Oracle® Retail Merchandising System

Installation Guide
Release 19.0.1
F44216-04

February 2025

ORACLE

Oracle® Retail Merchandising System Installation Guide, Release 19.0.1

Primary Author:

Contributors:

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 Merchandising System, Installation Guide, Release 19.0.1

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 and 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, 19.0) or a later patch release (for example, 19.0.1). 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 Help Center

Documentation is packaged with each Oracle Retail product release. Oracle Retail product documentation is also available on the following Web site:

https://docs.oracle.com/en/industries/retail/index.html

(Data Model documents are not available through Oracle Help Center. These documents are packaged with released code, or you can obtain them through My Oracle Support.)

> **Note:** In the images or examples below, user details / company name / address / email / telephone number represent a fictitious sample. Any similarity to actual persons, living or dead is purely coincidental and not intended in any manner.

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 chapter includes tasks to complete before installation.

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

Installation Terminology

STAGING_DIR – The directory where the rms19installer.zip is copied and extracted locally.

RETAIL_HOME – The directory where Database Files are stored, and Batch and Application are installed. This will contain the orpatch directory.

- Database RETAIL_HOME – The location where RMS Database Files are stored. This location will be used during the subsequent patching of the RMS.
- Batch RETAIL_HOME – This is the Batch installation directory, where the RMS Batch Files are installed.
- Application RETAIL_HOME – This is the Application installation directory, where the RMS application files are installed and staged for WebLogic deployment.

> **Note:** The RETAIL_HOME for database, batch, and application can be the same.

Architecture & Capacity Planning

Retailers have many options in designing the physical architecture. Every retailer will have different physical architecture requirements based on:

- Budget
- HA and DR requirements
- Governance practices in the territories they trade in
- Constraints imposed by 3rd party systems that will integrate with Merchandising
- Larger corporate IT strategies
- Fundamental corporate business strategy

The information in this document not intended to replace the advice of retailer IT staff or a knowledgeable System Integration partner. This architecture information is only intended to introduce new customers to deployment topics that retailers that affect the installation process.

Environment Strategy

Retailers implementing Merchandising should plan to have multiple environments. Fundamental corporate requirements and the larger context of the implementation program generally drive environment strategy. Common examples of environments and usage include, but are not limited to:

- Development – Generally used to develop integrations. Development environments are often short lived, not sized for production loads and not clustered for HA. Development environments may or may not be fully integrated via Single Sign ON (SSO) and federated identity management.

- User Acceptance Test – Generally used for end user training and testing. UAT environments generally do not have HA and DR requirements, but may be fully integrated via SSO with federated identity management to facilitate full business user training.
- System/Integration Test – System/Integration Test environments are often longer lived, and may in fact be permanent and used after go live to test Merchandising patches. System/Integration test environments may be fully sized so they can also be used as performance testing environments.
- Production – Actual production environment, which will be used for day to day Merchandising business activities.
- Disaster Recovery – Disaster Recovery (DR) environment is used when a production environment fails. DR environment requirements depend on corporate SLAs.

Different types of environments generally have different integration, HA and DR requirements. These requirements will drive capacity planning and physical architecture. These requirements also affect the Merchandising installation process.

High Availability & Disaster Recovery

Retailers will need to determine their availability and disaster recovery requirements before implementing Merchandising. These architectural decisions will affect both capacity planning and the installation process.

Oracle's Maximum Availability Architecture (MAA) provides blue prints for achieving various levels of HA and DR. Key technologies involved in MAA include

- Data Tier
  - Oracle Real Application Clusters (RAC)
  - Active Data Guard
- Application Tier
  - WebLogic Clustering

https://www.oracle.com/database/technologies/high-availability/maa.html

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 infrastructure vendor to request a storage 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
- Data sparsity
- Application features utilized
- Length of time history is retained

Additional considerations during this process include your HA requirements, backup policies and DR requirements.

Additional Components

This document discusses the requirements for the core database and application servers required for Merchandising. Additional technical components are also required for production. Most retailers will already have these some of these components (SFTP

server, federated IDM, and so on) in place, and the installation and implementation activity will be to link these components to Merchandising.

In other cases, retailers may need to procure or assign infrastructure to Merchandising. These dedicated Merchandising components should be considered in capacity planning.

Technical Requirements

Oracle Applications are developed and tested on Oracle Linux, which is optimized for performance, stability, and security.

Web Tier

Oracle maintains a consistent web browser support policy for all applications.

https://www.oracle.com/technetwork/indexes/products/browser-policy-2859268.html

Per this policy, at the time of this document (January 2020), Merchandising supports the following browsers:

Browser	Versions
Mozilla Firefox	ESR 78.11.0
Microsoft Internet Explorer	11
Microsoft Edge	91.0.864.54
Google Chrome	91.0.4472.106
Apple Safari	N/A

Application Tier

The Merchandising applications require application servers. General requirements for an application server running Merchandising include the following

Supported on:	
Middleware	Oracle Fusion Middleware 12.2.1.4.0 ▪ FMW 12.2.1.4.0 Infrastructure (WLS and ADF included) ▪ Oracle Enterprise Manager Fusion Middleware Control 12.2.1.4.0 ▪ Oracle Analytic Server 5.5 for legacy reports ▪ Oracle Identity Management 12c(12.2.1.4.0) Oracle DB Client 19.3+ (including its dependencies) Java JDK 1.8 Perl interpreter 5.0 or later C/C++ libraries (GCC) Optional (required for SSO) ▪ Oracle WebTier (12.2.1.4.0) Oracle Access Manager 12c Release (12.2.1.4.0) ▪ Oracle Access Manager Agent (WebGate) 12c Release (12.2.1.4.0) ▪ **Note:** Oracle Internet Directory (OID) is the supported LDAP directory for Oracle Retail products. For alternate LDAP directories, refer to Oracle WebLogic documentation set.
Deployment	On Premise, Oracle Cloud Infrastructure (OCI) or other Cloud Provider capable of providing supported Oracle Fusion Middleware
OS (On Premise Application Server only)	OS certified with Oracle Fusion Middleware 12.2.1.4.0 Options include: ▪ Oracle Linux 7 ▪ AIX 7.2+ ▪ Solaris 11.2+ ▪ HP-UX Itanium https://docs.oracle.com/en/middleware/lifecycle/12.2.1.4/sysrs/system-requirements-and-specifications.html

Note: Merchandising has been validated to run with WebLogic Clustering. Clustering for WebLogic Server 12.2.1.4.0 is managed as an Active-Active cluster accessed through a Load Balancer. Validation has been completed utilizing a RAC 19.3.0.0 Oracle Internet Directory database with the WebLogic 12.2.1.4.0 cluster. It is suggested that a Web Tier 12.2.1.4 installation be configured to reflect all application server installations if SSO will be utilized. For more information, see:

Oracle Fusion Middleware High Availability Guide, 12c Part Number E95104-03

Data Tier

The Merchandising applications require an Oracle RDBMS 19c. General requirements for an application server capable of running Merchandising include the following.

Supported on:	
Middleware	Oracle Database 19c (19.3.0+) with additional features/options: ▪ Partitioning ▪ Advanced Compression ▪ Tablespace Encryption **Note –** Installation assumes Oracle Multitenant, but limited use license covers one PDB for Merchandising if the customer does not desire additional database consolidation.
Deployment	On Premise, Oracle Cloud Infrastructure (OCI) or other Cloud Provider capable of providing supported Oracle Database 19c Detailed Oracle offerings described below
Database OS (On Premise DB only)	OS certified with Oracle Database 19c Options include: Oracle Linux 7 AIX 7.2+ Solaris 11.2+ HP-UX Itanium For more details, see https://docs.oracle.com/en/database/oracle/oracle-database/19/install-and-upgrade.html

Merchandising is supported on the following Oracle Database Offerings, as long as the required feature/options are available:

> **Note:** Some database offerings are licensing supersets, so additional requirement are included in the offer. In other cases, customers may need to license additional database components. Licensing information is current as of January 2020.

Oracle Database Offering	Offering Type	Feature/Option/Pack			
		Oracle Multitenant	**Oracle Advanced Compression**	**Partitioning**	**Tablespace Encryption**
Oracle Database Enterprise Edition (EE)	On Premise	Extra Cost Option for more than one PDB (A restricted use license for Oracle Multitenant is included with all Oracle Database offerings. Only one PDB is required for Merch install, but this limits further database consolidation)	Extra cost option	Extra cost option	Extra cost option Requires the Oracle Advanced Security option
Oracle Database Enterprise Edition on Oracle Exadata Database Machine (EE-Exa)	On Premise	Extra Cost Option for more than one PDB (A restricted use license for Oracle Multitenant is included with all Oracle Database offerings. Only one PDB is required for Merch install, but this limits further database consolidation)	Extra cost option	Extra cost option	Extra cost option Requires the Oracle Advanced Security option
Oracle Database Cloud Service Enterprise Edition - High Performance (DBCS EE-HP)	Cloud on OCI	Included	Included	Included	Included

Oracle Database Offering	Offering Type	Feature/Option/Pack			
		Oracle Multitenant	**Oracle Advanced Compression**	**Partitioning**	**Tablespace Encryption**
Oracle Database Cloud Service Enterprise Edition - Extreme Performanc e (DBCS EE-EP)	Cloud on OCI	Included	Included	Included	Included
Oracle Database Exadata Cloud Service (ExaCS)	Cloud on OCI	Included	Included	Included	Included

Additional Data Tier Recommendations

While not strictly required, the following data tier options are recommended for high availability and disaster recovery of production environments.

Oracle Database Offering	Offering Type	Feature/Option/Pack		
		Oracle Diagnostics Pack	**Oracle Real Application Clusters**	**Oracle Active Data Guard**
Oracle Database Enterprise Edition (EE)	On Premise	Extra cost option	Extra cost option	Extra cost option*
Oracle Database Enterprise Edition on Oracle Exadata Database Machine (EE-Exa)	On Premise	Extra cost option	Extra cost option	Extra cost option*
Oracle Database Cloud Service Enterprise Edition - High Performance (DBCS EE-HP)	Cloud	Included	Included	Included
Oracle Database Cloud Service Enterprise Edition - Extreme Performance (DBCS EE-EP)	Cloud	Included	Included	Included

Oracle Database Offering	Offering Type	Feature/Option/Pack		
		Oracle Diagnostics Pack	**Oracle Real Application Clusters**	**Oracle Active Data Guard**
Oracle Database Exadata Cloud Service (ExaCS)	Cloud	Included	Included	Included

Note: When using a RAC database, all JDBC connections should be configured to use THIN connections rather than OCI connections. Additional information can be found in:

Oracle Real Application Clusters Administration and Deployment Guide
19C (19.3.0.0) E95728-04

Note: As of January 2020, a license to Oracle GoldenGate includes a license to Oracle Active Data Guard. If you are licensing GoldenGate to implement Merchandising and DAS on Oracle Database Enterprise Edition (EE) or Oracle Database Enterprise Edition on Oracle Exadata Database Machine (EE-Exa), Active Data Guard may not require additional cost.

Additional Data Tier Options

Optional - Replication to Data Access Schema (DAS) via GoldenGate

Merchandising can replicate a subset of business data to a read only Data Access Schema (DAS). The data access schema includes a layer of views that provide integration contracts to the data below. This allows customers to build customer code against these versioned views without worrying about the impact of base Merchandising schema changes in later patches and updates.

Data Access Schema (DAS) is populated via simple, one-way replication from Merchandising.

Oracle Retail does not prescribe a specific replication technology. Customers can use any preferred replication technology (for example, Oracle GoldenGate) that supports basic unidirectional data and DDL replication. Merchandising does provide Oracle GoldenGate based source and target configurations. If Customers use Oracle GoldenGate, they should use

Supported on:	
GoldenGate version	**18.1**
Deployment	Microservices Architecture

https://www.oracle.com/middleware/technologies/goldengate.html

Use of DAS and replication via Goldengate are strictly optional. There is no impact to base Merchandising processes if DAS is not implemented. DAS is often useful for customers with complex custom integration to third party systems

DAS is also very useful if a retailer wants to run Merchandising and DAS and third party integrations/reporting in different locations. Customers should consider the use cases for DAS when determining the best topology.

Examples of supported topologies include:

- Merchandising On Premise/Customer Data Center to On Premise/Customer Data Center DAS
- Merchandising On Premise/Customer Data Center to DAS on OCI
- Merchandising On OCI to DAS on OCI
- Merchandising On OCI to DAS On Premise/Customer Data Center

GoldenGate itself is available as an on premise license, which can be installed either in the customer's data center or as BYOL in OCI.

As of initial publication of this document (January 2020), customers may also choose to use the GoldenGate image on OCI Marketplace if their DAS target will be in Oracle Cloud Infrastructure (OCI).

https://blogs.oracle.com/dataintegration/free-goldengate-software-on-oci-marketplace

The GoldenGate image on OCI Marketplace is an excellent option for development and test environments if the target DAS database will be in OCI.

Customers should evaluate their HA and DR requirements for production DAS during their planning process. HA and DR requirements are often a deciding factor in whether the full Maximum Availability Architecture (MAA) architecture for GoldenGate, and therefore the on premise license, is required for production.

Requesting Infrastructure Software

If you are unable to find the necessary version of the required Oracle infrastructure software (database server, application server, WebLogic, etc.) on the Oracle Software Delivery Cloud, you should file a non-technical 'Contact Us' Service Request (SR) and request access to the media. For instructions on filing a non-technical SR, see My Oracle Support Note 1071023.1 – Requesting Physical Shipment or Download URL for Software Media.

Verify Single Sign-On

If RMS is not being deployed in a Single Sign-On environment, skip this section.

If Single Sign-On is to be used, verify the Oracle Identity Management 12c version 12.2.1.4 has been installed along with the components listed in the above Application Server requirements section. Verify the HTTP Server is registered with the Oracle Access Manager (OAM) 12c as a partner application.

Supported Merchandising On-Premise Products

The Oracle Retail Merchandising System supports integration with the following on premise Oracle Retail Merchandising applications.

Product	Version
Oracle Retail Pricing	19.0.1
Oracle Retail Allocation	19.0.1
Oracle Retail Invoice Matching	19.0.1
Oracle Retail Sales Audit	19.0.1
Oracle Retail Trade Management	19.0.1

Supported Oracle Retail Merchandising On-Premise Products

The Oracle Retail Merchandising System supports integration with the following on premise Oracle Retail applications.

Product	Version
Oracle Retail Xstore Suite	19.0
Oracle Retail Store Inventory Management(SIM)	16.0.2+
Oracle Retail Warehouse Management System (RWMS)	16.0.2+
Oracle Retail Advanced Inventory Planning (AIP)	16.0.3
Oracle Retail Customer Engagement	18.0+
Oracle Retail Order Broker	19.0
Oracle Retail Order Management System	19.0

Supported Oracle Retail Integration on Premise Technologies

The Oracle Retail Merchandising System supports integration with the following on premise Oracle Retail Integration products.

Integration Technology	Version
Oracle Retail Integration Bus (RIB)	19.0.1

Supported Oracle Retail Integration Cloud Technologies

The Oracle Retail Merchandising System supports integration with the following Oracle Retail Integration Cloud Services.

Integration Technology	Version
Oracle Retail Integration Cloud Service	16.0.030+, 19.0

Supported Oracle Retail Cloud Service Products

The Oracle Retail Merchandising System supports integration with the following Oracle Retail Cloud Services.

Product	Version
Merchandise Financial Planning Cloud Service	18.0+
Oracle Retail Merchandise Financial Planning Enterprise Edition Cloud Service	18.0+
Oracle Retail Assortment and Item Planning Cloud Service	18.0+
Oracle Retail Assortment and Item Planning Enterprise Edition Cloud Service	18.0+
Oracle Retail Demand Forecasting Cloud Service	18.0+
Oracle Retail Predictive Application Server Cloud Edition	18.0+
Oracle Retail Order Broker Cloud Service	18.1+
Oracle Retail Order Management System Cloud Service	18.2+
Oracle Retail Customer Engagement Cloud Service	18.0+
Oracle Retail Insights Cloud Service Suite	16.0.2+, 18.0+
Oracle Retail Store Inventory Operations Cloud Service	18.1+

Supported Oracle Applications

The Oracle Retail Merchandising System supports integration with the following Oracle applications.

Requirement	Version
Oracle E-Business Suite Financials	12.2
Oracle PeopleSoft Financials	9.2
Oracle Cloud Financials	Most Current Cloud Service Release
Oracle Warehouse Management Cloud (WMS)	19C+

UNIX User Account Privileges to Install the Software

A UNIX user account is needed to install the software. The UNIX user that is used to install the software should have write access to the WebLogic server installation files.

For example, “oretail.”

Note: Installation steps will fail when trying to modify files under the WebLogic installation, unless the user has write access.

Verify RMS and SIM Inventory Adjustment Reason Codes

SIM and RMS must have the same inventory adjustment reason codes to work properly.

New DB Directory for Purge and Archive Setup

As part of 19.0 , archive feature of Purged RMS transaction data has been added and this requires a new DB directory to be created which points to a physical shared mount point which is accessible from both DB server and Application server (if batch is planned to be installed in Application server). Please follow the steps for creating a New DB directory:

This DB directory is used for exporting RMS transaction data archive dump file using data pump.

1. Create a new DB Directory with the directory name as “**DPUMP_PRGARCH_DIR**” by executing below command:
2. CREATE DIRECTORY DPUMP_PRGARCH_DIR AS <path of physical shared mount location>;
3. Provide necessary grants on DB directory to RMS Schema Owner by executing the below command:
4. `Grant read, write on directory DPUMP_PRGARCH_DIR to < RMSSchemaOwner:RMS19DEV>;`
5. Make sure to have the physical shared mount has restrictive permissions which is accessible from both DB server and Application server (if batch is planned to be installed in Application server).
6. `chmod –R 755 <DPUMP_PRGARCH_DIR>`

2

Database Installation Tasks

This chapter describes the tasks required for a full database installation.

Data Access Schema

Data Access Schema (DAS) exposes a subset of core RMS data to external applications via database replication. DAS allows these applications read only access RMS data as they need it. The use of a separate schema insulates core RMS processes from outside requests for information. If you choose to implement the DAS schema, refer to the DAS implementation section.

RMS Database Schema Distribution – Oracle Retail Applications Included

The RMS 19.0.1 release contains an installer package that can be used to install the database objects for the following products: RMS, ReSA, RTM, RPM, ReIM, and Allocation.

> **Note:** The Java application installers for RPM, ReIM, ReSA, and Allocation are separately downloadable under their respective products. It is only the database schema component of these applications that is included with the RMS release.

Create Staging Directory for RMS Installer

To create the staging directory for RMS installer, complete the following steps.

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

1. Login to the database server with a login that can connect to the RMS database.
2. Create a staging directory for the RMS installation software.
3. Copy the rms19installer.zip file from the RMS 19.0.1 release to the staging directory. This is referred to as STAGING_DIR when installing database software.
4. Change directories to STAGING_DIR and extract the rms19installer.zip file. This creates an rms/installer/ subdirectory under STAGING_DIR.

> **Note:** The DB Schema and Batch install can be run at the same time, with the same installer, since they are configured to run from the database server. To run both, please follow instructions from the DB Schema Full install and Batch Full install sections of the install guide. This will ensure that both DB Schema and batch have the same RETAIL_HOME. When running the installer, select the Install Schema and Install batch check boxes.

Establish a Database Partitioning Strategy

Partitioning is mandatory for specific tables. Review this entire section before proceeding with the installation.

> **Note:** Ensure the installer is used to automatically run the partition.ksh script when using the Sample Partitioning strategy. **Do not** run partion.ksh manually unless steps 1 and 2 below have been completed fully for the tables you wanted partitioned.

Sample Partitioning

The RMS 19.0.1 database schema installation runs the partitioning script (partition.ksh) automatically using a sample partitioning strategy if you do not run the partition script yourself. This is acceptable for development or demo installations and allows for a simpler installation. However, the resulting partitioning strategy is **not** suitable for production environments. It is highly recommended that the Production Partitioning section below be followed rather than allowing the installer to implement the sample strategy. The installer can be used to install the RMS database schema regardless of the choice made here.

Production Partitioning

Requirements for mandatory and optional partitioning are defined in the Microsoft Excel spreadsheet located in STAGING_DIR/rms/installer/mom/Cross_Pillar/partitioning/source/RMS_partition_definition.xlsx. Since partitioning strategies are complex, this step should be implemented by an experienced individual who has a thorough understanding of partitioning principles and the data to be partitioned.

Use the Microsoft Excel spreadsheet to determine an appropriate partitioning strategy (STAGING_DIR/rms/installer/mom/Cross_Pillar/partitioning/source/RMS_partition_definition.xlsx). The Partition Method column indicates the recommended partitioning options for each table. Refer to the information in this file to modify the DDL for partitioned tables. This can be done by manually changing the file STAGING_DIR/rms/installer/mom/Cross_Pillar/ddl/1_rms_tab_ddl.sql or by implementing the process defined below.. This file will be used later in the installation process.

> **Note:** Refer to Oracle Database Concepts 12 c Release 1 (12.1) Chapter 4 “Partitions, Views, and Other Schema Objects” for further details regarding partitioning concepts.

Beginning with hash partitions, complete the following process.

Hash partitions: o calculates the number of hash partitions and sub-partitions, enter values for the three parameters highlighted in yellow at the top of the RMS worksheet. Altering these values will update the Number of Partitions column for HASH partitioned/sub-partitioned tables. The values in these columns indicate the number of hash partitions/sub-partitions to create. Keep in mind that the number of hash partitions should be a power of 2.

Partition Factor: This value is used to adjust the number of hash partitions. It is based on the number of active items per location and transactions per location/day. If the number of items/location and/or transactions/store/day is low, the value of partition factor should be high. This will calculate fewer hash partitions. The typical factor value ranges from 2 to 4; and in some cases, it can be 10 or more.

Note: Changing the items/location and transactions/store/day fields on the worksheet does not automatically impact the factor value. They are used as a point of reference only.

Sub-Partition Factor: This value is used to adjust the number of hash sub-partitions. The partition strategy for historical information determines the value of this number. If the number of range partitions is high, the value of sub-partition factor should be high to control the number of sub-partitions. Typically, this value is 2.

Locations: The total number of active stores and warehouses.

Range partitions: Determine the purging strategy for all of the tables that are RANGE partitioned. Each partition should have a range of multiple key values. For example, if the strategy were to have data available for one year and to purge it every three months, five partitions would be created. In this case, four 3-month partitions and a max value partition to contain all data greater than the defined ranges would result. Refer to the Comments column and update the value in the Number of Partitions column. The value in this column indicates the number of range partitions to create.

