretail/tst
   ```

3. Set the PATH environment variable to include the orpatch/bin directory
   ```
   export PATH=$RETAIL_HOME/orpatch/bin:$PATH
   ```

4. Create a staging directory to contain the patches.

   ```
   mkdir –p $RETAIL_HOME/stage/merge/src
   ```

5. Download the patches to the staging directory and unzip them so that each patch is in a separate subdirectory.

6. Review the README.txt included with each patch to identify additional manual steps that may be required. If manual steps are specified in any patch, execute them at the appropriate time when applying the merged patch.

7. Create a destination directory to contain the merged patches.

   ```
   mkdir –p $RETAIL_HOME/stage/merge/dest
   ```

8. Execute ORMerge to merge the patches.

   ```
   Ormerge –s $RETAIL_HOME/stage/merge/src –d $RETAIL_HOME/stage/merge/dest –name
   merged_patch
   ```

The merged patch can now be applied as a single patch to the product installation using ORPatch.

Compiling Application Components

In some cases it may be desirable to recompile RWMS Forms or RMS Batch outside of a product patch. The ORCompile utility is designed to make this easy and remove the need to manually execute 'make' or 'frmcmp' commands which can be error-prone. ORCompile leverages ORPatch functions to ensure that it compiles forms and batch exactly the same way as ORPatch. In addition ORCompile offers an option to compile invalid database objects using ORPatch logic.

ORCompile takes two required command line arguments each of which take an option. Arguments and options can be specified in upper or lower case.

ORCompile Command Line Arguments

Argument	Description
-a <app>	The application to compile.
-t <type>	The type of application objects to compile

ORCompile Argument Options

Application	Type	Description
RMS	BATCH	Compile RMS Batch programs
RWMS	FORMS	Compile RWMS Forms
RMS	DB	Compile invalid database objects in the primary RMS schema
ALLOC	DB-ALC	Compile invalid database objects in the Allocations user schema
ALLOC	DB-RMS	Compile invalid database objects in the RMS schema
REIM	DB	Compile invalid database objects in the RMS schema
RME	DB	Compile invalid database objects in the RME schema
ASO	DB	Compile invalid database objects in the ASO schema
RI	DB-DM	Compile invalid database objects in the RI DM schema
RI	DB-RIBATCH	Compile invalid database objects in the RI batch schema
RI	DB-RMSBATCH	Compile invalid database objects in the RI RMS batch schema
RI	DB-FEDM	Compile invalid database objects in the RI front-end schema
RDE	DB-DM	Compile invalid database objects in the RDE DM schema
RDE	DB-RDEBATCH	Compile invalid database objects in the RDE batch schema
RDE	DB-RMSBATCH	Compile invalid database objects in the RDE RMS batch schema

> **Note:** Compiling RMS type DB, ReIM type DB, and Allocation type DB-RMS, are all identical as they attempt to compile all invalid objects residing in the RMS schema.

Running the ORCompile utility

To compile files, perform the following steps:

1. Log in as the UNIX user that owns the product installation.
2. Set the RETAIL_HOME environment variable to the top-level directory of your product installation.
   ```
   export RETAIL_HOME=/u00/oretail/tst
   ```
3. Set the PATH environment variable to include the orpatch/bin directory
   ```
   export PATH=$RETAIL_HOME/orpatch/bin:$PATH
   ```
4. Execute orcompile to compile the desired type of files.
   ```
   Orcompile –a <app> -t <type>
   ```

ORCompile Examples

Compile RMS Batch.

```
Orcompile –a RMS –t BATCH
```

Compile RWMS Forms.

```
Orcompile –a RWMS –t FORMS
```

Compile invalid objects in the RA DM schema.

```
Orcompile –a RI –t DB-DM
```

Compile invalid objects in the RMS owning schema.

```
Orcompile –a RMS –t DB
```

Deploying Application Components

In some cases it may be desirable to redeploy Java applications outside of a product patch.   For example, when troubleshooting a problem, or verifying the operation of the application with different WebLogic settings. Another situation might include wanting to deploy the application using the same settings, but without customizations to isolate behavior that could be related to customized functionality.

The ordeploy utility is designed to make this easy and remove the need to re-execute the entire product installer when no configuration needs to change. ORDeploy leverages Oracle Retail Patch Assistant functions to ensure that it deploys applications exactly the same way as ORPatch. In addition ORDeploy offers an option to include or not include custom Java files, to ease troubleshooting.

ORDeploy takes two required command line arguments each of which take an option. Arguments and options can be specified in upper or lower case.

ORDeploy Command Line Arguments

Argument	Description
-a <app>	The application to deploy.
-t <type>	The type of application objects to deploy

ORDeploy Argument Options

Application	Type	Description
ALLOC	JAVA	Deploy the Allocations Java application and Java batch files, including any custom Java files.
ALLOC	JAVANOCUSTOM	Deploy the Allocations Java application and Java batch files, **NOT** including any custom Java files.
REIM	JAVA	Deploy the REIM Java application and Java batch files, including any custom Java files.
REIM	JAVANOCUSTOM	Deploy the REIM Java application and Java batch files, **NOT** including any custom Java files.
RESA	JAVA	Deploy the RESA Java application, including any custom Java files.

Application	Type	Description
RESA	JAVANOCUSTOM	Deploy the RESA Java application, **NOT** including any custom Java files.
RPM	JAVA	Deploy the RPM Java application and Java batch files, including any custom Java files.
RPM	JAVANOCUSTOM	Deploy the RPM Java application and Java batch files, **NOT** including any custom Java files.

Running the ORDeploy utility

To deploy Java applications, perform the following steps:

1. Log in as the UNIX user that owns the product installation.

2. Set the RETAIL_HOME environment variable to the top-level directory of your product installation.

   ```
   export RETAIL_HOME=/u00/oretail/tst
   ```

3. Set the PATH environment variable to include the orpatch/bin directory

   ```
   export PATH=$RETAIL_HOME/orpatch/bin:$PATH
   ```

4. Execute ORDeploy to deploy the desired Java application.

   ```
   ordeploy –a <app> -t <type>
   ```

ORDeploy Examples

Deploy RPM.

```
ordeploy -a RPM -t JAVA
```

Deploy ReIM without including Java customizations.

```
ordeploy -a REIM -t JAVANOCUSTOM
```

Maintenance Considerations

The additional information stored within the RETAIL_HOME and within database schemas adds some considerations when performing maintenance on your environment.

Database Password Changes

Oracle wallets are used to protect the password credentials for connecting to database schemas. This includes all database schemas used during an install. If the password for any of these users is changed the wallet's entry must be updated.

The wallet location is configurable but by default is in the following locations:

Location	Installation Type
$RETAIL_HOME/orpatch/rms_wallet	RMS Database RMS Batch
$RETAIL_HOME/orpatch/rwms_wallet	RWMS Database
$RETAIL_HOME/orpatch/rwms_wallet_app	RWMS Forms
$RETAIL_HOME/orpatch/oraso_wallet	ASO Database
$RETAIL_HOME/orpatch/orme_wallet	RME Database
$RETAIL_HOME/orpatch/ra_wallet	RI (Previously RA) Database

Location	Installation Type
$RETAIL_HOME/orpatch/rde_wallet	RDE Database

The wallet alias for each schema will be <username>_<dbname>. Standard mkstore commands can be used to update the password.

For example:

```
mkstore -wrl $RETAIL_HOME/orpatch/rms_wallet –modifyCredential rms_rmsdb rms01
rmspassword
```

This command will update the password for the RMS01 user to 'rmspassword' in the alias 'rms_rmsdb'.

The Oracle wallets are required to be present when executing ORPatch. Removing them will prevent you from being able to run ORPatch successfully. In addition the Oracle wallet location is referenced in the RMS batch.profile, and in the default RWMS Forms URL configuration, so removing them will require reconfiguration of batch and forms. If batch and forms were reconfigured after installation to use other wallet files, it is possible to backup and remove the wallets, then restore them when running ORPatch.

WebLogic Password Changes

Java wallets are used to protect the password credentials used when deploying Java products. This includes the WebLogic administrator credentials, LDAP connection credentials, batch user credentials and any other credentials used during an install. If the password for any of these users is changed the wallet's entry must be updated, or the Java product installation can be run again.

The wallet location is in the following locations:

Location	Installation Type
$RETAIL_HOME/orpatch/config/javapp_rpm	RPM Java
$RETAIL_HOME/orpatch/config/javapp_reim	ReIM Java
$RETAIL_HOME/orpatch/config/javapp_alloc	Allocation Java
$RETAIL_HOME/orpatch/config/javapp_resa	RESA Java
$RETAIL_HOME/orpatch/config/javaapp_rasrm	ORAAC (Previously RASRM) Java
$RETAIL_HOME/orpatch/config/javaapp_rms	RMS Java

The wallet aliases will be stored in the retail_installer partition. The names of the aliases will vary depending on what was entered during initial product installation.

The dump_credentials.sh script can be used to list the aliases in the wallet.

For example:

```
cd $RETAIL_HOME/orpatch/deploy/retail-public-security-api/bin
./dump_credentials.sh $RETAIL_HOME/orpatch/config/javapp_alloc

Apapplication level key partition name:retail_installer
User Name Alias:dsallocAlias User Name:rms01app
User Name Alias:BATCH-ALIAS User Name:SYSTEM_ADMINISTRATOR
User Name Alias:wlsAlias User Name:weblogic
```

The easiest way to update the credential information is to re-run the Java product installer. If you need to manually update the password for a credential, the save_credential.sh script can be used.

For example:

```
cd $RETAIL_HOME/orpatch/deploy/retail-public-security-api/bin
./save_credential.sh —l $RETAIL_HOME/orpatch/config/javapp_alloc —p
retail_installer —a wlsAlias —u weblogic
```

This command will prompt for the new password twice and update the aslias wlsAlias, username weblogic with the new password.

Infrastructure Directory Changes

The RETAIL_HOME/orpatch/config/env_info.cfg file contains the path to the database ORACLE_HOME on database or RMS Batch installations, to the WebLogic Forms and Reports ORACLE_HOME and ORACLE_INSTANCE on RWMS Forms installations, and to the WEBLOGIC_DOMAIN_HOME, WL_HOME and MW_HOME on Java product installations. If these paths change, the related configuration variables in the env_info.cfg file must be updated.

DBManifest Table

The table dbmanifest within Oracle Retail database schemas is used to track the database scripts which have been applied to the schema. It is critical not to drop or truncate this table. Without it, ORPatch will attempt to re-run scripts against the database which have already been applied which can destroy a working environment. Similarly, if copying a schema from one database to another database, ensure that the dbmanifest table is preserved during the copy.

RETAIL_HOME relationship to Database and Application Server

The RETAIL_HOME associated with an Oracle Retail product installation is critical due to the additional metadata and historical information contained within it. If a database or application installation is moved or copied, the RETAIL_HOME related to it should be copied or moved at the same time.

Jar Signing Configuration Maintenance

The RPM product installation includes an option to configure a code signing certificate so that jar files modified during installation or patching are automatically re-signed. This configuration is optional, but recommended. If it is configured, the code signing keystore is copied during installation to $RETAIL_HOME/orpatch/config/jarsign/orpkeystore.jks. The keystore password and private key password are stored in a Java wallet in the $RETAIL_HOME/orpatch/config/jarsign directory. The credentials are stored in a wallet partition called orpatch:

Alias	Username	Description
storepass	discard	Password for the keystore
keypass	discard	Password for the private key

The keystore file and passwords can be updated using the product installer. This is the recommended way to update the signing configuration.

If only the credentials need to be updated, the sign_jar.sh script can be used.

1. Log in as the UNIX user that owns the product installation.

2. Set the RETAIL_HOME environment variable to the top-level directory of your installation.

    ```
    export RETAIL_HOME=/u00/oretail/tst
    ```

3. Change directories to the location of sign_jar.sh

    ```
    cd $RETAIL_HOME/orpatch/deploy/bin
    ```

4. Execute sign_jar.sh

    ```
    sign_jar.sh changepwd
    ```

5. When prompted, enter the new keystore password

6. When prompted, enter the new private key password

Customization

Patching Considerations with Customized Files and Objects

In general, the additional capabilities provided by the ORPatch should make it easier to evaluate the potential impacts of patches to your customizations of Oracle Retail products. However, the additional metadata maintained by the Oracle Retail patching utilities does add some considerations when making customizations.

General Guidelines

It is always preferred to customize applications by extension rather than by direct modification. For example, adding new database objects and forms rather than modifying existing Oracle Retail objects and forms. You can also leverage built-in extension points such as User Defined Attributes, the Custom Flexible Attribute Solution, or seeded customization points in ADF Applications.

It is strongly discouraged to directly modify Oracle Retail database objects, especially tables, as your changes may be lost during patching or may conflict with future updates. When adding or modifying database objects, Oracle Retail recommends that all objects be added with scripts to ensure that they can be rebuilt if necessary after a patch.

Custom Database Objects

When you create new database objects, Oracle Retail recommends placing them in an Oracle database schema specifically for your customizations. You must use synonyms and grants to allow the Oracle Retail product schema owner and other users to access your objects, and use synonyms and grants to allow your customizations to access Oracle Retail objects. A separate schema will ensure that your customizations are segregated from base Oracle Retail code.

ORPatch expects that there will be no invalid objects in the database schemas it manages after a patch is applied. For this reason adding extra objects to the product schema could result in failures to apply patches as changes to base objects may cause custom objects to go invalid until they are updated. In this situation, manually update the custom objects so that they compile, and restart the patch.

Custom Forms

When creating new custom forms, Oracle Retail recommends placing them in a separate directory specifically for your customizations. This directory should be added to the FORMS_PATH of your RWMS Forms URL configuration to allow the forms to be found by the Forms Server. This will ensure that your customizations are segregated from base Oracle Retail code. If you choose to place customizations in the Forms bin directory, then your custom forms will need to be recopied each time Forms are fully recompiled.

ADF Application Customization

Oracle Retail ADF-based applications such as Allocation and ReSA can be customized using a process called 'seeded customization'. The customization process involves using JDeveloper in Customizer mode to create changes to product configurations, and then building a MAR archive containing the changes. The generated MAR is deployed to the MDS repository used by the application and applied to the application at runtime. These types of customizations are handled outside of ORPatch and are not reported during patch analysis or tracked by the custom file registration utility. More information can be found in the respective product customization guides.

Custom Compiled Java Code

When customizing Oracle Retail Java-based products such as RPM and ReIM via product source code, ORPatch supports automatically adding compiled customizations into the application ear file prior to deployment. This allows customizations to be applied to the application without directly modifying the base product ear, enabling customizations and defect hotfixes to co-exist when they do not change the same file or a dependent file. See the later "Custom Compiled Java Code" section for additional information and considerations.

Analyze Patches when Customizations are Present

Whenever you have customized a product by directly modifying Oracle Retail files or database objects, it is important to ensure you analyze each the files that will be updated by a patch before applying the patch. This will allow you to identify any customized files which may be overwritten by the patch and either merge your customization with the new version of the file, or re-apply the customization after applying the patch.

Manifest Updates

If you choose to customize Oracle Retail files directly, it is extremely important **not** to update the revision number contained in the env_manifest.csv. This could cause future updates to the file to be skipped, invalidating later patch applications as only a partial patch would be applied. The customized revision number for modified files will need to be tracked separately.

Registering Customized Files

The ORPatch contains utilities and functionality to allow tracking of files that have been customized through direct modification. This process is referred to as 'registering' a customized file. Registration only works for files which are shipped by Oracle Retail. It is not possible to register new files created in the environment as part of extensions or customizations.

When patches are analyzed with ORPatch, special reporting is provided if any registered files would be updated or deleted by the patch. Customized files impacted by the patch are listed at the end of the analysis report from ORPatch. The detail files generated during the analyze will contain a column called 'customized' which will have a Y for any files which were registered as customized. This allows easier identification of customizations which will be overwritten by a patch.

All files delivered by Oracle Retail are considered 'base' and so when they are applied to an environment any registrations of those files as customized will revert back to un-customized. **Each time a patch overwrites customized files, you must re-register the files as customized once you have applied customizations.**

To register customized files, use the $RETAIL_HOME/orpatch/bin/orcustomreg script.

The orcustomerg script operates in one of two modes: registration and list.

- Registration mode registers or unregisters one or more files as customized.
- List mode lists all files in the environment that are registered as customized.

Command Line Arguments for Registration Mode

Argument	Description
-f <file>	Adds <file> to the list of files that will be registered. Can be specified more than once.

Argument	Description
-bulk <file>	Specifies a file to read, containing one filename per line. All filenames listed inside <file> will be registered.
-register	Files specified with -f or -bulk will be registered as 'customized'
-unregister	Files specified with -f or -bulk will be registered as 'base'

Notes:

- At least one of -f or -bulk is required.

- If neither -register nor -unregister is specified, the default is '-register'.

- File names specified with -f must either be fully-qualified or be relative to RETAIL_HOME. The same is true for filenames specified within a -bulk file.

Command Line arguments for list mode

Argument	Description
-list	List all files in the environment registered as customized

Running the orcustomreg Script

Perform the following procedure to run the orcustomreg script:

1. Log in as the UNIX user that owns the product installation.

2. Set the RETAIL_HOME environment variable to the top-level directory of your product installation.

   ```
   export RETAIL_HOME=/u00/oretail/tst
   ```

3. Set the PATH environment variable to include the orpatch/bin directory

   ```
   export PATH=$RETAIL_HOME/orpatch/bin:$PATH
   ```

4. Execute orcustomreg script to register the desired file(s).

   ```
   orcustomreg –register –f <file>
   ```

Examples of using the orcustomreg Script

Register $RETAIL_HOME/dbsql_rms/Cross_Pillar/control_scripts/source/oga.sql as customized.

```
orcustomreg –f dbsql_rms/Cross_Pillar/control_scripts/source/oga.sql
```

Unregister customizations for
$RETAIL_HOME/dbsql_rwms/Triggers/Source/TR_WAVE.trg

```
orcustomreg –unregister –f $RETAIL_HOME/dbsql_rwms/Triggers/Source/TR_WAVE.trg
```

Bulk register several files as customized.

```
echo "$RETAIL_HOME/oracle/proc/src/mrt.pc" > custom.txt
echo "$RETAIL_HOME/oracle/proc/src/saldly.pc" >> custom.txt
echo "$RETAIL_HOME/oracle/proc/src/ccprg.pc" >> custom.txt
orcustomreg –bulk custom.txt
```

List all files registered as customized.