Interval partitions: Interval partitioning is an extension of range partitioning wherein the database automatically creates interval partitions as data for that partition is inserted. Determine the purging strategy for all of the tables that are INTERVAL partitioned. Each partition should have a range of multiple key values. For example, if the strategy were to have data available for 90 days and to purge it every week, you can create one 7-day partition, with an interval of 7 days. In this case, one 7-day partition would be created and any data that is inserted past the initial 7-day range will have a new partition automatically create to store the new data. Refer to the Comments column and update the value in the Number of Partitions column. The value in this column indicates the number of initial range partitions to create.

List partitions: The DAILY_ITEM_FORECAST, ITEM_FORECAST, DEAL_ITEMLOC_DCS, DEAL_ITEMLOC_DIV_GRP, DEAL_ITEMLOC_ITEM, AND DEAL_ITEMLOC_PARENT_DIFF must be LIST partitioned. If number of partition keys is relatively static, change the value in the Partition Method column to LIST where allowed. This method will ensure that each partition key has a separate partition and that none are empty. The Number of Partitions column will be automatically updated with the proper number of locations in the event the partition method is changed. The value in this column indicates the number of list partitions to create.

Step 1: Modify partition_attributes.cfg

Modify STAGING_DIR/rms/installer/mom/Cross_Pillar/partitioning/source/partition_attributes.cfg based on the partitioning strategy defined in RMS_partition_definition.xlsx. Changes to this file should be made only as indicated.

partition_attributes.cfg file: (file is comma-delimited)

Sample Entry:

```
ITEM_LOC_HIST,EOW_DATE,RANGE,item_loc_hist.eow_date.date,64,LOC,HASH,item_loc_hist.loc.number,64,RETAIL_DATA
```

- Field 1: `Table Name` - Do not modify
- Field 2: `Partition Key` - Do not modify
- Field 3: `Partition Method` - Modify based on value in Partition Method column in RMS_partition_definition.xlsx - Valid values are RANGE, LIST, HASH, or INTERVAL (case sensitive)

- Field 4: `Partition Data Definition Filename` - Do not modify - This field is ignored if Partition Method is not RANGE or LIST or INTERVAL
- Field 5: `Partition Hash Count` – Modify based on value in Hash Partitions Calculated column in RMS_partition_definition.xlsx. In case of INTERVAL partition, this field will contain a partition interval value (e.g. 7 days in one partition). This field is ignored if Partition Method is not HASH or INTERVAL.
- Field 6: `Interval Unit` – Used and required for INTERVAL partition only. Expected values are 'DAY' or 'MONTH'.
- Field 7: `Sub-Partition Key` - Do not modify
- Field 8: `Sub-Partition Method` - Modify based on value in Sub-partition Method column in RMS_partition_definition.xlsx - Valid values are LIST or HASH (case sensitive)
- Field 9: `Sub-Partition Data Definition Filename` - Do not modify - This field is ignored if Sub-Partition Method is not RANGE, LIST, or INTERVAL
- Field 10: `Sub-Partition Hash Count` - Modify based on value in Hash Sub-partitions Calculated column in RMS_partition_definition.xls. This field is ignored if Sub-Partition Method is HASH
- Field 11: `Tablespace Name` - Optional. Default is RETAIL_DATA

Step 2: Modify Data Definition Files

Tables partitioned or sub-partitioned by RANGE, INTERVAL or LIST have a corresponding data definition file in the STAGING_DIR/rms/installer/mom/Cross_Pillar/partitioning/source/data_def" directory and should not be removed or renamed. These files are used to define the data boundaries for each partition. Values must be entered in each file based on the data type of the Partition Key column in RMS_partition_definition.xls. Refer to the Comments column in this file for additional information. The value in the Number of Partitions column indicates the number of entries to place in the data definition file. For INTERVAL partitioning, a single entry in the data definition file will be sufficient.

The format of a data definition file name is <table name>.<partition key column>.<partition key data type> (for example, item_loc_hist.eow_date.date). When placing data into these files, enter one data partition value per line.

When entering varchar2 values in a data definition file, do not use quotation marks. When defining date values, use the DDMMYYYY format.

sampletable.action_date.date:

```
01012004
01012005
```

sampletable.state.varchar2:

```
Minnesota
Iowa
```

sampletable.location.number:

```
1000
2000
```

When using RANGE partitioning, the data definition files will use the value less than concept. For example, in sampletable.action_date.date above, the first partition will contain all data less than 01012004. The second partition will contain all data greater than or equal to 01012004 and less than 01012005. A third MAXVALUE partition will automatically be created for all data greater than or equal to 01012005.

When using INTERVAL partitioning, the data definition file can be populated with one date entry to create the first range. Future partitions will be added automatically when

data is inserted into the table for dates greater than the defined range and corresponding interval.

When using LIST partitioning, the data definition files will use the value equal to concept. For example, in sampletable.state.varchar2 above, the first partition will contain all data equal to Minnesota. The second partition will contain all data equal to Iowa.

Step 3: Generate DDL for Tables – Run partition.ksh

1. Copy STAGING_DIR/rms/installer/mom/Cross_Pillar/ddl/1_rms_tab_ddl.sql to STAGING_DIR/rms/installer/mom/Cross_Pillar/partitioning/rms.tab.
2. Execute STAGING_DIR/rms/installer/mom/Cross_Pillar/partitioning/source/partition.ksh at the UNIX command prompt. This script reads configuration information from the partition_attributes.cfg file and generates the partitioned DDL file STAGING_DIR/rms/installer/mom/Cross_Pillar/partitioning/rms_part.tab. This file is used later during the installation process.

   Sample output from partition.ksh:

```
STAGING_DIR/installer/mom/Cross_Pillar/partitioning/source >
./partition.ksh
#########################################################################
###
# partition.ksh:
# This script will read the partition_attributes.cfg file and any
referenced
# data definition files and generate partitioned DDL.
#########################################################################
###
# The non-partitioned DDL file is ../rms.tab.
# The partitioned DDL file that will be generated is ../rms_part.tab.
#########################################################################
###
Checking partition_attributes.cfg for errors
Generating Partitioned DDL for DAILY_DATA
Generating Partitioned DDL for DAILY_ITEM_FORECAST
Generating Partitioned DDL for DAILY_SALES_DISCOUNT
…
partition.ksh has generated the DDL for partitioned tables in the
../rms_part.tab file.
Completed successfully
```

Create the RMS Database

It is assumed that Oracle Database 19c, with appropriate patches, has already been installed. If not, refer to Data Tier Requirements in Chapter 1 before proceeding**.** Additionally, STAGING_DIR in this section refers to the directory created in Create Staging Directory for RMS Database Schema Files in Part I, Chapter 1.

Review the Establish Database Partitioning Strategy section before continuing.

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. Refer to appendices A, B, C in this document.

If a database instance has not been created, create one using database creation templates via DBCA in silent mode.

Create the Database Instance Using Oracle Generic Template

Prerequisites:

- 19.3.0+ binary must have already been installed along with the appropriate one-off patches. Refer to the Database Server Preinstallation section for all the required one-off patchesBackground
- Oracle Retail no longer deliver custom database template files. Instead, databases can be created using the generic Oracle delivered template in the directory$ORACLE_HOME/assistant/dbca/template.

```
$ORACLE_HOME/assistantsdbca/templates>
--> ls -l General_Purpose.dbc
-rw-r----- 1 rgbuora dba 4768 Apr 17  2019 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=<Location for Oracle Home >
   export ORACLE_BASE=<Location for Oracle Base>
   export PATH=$ORACLE_HOME/bin:$PATH
   ```

   .cd into $ORACLE_HOME/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
   <Datafile Location> -characterSet AL32UTF8 -nationalCharacterSet
   AL16UTF16 -redoLogFileSize 100 -initParams nls_date_format=DD-MON-
   RR,nls_language=AMERICAN,nls_calendar=GREGORIAN,fast_start_mttr_ta
   rget=900
   ```

   The above will create a container database using all the default parameters set by dbca. Please replace the pfile by taking a copy from Appendix: Oracle 19cR1 Database Parameter File 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 command 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=('<Old
   Locationof PDB Datafiles>','<New Location for PDB Datafiles>');

   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. 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 to create the tablespaces.

Create Required RMS Tablespaces

Release 19.0.1 uses the tablespaces RETAIL_DATA, RETAIL_INDEX. ENCRYPTED_RETAIL_DATA, ENCRYPTED_RETAIL_INDEX and FLASHBACK_DATA.

The ENCRYPTED_RETAIL_DATA and ENCRYPTED_RETAIL_INDEX tablespaces hold data, which may include Personally Identifiable Information data (PII Data). If you hold the Advanced Security Option license, you can choose to create these two tablespaces with TDE tablespace encryption to protect the PII data at rest. If you do not hold an Advanced Security Option license, you can create the tablespaces as normal tablespaces. The tablespace names must always be ENCRYPTED_RETAIL_DATA and ENCRYPTED_RETAIL_INDEX regardless of whether TDE encryption is used, because the table and index creation scripts look for these specific names.

1. Modify STAGING_DIR/rms/installer/create_db/create_rms_tablespaces.sql. The table below shows the default initial sizes.
2. Once this script has been modified, execute it in SQL*Plus as sys.
   - For Example: SQL> @create_rms_tablespaces.sql
3. Review create_rms_tablespaces.log for errors and correct as needed.
4. If you do not wish to use TDE tablespace encryption follow below steps else for TDE encryption skip to step 5.
   a. Modify STAGING_DIR/rms/installer/create_db/create_encrypted_ tablespaces_no_TDE.sql.
   b. Run the script using SQL*Plus as sys.
   c. Review Create_encrypted_retail_tablespaces_no_TDE.log for errors and correct as needed.
5. If you hold an Advanced Security Option license and wish to use TDE tablespace encryption
   a. Modify STAGING_DIR/rms/installer/create_db/create_encrypted_ tablespaces_TDE.sql.
   b. Run the script using SQL*Plus as sys.
   c. Review Create_encrypted_retail_tablespaces_TDE.log for errors and correct as needed.
   d. Refer to Appendix: Tablespace Creation for details about how to create tablespaces in an encrypted format.

> **Note:** The partitioning strategy determines the size of RMS tablespaces. Be aware that increasing the number of partitions may necessitate an increase in the size of the required tablespaces. It is important to be accurate when sizing tablespaces prior to the installation of RMS. Failure to do so results in “insufficient space” errors, which require a complete re-install of RMS.

The standard tablespace scripts contain the DDL for creating the required tablespaces, which can extend up to the following sizes:

TABLESPACE_NAME	Size
ENCRYPTED_RETAIL_INDEX	12G
ENCRYPTED_RETAIL_DATA	10G

RETAIL_INDEX	10G
RETAIL_DATA	8G
LOB_DATA	2G
USERS	2G
FLASHBACK_DATA	2G

These sizes are sufficient if the initial values in the STAGING_DIR/rms/installer/mom/Cross_Pillar/partitioning/source/RMS_partition_definition.xls spreadsheet are used without modifications. Although using the initial values is not recommended for a production environment, it is possible to use them for creating a small test environment. For additional assistance with production database sizing, please work with your implementation partner or contact Oracle Retail Consulting.

Create the Schema Owner for RMS

Create an Oracle schema that will own the RMS application.

> **Note:** The RMS schema owner must be created prior to running the RMS database schema installation. The installer will validate this user before proceeding with installation.

1. Change directories to STAGING_DIR/rms/installer/create_db.
2. The create_user script relies on empty roles, being created. Log into sqlplus sys as sysdba and run the following commands to create the roles.

   ```
   SQL> @create_roles.sql
   ```

3. Enter the following command to create the schema owner:

   ```
   SQL> @create_user.sql
   ```

   The following prompts will occur:

   - Schema Owner – the Oracle user that will own all RMS objects. Referred to in this install guide as RMS19DEV
   - Password – the password for RMS19DEV
   - Temp Tablespace – the temporary tablespace for RMS19DEV
4. Check the log file create_<Schema Owner>.lst for any errors.

Create the Database User for BDI RMS INT SCHEMA

1. Enter the following command to create the BDI RMS Integration Schema:

   ```
   SQL>@create_bdi_int_user.sql
   ```

   The following prompts will occur:

   - Please enter the BDI INT schema: The BDI RMS Integration Schema is referred to in this install guide as BDI_RMS_INT_SCHEMA
   - Please enter the password for the user: the password for BDI_RMS_INT_SCHEMA user
   - Please enter the temporary tablespace for the user: the temporary tablespace for BDI_RMS_INT_SCHEMA
2. Check the log file create_BDI_RMS_INT_SCHEMA.lst for any errors.

Create the Database User for Allocation (Optional)

1. To create the database user for where Allocation temporary tables will be stored, complete the following steps.
2. Change directories to STAGING_DIR/rms/installer/create_db
3. Log into sqlplus as sysdba and run the following command:

   ```
   SQL> @create_user_generic.sql
   ```

   The following prompts will occur:

   - Schema Name – The name of the Allocation database user. Referred to in this install guide as ALLOC19DEV
   - Password – the password for ALLOC19DEV
   - Temp Tablespace – the temporary tablespace for ALLOC19DEV

Create the Database User for Demo Data (Optional)

The RMS demo data user is only required if you will be seeding RMS during installation with optional demo data. To create the demo data user, complete the following steps.

1. Change directories to STAGING_DIR/rms/installer/create_db
2. Log into sqlplus as sysdba and run the following command:

   ```
   SQL>@create_user_generic.sql
   ```

   The following prompts will occur:

   - Schema Name – The name of the Demo database user. Referred to in this install guide as RMS19DEMO
   - Password – the password for RMS19DEMO
   - Temp Tablespace – the temporary tablespace for RMS19DEMO

Run the RMS Database Schema Installation

Note: See Appendix: RMS Database Schema Installer Screens for details on the RMS Database Schema installation screens and fields in the installer.

Note: The Schema and Batch installation to be done at the same time and recommendation is to use the same path for RETAIL_HOME. See next section for batch installation steps

Note: If dynamic hierarchy is being used, as a pre-installation task, update the script <STAGING_DIR>/rms/installer/mom/Cross_Pillar/control_scripts/source/dynamic_hier_token_map.sql and its language files <STAGING_DIR>/rms/installer/mom/Cross_Pillar/languages/xx/dynamic_hier_token_map_xx.sql to provide the client name value. Refer to Merch Implementation guide for details on dynamic hierarchy.

1. Change directories to STAGING_DIR/rms/installer.

2. Source the oraenv script to set up the Oracle environment variables (ORACLE_HOME, ORACLE_SID, PATH, and so on).

   Example: prompt$. oraenv

   ORACLE_SID = [] ? mydb

   prompt$

3. Verify the ORACLE_HOME and ORACLE_SID variables after running this script.

   Example: prompt$ echo $ORACLE_HOME

   /u00/oracle/product/mydbversion

   prompt$ echo $ORACLE_SID

   mydb

4. Set and export the following environment variables. These variables are needed in addition to the environment variables set by the oraenv script above.

Variable	Description	Example
JAVA_HOME	Java home needed to run the GUI. Java 1.8 is required	JAVA_HOME=/usr/java/jdk1.8.64bit export JAVA_HOME
NLS_LANG	Locale setting for Oracle database client	NLS_LANG=AMERICAN_AMERICA.AL32UTF8 export NLS_LANG
DISPLAY	Address and port of X server on desktop system of user running install. Optional for dbschema installation	DISPLAY=<IP address>:0.0 export DISPLAY

   Note: Unset NLS_DATE_FORMAT before running the installer. If NLS_DATE_FORMAT is set as YYYY-MM-DD:HH24:MI:SS, the installer will fail.

5. 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.
6. Run the install.sh script to start the installer.

   Note: Below are the usage details for install.sh. The typical usage for GUI mode is no arguments.

   ./install.sh [text | silent]

7. Verify that the installer reports “SUCCESS” for the Database Preinstall Check. If it reports “FAILED,” check for errors in the output under the “Checking environment for Database installation” section, and verify that your environment variables are set properly.
8. For the initial RMS database installation, select the Full option on the Full Install or Patch screen.
9. Check the Install DB Objects checkbox and continue with installer. If the Batch and Database objects reside on the same RETAIL_HOME then click on the Batch also.

10. The RMS Installer provides the option of installing the Invoice Matching (ReIM) and Allocation database objects in addition to the RMS objects.
11. After the installer is complete, you can check its log file: rms-install.<timestamp>.log.
12. 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 installer encounters any errors, it halts execution immediately and prints to the screen which SQL script it was running when the error occurred. Please view the log files in $RETAIL_HOME/orpatch/logs. Additional error information for invalid objects can be found in $RETAIL_HOME/orpatch/logs/detail_logs/dbsql_{schema}/invalids. The {schema} refers to rms, rmsbdiint, raf, reim, rpm, alloc.

See Appendix: Common Installation Errors in this document for a list of common installation errors.

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.

In case if you decided to drop the schemas and start the install from scratch, then make sure the RETAIL_HOME is also removed.

Set Up Additional RMS Users

1. Few sample scripts to create application roles and database user are available in the following location: <STAGING_DIR>/rms/installer/mom/Cross_Pillar/utility_files. Review the scripts as per your company's security regulation to restrict the access based on user responsibility.
   - create_ORMS_business_user_role.sql can be referred to create a new DB role having access to objects owned by the schema owner.
   - create_ORMS_business_user.sql can be referred to create a new DB user and granted the role created by create_ORMS_business_user_role.sql script.
   - create_roles.sql creates sample roles.
   - create_user_generic.sql is a generic script to create DB user having extensive access and are assigned the roles created using create_roles.sql

     Note: Evaluate the use of multiple roles and assign appropriately to users, based on user responsibilities.
2. After users are set up, create synonyms to the owner schema for all tables, views, sequences, functions, procedures, packages and types to which the user has access.
3. For information, see "Appendix: Creating User Synonyms."

Note: create_ORMS_business_user_role.sql and create_ORMS_business_user.sql can be referred to create RMS user with restricted privileges. Please refer to the *Oracle Retail Merchandising Operations Management Security Guide* for details.