```
orcustomreg –list
```

Custom Compiled Java Code

When customizing Oracle Retail Java-based products such as RPM and ReIM via product source code, ORPatch supports automatically adding compiled customizations into the application ear file prior to deployment. This allows customizations to be applied to the application without directly modifying the base product ear, enabling customizations and defect hotfixes to co-exist when they do not change the same file or a dependent file

This functionality is enabled by creating a directory called $RETAIL_HOME/javaapp_<app>/custom, where <app> is the application the customizations apply to. Files stored within this directory will be combined with the base product ear files before the application is deployed to WebLogic. ORPatch will attempt to consider customizations stored within the 'custom' directory during patch analysis by triggering more detailed ear file change analysis to assist with identifying which customizations might be impacted by changes in the patches.

> **Note:** It is not possible, nor necessary, to register compiled Java customizations with the orcustomreg tool.

As with other customization techniques for other technologies, Oracle Retail recommends making Java customizations in new files as much as possible, versus overwriting base product or configuration files. In the past it was necessary to build complete replacement product ear files, but this method of customization is no longer required nor recommended. Replacement ear and jar files will not contain the META-INF/env_manifest.csv files which are required in order to be able to apply incremental patches. Instead, compile the specific Java classes being customized and place them along with any custom configuration files in $RETAIL_HOME/javaapp_<app>/custom.

Building Deployable ear files

When constructing the product ear file to deploy to WebLogic, ORPatch applies changes to the ear file in a specific order, with files from later steps overwriting files in earlier steps. The resulting ear is stored in $RETAIL_HOME/javaapp_<app>/deploy, and then deployed to WebLogic.

Sequence for ORPatch Java Product ear file updates

Order	File Type	Location
1	Base product ear	$RETAIL_HOME/javaapp_<app>/base
2	Updated configuration files	$RETAIL_HOME/javaapp_<app>/config
3	Oracle Retail-supplied hotfixes	$RETAIL_HOME/javaapp_<app>/internal
4	Compiled customizations	$RETAIL_HOME/javaapp_<app>/custom

Merging Custom Files

When merging files from the custom directory with the product ear, ORPatch uses the directory path of the files within custom to calculate where the file should be stored within the ear. This allows arbitrary nesting of files, even when placing files within jars stored in jars, stored within the ear. The following examples below use RPM, but apply to adding compiled customizations to any Java-based product.

Custom directory location and product ear location Examples

File path within javaapp_<app>/custom/	Final Ear File Location
rpm.ear/company/ui/MyCustom.class	In rpm.ear: /company/ui/MyCustom.class
rpm.ear/rpm.jar/company/bc/MyCustom2.class	In rpm.ear: In rpm.jar: /company/bc/MyCustom2.class
rpm.ear/lib/ourcustomlibs.jar	In rpm.ear /lib/ourcustomlibs.jar
rpm.ear/WebLaunchServlet.war/lib/ rpm.jar/company/bc/MyCustom2.class	In rpm.ear: In WebLaunchServlet.war: In lib/rpm.jar: /company/bc/MyCustom2.class

Analyzing patches when customizations are present

When analyzing a patch which contains a base product ear and the custom directory contains files, ORPatch will automatically trigger a more detailed analysis of the changes coming in a patch. This includes calculating what files inside the product ear have been added, removed or updated and which files appear to be customized based on the contents of the 'custom' directory. The detailed results of the ear file comparison during patch analysis will be saved in javaapp_<app>_archive_compare_details.csv. Any custom files which appeared to be impacted by the patch are saved in javapp_<app>_archive_custom_impacts.csv. Both files will be in the $RETAIL_HOME/orpatch/logs/detail_logs/analyze/details directory.

> **Note:** This detailed analysis is not available when analyzing individual hotfixes, so special care must be taken when applying hotfixes to a customized product installation, to ensure there are no conflicts between customizations and hotfix changes.

Customizations and cumulative patches

By default, when applying a cumulative patch, ORPatch will not include customizations in the deployed product ear, even if they are present in the appropriate directory. This allows verification that the application is functioning properly using base code, before applying customizations. After verifying the initial deployment, use ORDeploy with the "-t JAVA" option to construct and deploy the product ear including customizations.

If customizations need to be removed outside of a patch, use ORDeploy with the "-t JAVANOCUSTOM" option to create and deploy an ear containing only Oracle Retail code. To force ORPatch to include customizations in the deployed ear even when applying a cumulative patch, set JAVAAPP_<app>_INCLUDE_CUSTOM=Y in the $RETAIL_HOME/orpatch/config/env_info.cfg file.

Changing configuration files

It is possible to directly change product configuration files in $RETAIL_HOME/javaapp_<app>/config. These updates can be deployed to the environment using the ORDeploy utility. However, the 'config' directory is completely recreated each time the product installer is used. This means that modifications will be

lost and must be manually reapplied after each installer run. It is recommended to make configuration changes via the installer where possible, and retain the ant.install.properties file for use in later installer sessions.

Extending Oracle Retail Patch Assistant with Custom Hooks

The default ORPatch actions and processing logic is sufficient to install and patch the base Oracle Retail product code. However there may be situations where custom processing is desired during patching activities such as executing a shell script prior to the start of patching, or running a SQL script at the end of the patch.

ORPatch supports extensions in the form of custom hooks. These hooks allow external scripts to be run at specific points during ORPatch processing.

ORPatch Processing

Action

ORPatch supports a variety of 'actions' which define the steps necessary to apply updates to a particular area of the Oracle Retail application. Each action is generally specific to updates to a single technology or logical component of the environment. For example, one action might handle making updates to the RMS database schema, while a separate action is responsible for compiling RWMS forms, and a different action deploys the RPM Java application. These actions are enabled and disabled within the environment configuration file, allowing ORPatch to determine what types of changes to apply to each product installation.

ORPatch Actions

Order	Action Name	Description
1	DBSQL_RMSBDIINT	Loads database objects into the RMS BDI Integration schema
2	DBSQL_RMSBDIINFR	Loads database objects into the RMS BDI Infrastructure schema
3	DBSQL_RAF	Loads Retail Application Framework database objects into the RMS schema
44	DBSQL_RMS	Loads RMS and RPM database objects into the primary RMS schema
5	DBSQL_REIM	Loads ReIM database objects into the RMS schema
6	DBSQL_ALCRMS	Loads Allocation database objects into the RMS schema
7	DBSQL_ALLOC	Loads Allocation database objects into the Allocation user schema
8	DBSQL_RMSDEMO	Used to create demo data in the RMS schema if demo data was selected during initial installation
9	DBSQL_RMSDAS	Loads database objects into the RMS Data Access Schema
10	RMSBATCH	Compiles RMS Batch
11	RMSRETLSCRIPTS	Copies Oracle Retail Extract and Load scripts for RMS
12	RMSDCSCRIPTS	Copies Oracle Retail Merchandising System data conversion scripts

Order	Action Name	Description
13	JAVAAPP_RMS	Deploys the RMS Java application
14	DBSQL_RWMS	Loads database objects into the primary RWMS schema
15	DBSQL_RWMSADF	Loads database objects into the RWMS ADF user schema
16	DBSQL_RWMSUSER	Loads database objects into the RWMS user schema
17	ORAFORMS_RWMS	Compiles RWMS Forms, copies RWMS batch scripts and reports to $RETAIL_HOME
18	JAVAAPP_RPM	Deploys the RPM Java application and batch scripts
19	JAVAAPP_REIM	Deploys the REIM Java application and batch scripts
20	JAVAAPP_ALLOC	Deploys the Allocation Java application and batch scripts
21	JAVAAPP_RESA	Deploys the ReSA Java application
22	JAVAAPP_RASRM	Deploys the ORAAC (previously called RASRM) Java application
23	DBSQL_RARMSBATCH	Loads database objects into the RMS Batch schema for RI (previously called RA)
24	DBSQL_RADM	Loads database objects into the RI (previously called RA) Data Mart schema
25	DBSQL_RAFEDM	Loads database objects into the RI (previously called RA) Front-end schema
26	DBSQL_RABATCH	Loads database objects into the RI (previously called RA) Batch schema
27	RACOREBATCH	Copies RA Core batch scripts and libraries
28	DBSQL_RDERMSBATCH	Loads database objects into the RMS Batch schema for RDE
29	DBSQL_RDEDM	Loads database objects into the RDE Data Mart schema
30	DBSQL_RDEBATCH	Loads database objects into the RDE Batch schema
31	RDECOREBATCH	Copies RDE Core batch scripts and libraries
32	DBSQL_RASECORE	Loads core database objects into the ORASE schema
33	DBSQL_RASEASO	Loads ASO database objects into the ORASE schema
34	DBSQL_RASERL	Loads RL database objects into the ORASE schema
35	DBSQL_RASECDT	Loads CDT database objects into the ORASE schema
36	DBSQL_RASECIS	Loads CIS database objects into the ORASE schema
37	DBSQL_RASEDT	Loads DT database objects into the ORASE schema
38	DBSQL_RASEAE	Loads AE database objects into the ORASE schema
39	DBSQL_RASEMBA	Loads MBA database objects into the ORASE schema
40	RASECOREBATCH	Copies ORASE core batch scripts and libraries
41	RASEASOBATCH	Copies ORASE ASO batch scripts and libraries
42	RASERLBATCH	Copies ORASE RL batch scripts and libraries
43	RASECDTBATCH	Copies ORASE CDT batch scripts and libraries

Order	Action Name	Description
44	RASECISBATCH	Copies ORASE CIS batch scripts and libraries
45	RASEDTBATCH	Copies ORASE DT batch scripts and libraries
46	RASEAEBATCH	Copies ORASE AE batch scripts and libraries
47	RASEMBABATCH	Copies ORASE MBA batch scripts and libraries
48	DBSQL_RFM	Loads RFM database objects into the RMS schema

Phase

ORPatch processes patches in phases. Each action relevant to a patch and host is provided an opportunity to process the patch for each phase. The standard phases which allow hooks are:

Restart Phase Number	Phase Name	Description
N/A	PRECHECK	Actions verify that their configuration appears complete and correct. This phase and the associated hooks will be run every time orpatch is executed, even if processing will be restarted in a later phase.
10	PREACTION	Actions do processing prior to when files are copied to the environment. Files are deleted during this phase.
20	COPYPATCH	Actions copy files included in a patch into the destination environment and the environment manifest is updated.
30	PATCHACTION	Actions take the more detailed steps necessary to apply the new files to the environment. For database actions in particular, this is the phase when new and updated sql files are loaded into the database.
40	POSTACTION	Actions do processing after files have been copied and PatchActions are completed. The Forms actions, for example, use this phase to compile the forms files as this must happen after database packages are loaded.
50	CLEANUP	Actions do any additional processing. Currently no actions implement activities in this phase.

Configuring Custom Hooks

Custom hooks are configured in a configuration file RETAIL_HOME/orpatch/config/custom_hooks.cfg. The configuration file is a simple text file where blank lines and lines starting with # are ignored and all other lines should define a custom hook.

To define a custom hook, a line is added to the file in the form:

```
<hook name>=<fully qualified script>
```

The hook name must be in upper case and is in the form:

```
<action name>_<phase name>_<sequence>
```

The action name is any action name understood by ORPatch. The phase name is one of the five phase names from the table above. The sequence is either 'START' or 'END'. Hooks defined with a sequence of 'START' are run before the action's phase is invoked. Hooks defined with a sequence of 'END' are run after the action's phase is invoked.

Multiple scripts can be associated with a single hook by separating the script names with a comma. If a hook name appears in the configuration file multiple times only the last entry will be used.

The script defined as a custom hook must be an executable shell script that does not take any arguments or inputs. The only environment variable that is guaranteed to be passed to the custom hook is RETAIL_HOME. The script must return 0 on success and non-zero on failure.

If an action is a DBSQL action (i.e. has a name like DBSQL_), the custom hook can optionally be a .sql file. In this case the SQL script will be run against the database schema that the DBSQL action normally executes against. The SQL script must not generate any ORA- or SP2- errors on success. In order to be treated as a database script, the extension of the file defined as the custom hook must be .sql in lower-case. Any other extension will be treated as if it is a shell script. If you have database scripts with different extensions, they must be renamed or wrapped in a .sql script.

When using the PRECHECK phase and START sequence, please note that the custom hook will be executed prior to any verification of the configuration. Invalid configuration, such as invalid database username/password or a non-existent ORACLE_HOME, may cause the custom hook to fail depending on the actions it tries to take. However in these cases, the normal orpatch PRECHECK activities would likely have failed as well. All that is lost is the additional context that orpatch would have provided about what was incorrect about the configuration.

Restarting with Custom Hooks

If a custom hook fails, for example a shell script hook returns non-zero or a sql script generates an ORA- error in its output, the custom hook will be treated as failing. A failing custom hook causes ORPatch to immediately stop the patching session.

When ORPatch is restarted it always restarts with the same phase and action, including any START sequence custom hooks. If the START sequence custom hook fails, the action's phase is never executed. With an END sequence custom hook, the action's phase is re-executed when ORPatch is restarted and then the custom hook is re-executed. When an action's phase is costly, for example the DBSQL_RMS action which does a lot of work, this can mean a lot of duplicate processing.

For this reason it is preferred to use START sequence custom hooks whenever possible. If necessary, use a START sequence hook on a later phase or a later action, rather than an END sequence custom hook.

Patch-level Custom Hooks

In addition to action-specific hooks, there are two patch-level hook points available. These hooks allow scripts to be run before any patching activities start and after all patching activities are completed. The hooks are defined in the same configuration file, with a special hook name.

To run a script before patching, define:

```
ORPATCH_PATCH_START=<fully qualified script>
```

To run a script after patching, define:

```
ORPATCH_PATCH_END=<fully qualified script>
```

These hooks only support executing shell scripts, database scripts must be wrapped in a shell script. It is also important to note that these hooks are run on every execution of ORPatch to apply a patch, even when restarting a patch application. If the START sequence patch-level hook returns a failure, patching is aborted. If the END sequence

patch-level hook returns a failure, it is logged but ignored as all patching activities have already completed.

Please note that the ORPATCH_PATCH_START hook is executed prior to any verification of the configuration. Invalid configuration may cause the custom hook to fail depending on the actions it tries to take. However in these cases, the normal ORPatchactivities would likely fail as well.

Example Custom Hook Definitions

A shell script that is executed prior to the Pre-Action phase of RMS Batch:

`RMSBATCH_PREACTION_START=/u00/oretail/prepare_custom_header.sh`

A shell script that is executed after RETL script files are copied into the RETAIL_HOME:

`RETLSCRIPTS_COPYPATCH_END=/u00/oretail/copy_custom_files.sh`

A SQL script that is executed against the RWMS owning schema at the start of the Clean-up Phase:

`DBSQL_RWMS_CLEANUP_START=/dba/sql/recompile_synonyms.sql`

Troubleshooting Patching

There is not a general method for determining the cause of a patching failure. It is important to ensure that patches are thoroughly tested in a test or staging system several times prior to attempting to apply the patch to a production system, particularly if the patch is a large cumulative patch. After the test application is successful, apply the patch to the production system.

ORPatch Log Files

ORPatch records extensive information about the activities during a patch to the log files in RETAIL_HOME/orpatch/logs. This includes a summary of the actions that are planned for a patch, information about all files that were updated by the patch, and detailed information about subsequent processing of those files. The ORPatch log files also contain timestamps to assist in correlating log entries with other logs.

Even more detailed logs are available in RETAIL_HOME/orpatch/logs/detail_logs for some activities such as forms compilation, invalid database object errors, and output from custom hooks. If the standard ORPatch log information is not sufficient, it might be helpful to check the detailed log if it exists.

Restarting ORPatch

The restart mechanism in ORPatch is designed to be safe in nearly any situation. In some cases to ensure this, a portion of work may be redone. If the failure was caused by an intermittent issue that has been resolved, restarting ORPatch may be sufficient to allow the patch to proceed.

Manual DBManifest Updates

A possible cause for database change script failures is that a database change was already made manually to the database. In this event, you may need to update the dbmanifest table to record that a specific script does not need to be run. Before doing this, it is extremely important to ensure that all statements contained in the script have been completed.

Use the $RETAIL_HOME/orpatch/bin/ordbmreg script to register database scripts in the dbmanifest table.

Command Line Arguments for ordbmreg

Argument	Description
-f <file>	Adds <file> to the list of files that will be registered. Can be specified more than once.
-bulk <file>	Specifies a file to read, containing one filename per line. All filenames listed inside <file> will be registered.
-register	Files specified with -f or -bulk will be registered in the dbmanifest table
-unregister	Files specified with -f or -bulk will be removed from the dbmanifest table

> **Notes:**
>
> - At least one of -f or -bulk is required.
>
> - If neither -register nor -unregister is specified, the default is '-register'.
>
> - File names specified with -f must either be fully-qualified or be relative to RETAIL_HOME. The same is true for filenames specified within a -bulk file.
>
> - Registering a file in the dbmanifest table will cause it to be completely skipped.   Before doing so, ensure that all commands contained in it have been completed.
>
> - Removing a file from the dbmanifest table will cause it to be run again. This will fail if the commands in the script cannot be re-run. For example if they create a table that already exists.

Running the ordbmreg Script

Perform the following procedure to run the ordbmreg script:

1. Log in as the UNIX user that owns the product installation.
2. Set the RETAIL_HOME environment variable to the top-level directory of your product installation.
   ```
   export RETAIL_HOME=/u00/oretail/tst
   ```
3. Set the PATH environment variable to include the orpatch/bin directory
   ```
   export PATH=$RETAIL_HOME/orpatch/bin:$PATH
   ```
4. Execute ordbmreg script to register the desired file(s).
   ```
   ordbmreg –register –f <file>
   ```

Examples of using the ordbmreg Script

Register $RETAIL_HOME/dbsql_rms/Cross_Pillar/db_change_scripts/source/000593_system_options.sql with the dbmanifest table.
```
ordbmreg –f
dbsql_rms/Cross_Pillar/db_change_scripts/source/000593_system_options.sql
```

Remove the dbmanifest row for $RETAIL_HOME/dbsql_radm/rde_db/radm/database_change_scripts/000035_s12733240_w_party_per_d.sql.
```
ordbmreg –unregister –f
$RETAIL_HOME/dbsql_radm/rde_db/radm/database_change_scripts/000035_s12733240_w_party_per_d.sql
```

Bulk register several files in the dbmanifest table.
```
echo "$RETAIL_HOME/dbsql_rwms/DBCs/Source/000294_container.sql" > dbcs.txt
echo "$RETAIL_HOME/dbsql_rwms/DBCs/Source/000457_drop_object.sql" >> dbcs.txt
ordbmreg –bulk dbcs.txt
```

Restarting after registration

Once the row has been added to the dbmanifest table, restart ORPatch and the script will be skipped. If the file is not skipped there are several possibilities:

- The script registered is not the failing script.
- The file type is not a type that is filtered by the dbmanifest. The only file types that skip files listed in the dbmanifest are:
  - Initial install DDL Files
  - Installation scripts that cannot be rerun
  - Database Change Scripts

Manual Restart State File Updates

Oracle Retail strongly discourages manually updating the ORPatch restart state files. Updating the file improperly could cause necessary steps in the patching process to be skipped or patches to be incorrectly recorded as applied.

DISPLAY Settings When Compiling Forms

When compiling RWMS forms, it is necessary to have a valid X-Windows Display. ORPatch allows this setting to come from one of two places:

- DISPLAY environment variable set before executing ORPatch

  or

- DISPLAY setting in RETAIL_HOME/orpatch/config/env_info.cfg

The DISPLAY variable in the environment overrides the env_info.cfg, if both are set. The destination X-Windows display must be accessible to the user running ORPatch, and for best compilation performance it should be on the network 'close' to the server where Forms are installed and compiled. Using a local display or VNC display is preferred. Compiling forms across a Wide-Area Network will greatly increase the time required to apply patches to environments.

JAVA_HOME Setting

When working with Java application jar, ear or war files, it is necessary to have a valid JAVA_HOME setting. ORPatch allows this setting to come from one of two places:

- JAVA_HOME environment variable set before executing ORPatch

  or

- JAVA_HOME setting in RETAIL_HOME/orpatch/config/env_info.cfg

The JAVA_HOME variable in the environment overrides the env_info.cfg, if both are set. The specified Java home location must be accessible to the user running ORPatch and be a full Java Development Kit (JDK) installation. The JAVA_HOME must contain the jar utility and if automatic Jar file signing is configured, must also contain the keytool and jarsigner utilities.

Patching Prior to First Install

In some situations, it may be necessary to apply a patch to product installation files before the initial install. For example, if there is a defect with a script that would be run during the install and prevent proper installation. In this rare situation, it may be necessary to apply a patch to the installation files prior to starting installation.

> **Note:** These steps should only be undertaken at the direction of Oracle Support.

Perform the following steps to patch installation files prior to starting an installation. The steps assume an RMS installation, but apply to any product supported by ORPatch:

1. Unzip the installation files to a staging area.

   > **Note:** The following steps assume the files are in /media/oretail

2. Locate the patch_info.cfg within the product media. The directory it resides in will be used for later steps.

3. find /media/oretail/rms/installer –name patch_info.cfg

4. Output Example:

5. /media/oretail/rms/installer/mom/patch_info.cfg

6. Get the PATCH_NAME for the standard product installation. The patch name to use in subsequent steps will be the portion following the "=" sign.

   ```
   grep "PATCH_NAME=" /media/oretail/rms/installer/mom/patch_info.cfg
   ```

   Output Example:

   PATCH_NAME=**MOM_16_0_0_0**

7. Create a directory that will contain the patch that must be applied, next to the directory with the product installation files.

   > **Note:** The following steps assume this directory is in /media/patch.

8. Unzip the patch into the directory created in step 2.

   > **Note:** This should place the patch contents in /media/patch/<patch num>.

9. Export RETAIL_HOME to point within the installation staging area.

   ```
   export RETAIL_HOME=/media/oretail/rms/installer/mom/Build
   ```

10. Create a logs directory within the installation staging area

    ```
    mkdir $RETAIL_HOME/orpatch/logs
    ```

11. Ensure the ORMerge shell script is executable.

    ```
    chmod u+x $RETAIL_HOME/orpatch/bin/ormerge
    ```

12. Run ORMerge to apply the patch to the installation media, using a –name argument that is the same as what was found in step 3.

    ```
    $RETAIL_HOME/orpatch/bin/ormerge –s /media/patch –d
    /media/oretail/rms/installer/mom –name MOM_16_0_0_0 –inplace
    ```

    > **Note:** The –inplace argument is critical to ensure that the patching replaces files in the mom15 directory.

13. Unset the RETAIL_HOME environment variable.

    ```
    unset RETAIL_HOME
    ```

At this point, the installation files will have been updated with the newer versions of files contained within the patch. Log files for the merge will be in /media/oretail/rms/installer/mom/Build/orpatch/logs.

Providing Metadata to Oracle Support

In some situations, it may be necessary to provide details of the metadata from an environment to Oracle support in order to assist with investigating a patching or application problem. ORPatch provides built-in functionality through the 'exportmetadata' action to extract and consolidate metadata information for uploading to

Oracle Support or for external analysis. For more information, see the ORPatch
'Exporting Environment Metadata' section.

Known Issues/Limitations

Error Messages when Importing ODI Work Repository

The following messages may be displayed; this is a known issue and does not impact the success on the import:

<<Date and Time>> NOTIFICATION ==>> DwgDuplicationTools.importOtherObjects msg = ODI-10036: Importing object SNP_MODEL : RDE Processing Views (with internal id: 22002) failed.

<<Date and Time>> NOTIFICATION Msg = ORA-02291: integrity constraint (ODI_RDE_WREP.FK_MODEL_KDM) violated - parent key not found

<<Date and Time>> NOTIFICATION ==>> DwgDuplicationTools.importOtherObjects msg = ODI-10036: Importing object SNP_MODEL : RMS (with internal id: 12002) failed.

<<Date and Time>> NOTIFICATION Msg = ORA-02291: integrity constraint (ODI_RDE_WREP.FK_MODEL_KDM) violated - parent key not found

Temp Space Issue during ODI Batch Execution

In case if customer experience the following type of error for temp segment while executing the scenario please consider the following recommendation and check with your DBA on the table space availability.
Caused By: java.sql.SQLException: ORA-01652: unable to extend temp segment by 128 in tablespace TEMP1

```
SQL> exec dbms_stats.gather_table_stats('RDE_RMS01', 'W_RTL_NCOST_IT_LC_DY_TMP',
cascade => true);
```

After Analyzing the query using dbms_stats (Stats=Gathered), the plan is:

```
//Good Plan avoiding TEMP issue//

-----------------------------------------------------------------------------------
------------------------------
| Id  | Operation                      | Name                       | Rows  |
Bytes | Cost (%CPU)| Time     |
-----------------------------------------------------------------------------------
------------------------------
|   0 | SELECT STATEMENT               |                            |     3 |
|  1032 |    32   (7)| 00:00:01 |
|*  1 |   HASH JOIN                    |                            |     3 |
|  1032 |    32   (7)| 00:00:01 |
|*  2 |    HASH JOIN                   |                            |     3 |
|   807 |    12   (9)| 00:00:01 |
```