Note: Users created with these scripts will be granted with selective privileges on each database object. A new object addition/patch that contains new objects will need attention from customer database administrator. Either grant selective privileges to the individual database objects or re-create the role with create_ORMS_business_user_role.sql, which will grant privileges to new objects for the users.

PRODUCT_VERS_CONFIG_OPTIONS

1. Run the ad-hoc script as RMS Schema Owner **STAGING_DIR/rms/installer /mom/Cross_Pillar/install_scripts/source/sys_update_prod_vers.sq**l to update the PRODUCT_VERS_CONFIG_OPTIONS table. It updates the patch version of the other MOM products installed if any. It accepts seven values as user input:
   - first input as Allocation version
   - second input as RWMS version
   - third input as REIM version
   - fourth input as SIM version
   - fifth input as AIP version
   - sixth input as RPM version
   - seventh input as ReSA version

Batch Security Setup

If RMS was installed without DEMO Data, additional data setup is required to be able to run batch programs. SEC_USER, SEC_GROUP, and SEC_USER_GROUP need to be populated using the below scripts.

1. Log on to sqlplus as the RMS schema owner.
2. Insert into SEC_GROUP and entry for Super Group:
   ```
   @<STAGING_DIR>/rms/installer/create_db/superGroup.sql
   ```
3. Insert the following row into SEC_USER and SEC_USER_GROUP for the schema owner:
   ```
   @<STAGING_DIR>/rms/installer/create_db/superUser.sql
   ```

3

Batch Installation Tasks

This section includes steps for batch installation.

Create Staging Directory for RMS Installer

To create the staging directory for RMS installer, complete the following steps.

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

1. Log into the database server as a user that can connect to the RMS database.
2. Create a staging directory for the RMS installation software.
3. Copy the rms19installer.zip file from the RMS 19.0.1 release to the staging directory. This is referred to as STAGING_DIR when installing batch software.
4. Change directories to STAGING_DIR and extract the rms19installer.zip file. This creates an rms/installer/ subdirectory under STAGING_DIR.

> **Note:** Refer to the following My Oracle Support note if the operating system platform is Linux:
>
> Doc ID 102288.1 – Precompiling Sample Pro*C Programs on Linux Fails with PCC-02015 and PCC-02201 (Doc ID 102288.1

To fix the issue – Example:

1. Compare the paths in the installer pcscfg.cfg to the paths for pcscfg.cfg that the Linux OS has. The paths in the installer pcscfg.cfg are that may be invalid are
   - /usr/lib/gcc/x86_64-redhat-linux/4.1.2/include
   - /usr/lib/gcc/x86_64-redhat-linux/4.4.6/include
2. Find the pcscfg.cfg file in the correct path in the Linux OS. The path is
   - /usr/lib/gcc/x86_64-redhat-linux/4.4.4
   - /usr/lib/gcc/x86_64-redhat-linux/4.4.7 -> 4.4.4
3. Back up the pcscfg.cfg file.
4. Edit the pcscfg.cfg file.
5. Change the following in the pcscfg.cfg file:

   /usr/lib/gcc/x86_64-redhat-linux/4.4.6/include to /usr/lib/gcc/x86_64-redhat-linux/4.4.7/include
6. Run the batch installer.

Run the RMS Installer

To run the RMS Installer, complete the following steps:

Note: If Batch is installed along with Database installation then this step can be skipped.

Note: See Appendix: RMS Batch Installation Screens for details about the RMS Batch installation screens and fields in the installer.

1. Change directories to STAGING_DIR/rms/installer.
2. Source the oraenv script to set up the Oracle environment variables (ORACLE_HOME, ORACLE_SID, PATH, and so on).

   Example: prompt$. oraenv

   ORACLE_SID = [] ? mydb

   prompt$

3. Verify the ORACLE_HOME and ORACLE_SID variables after running this script.

   Example: prompt$ echo $ORACLE_HOME

   /u00/oracle/product/mydbversion

   prompt$ echo $ORACLE_SID

   mydb

4. Verify that the following executables are available from PATH: make, makedepend, cc, ar.

   Example: Here are some locations where makedepend is commonly found:

   Linux: /usr/X11R6/bin

5. Set and export the following environment variables. These variables are needed in addition to the environment variables set by the oraenv script above.

Variable	Description	Example
JAVA_HOME	Java home needed to run the GUI. Java 1.8is required	JAVA_HOME=/usr/java/jdk1.864bit
NLS_LANG	Locale setting for Oracle database client	NLS_LANG=AMERICAN_AMERICA .AL32UTF8 export NLS_LANG
DISPLAY	Address and port of X server on desktop system of user running install. Optional for batch installation	DISPLAY=<IP address>:0 export DISPLAY

6. 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.
7. Run the install.sh script to start the installer.

Note: Below are the usage details for install.sh. The typical usage for GUI mode is no arguments.

./install.sh [text | silent]

8. Verify that the installer reports “SUCCESS” for the Batch preinstall check. If it reports “FAILED,” check for errors in the output under the “Checking environment for Batch installation” section, and verify that your environment variables are set properly.
9. Check the Install Batch checkbox and continue with installer.
10. Depending on system resources, a typical RMS batch installation takes around 30 minutes. After the installer is complete, you can check its log file in the “logs” directory: rms-install.<timestamp>.log.
11. 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 Batch Installation

The RMS batch installation is a full install that starts from the beginning each time it is run. If you encounter errors in your environment, after resolving the issue you can safely run the batch installation again to attempt another installation. Log files for the batch compilation can be found in the $RETAIL_HOME/orpatch/logs/rmsbatch/{lib,proc}

RETL

The RMS batch installation installs the RETL files under RETAIL_HOME.

See Appendix: RMS RETL Instructions in this document for more information about RETL.

4

Application Server Installation Tasks

Before proceeding, you must install Oracle WebLogic Server 12.2.1.4.0 with ADF and any patches listed in the Chapter 1 of this document. The Oracle Retail Merchandising System is deployed to a WebLogic Managed server within the WebLogic installation. It is assumed Oracle Database has already been configured and loaded with the appropriate schemas for your installation.

Installing a separate domain is mandated. It can be called “RMSDomain” (or something similar) and will be used to install the managed servers. The ADF libraries should be extended to this domain and the Enterprise Manager application should be deployed.

Note: If this domain is to be setup in a secure mode. Please set up weblogic as SSL and refer to ORACLE Retail Merchandising Security Guide for details on all items to change to be in secure mode. This would best be done before domain and application install. The domain example below is for unsecured setup.

Middleware Infrastructure and WebLogic Server12c (12.2.1.4.0) Installation

Create a directory to install the WebLogic (this will be the ORACLE_HOME):

Example: mkdir -p /u00/webadmin/products/wls_retail

1. Set the ORACLE_HOME, JAVA_HOME and DOMAIN_HOME environment variables:
   - ORACLE_HOME should point to your WebLogic installation.
   - JAVA_HOME should point to the Java JDK 1.8+. This is typically the same JDK which is being used by the WebLogic domain where application is getting installed.

   Example:

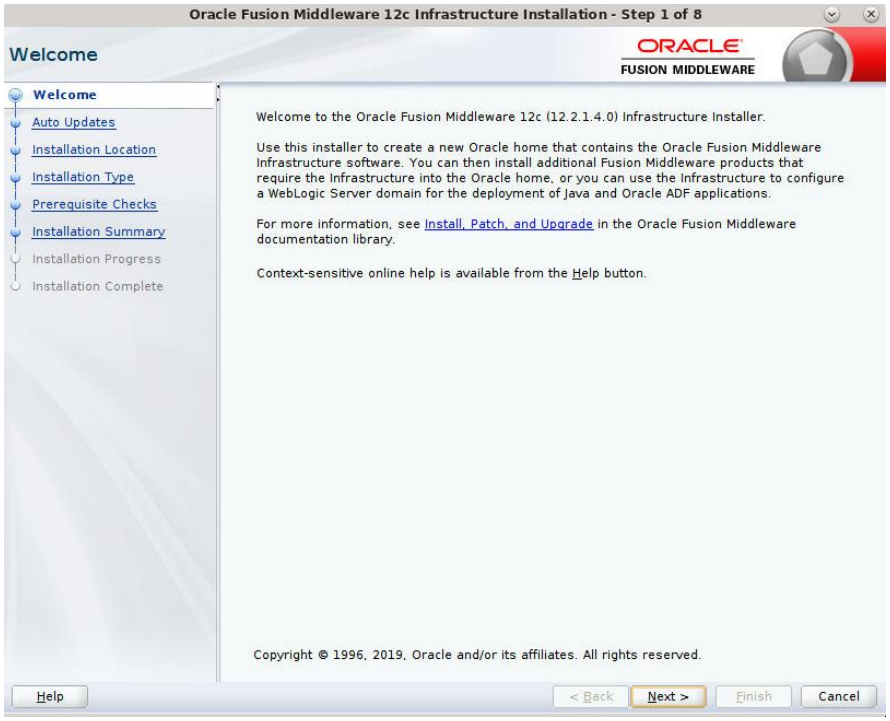
   ```
   $export ORACLE_HOME=/u00/webadmin/products/wls_retail
   $export JAVA_HOME=/u00/webadmin/products/jdk_java
   (This should point to the Java which is installed on your server)
   $export PATH=$JAVA_HOME/bin:$PATH
   ```

   Going forward we will use the above references for further installations.
2. Go to location where the WebLogic jar is downloaded and run the installer using the following command:

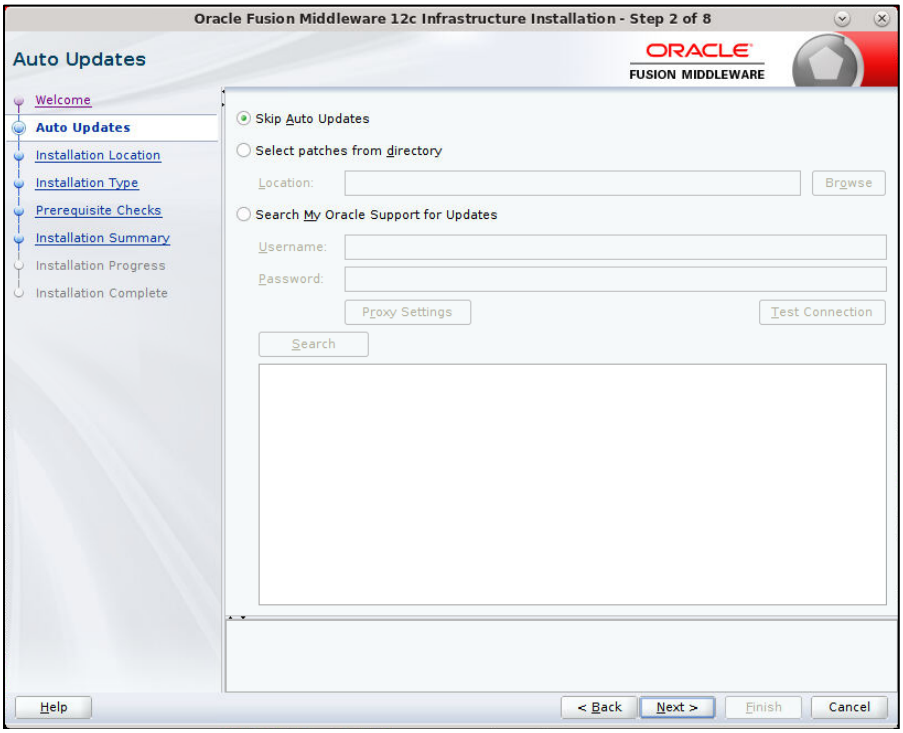
   ```
   java –jar ./fmw_12.2.1.4.0_infrastructure.jar
   ```

3. Click **Next**.

4. Welcome screen appears. Click **Next**.



5. Click Next

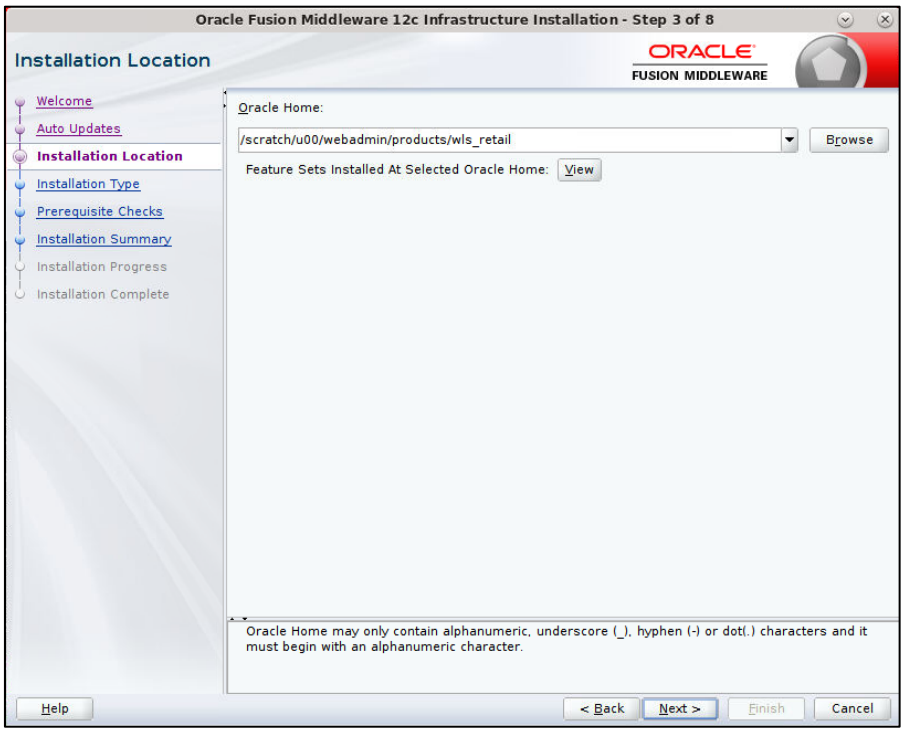


6. Enter the following and click **Next**.

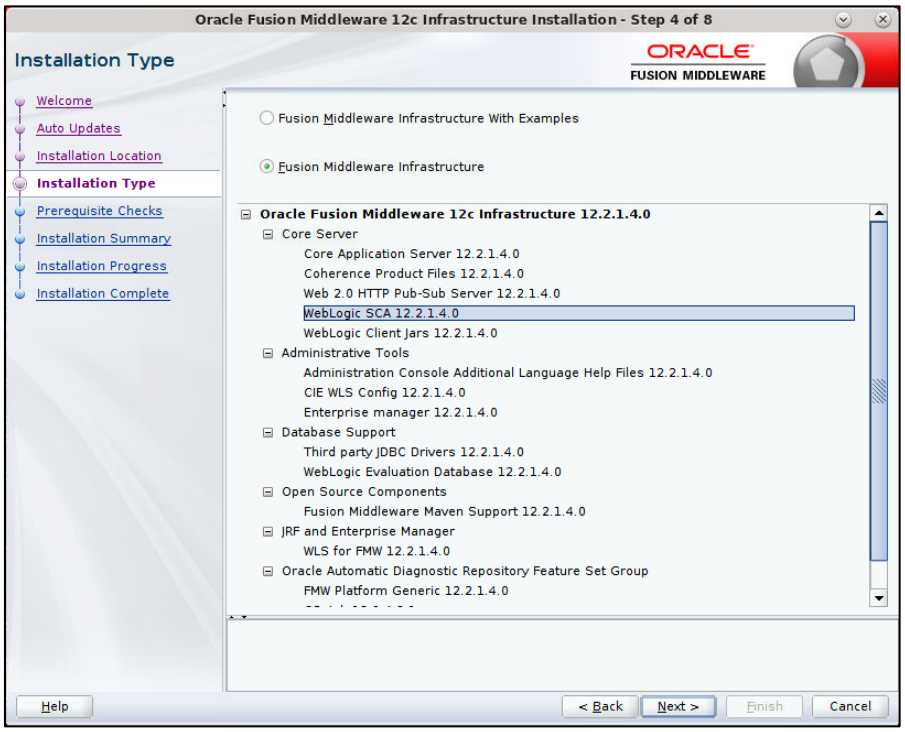
   `Oracle home =<Path to the ORACLE_HOME>`

   Example:

   `/u00/webadmin/products/wls_retail`

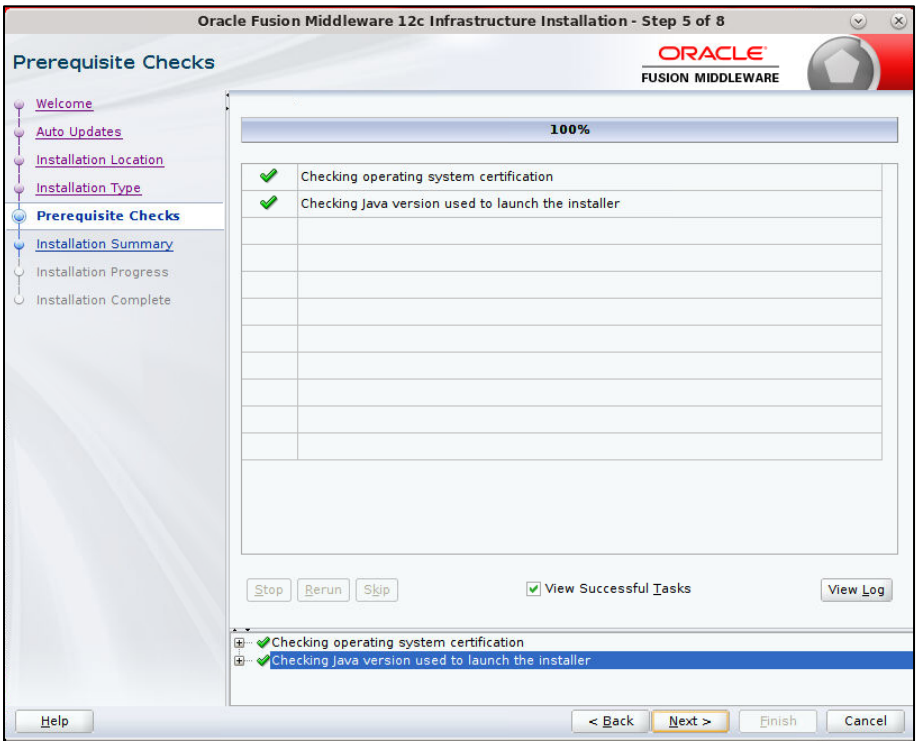


7. Select install type 'Fusion Middleware Infrastructure'. Click **Next**.

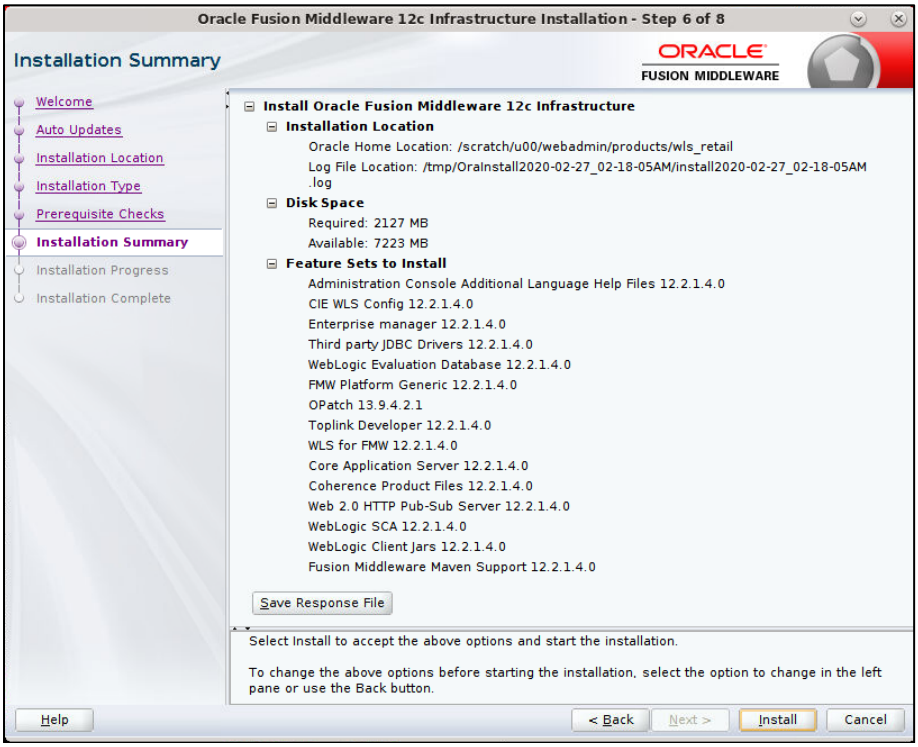


This screen will verify that the system meets the minimum necessary requirements.

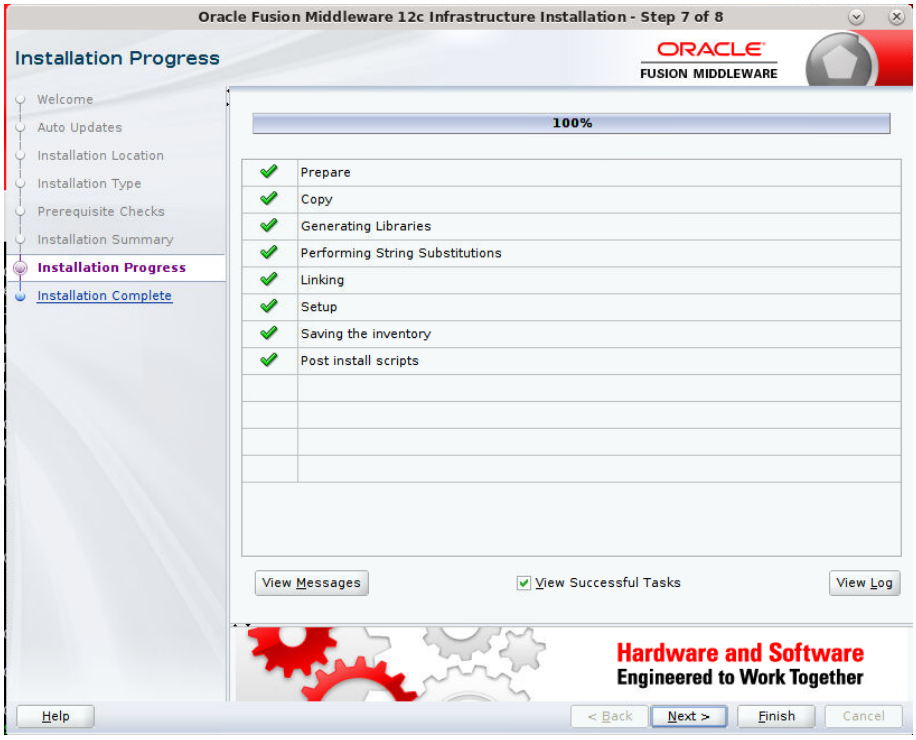
8. Click **Next**.



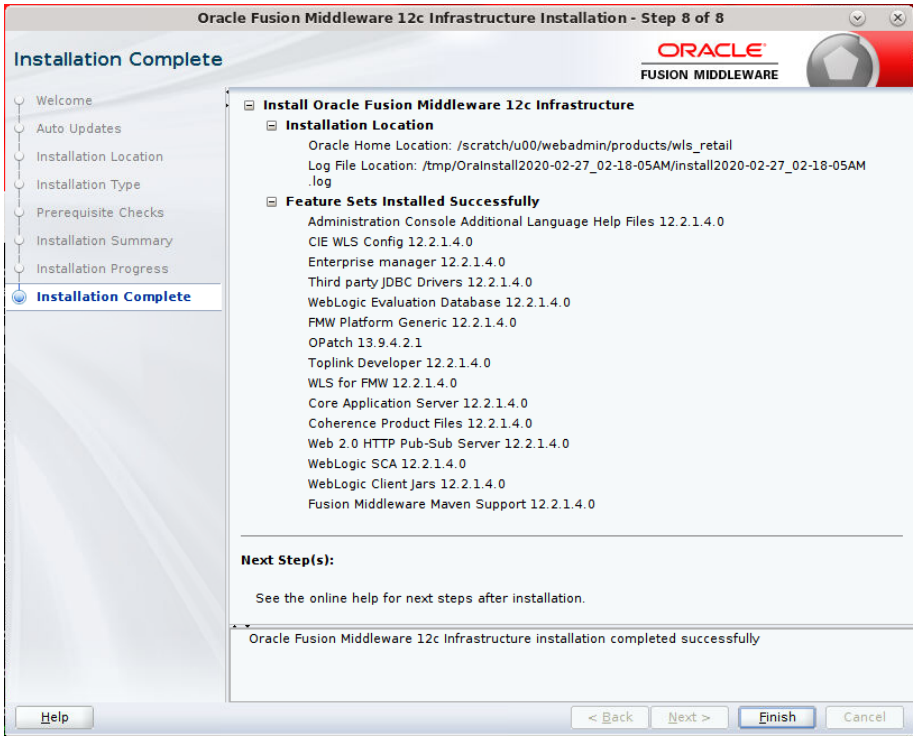
9. Click **Install**.



10. Click Next



11. Click Finish



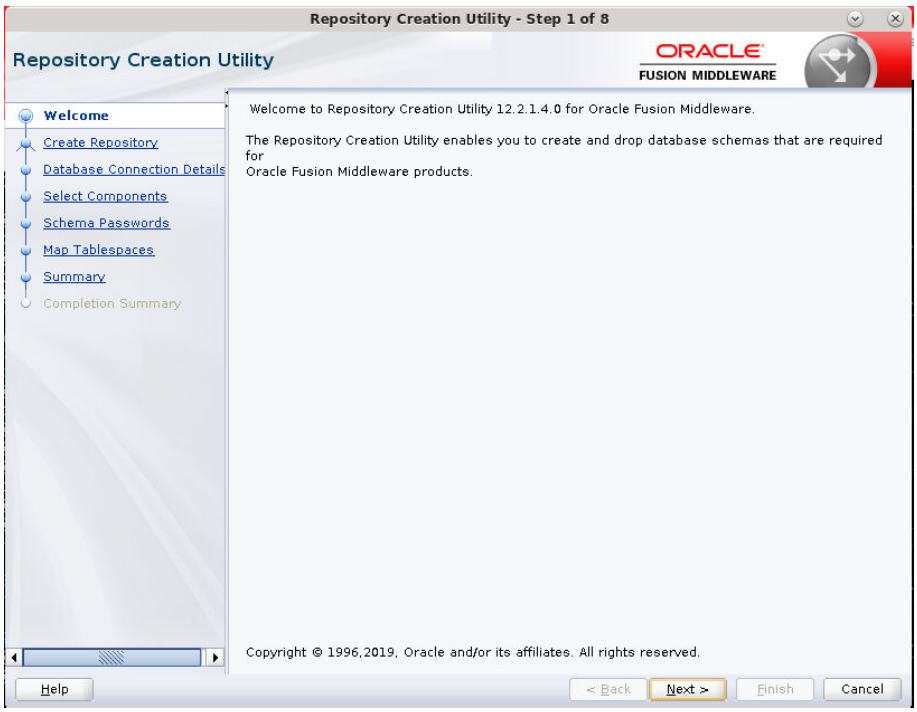
Install RCU Database Schemas

The RCU database schemas are required for the installation of configuration of domain and retail application.

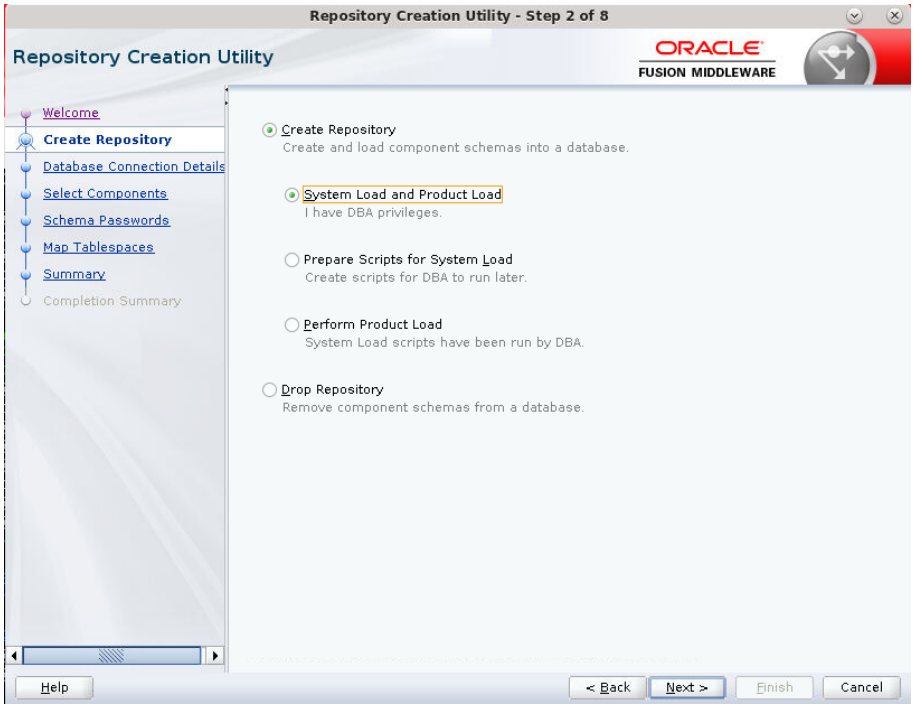
> **Note:** Need user which have sys admin privileges to install the RCU database schemas.

The following steps are provided for the creation of the database schemas:

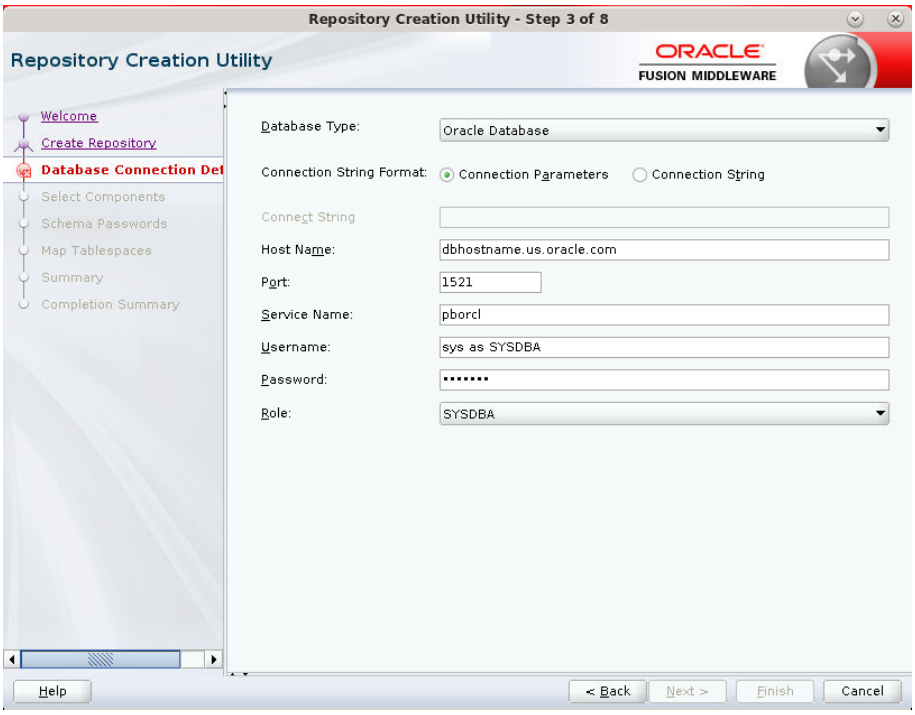
1. Navigate to the directory into which RCU is installed. For example:
   ```
   <ORACLE_HOME>/oracle_common/bin/
   Run "./rcu"
   ```
2. Click **Next**.



3. Select Create Repository and System Load and Product Load. Click **Next**.

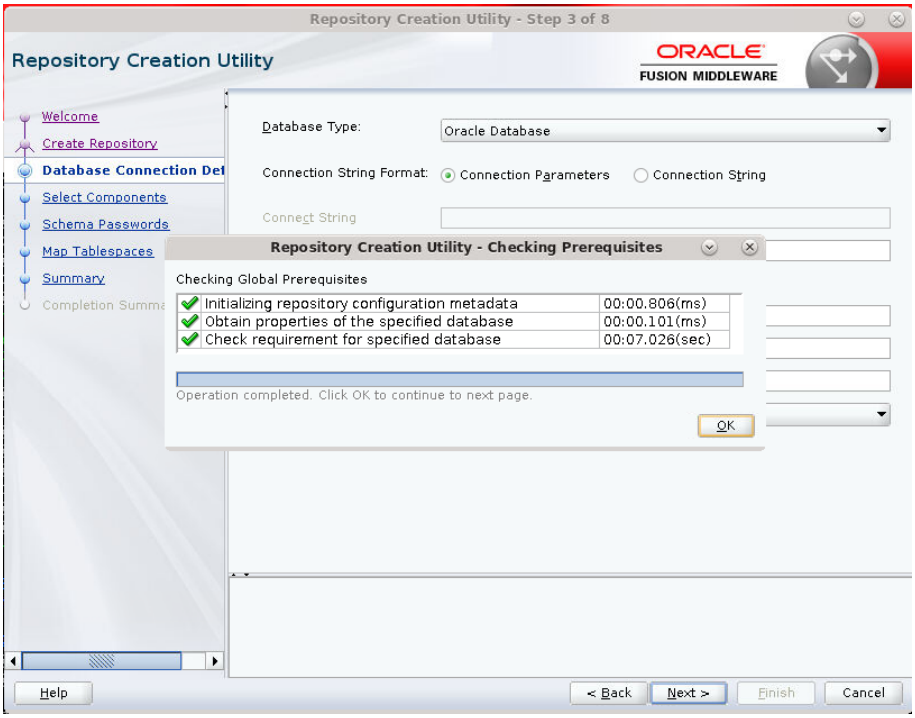


4. Enter database connection details:
   - Database Type: Oracle Database
   - Host Name: dbhostname.us.oracle.com
   - Port: 1521
   - Service Name: dbservicename
   - Username: sys
   - Password: <syspassword>
   - Role: SYSDBA



5. Click **Next**. The Installer checks prerequisites.

6. When the prerequisite checks are complete, click **OK**. Click **Next**.



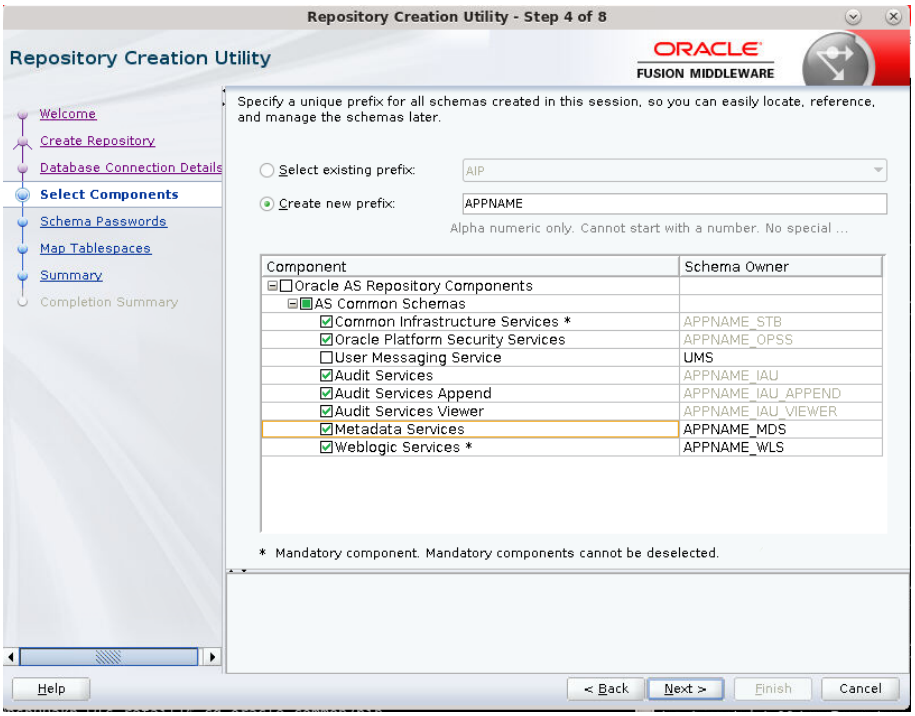
7. Click the **Create a new prefix** option, the prefix name for your schemas should be unique to your application environment.

   Example: ReIM, ALLOC, ReSA, etc

8. Select the components to create:
   - Meta Data Services
   - Oracle Platform Security Services

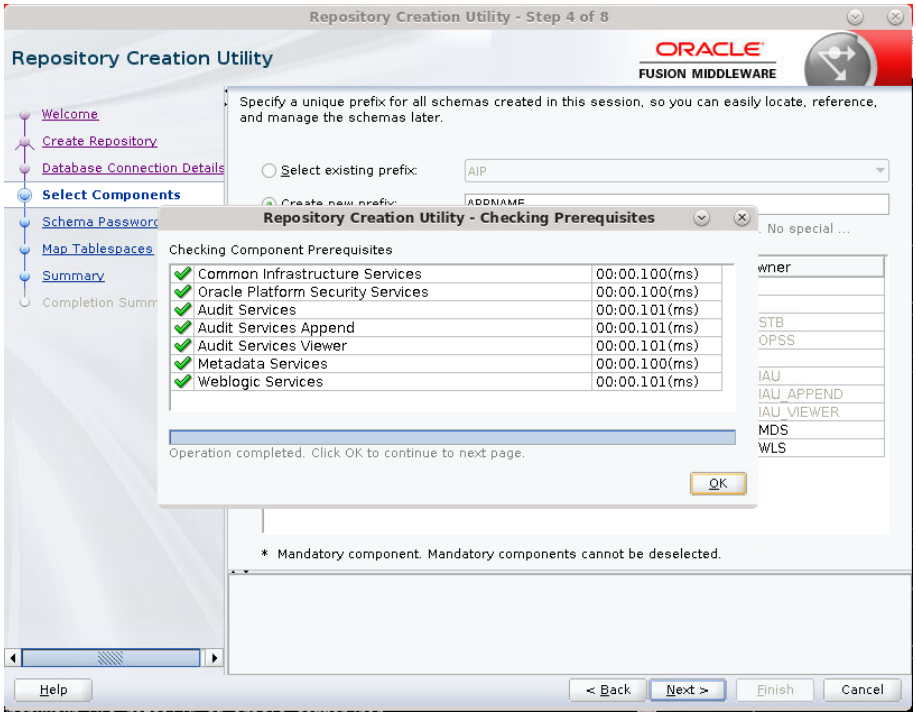
> **Note:** Once OPSS schema is selected, the following dependent schemas will get selected automatically.
>
> Audit Services
>
> Audit Services Append
>
> Audit Services Viewer

> **Note:** STB schema will be already selected as part of the Common Infrastructure component.



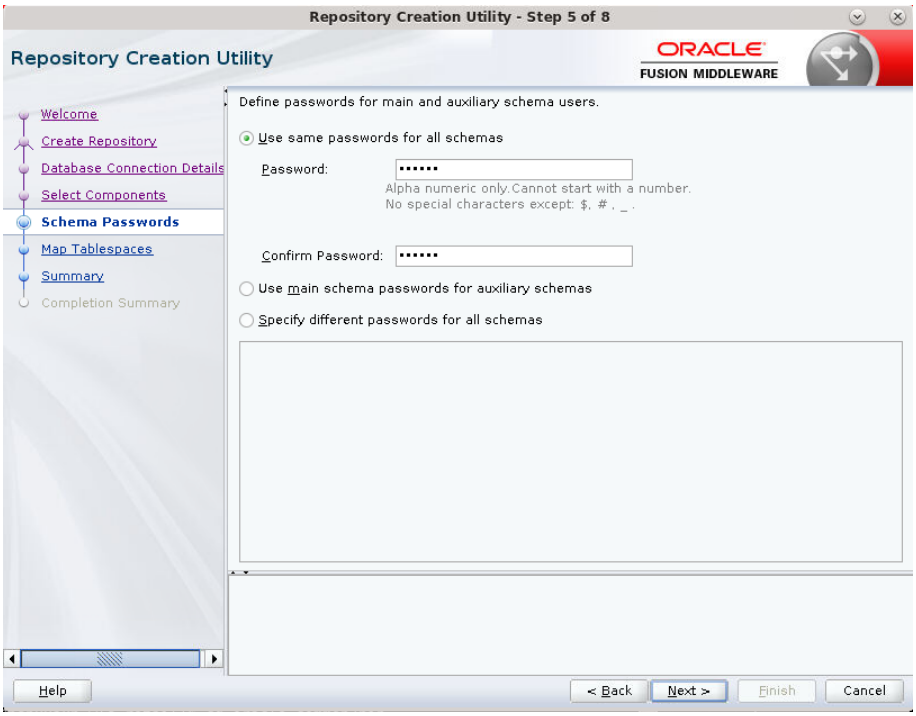
9. Click **Next**.

10. Click OK when the repository Creation Utility is done checking prerequisites.



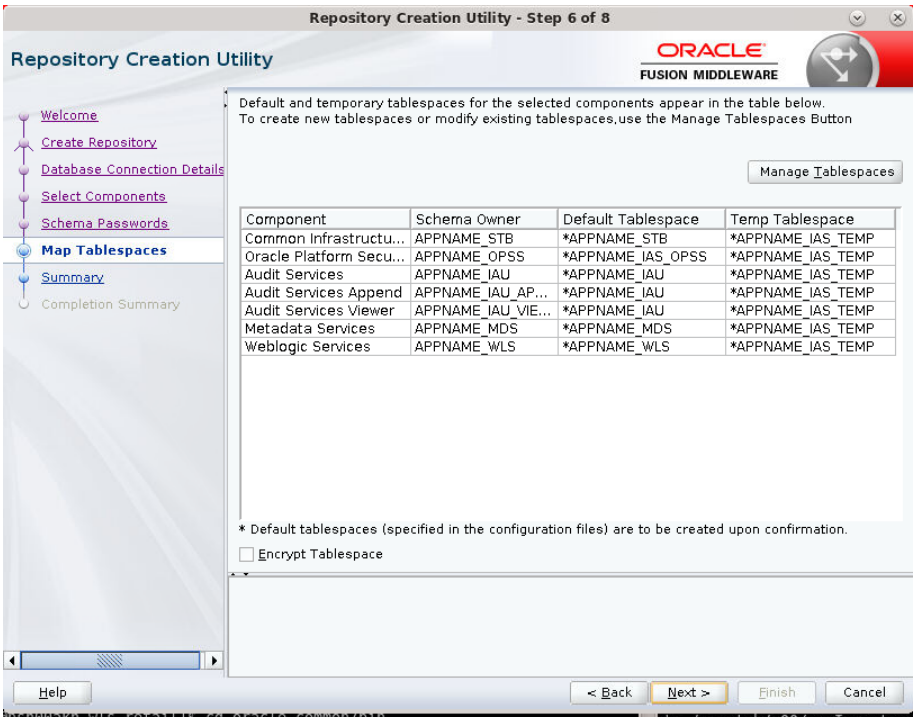
11. Enter password of your choice.

> **Note:** This password is needed at the time of ADF domain creation.

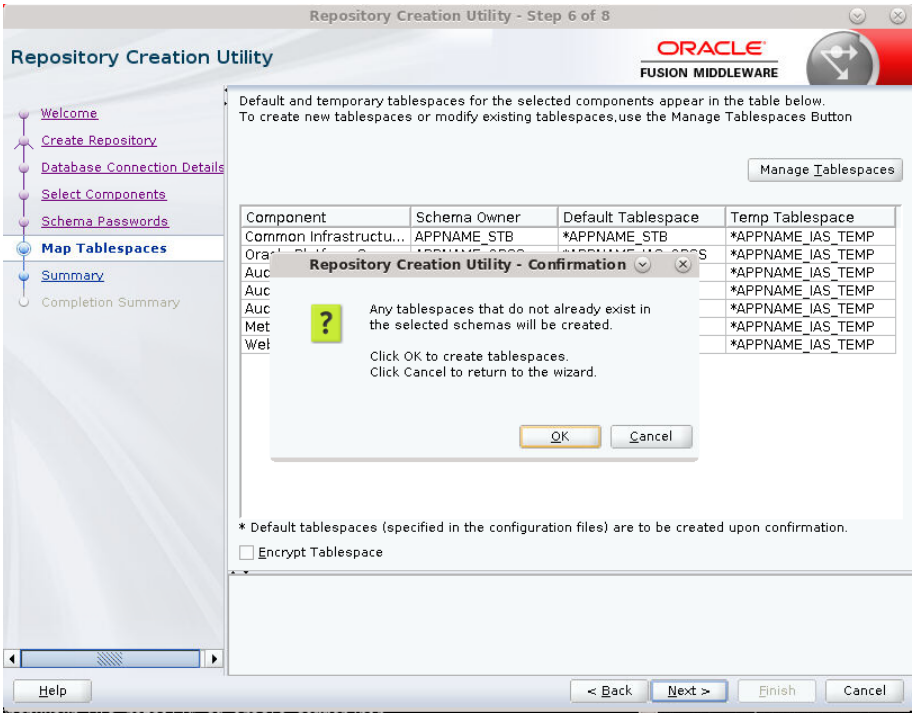


12. Click Next

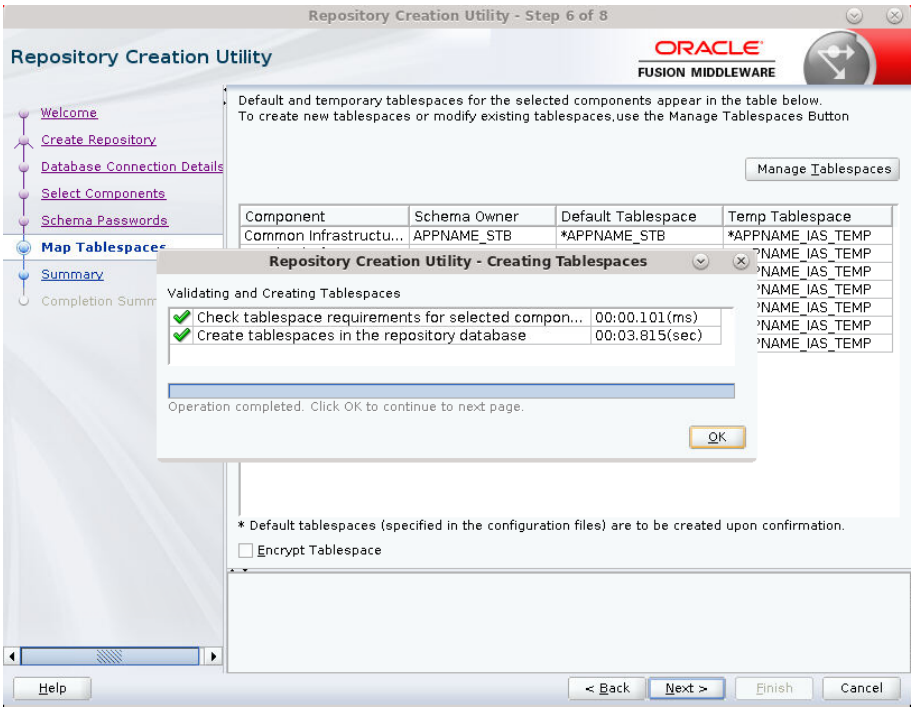
13. Click '**Next**'.



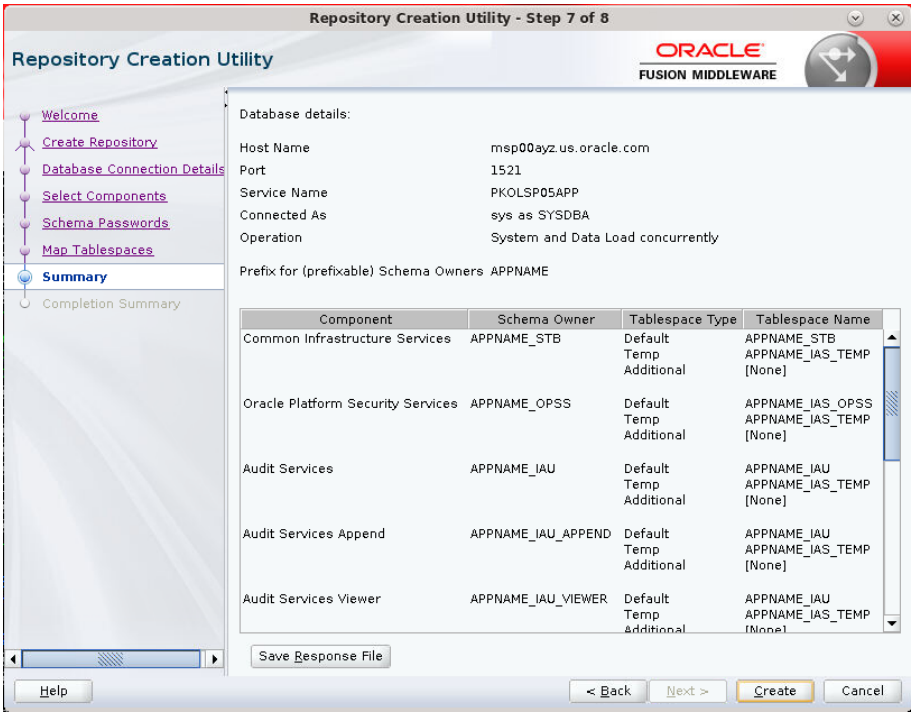
14. A Repository Creation notification will appear. Click **OK**.



15. Tablespaces are created, and the progress will be displayed in a pop-up notification. When the operation is completed, click **OK**.

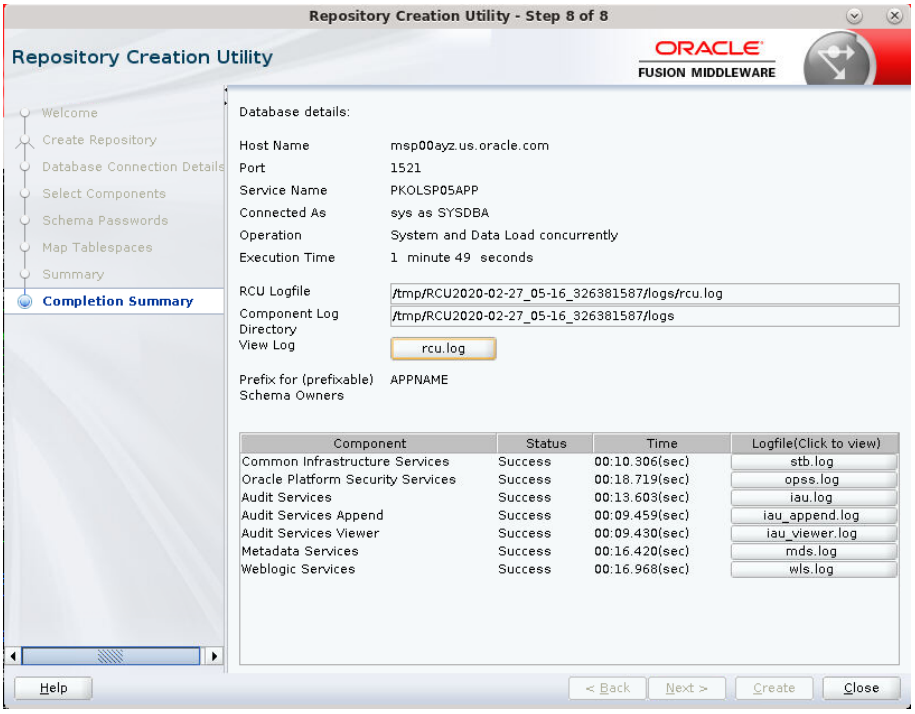


16. Click **Create**. The schema is created.



Upon successful creation of database schemas, a screen will appear with all the schemas created.

17. Click **Close**.



Create a New ADF Domain (with managed server and EM)

To create a new domain and managed server with ADF libraries and EM, follow the below steps:

1. Set the environment variables:

```
export JAVA_HOME=<JDK_HOME>
    (Example:/u00/webadmin/products/jdk_java) [JDK_HOME is the location where
jdk has been installed)
export PATH=$JAVA_HOME/bin:$PATH
export ORACLE_HOME=<ORACLE_HOME>/
  (Example:/u00/webadmin/products/wls_retail)

cd $ORACLE_HOME/oracle_common/common/bin
    (ORACLE_HOMEis the location where Weblogic has been installed.)
```

2. Run the following command:

```
./config.sh
```

3. Select Create a new domain.

   Domain location: Specify the path to the <DOMAIN_HOME>
   Example:/u00/webadmin/config/domains/wls_retail/APPNAMEDomain

4. Click **Next**.

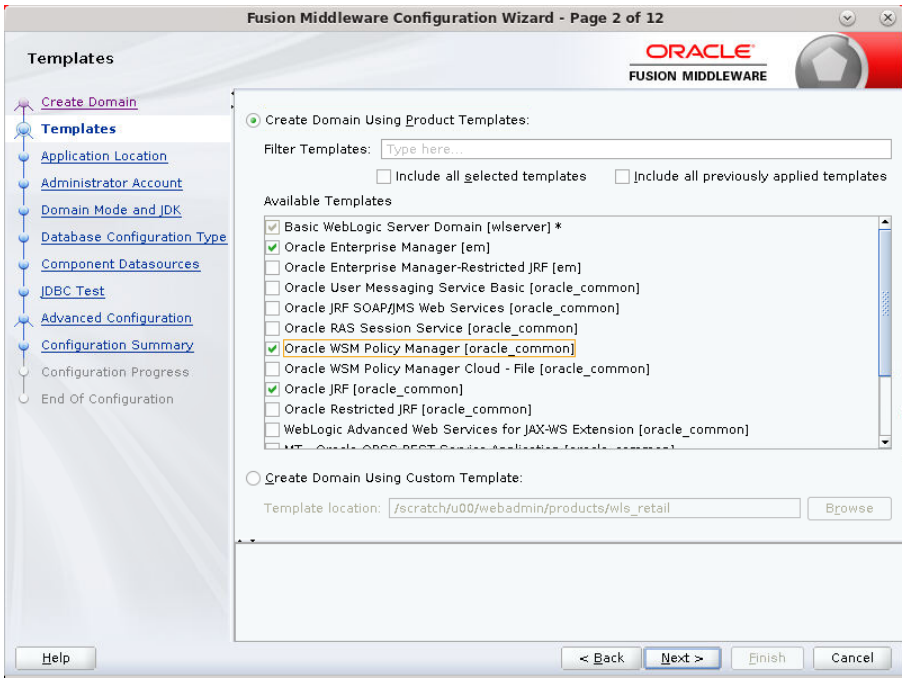
5. Select Create Domain Using Product Templates.
6. Check the following components:
   Oracle Enterprise Manager
   Oracle WSM Policy Manager

   Note: When Oracle Enterprise Manager component is selected, the following dependent components are selected automatically:

   Oracle JRF

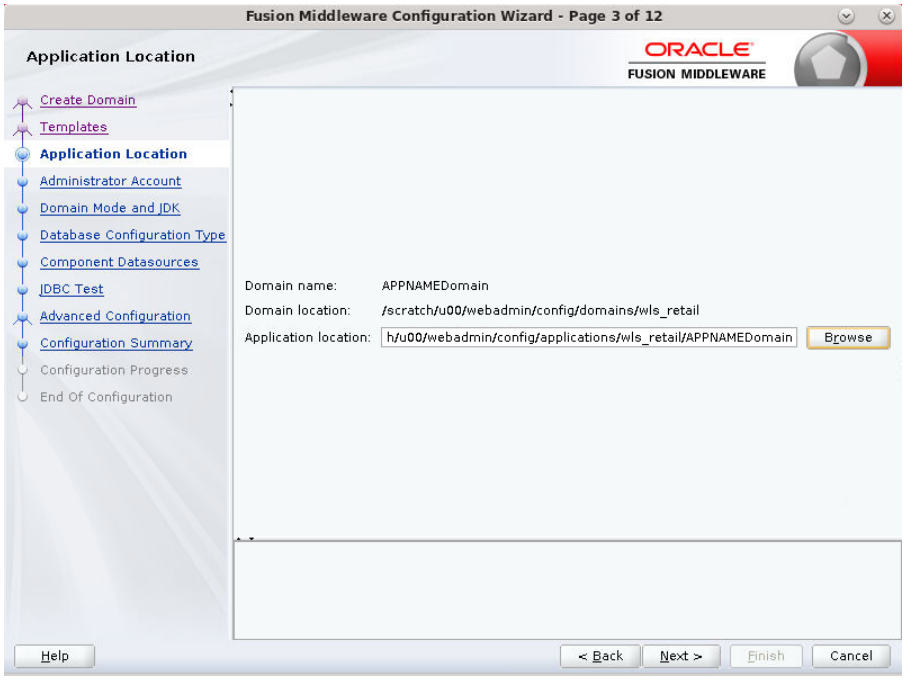
   Weblogic Coherence Cluster Extension

7. Click **Next**.

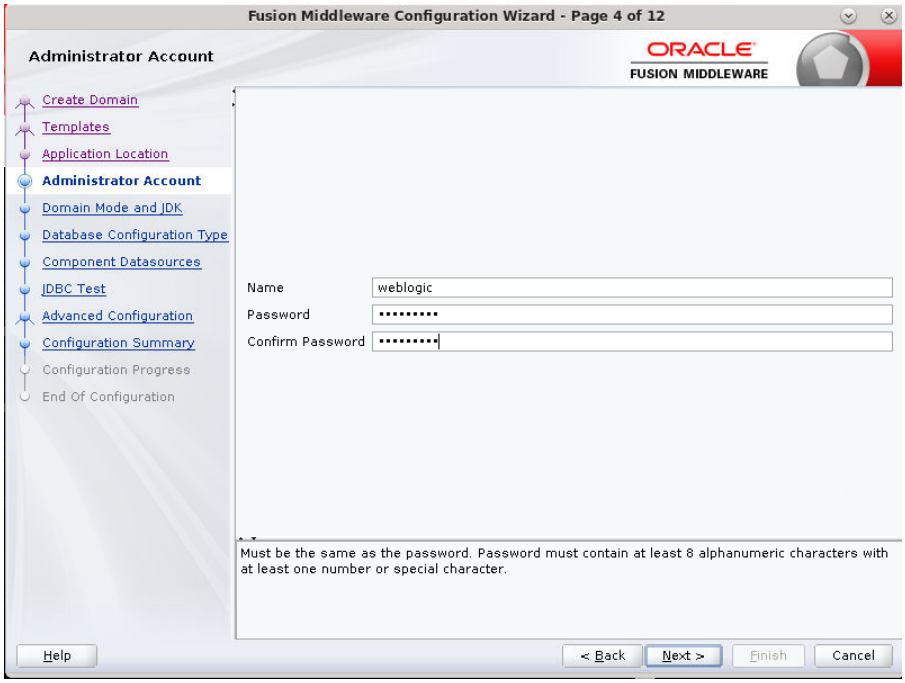


Application location: Application directory location. Example: /u00/webadmin/config/applications/wls_retail/APPNAMEDomain

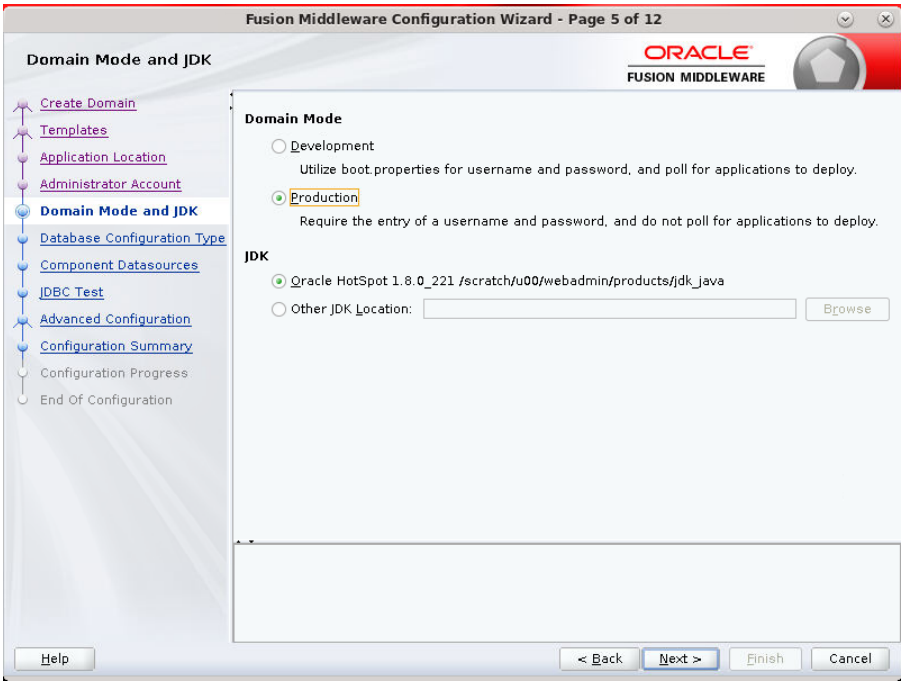
8. Click **Next**.



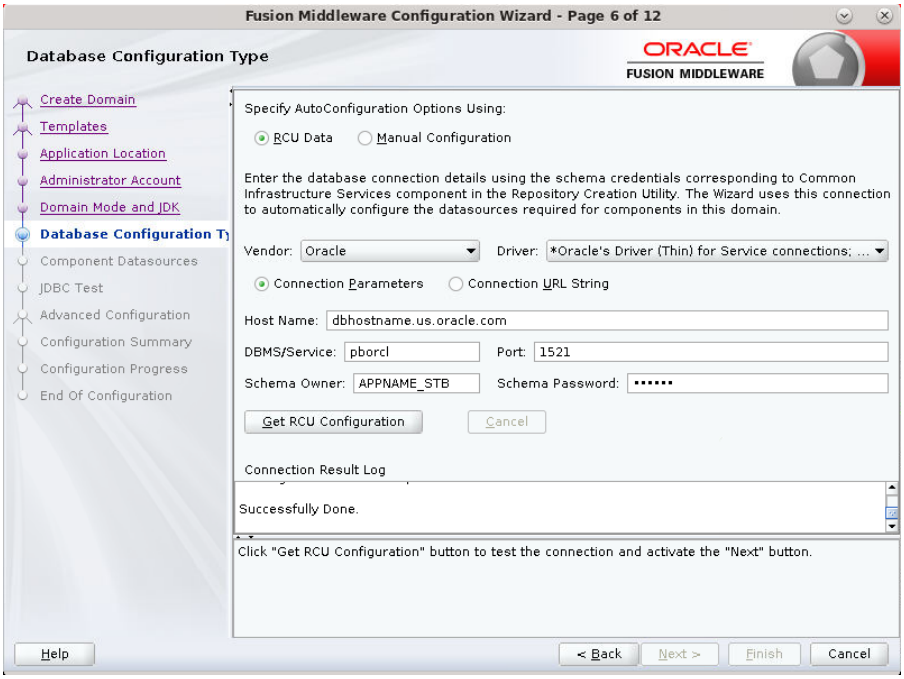
9. Provide the WebLogic administrator credentials and click **Next**:
   - Username: weblogic
   - Password: <Password>



10. Select Domain Mode as Production and the JDK to use (as applicable) and click **Next.**

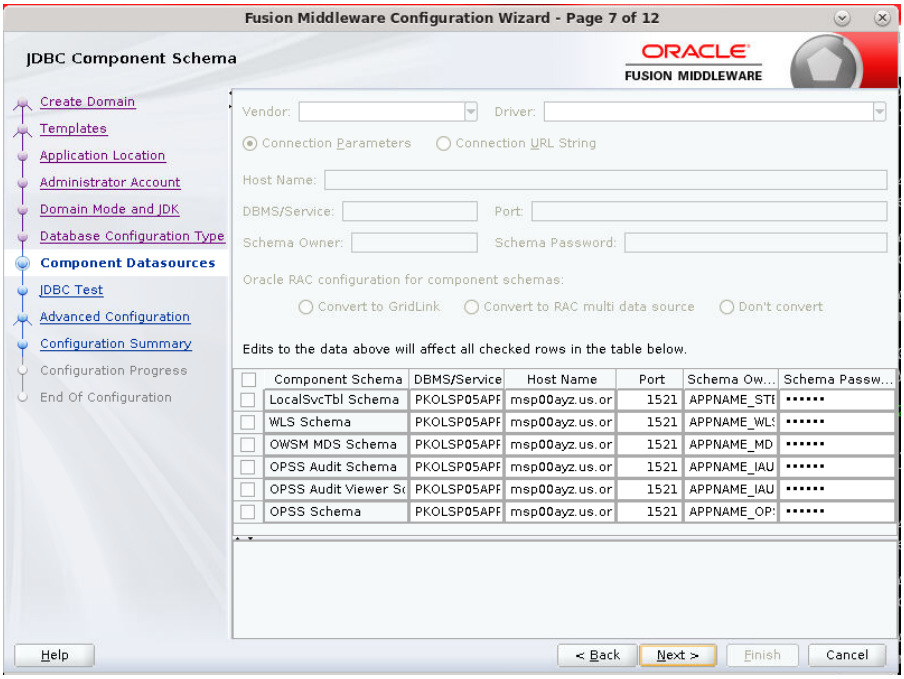


11. Select **RCU Data**.
    - Vendor: Oracle
    - DBMS/Service: dbservicename
    - Host Name: dbhostname.us.oracle.com
    - Port: 1521
    - Schema Owner: APPNAME_STB (Example: ALLOC_STB, ReSA_STB, etc)
    - Password: <Password>. This is the password that was used for RCU schema creation.

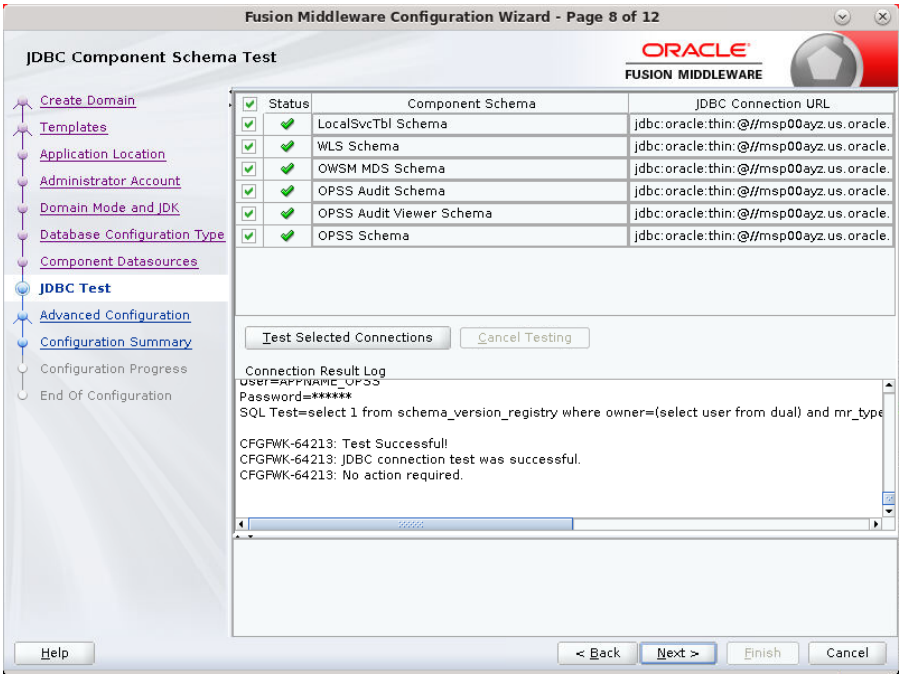


12. Click the **Get RCU Configuration** button.

13. Click **Next**.



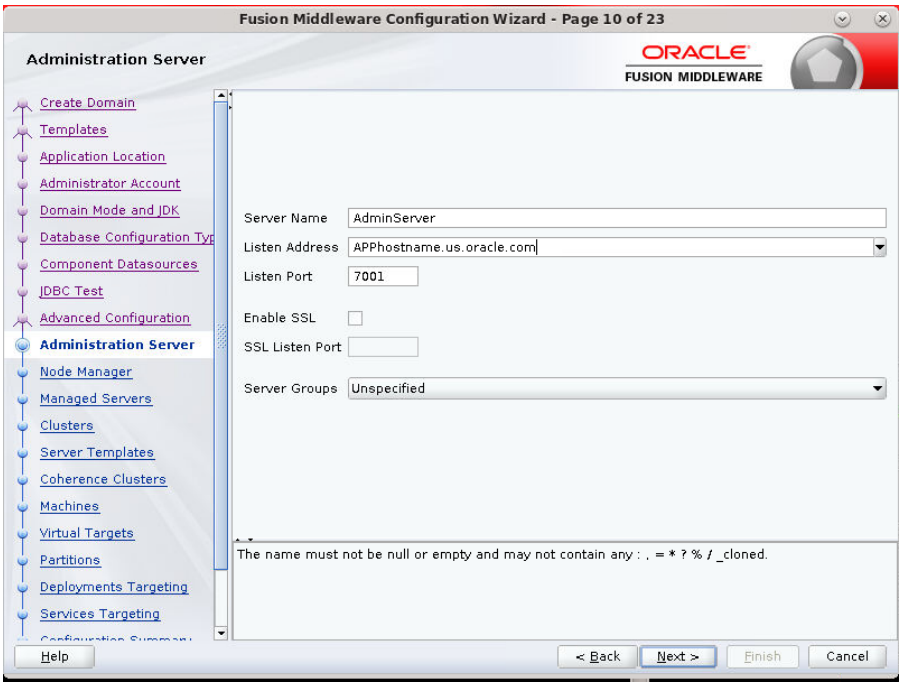
14. Click **Next** to test to make sure it can connect to your datasources.



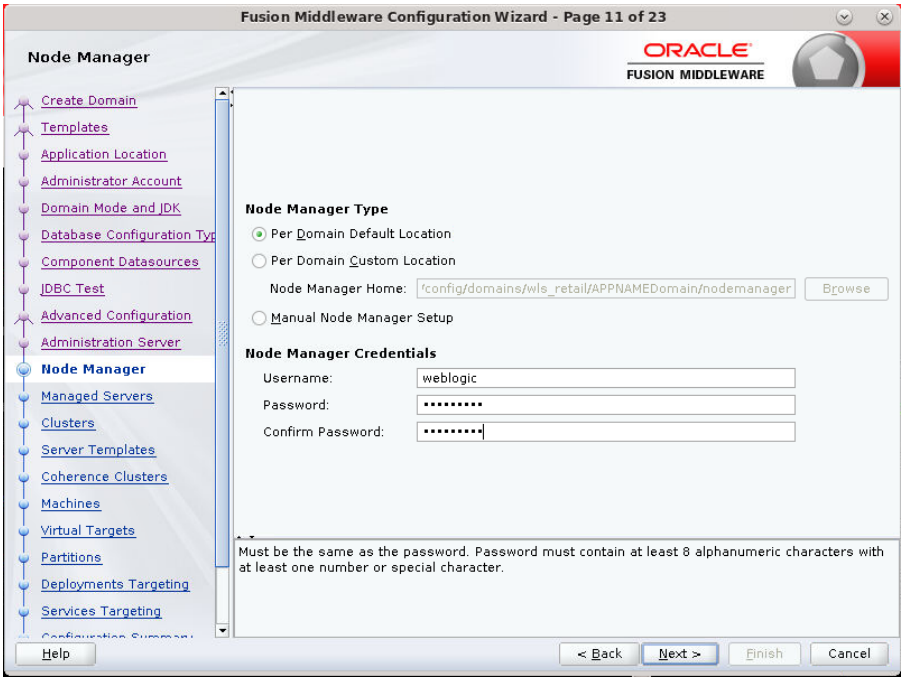
15. Click **Next** to continue.

16. Select advanced configuration for:
    - Administration Server
    - Node manager
    - Topology: Managed Servers, Clusters and Coherence
    - Deployments and Services

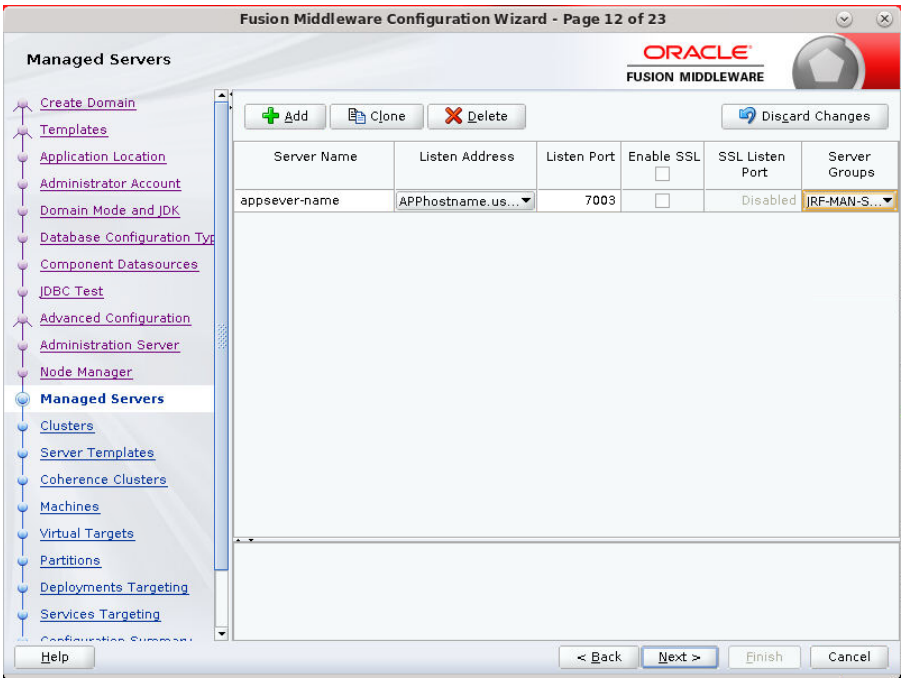
17. Configure the Administration Server:
    - Server Name: <APP name>_AdminServer
    - Listen address: Appserver Hostname or IPAddress of the Appserver Host.
    - Listen port: <Port for Admin Server> Note: The port that is not already used.
    - Server Groups: Unspecified



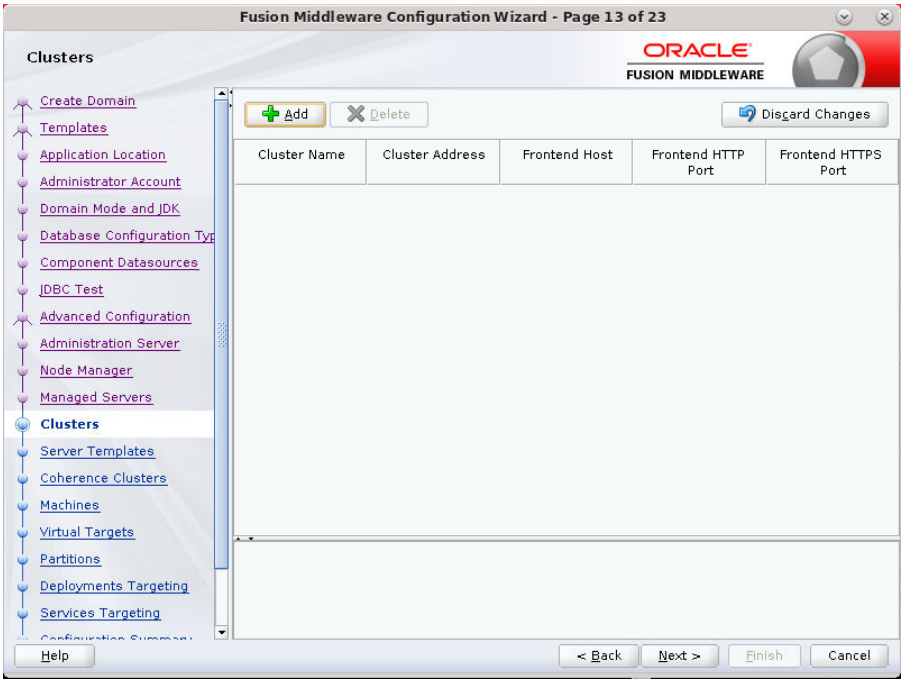
18. Configure Node Manager:
    - Node manager type: Per domain default location
    - Username: weblogic
    - Password: <Password for weblogic>



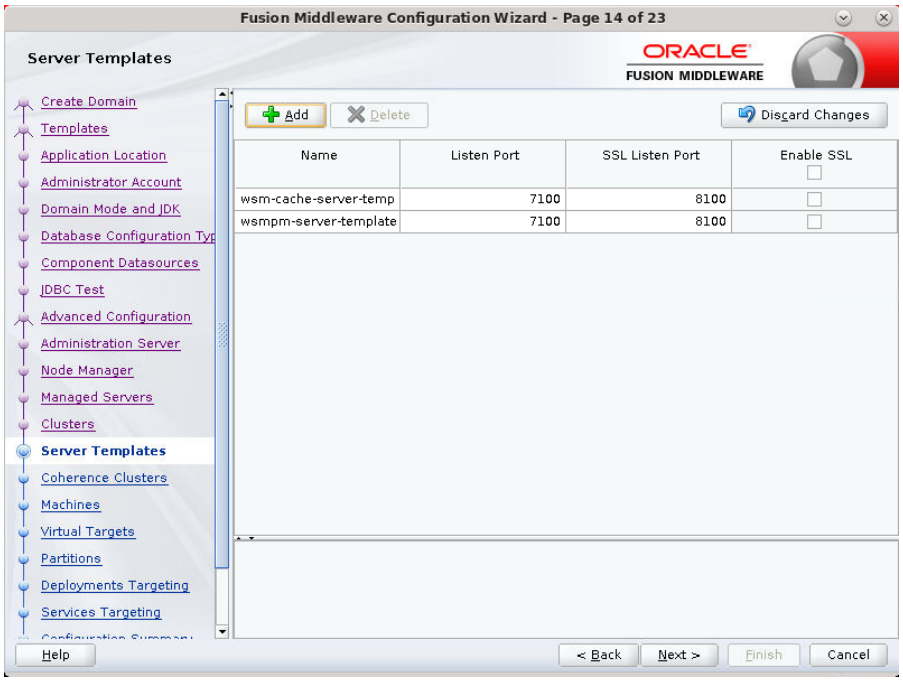
19. Click the **Add** button.
    - Server Name: <appname-server>
    - Listen address: Appserver Hostname or IPAddress of the Appserver Host
    - Listen port: <Port for Managed Server> Note: The port used here must be a free port.
    - Server Groups: JRF-MAN-SVR



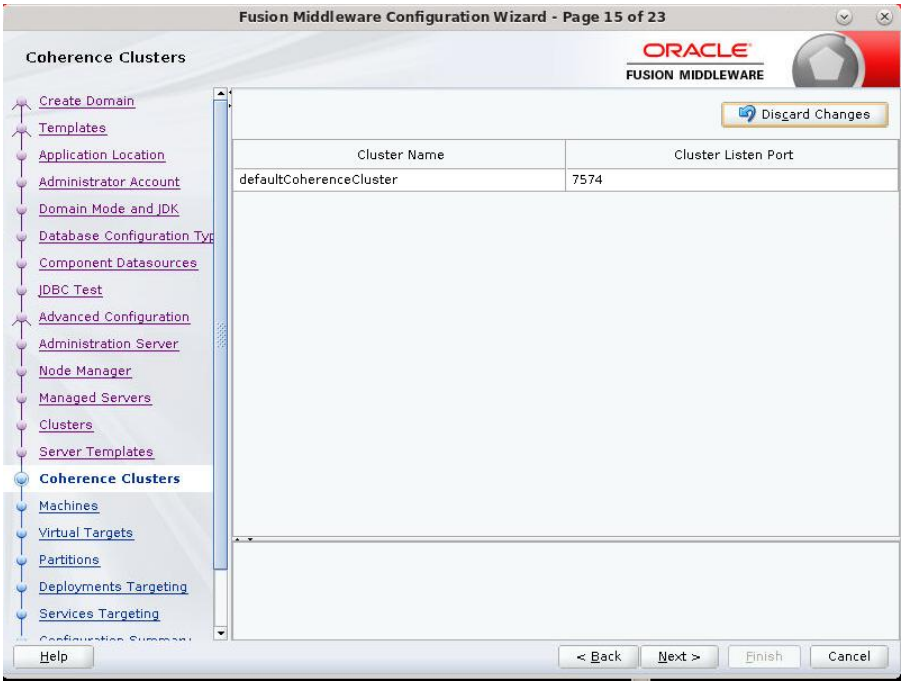
20. Skip Configure Clusters and click **Next**.



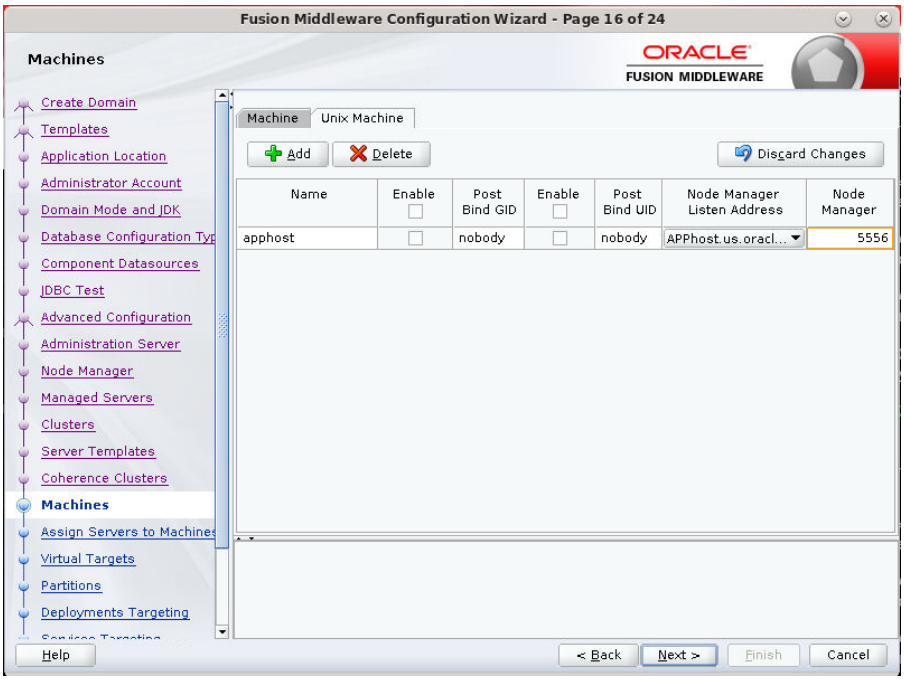
21. Do not change anything and click **Next**.



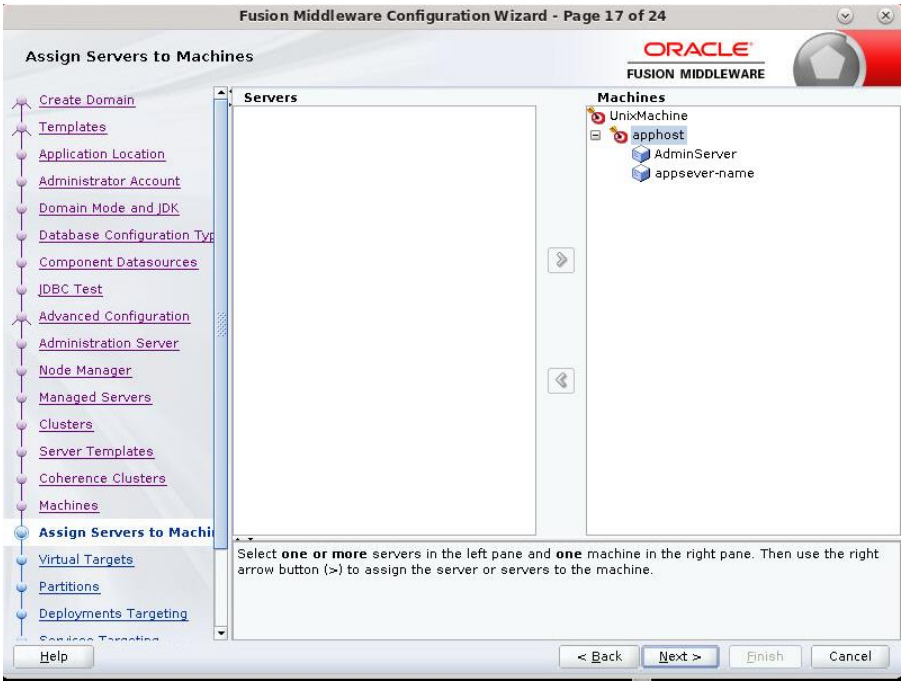
22. Click Next.



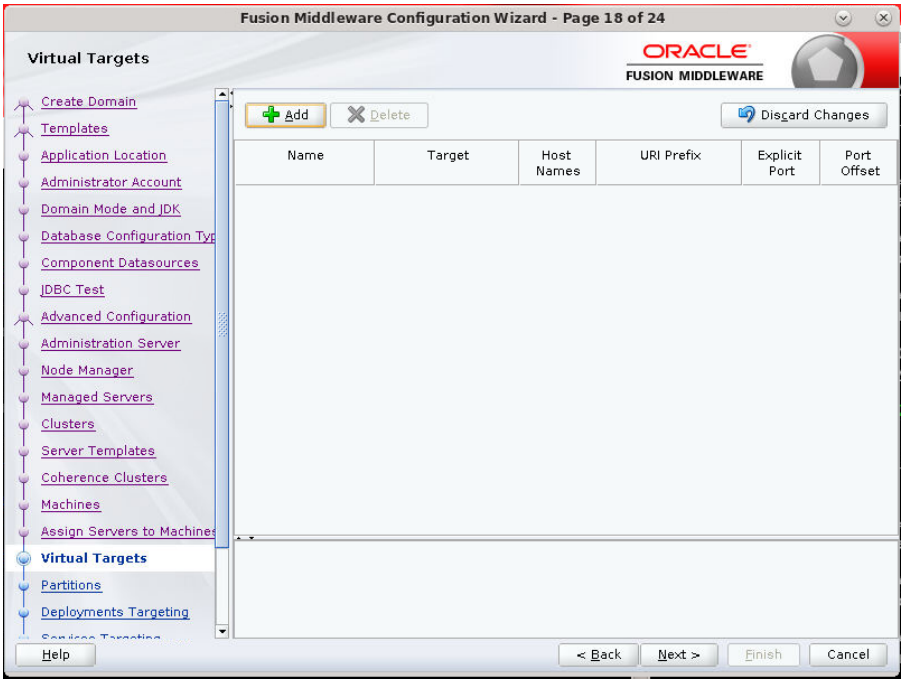
23. Configure Machines by selecting Uunix Machine :
24. Click the **Add** button.
    - Name: apphostname_MACHINE
    - Listen address: apphostname or IPAddress
    - Listen port: <Port for node manager> Note: The port used here must be a free port.



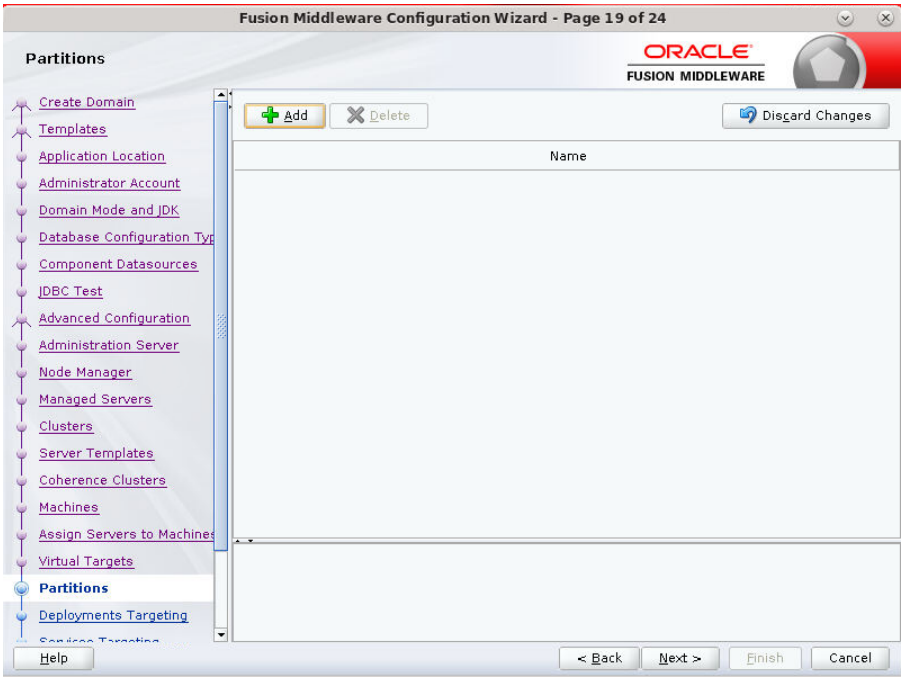
25. Assign the configured Admin server and managed servers to the new machine.



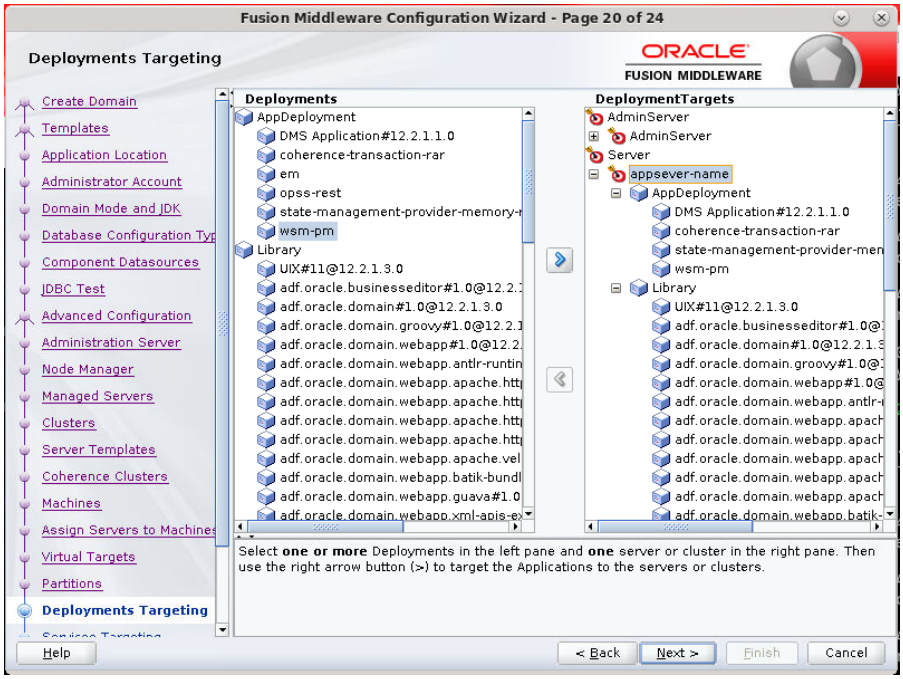
26. Skip Virtual Targets. Click **Next.**



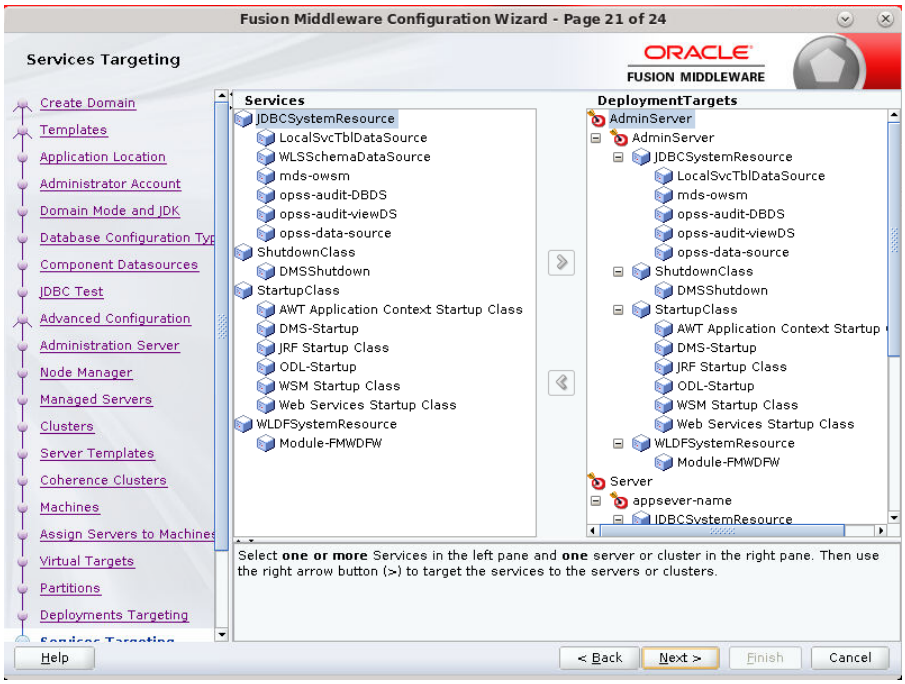
27. Skip Partitions. Click Next.



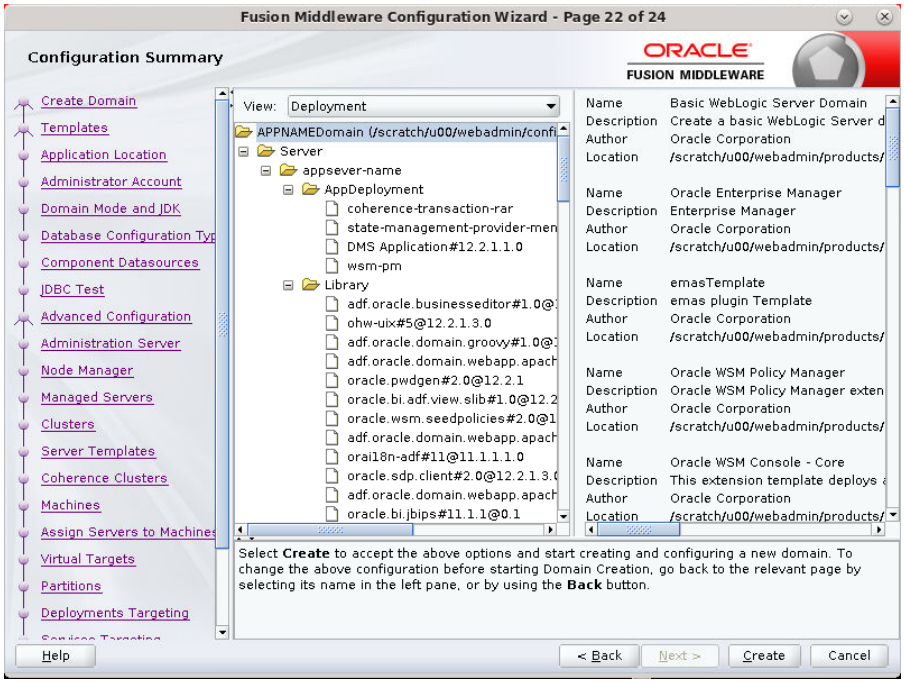
28. Target the “wsm-pm” deployment to APPNAME_AdminServer:



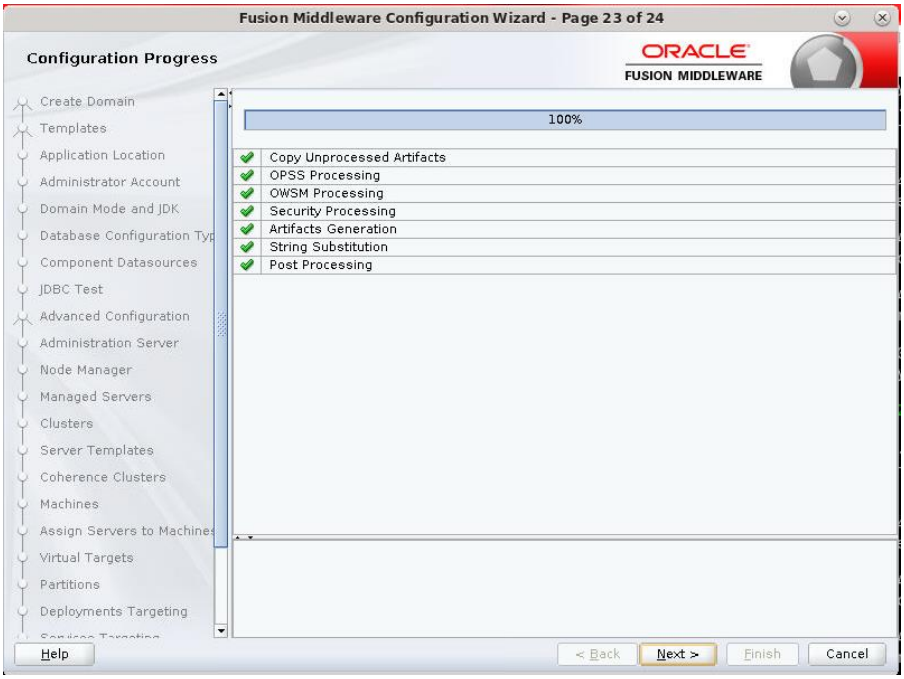
29. Click Next.



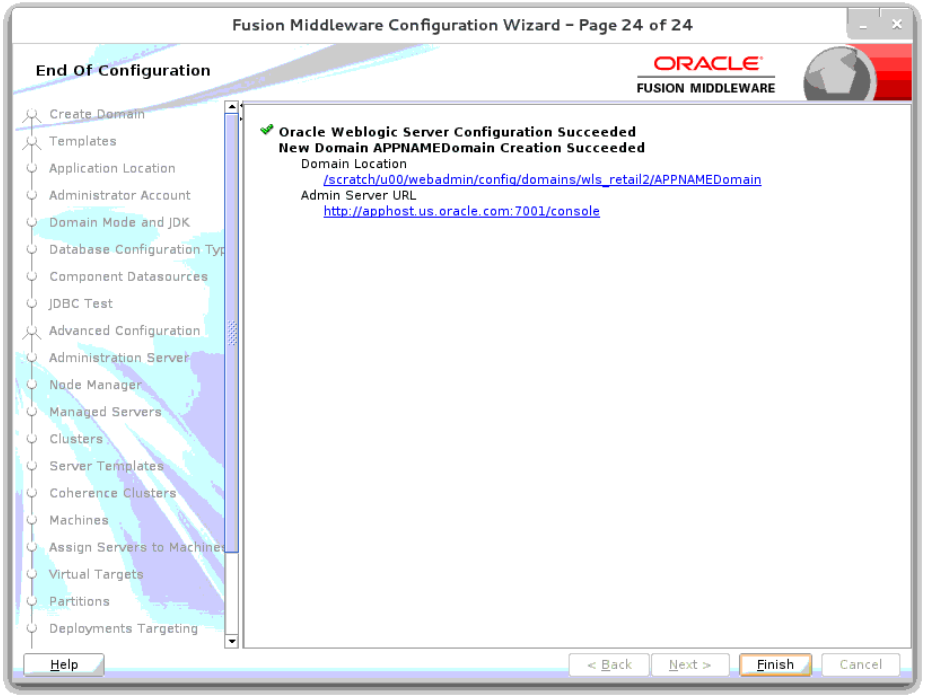
30. Click **Create**.



31. Click **Next**.



32. When the process completes, click **Finish**.



Start the Node Manager

1. Start the nodemanager from <DOMAIN_HOME>/bin using the following script:

   ```
   nohup ./startNodeManager.sh &
   ```

Start the AdminServer (admin console)

1. Configure boot.properties for starting the Weblogic domain without prompting to username and password using the following command:
2. Create security folder at <DOMAIN_HOME>/servers/<AdminServer>/ and create boot.properties file under <DOMAIN_HOME>/servers/<AdminServer>/security

   The file 'boot.properties' should have the following:

   ```
   ---------------------------------
   username=weblogic
   password=<password>
   ---------------------------------
   ```

   In the above, the password value is the password of WebLogic domain which is given at the time of domain creation.

   Save the boot.properties file and start WebLogic server.
3. Start the WebLogic Domain (Admin Server) from <DOMAIN_HOME> using the following:

   ```
   nohup ./startWebLogic.sh &
   ```

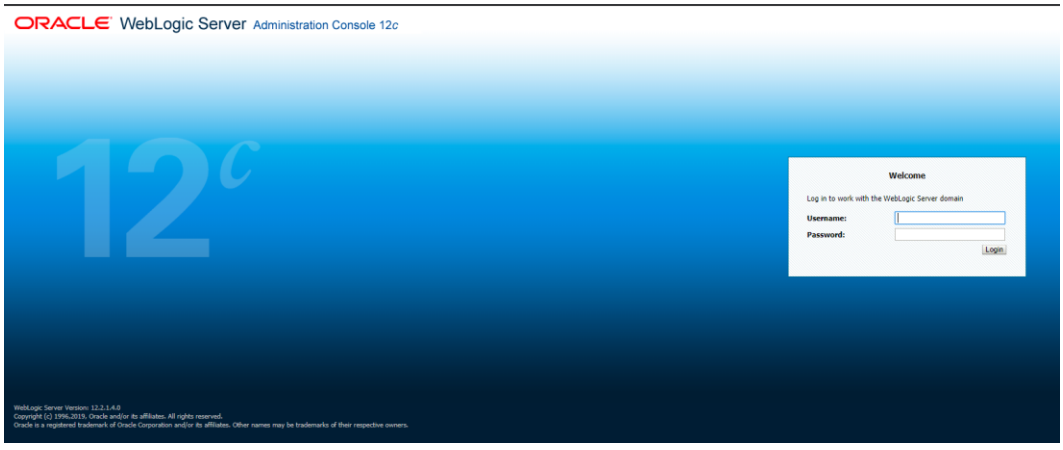
   Example:

   ```
   nohup /u00/webadmin/config/domains/wls_retail/RPMdomain/ startWebLogic.sh
   ```