```
|*  3 |     HASH JOIN                     |                               |     3
|   795 |     9  (12)| 00:00:01 |
|   4 |       TABLE ACCESS BY INDEX ROWID| W_RTL_CO_STATUS_IT_LC_CH_DY_A  |     3
|   600 |     5   (0)| 00:00:01 |
|*  5 |         INDEX SKIP SCAN          | PK_W_RTL_CO_STATUS_IT_LC_CH_DY |     3
|     |     1   (0)| 00:00:01 |
|*  6 |       TABLE ACCESS FULL          | W_INT_ORG_DH_RTL_TMP          |   406 |
26390 |     3   (0)| 00:00:01 |
|   7 |       TABLE ACCESS FULL          | W_INT_ORG_D_RTL_TMP           |   485
|  1940 |     3   (0)| 00:00:01 |
|*  8 |     TABLE ACCESS FULL            | W_PRODUCT_D_RTL_TMP           |  4852
|  355K|    19   (0)| 00:00:01 |
```

Creating ODI Master and Work Repositories

RDE ODI Master and Work Repositories should be created manually by RCU for installing Retail Data Extractor.

This process should be done manually by the repository creation wizard. Please refer to Installing a Database and Creating the ODI Database Schemas chapter of the *Oracle Fusion Middleware Installation Guide for Oracle Data Integrator* for creating the Master and Work Repository manually via RCU.

Appendix: Oracle Database 12cR1 Parameter File

```
###############################################################################
# Copyright (c) 2015 by Oracle Corporation
# Oracle 12.1.0.x Parameter file
# NOTES: Before using this script:
#        1.  Change <datafile_path>, <admin_path>, <utl_file_path>, <diag_path>
and <hostname>
#             values as appropriate.
#        2.  Replace the word SID with the database name.
#        3.  Size parameters as necessary for development, test, and production
environments.
# --------------------------------------------------------------------
*.audit_file_dest=full_path_of_audit_dir
*.audit_trail='db'
*.compatible='12.1.0.2'
*.control_files='full_path_of_controlfile_1','full_path_of_controlfile_2'
###########################################
# Memory Settings:
# xxxM = Some reasonable starting value for your environment.
###########################################
*.db_block_size=xxxM
*.db_cache_size=xxxM
*.java_pool_size=xxxM
*.memory_target=xxxM
*.pga_aggregate_target=xxxM
*.shared_pool_size=xxxM
*.streams_pool_size=xxxM

###########################################

*.db_block_size=8192
*.db_domain=''
*.db_name='dbName'
*.diagnostic_dest='full_path_of_diag_dir'
*.enable_pluggable_database=true|false
*.fast_start_mttr_target=900
*.nls_calendar='GREGORIAN'
*.nls_date_format='DD-MON-RR'
*.nls_language='AMERICAN'
*.nls_numeric_characters='.,'
*.nls_sort=BINARY
*.open_cursors=900
*.os_authent_prefix=''
*.plsql_optimize_level=2
*.processes=2000
*.query_rewrite_enabled='true'
*.remote_dependencies_mode='SIGNATURE'
*.remote_login_passwordfile='EXCLUSIVE'
*.remote_os_authent=true
*.sec_case_sensitive_logon=false
```

```
*.undo_tablespace='UNDOTBS1'
```

```
*.undo_tablespace='UNDOTBS1'
```

Appendix: CreateDatabase Instance Using an Oracle GenericTemplate – Example

Instance Creation Using a Generic Template via DBCA

Prerequisites:

- 12.1.0.2 binary must have already been installed along with the required oneoff patches. Refer to the Database Server Preinstallation section for all the required oneoff patches.
- Ensure that the template file in $ORACLE_HOME/assistant/template exists:

  msp52833:[dvols179] /u00/oracle/product/12.1.0.2/assistants/dbca/templates>
  --> ls -l General_Purpose.dbc
  -rw-r--r-- 1 rgbuora dba 4908 Jul 7 2015 General_Purpose.dbc

Instance Creation Using the Generic Template via DBCA

1. Ensure ORACLE_HOME and ORACLE_BASE is in the path:
   ```
   export ORACLE_HOME=/u00/oracle/product/12.1.0.2
   export ORACLE_BASE=/u00/oracle
   export PATH=$ORACLE_HOME/bin:$PATH
   ```
   .cd into /u00/oracle/product/12.1.0.2/assistants/dbca/templates

2. Execute the following command to create an instance:
   ```
   $ORACLE_HOME/bin/dbca -silent -createDatabase -templateName
   General_Purpose.dbc -gdbName DB_NAME -sid DB_SID  -createAsContainerDatabase
   true -SysPassword oracle1 -SystemPassword oracle1 -emConfiguration NONE -
   datafileDestination /u02/oradata -characterSet AL32UTF8 -nationalCharacterSet
   AL16UTF16 -redoLogFileSize 100 -initParams nls_date_format=DD-MON-
   RR,nls_language=AMERICAN,nls_calendar=GREGORIAN,fast_start_mttr_target=900
   ```

   The above will create a container database using all the default parameters set by dbca. Replace the pfile by taking a copy from Appendix A but customize the values according to the need of your environment.

   If you wish to create a non-container database, replace [-createAsContainerDatabase **true**] with [-createAsContainerDatabase **false**].

3. Execute the following commands to create a pluggable database if this is a container environment.
   ```
   CREATE PLUGGABLE DATABASE PDB_NAME ADMIN USER PDBADMIN
   IDENTIFIED BY pdbadmin_pwd ROLES=(CONNECT)
   file_name_convert=('/u02/oradata/cdb_name/pdbseed','/u02/oradata/pdb_name');

   alter pluggable database pdb_name open;

   alter system register;
   ```

4. Post Database Creation Setup

   The above commands create a database with all files in one directory, ie, /u02. Multiplex the redo logs and the control files following the OFA architecture.

5. Configure the listener and the tnsnames entry.

6. Log into the pluggable database to create the required tablespaces accordingly. For non-container databases, log into the database as normal and create the tablespaces.

Appendix: Tablespace Creation Scripts

```
-------------------------------------------------------------------------------
---
--- Script:         create_rde_tablespaces.sql
--- Execute as:     sysdba
--- Note:           Before running this script:
---                     Modify <datafile_path> values.
---                     Modify datafile storage parameters and sizes based on
partitioning strategy.
-------------------------------------------------------------------------------
spool create_ra_tablespaces.log

set echo on
CREATE TABLESPACE DM_DIM_DATA
   DATAFILE '<datafile_path>/dm_dim_data01.dbf' SIZE 300M    AUTOEXTEND
ON NEXT 100M MAXSIZE 2000M
     EXTENT MANAGEMENT LOCAL
     SEGMENT SPACE MANAGEMENT AUTO
;

CREATE TABLESPACE DM_DIM_INDEX
DATAFILE '<datafile_path>/dm_dim_index01.dbf' SIZE 300M AUTOEXTEND ON
NEXT 100M MAXSIZE 2000M
     EXTENT MANAGEMENT LOCAL
     SEGMENT SPACE MANAGEMENT AUTO
;

CREATE TABLESPACE DM_FACT_DATA
   DATAFILE '<datafile_path>/ dm_fact_data01.dbf' SIZE 300M
     AUTOEXTEND ON NEXT 100M MAXSIZE 2000M
     EXTENT MANAGEMENT LOCAL
     SEGMENT SPACE MANAGEMENT AUTO
;

CREATE TABLESPACE DM_FACT_INDEX
   DATAFILE '<datafile_path>/dm_fact_index01.dbf'  SIZE 300M
     AUTOEXTEND ON NEXT 100M MAXSIZE 2000M
     EXTENT MANAGEMENT LOCAL
     SEGMENT SPACE MANAGEMENT AUTO

;

CREATE TABLESPACE RETAIL_INDEX
   DATAFILE '<datafile_path>/retail_index01.dbf'  SIZE 100M
     AUTOEXTEND ON NEXT 100M MAXSIZE 2000M
     EXTENT MANAGEMENT LOCAL
     SEGMENT SPACE MANAGEMENT AUTO
;

CREATE TABLESPACE RETAIL_DATA
   DATAFILE '<datafile_path>/retail_data01.dbf'  SIZE 100M
     AUTOEXTEND ON NEXT 100M MAXSIZE 2000M
```

```
            EXTENT MANAGEMENT LOCAL
            SEGMENT SPACE MANAGEMENT AUTO
        ;