4. Access the Weblogic Admin console

   Example: **Error! Hyperlink reference not valid.**

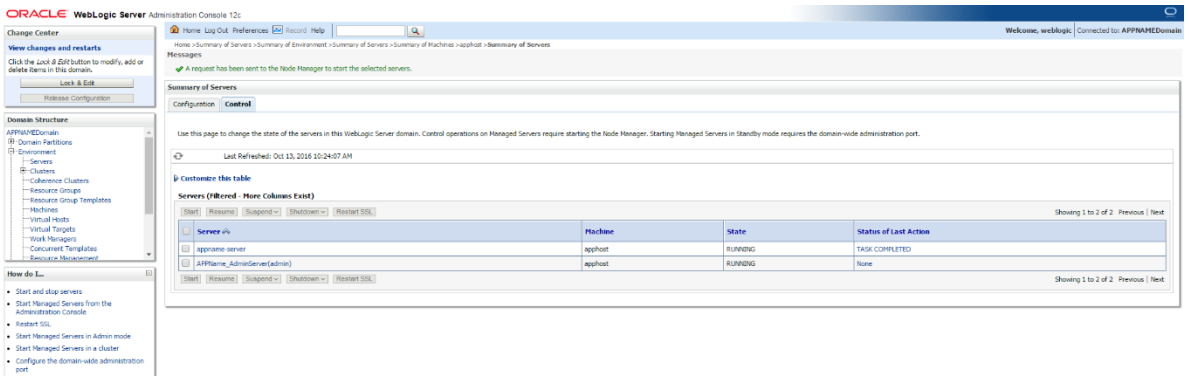
   In the below screen, provide username=weblogic and password=<weblogic password>



Start the Managed Server

After NodeManager is started, the managed servers can be started via the admin console.

1. Navigate to Environments -> Servers and select the Control tab. Select appname-server and click **Start**.

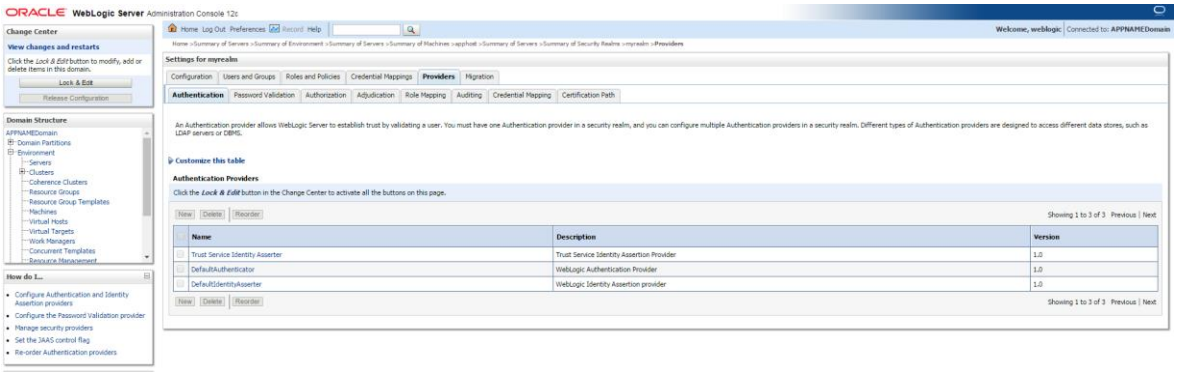


The Managed Server should be up and running before configuring further steps

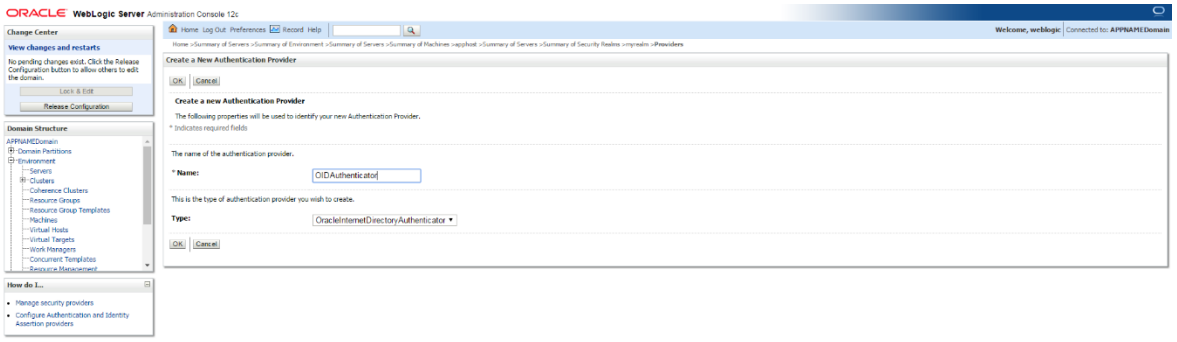
Configuration of OID LDAP Provider in WebLogic Domain:

Perform the following procedure to create LDAP providers in the domains created in the previous steps

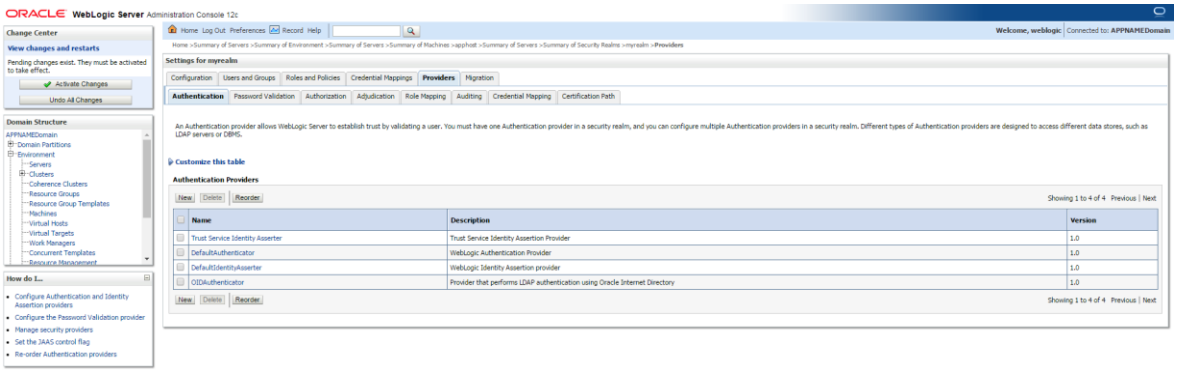
1. Log in to the Administration Console.
   `http://<HOSTNAME>:<ADMIN_PORT>/console`
2. In the Domain Structure frame, click Security Realms.
3. In the Realms table, click myrealm. The Settings for myrealm page is displayed.
4. Click the Providers tab.



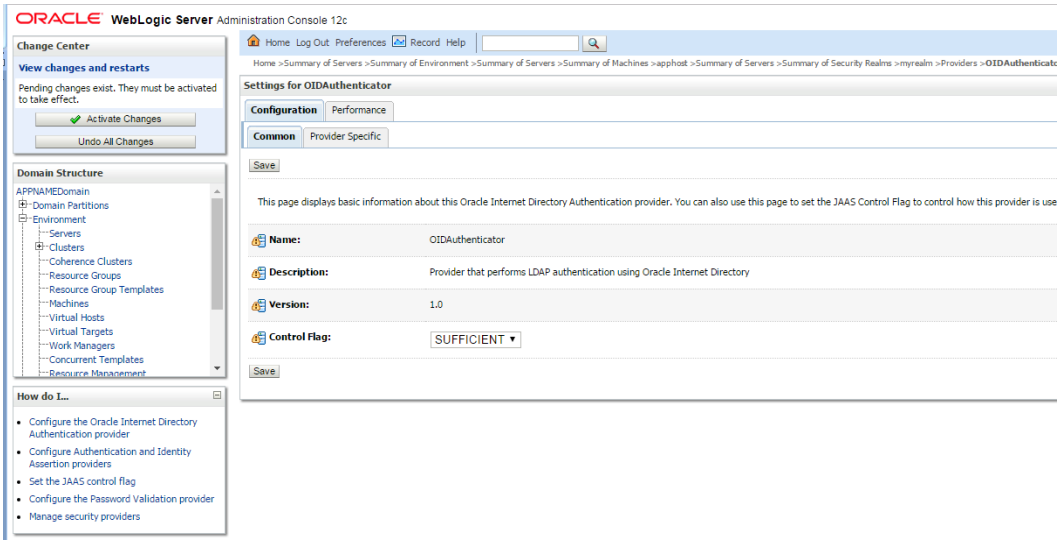
5. Click **Lock & Edit** and then click **New**. The 'Create a New Authentication Provider' page is displayed.



6. Enter OIDAuthenticator in the Name field and select OracleInternetDirectoryAuthenticator as the type. Click **OK**.

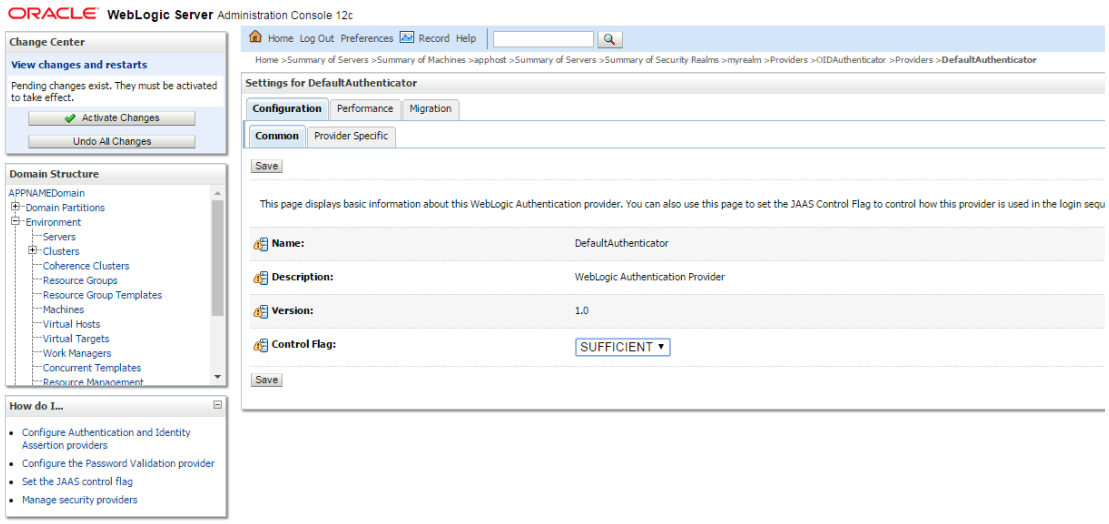


7. All providers are displayed. Click **OID Authenticator**. The Settings for the OID Authenticator are displayed.



8. Set the Control Flag field to SUFFICIENT and click **Save**.

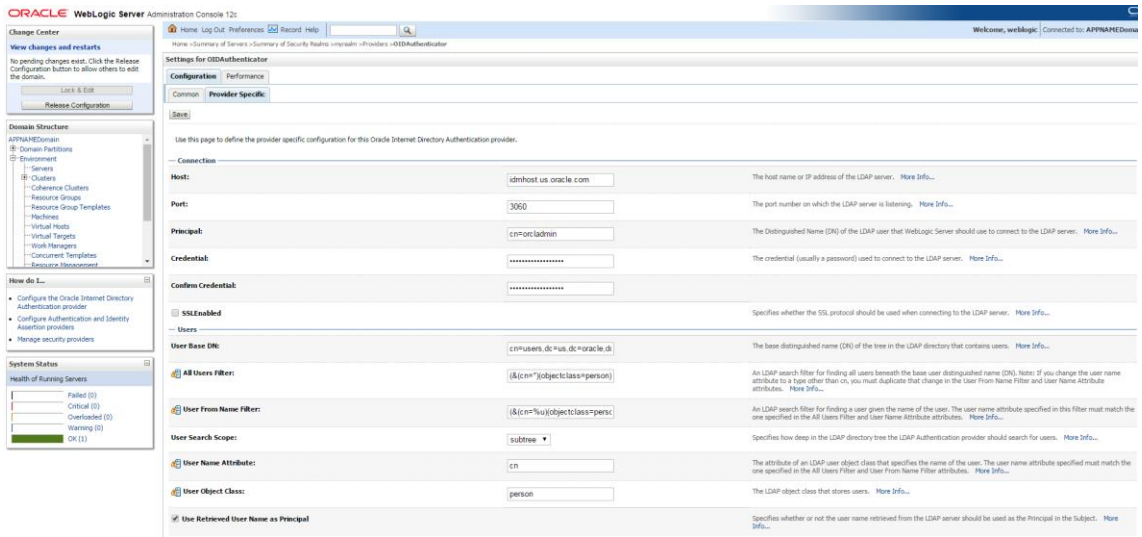
9. From the Providers tab, click on DefaultAuthenticator -> Configuration tab -> Common tab. Update the Control Flag to SUFFICIENT.
10. Click **Save**.



11. From the Providers tab, click the “OIDAuthenticator” (you just created), in the configuration -> Provider Specific tab enter your LDAP connection details:

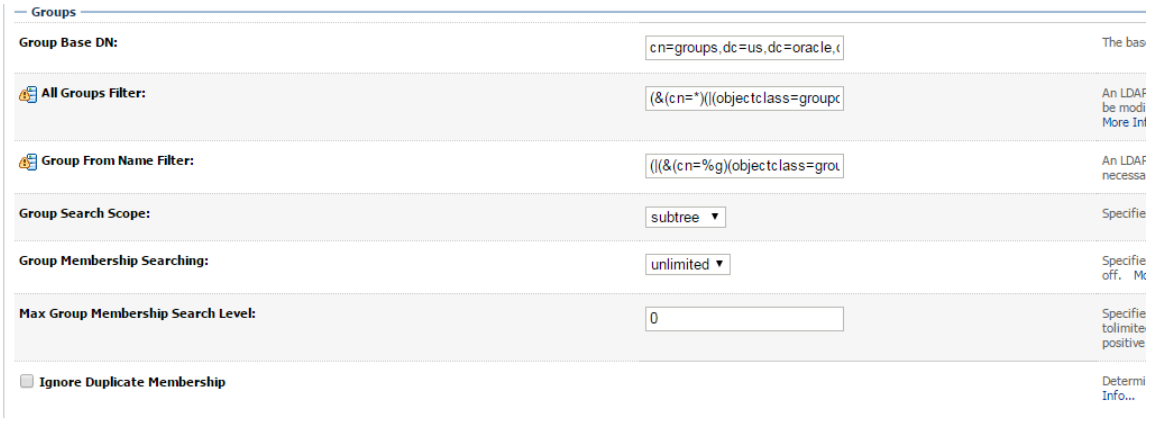
    The values shown below are examples only. You should match the entries to your OID.

    - Host: <*oidhost*>
    - Port: <*oidport*>
    - Principal: cn=orcladmin
    - Credential: <*password*>
    - Confirm Credential: <*password*>
    - User Base DN: cn=users,dc=us,dc=oracle,dc=com
    - Enable ‘Use Retrieved User Name as principal.’

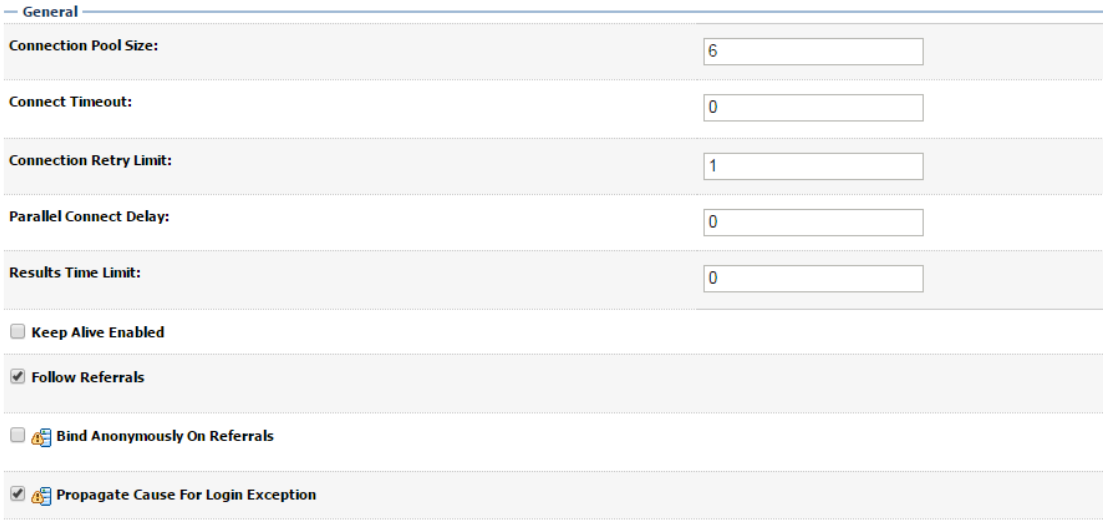


12. Modify the following:

- Group Base DN: cn=Groups,dc=us,dc=oracle,dc=com

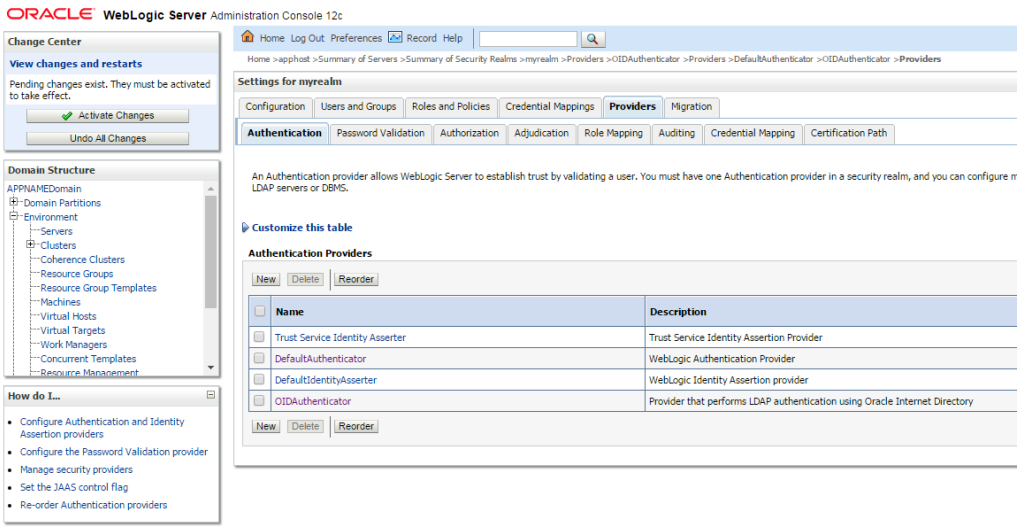


13. Check Propagate Cause For Login Exception

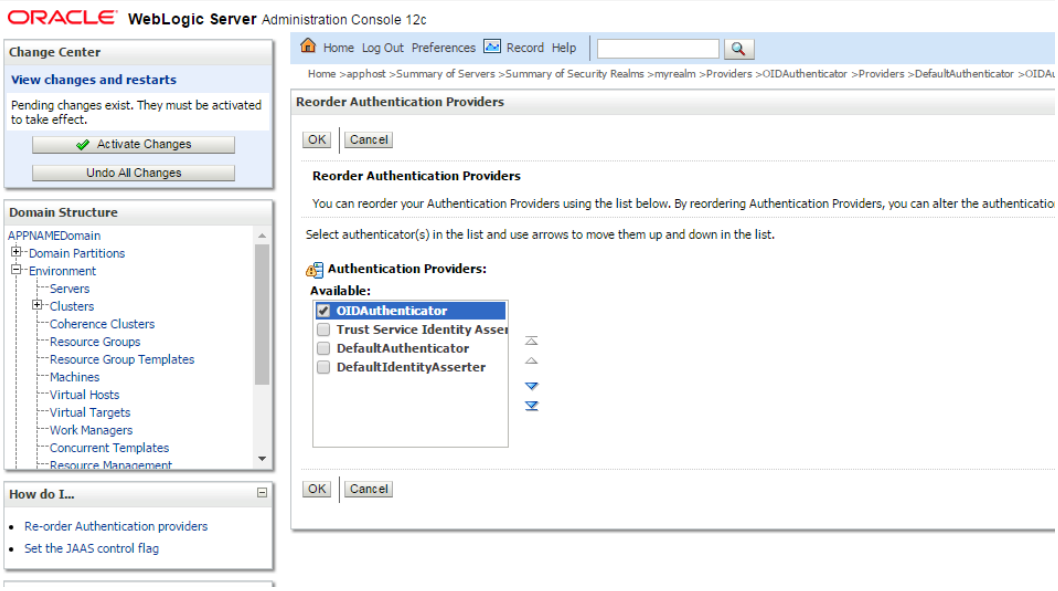


14. Click **Save**.

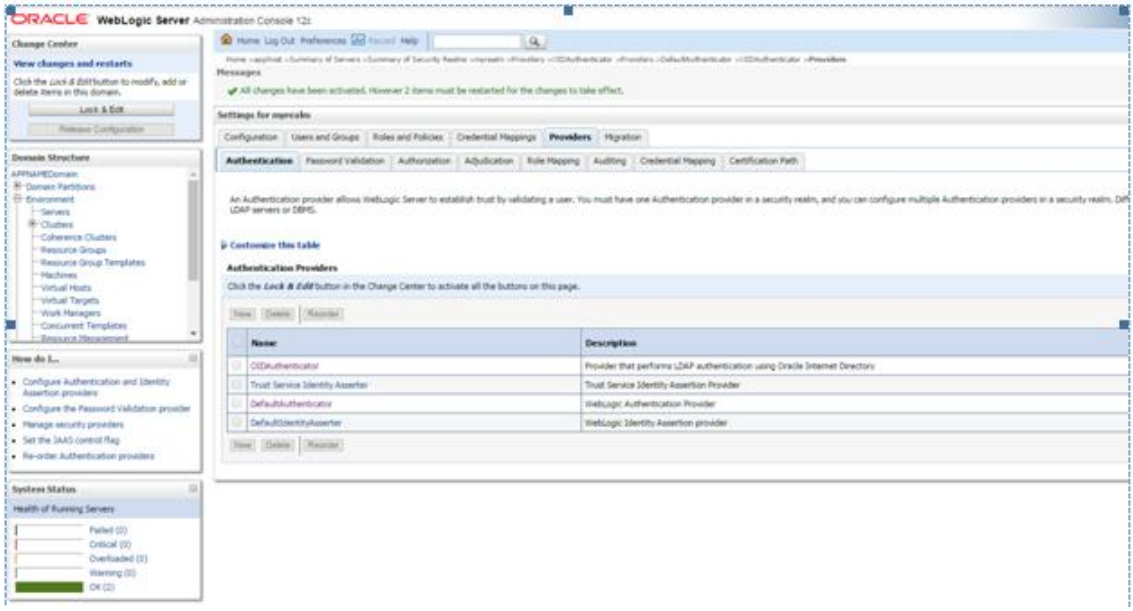
15. Click the Providers tab.



16. Click **Reorder**.