        CREATE TABLESPACE USERS
            DATAFILE '<datafile_path>/users01.dbf'  SIZE 100M
            AUTOEXTEND ON NEXT 100M MAXSIZE 1000M
            EXTENT MANAGEMENT LOCAL
            SEGMENT SPACE MANAGEMENT AUTO
        ;
        spool off
```

D

Appendix: Retail Data Extractor Application Installer Screens

Understanding of the following details about your environment is required to ensure the installer successfully deploys the Retail Data Extractor application. Depending on the options you select, you may not see some screens or fields.

> **Note:** The values shown in the text fields in the screenshots and in examples are sample values. Enter appropriate values for your organization for all text fields as you go through the UI screens.
>
> **Note:** When running in text mode, and when you are presented with questions having to do with making choices, you are expected to fully spell out the selected item. For example, a checkbox control in graphical user interface will appear as (yes/no) question in text mode. You must enter Yes or No. Entering anything other than Yes in this case will result in No.
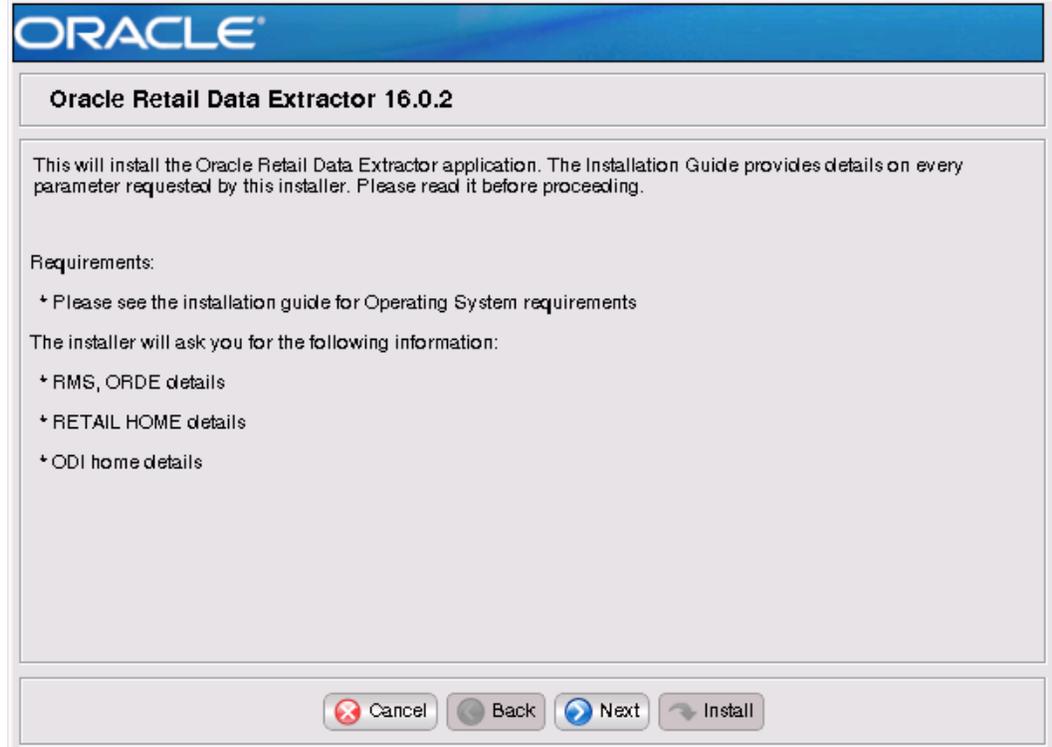
A separate section is provided below for full installations. Please refer to the appropriate sections for installation.

Installer Screens for Analyze tool:
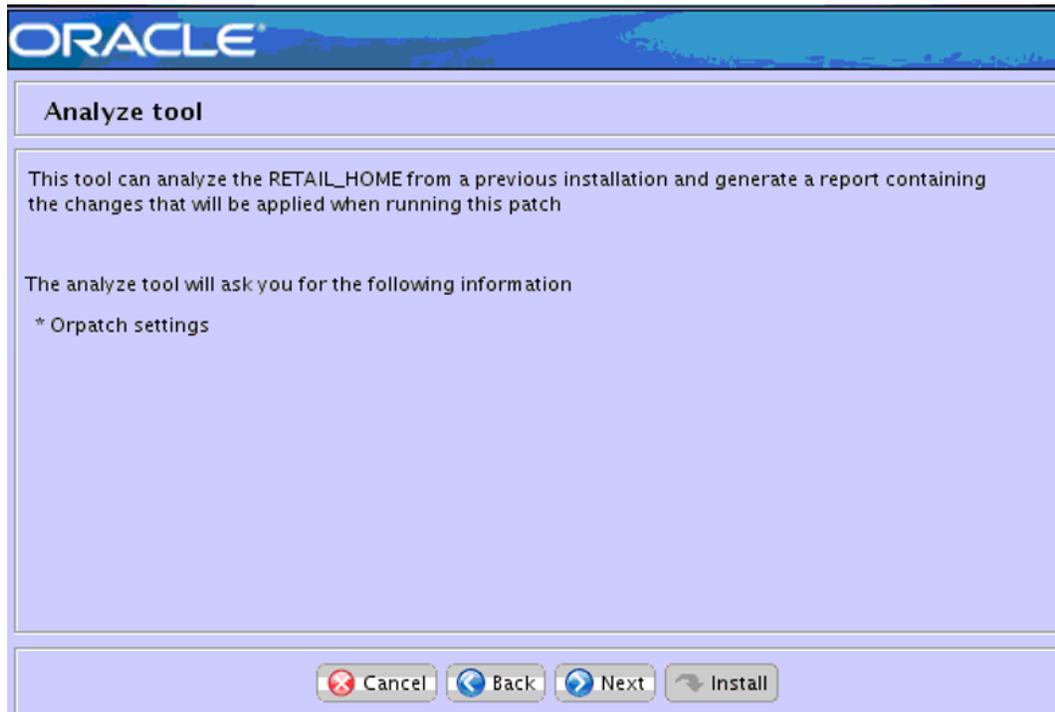
This section is only valid if ORPATCH_ANALYZE is set to true with the environment variables.

Screen: Oracle Retail Data Extraction 16.0.2

Screen: Analyze tool

This screen displays the information about analyze tool

Screen: RETAIL_HOME to Analyze

This screen asks for the retail home to analyze

Field Title	RDE Application RETAIL_HOME
Field Description	Retail home path
Notes	Retail Home to analyze
Example	/u00/oracle/retail_home

Installer Screens for Full installation:

Screen: Oracle Retail Data Extractor 16.0.2

This screen displays the requirements for installing Retail Data Extractor 16.0.2
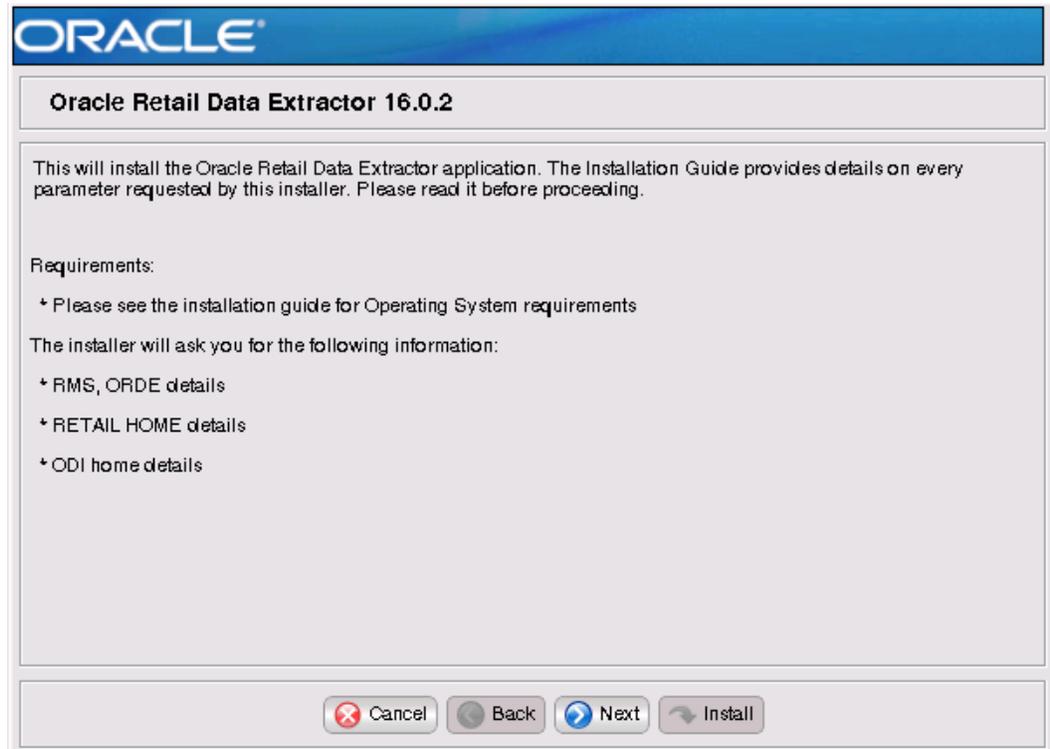
Screen: ORDE Application RETAIL_HOME

This screen will ask for RETAIL_HOME

Field Title	ORDE Application RETAIL_HOME
Field Description	Retail home path
Notes	Retail Home is used to keep Orpatch related files and the mmhome files by default. Please keep track of this directory, it should remain in place after installation and will be used to apply future patches. The mmhome stores all shell and DDL scripts.
Example	/u00/webadmin/retail_home

Screen: ODI Home Directory

This screen is displayed only if you had checked off the Install ODI files box from the Component Selection screen. Field Title	ODI Home directory
Field Description	ODI Home directory
Example	/u01/product/fmw/wls_odi/odi/agent
Notes	▪ This directory must be the ODI Home where ODI is installed ▪ You must run the installer as the same OS user that installed ODI. Due to the installer characteristics, the installer files must be owned by this same user also.

Screen: ODI Supervisor Details

Field Title	ODI Supervisor Username
Field Description	This is the ODI supervisor user which is used to connect to ODI GUI.
Example	SUPERVISOR

Field Title	ODI Supervisor Password
Field Description	ODI Supervisor Password

Field Title	Supervisor user alias
Field Description	ODI Supervisor alias

Screen: Work Repository Details

This screen is displays only if you had checked the **Do you wish to enter MREP and WREP details checkbox from Select to Install ODI files and Create Repository.**

Field Title	Work Repository Username
Field Description	The ODI work repository user for Retail Data Extractor.
Example	ODI_WREP_USER

Field Title	Work Repository Password
Field Description	Provide the work repository password.

Field Title	ODI WREP Alias
Field Description	Provide the work repository alias used while storing password in java wallet as a part of pre-installation tasks

Screen: ODI Domain Home and Agent Details

Field Title	ODI Domain home
Field Description	ODI domain path
Example	/u00/webadmin/config/domains/wls_odi/

Field Title	ODI Hostname
Field Description	Hots name with odi installation
Example	msp000adg.us.oracle.com

Field Title	ODI agent name
Field Description	ODIAgent name to launch the odi import files
Example	ODIInvoke

Field Title	ODI Agent port number
Field Description	ODIAgent port number
Example	20910

Screen: ORDE RMS Oracle Retail Data Extractor Host and Schema Details

This screen is displayed only if you had checked off the **Install RMS objects for Oracle Retail Data Extractor** box from the Component Selection screen

Field Title	RMS ORDE DB Hostname
Field Description	RMS hostname on which the RMS master schema is created.
Example	msp00ayx.us.oracle.com

Field Title	RMS Master schema name
Field Description	RMS master schema username
Example	RMS01

Field Title	RMS Master schema password
Field Description	RMS master schema password for authentication
Example	password

Field Title	RMS DB user alias
Field Description	DB alias

Field Title	RMS ORDE schema service name
Field Description	Schema service name
Example	Pkolsp04app

Field Title	RMS ORDE DB port
Field Description	DB port to connect to the database host
Example	1521

Screen: ORDE RMS Oracle Retail Data Extractor JDBC URL

This screen is displayed only if you had checked the **Install Oracle Retail Data Extractor** box from the Component Selection screen.

Field Title	RMS master jdbc url
Field Description	RMS JDBC connection url
Example	Pkolsp04app

Field Title	ORDE Datamart Schema SERVICE_NAME
Field Description	Oracle Retail Data Extractor Datamart schema SERVICE_NAME
Example	rdeservicename

Field Title	ORDE JDBC URL
Field Description	Oracle Retail Data Extractor JDBC URL
Example	jdbc:oracle:oci:@rdeservicename

Field Title	Test Data Source?
Field Description	Check the box if you want the installer to test the connection to the schema upon clicking Next. Clear the box to bypass the validation.

Screen: Oracle Retail Data Extractor Host and Schema Details

This screen is displayed only if you had checked the **Install Oracle Retail Data Extractor** box from the Component Selection screen.

Field Title	ORDE Batch schema Username
Field Description	Oracle Retail Data Extractor Batch schema name
Example	RDE_RMS01

Field Title	ORDE Batch Schema password
Field Description	Oracle Retail Data Extractor Batch schema password

Field Title	ORDE Batch User alias
Field Description	Oracle Retail Data Extractor Batch user alias

Screen: Oracle Retail Data Extractor Datamart Schema Details

This screen is displayed only if you had checked the **Install Oracle Retail Data Extractor** box from the Component Selection screen.

Field Title	ORDE Datamart Schema Username
Field Description	Oracle Retail Data Extractor Datamart schema name
Example	RDE_DM01

Field Title	ORDE Datamart Schema Password
Field Description	Oracle Retail Data Extractor Datamart schema password

Field Title	ORDE Datamart Schema Alias
Field Description	Oracle Retail Data Extractor Datamart schema alias

Screen: Oracle Wallet

An oracle wallet is an encrypted container used to store and retrieve sensitive information, such as user credentials. This screen will ask you for a new password for wallet if you have opted to install db components.

Field Title	Oracle Wallet password
Field Description	The wallet password
Notes	The password for wallet to store the credentials of db components. The password must have a minimum length of eight characters and contain alphabetic character combined with numbers or special characters. Note: If a wallet already exists for an MMHOME you have selected, this password must match the password for the existing wallet. Make sure this password is kept as it will be needed for future patches.

Field Title	Oracle Wallet alias
Field Description	The wallet alias
Notes	The alias stored in wallet created during pre-installation step for silent install

Screen: Installation Summary

This screen shows the selections you have made so far. Not all fields will be displayed. For example, DB schema user passwords are not displayed regardless of whether you selected to do DB schema installation.
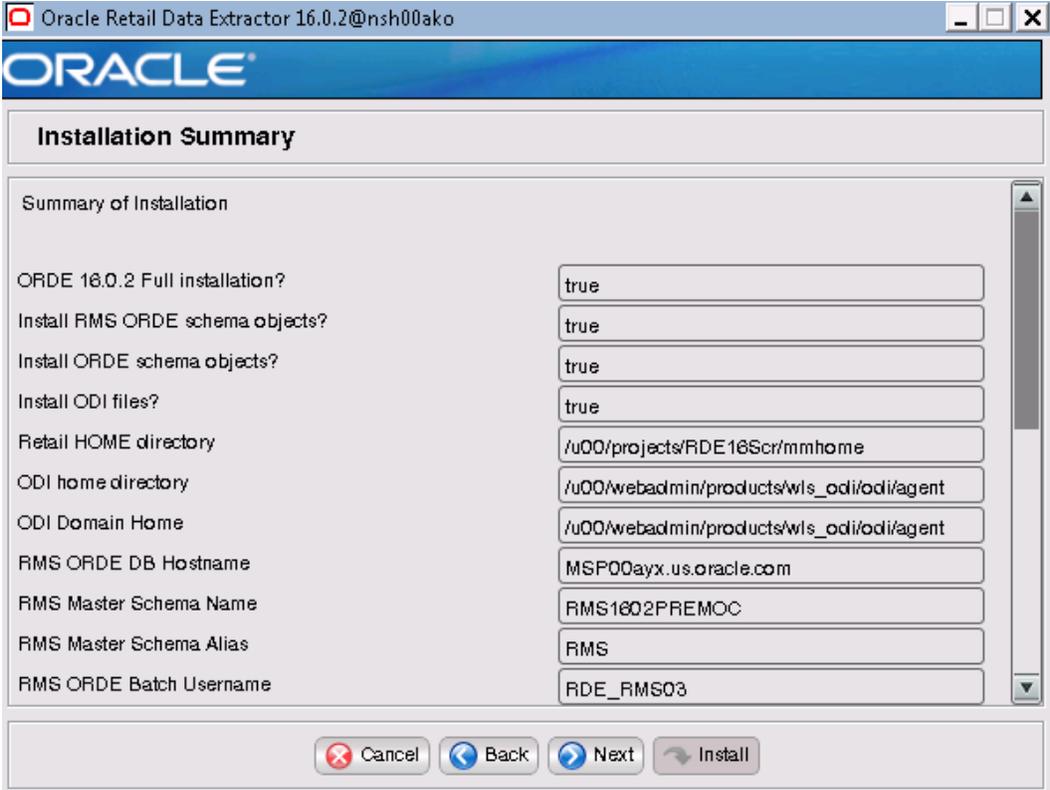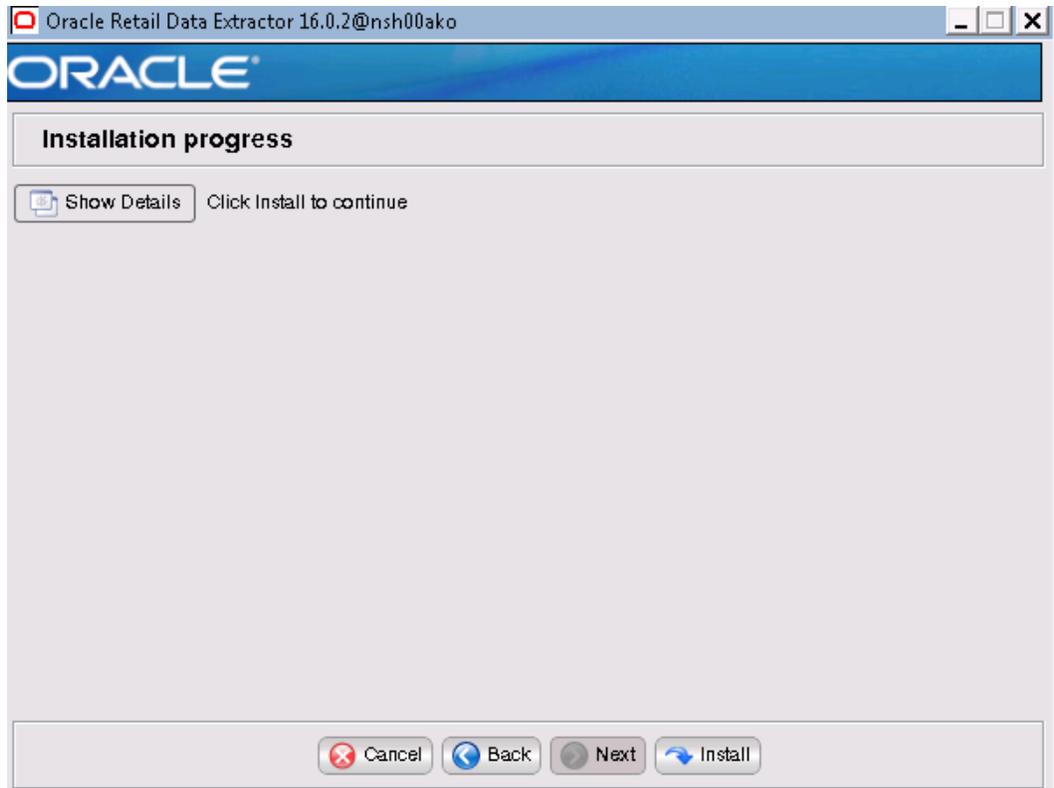
Installation Trail File

When the installer exits (at the end of successful/failed installation), it produces a trail file called ra.<version>install.trail.properties in <STAGING_DIR>/ora/installer, (for example,. ra.16.0.2.0.install.trail.properties). The file content reflects the choices you made and the statuses of the various components you have or have not chosen (MMHOME listed in the file is actually RETAIL_HOME/mmhome). For example:

```
#Thu, 15 May 2015 22:57:30 -0500

Installation_Attempted_At=201511061631
Installation_Status=Success
RA_version=16.0.2.0.0
MMHOME_Installed_or_Upgraded=Yes
MMHOME=/u00/projects/rde16.0.2/qa8/mmhome
ODI_Files_Installed_or_Upgraded=Yes
ODI_Home=/u01/product/fmw/wls_odi/odi/agent
ODI_Repositories_Installation_Status=Yes
RA_RMS_User_Database_Installation_Status=Installed v16.0.2.0.0
RA_User_Database_Installation_Status=Installed v16.0.2.0.0
```

If this file already exists when you run the installer (for example, running the installer for the second time), the installer will read in the existing file's settings as a basis to work with. This means, for example, if you had already installed ODI components the first time, and you are installing Retail Data Extractor database objects now, then the file will continue to reflect the status of ODI component and will update Retail Data Extractor database installation status.

Since this file is produced in <STAGING_DIR>/ora/installer for your installer, it reflects only the activities you carried out using your installer. If another person had another copy of the installer somewhere and already finished installing certain components, your trail file will not reflect that and it is easy for it to be out of sync with reality. Therefore, keep this in mind and use the file as a loose guide only. You may want to manually update your trail file from time to time to keep it accurate.

E

Appendix: Installer Silent Mode

In addition to the GUI and text interfaces of the Retail Data Extractor installer, there is a silent mode that can be run. This mode is useful if you want to run a repeat installation without retyping the most settings you provided in the previous installation. It is also useful if you encounter errors in the middle of an installation and want to continue.

The installer runs in two distinct phases. The first phase involves gathering settings from the user. At the end of the first phase, a properties file named ant.install.properties is created with the settings that were provided. Then the second phase begins, where this properties file is used to provide your settings for the installation.

> **Note:** Sensitive credential information such as schema passwords are deleted from ant.install.properties, so you will need to provide them again before running the installer in silent mode

To skip the first phase and re-use the ant.install.properties file from a previous run, follow these instructions:

1. Edit the ant.install.properties file and correct any invalid settings that may have caused the installer to fail in its previous run.

2. Look for duplicate properties in the ant.install.properties file. Some properties are set on multiple pages to ensure default values when a page is only displayed under certain conditions. For example, if there are two instances of input.property.name, remove all but the last one.

3. Run the installer again with the **silent** argument.

> **Example:** ksh install.sh silent

Appendix: Common Installation Errors

This section provides some common errors encountered during installation of Retail Data Extractor.

Installer Hangs on Startup

Symptom:

When the database schema installer is run, the following is written to the console and the installer hangs indefinitely:

```
Running pre-install checks
Running tnsping to get listener port
```

Solution:

The installer startup script is waiting for control to return from the **tnsping** command, but tnsping is hanging. Type Control+C to cancel the installer, and investigate and solve the problem that is causing the **tnsping <sid>** command to hang. This can be caused by duplicate database listeners running.

Unreadable Buttons in the Installer

If you are unable to read the text within the installer buttons, it probably means that your JAVA_HOME is pointed to a pre-1.7 JRE or JDK.

Set JAVA_HOME with the appropriate JDK (the same JDK that has been used by WebLogic Server).

Warning: Could not create system preferences directory

Symptom:

The following text appears in the installer Errors tab:

```
May 22, 2006 11:16:39 AM java.util.prefs.FileSystemPreferences$3 run
WARNING: Could not create system preferences directory. System preferences are
unusable.
May 22, 2006 11:17:09 AM java.util.prefs.FileSystemPreferences
checkLockFile0ErrorCode
WARNING: Could not lock System prefs. Unix error code -264946424.
```

Solution:

This is related to Java bug 4838770. The /etc/.java/.systemPrefs directory may not have been created on your system. See http://bugs.sun.com for details.

This is an issue with your installation of Java and does not affect the Oracle Retail product installation.

Warning: Couldn't find X Input Context

Symptom:

The following text appears in the console window during execution of the installer in GUI mode:

Couldn't find X Input Context

Solution:

This message is harmless and can be ignored.

Message: SP2-0734: unknown command beginning

Symptom:

Installation outputs the message that looks like:

SP2-0734: unknown command beginning "old   5: ..." – rest of line ignored.

or

SP2-0734: unknown command beginning "new   4: ..." – rest of line ignored.

Solution:

You can ignore this error. This warning is generated from invalids.sql if no objects need to be validated.

Message: Invalid Username/Password; Login Denied

Symptom:

Installation outputs the message:

Error: java.sql.SQLException: ORA-01017: invalid username/password; logon denied

Solution:

The installer cannot log into the database as one of the users you have supplied it. Verify that they have been created with SQL files in <STAGING_DIR>/ora/installer/create_db per "Create Retail Data Extractor Schema Owners" in Chapter 2 and verify that you can manually log into the database with them.

Message: Adding credentials to the wallet for … BUILD FAILED

Symptom:

Installer starts to add DB credentials to the wallet and then give out an error like this:

```
[echo] Adding credentials to wallet for ...
```

```
BUILD FAILED
/view_storage/mohammz_ra/rgbura/dist/ora/installer/build.xml:474: The following
error occurred while executing this line:
/view_storage/mohammz_ra/rgbura/dist/ora/installer/common-ora-install.xml:213:
exec returned: 1
```

Solution:

It is likely that you have run the installer in silent mode and ant.install.properites file did not have the password set for the DB schemas.

If you did fill out the passwords in a previous run, the installer had cleared the password fields for all schemas (e.g. input.rms.db.password, input.rdm.db.password, etc) when the installer starts to run. This is done for security So next time, you need to re-populate these fields before re-running the installation.

Another possible cause is if you entered an invalid character for schema username, password or SID. For example, if you mistakenly entered "<host>:<port>:<sid>" instead of "<sid>" for schema SID, then it will cause mkstore utility to crash, resulting in the error in discussion.

Message: Error Connecting to Database URL

Symptom:

After entering database credentials in the installer screens and hitting next, a message pops up with an error like this:

```
Error connecting to database URL <url> as user <user>
details...
```

The message prevents you from moving on to the next screen to continue the installation.

Solution:

This error occurs when the installer fails to validate the user credentials you have entered on the screen. Make sure that you have entered the credentials properly.

You may receive a message similar to this:

```
Error connecting to database URL <url> as user <user>
java.lang.Exception: UnsatisfiedLinkError encountered when using the Oracle
driver.
Please check that the library path is set up properly or switch to the JDBC thin
client.
```

This message means that bit-width for your Java and Oracle client libraries are not compatible with each other. Make sure that you are using only the 64-bit version of Java and Oracle client libraries.

Message: Cannot access NLS data files or invalid environment specified

Symptom:

After entering database credentials in the installer screens and hitting next, a message pops up with an error like this:

```
Error connecting to database URL jdbc:oracle:oci:@pkols07 as user XYZ
java.sql.SQLException: ORA-12705: Cannot access NLS data files or invalid
environment
```

Solution:

This error occurs if the NLS_LANG environment variable has not been set and exported with a valid value expected by the Oracle database server. See the section, Run the Retail Data Extractor Database Schema Installer, in Chapter 2.

Message: User XYZ lacks CREATE SESSION privilege; log on denied

Symptom:

After entering database credentials in the installer screens and hitting next, a message pops up with an error like this:

```
Error connecting to database URL jdbc:oracle:oci:@pkols07 as user rarms1
java.sql.SQLException: ORA-01045: user RDE_RMS01 lacks CREATE SESSION privilege;
logon denied
```

Solution:

This error occurs if the schema user in question has not been created properly. Make sure there was no error when you ran one of the schema user creation scripts in <STAGING_DIR>/ora/installer/create_db per "Create Retail Data Extractor Schema Owners" in Chapter 2. You may find it easier to drop the schema user and re-create it as opposed to manually granting the missing privilege(s).

Message: Some of the objects have errors

Symptom:

During DB schema object creations step for a given schema, after all objects have been successfully created, it gives "Some of the objects have errors" when it tries to compile any invalid objects.

```
    [exec] Execution of INV_OBJ_COMP script
    [exec] Some of the objects have errors. Open a sql session and run the
command below to find out the invalid objects
    [exec] ******************************************************
    [exec] * select * from user_objects where status != 'VALID'
    [exec] ******************************************************
```

Solution:

As the error message suggests, find out the invalid objects and resolve accordingly. This error can manifest itself for a variety of reasons but here are some possible causes:

- You created temporary invalid objects in the schema and forgot to drop them when the installer was run.
- If this happened in the RMS user schema, it is possible that the invalid objects reported came from the master RMS schema.
  - Someone might have created invalid objects in the master RMS schema that may not have anything to do with Retail Data Extractor, and may be outside of your control as the master RMS schema. If you want to resolve these invalid objects, consult someone responsible for maintaining the master RMS schema to resolve the invalid objects, and rerun the installation.

Alternatively, you may want to ignore this error message and continue with the rest of the schema objects installations. You can do this if, after reviewing the list of invalid objects that failed to compile, according to the log indicated by the installer, you determine they do not need to be recompiled during the installation. In this case you can rerun the installation by choosing to resume from the previous point of failure and clearing the "RMS User Schema Install Option" check box. In silent mode, you should set input.do.install.rms.db to "false" in the installer properties file, ant.install.properties, in order to not install RMS user schema objects. This will cause the RMS user schema object installation to be skipped, and the installation will continue with the next schema. For details on how to resume database schema object installations from the previous point of failure, see "Resuming from the Previous Point of Failure" in the section, Resolving Errors Encountered During Database Schema Installation.

WARNING: Expected * SYNONYM objects, found X

Symptom:

Toward the end of the installation involving database schema objects, the installer will issue: "WARNING: Expected * SYNONYM objects, found 9026" for RMS user schema.

Solution:

This warning can be ignored. The installer does not validate the number of SYNONYM objects for RMS user schema.

Fatal exception: Width (0) and height (0) cannot be <= 0 java.lang.IllegalArgumentException: Width (0) and height (0) cannot be <= 0

Symptom:

When running the installer in GUI mode, the screens fail to open and the installer ends, returning to the console without any error message. The ant.install.log file contains this error. This is an error encountered when the installer is used in GUI mode with certain X Servers.

Solution:

Until this is fixed permanently, employ the workaround solution below should you encounter this issue:

1. Copy ant.install.properties.sample to ant.install.properties
2. Re-run the installer

Installation Overview

Installing an Oracle Retail supported Single Sign-On installation using OAM11g requires installation of the following:

1. Oracle Internet Directory (OID) LDAP server and the Oracle Directory Services Manager. They are typically installed using the Installer of Oracle Identity Management. The ODSM application can be used for user and realm management within OID.

2. Oracle Access Manager 11gR2 has to be installed and configured.

3. Additional midtier instances (such as Oracle Forms 11gr2) for Oracle Retail applications based on Oracle Forms technologies (such as RMS). These instances must be registered with the OAM installed in step 2.

4. Additional application servers to deploy other Oracle Retail applications and performing application specific initialization and deployment activities must be registered with OAM installed in step 2.

Infrastructure Installation and Configuration

The Infrastructure installation for Oracle Access Manager (OAM) is dependent on the environment and requirements for its use. Deploying Oracle Access Manager (OAM) to be used in a test environment does not have the same availability requirements as for a production environment. Similarly, the Oracle Internet Directory (OID) LDAP server can be deployed in a variety of different configurations. See the *Oracle Identity Management Installation Guide11g.*

OID User Data

Oracle Internet Directory is an LDAP v3 compliant directory server. It provides standards-based user definitions out of the box.

Customers with existing corporate LDAP implementations may need to synchronize user information between their existing LDAP directory servers and OID. OID supports standard LDIF file formats and provides a JNDI compliant set of Java classes as well. Moreover, OID provides additional synchronization and replication facilities to integrate with other corporate LDAP implementations.

Each user ID stored in OID has a specific record containing user specific information. For role-based access, groups of users can be defined and managed within OID. Applications can thus grant access based on group (role) membership saving administration time and providing a more secure implementation.

User Management

User Management consists of displaying, creating, updating or removing user information. There are many methods of managing an LDAP directory including LDIF scripts or Oracle Directory Services Manager (ODSM) available for OID11g.

ODSM

Oracle Directory Services Manager (ODSM) is a Web-based application used in OID11g is designed for both administrators and users which enables you to configure the structure of the directory, define objects in the directory, add and configure users, groups, and other entries. ODSM is the interface you use to manage entries, schema, security, adapters, extensions, and other directory features.

LDIF Scripts

Script based user management can be used to synchronize data between multiple LDAP servers. The standard format for these scripts is the LDAP Data Interchange Format (LDIF). OID supports LDIF script for importing and exporting user information. LDIF scripts may also be used for bulk user load operations.

User Data Synchronization

The user store for Oracle Access Manager resides within the Oracle Internet Directory (OID) LDAP server. Oracle Retail applications may require additional information attached to a user name for application-specific purposes and may be stored in an application-specific database. Currently, there are no Oracle Retail tools for synchronizing changes in OID stored information with application-specific user stores. Implementers should plan appropriate time and resources for this process. Oracle Retail strongly suggests that you configure any Oracle Retail application using an LDAP for its user store to point to the same OID server used with Oracle Access Manager.

G

Appendix: Installation Order

This section provides a guideline as to the order in which the Oracle Retail applications should be installed. If a retailer has chosen to use some, but not all, of the applications the order is still valid less the applications not being installed.

> **Note:** The installation order is not meant to imply integration between products.

Enterprise Installation Order

1. Oracle Retail Merchandising System (RMS), Oracle Retail Trade Management (RTM)
2. Oracle Retail Sales Audit (ReSA)
3. Oracle Retail Extract, Transform, Load (RETL)
4. Oracle Retail Warehouse Management System (RWMS)
5. Oracle Retail Invoice Matching (ReIM)
6. Oracle Retail Price Management (RPM)
7. Oracle Retail Allocation
8. Oracle Retail Mobile Merchandising (ORMM)
9. Oracle Retail Customer Engagement (ORCE)
10. Oracle Retail Xstore Office
11. Oracle Retail Xstore Point-of-Service, including Xstore Point-of-Service for Grocery, and including Xstore Mobile
12. Oracle Retail Xstore Environment
13. Oracle Retail EFTLink
14. Oracle Retail Store Inventory Management (SIM), including Mobile SIM
15. Oracle Retail Predictive Application Server (RPAS)
16. Oracle Retail Predictive Application Server Batch Script Architecture (RPAS BSA)
17. Oracle Retail Demand Forecasting (RDF)
18. Oracle Retail Category Management Planning and Optimization/Macro Space Optimization (CMPO/MSO)
19. Oracle Retail Replenishment Optimization (RO)
20. Oracle Retail Regular Price Optimization (RPO)
21. Oracle Retail Merchandise Financial Planning (MFP)
22. Oracle Retail Size Profile Optimization (SPO)
23. Oracle Retail Assortment Planning (AP)
24. Oracle Retail Item Planning (IP)
25. Oracle Retail Item Planning Configured for COE (IP COE)
26. Oracle Retail Advanced Inventory Planning (AIP)
27. Oracle Retail Integration Bus (RIB)
28. Oracle Retail Service Backbone (RSB)
29. Oracle Retail Financial Integration (ORFI)

30. Oracle Retail Bulk Data Integration (BDI)

31. Oracle Retail Integration Console (RIC)

32. Oracle Commerce Retail Extension Module (ORXM)

33. Oracle Retail Data Extractor for Merchandising (RDE)

34. Oracle Retail Clearance Optimization Engine (COE)

35. Oracle Retail Analytic Parameter Calculator for Regular Price Optimization (APC-RPO)

36. Oracle Retail Insights (RI), including Retail Merchandising Insights (previously Retail Merchandising Analytics) and Retail Customer Insights (previously Retail Customer Analytics)

37. Oracle Retail Order Broker