17. Move OIDAuthenticator to the top of the providers list.



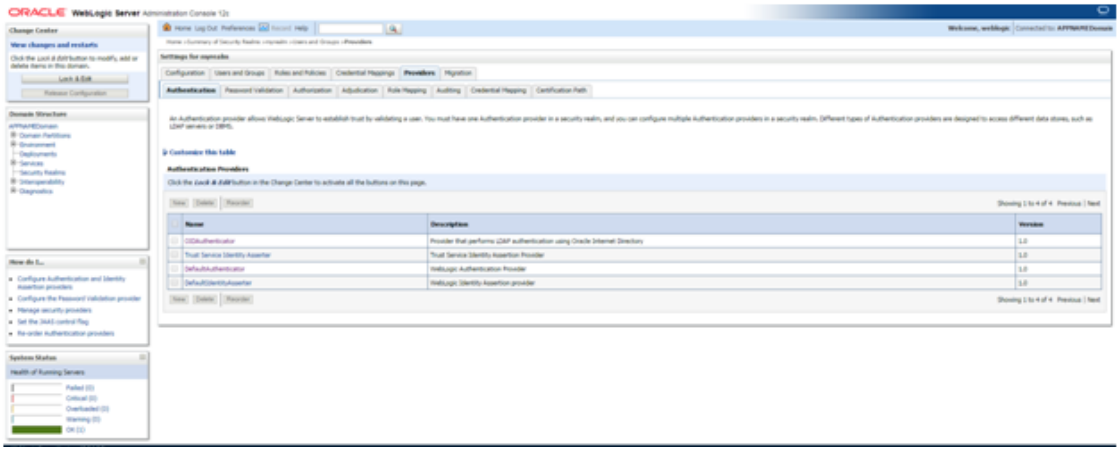
18. Click **OK**.
19. Once your changes are saved, click **Activate Changes**.



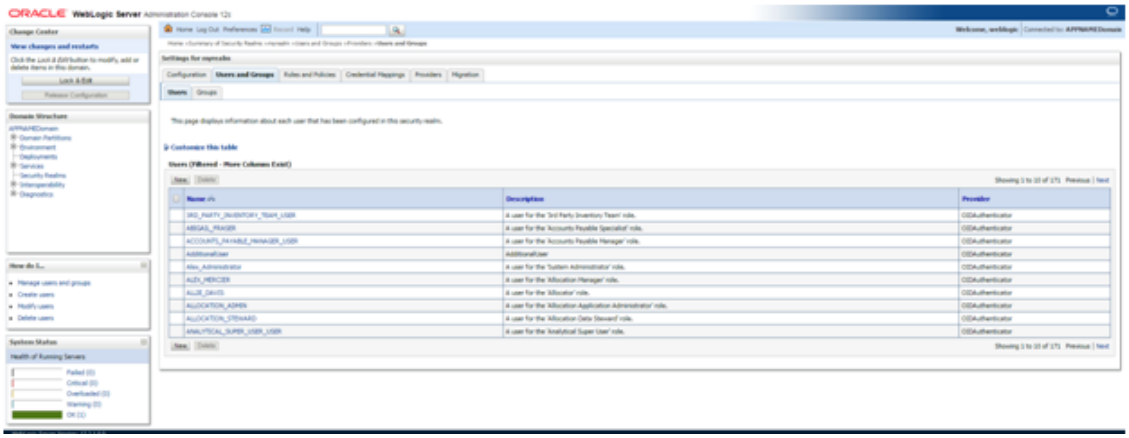
20. Shutdown all servers and restart the admin server using startWebLogic.sh script. Login to Admin Console and restart Managed server.

Verify OID Authenticator

1. Log in to the Administration Console.
   http://<HOST_NAME>:<ADMIN_PORT>/console/
2. In the Domain Structure frame, click Security Realms.
3. In the Realms table, click Default Realm Name. The Settings page is displayed.
4. Click the Providers tab. You must see the OID Provider in that list.



5. Click the Users and Groups tab to see a list of users and groups contained in the configured authentication providers.



Configure Oracle Single Sign-On

> **Note:** This procédure is only needed if RMS application setting up using Single Sign On (SSO) authentication. This can be skipped if SSO is not going to be used. The Oracle Access Manager must be configured and the Oracle http server (Webtier and webgate) must be registered into the Oracle Access Manager.

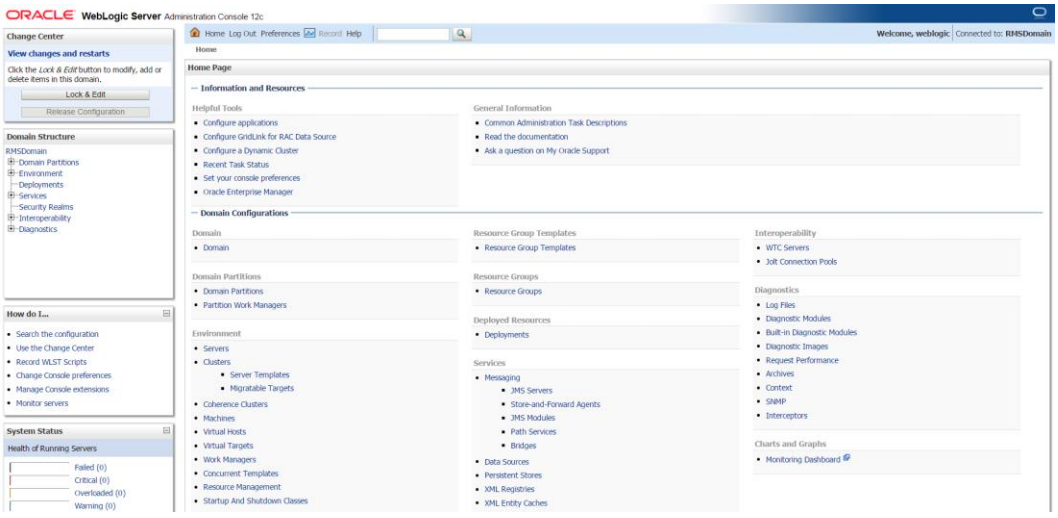
(Webtier and webgate) must be registered into the Oracle Access Manager

1. Log into the WebLogic console.
2. Navigate to: security realms -> myrealm (default realm) -> providers.
3. Start a Lock and Edit session.
4. Click New provider.
5. Set the provider name (Default: OAMIdentityAsserter).
6. Click **OK**.
7. Open the new provider configuration.
8. Under Common, set the Control Flag to REQUIRED.
9. On the provider list, click **Reorder**.
10. Move the OAMIdentityAsserter to the top of the list, or above the DefaultAuthenticator.
    a. Click **OK**.
    b. Click Activate Changes.
    c. Shutdown the domain.
    d. Start the admin and managed servers for the domain.

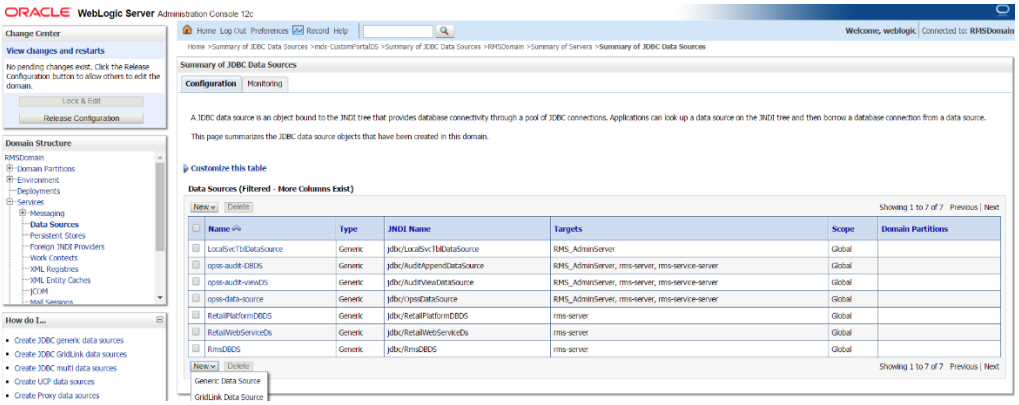
Create mds-CustomPortalDS Datasource using console

Follow below steps to create mds-CustomPortal datasource using console:

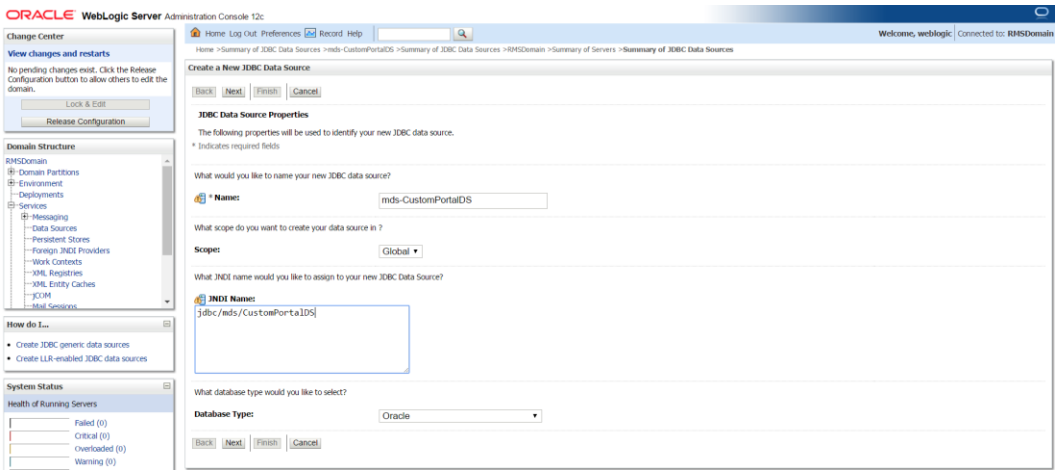
1. Login to Weblogic Admin console with Administrator user credentials.
   Error! Hyperlink reference not valid.console



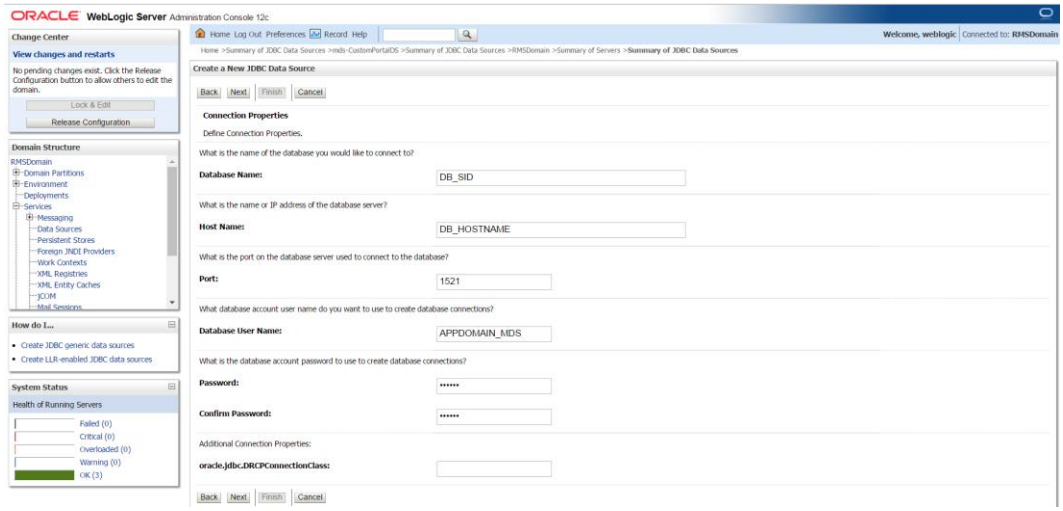
2. Take Lock & Edit and Navigate to Services→Data Sources and click on New→Generic Data Source.



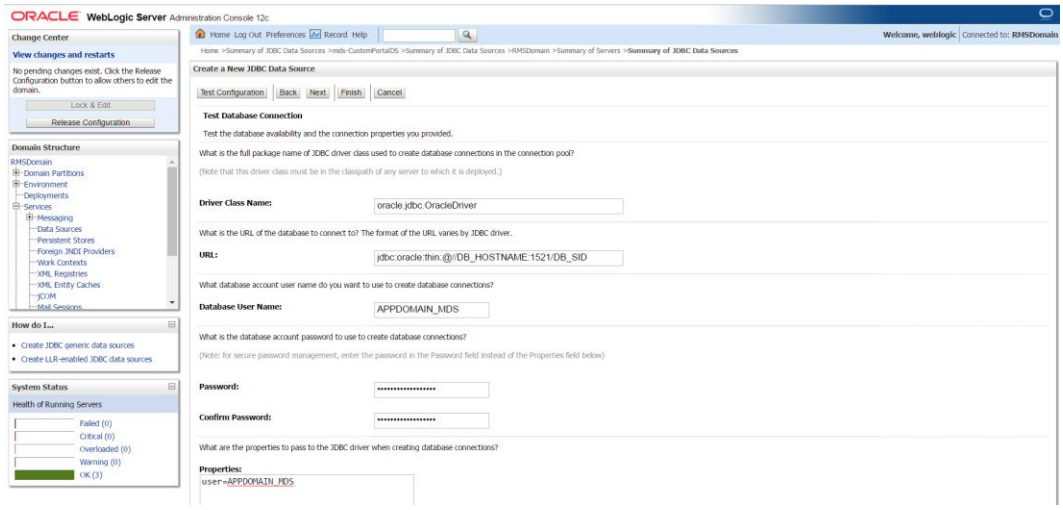
3. Provide mds-CustomPortalDS name, JNDI Name and Database Type



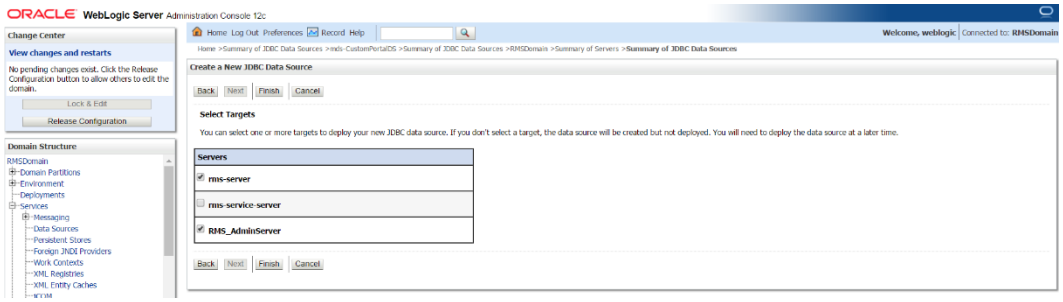
4. Select Oracle's (Thin) Driver Service connections and Click next. Input the details of Database Hostname, Port number and Service name. Provide Database username and password created during RCU installation. Click **Next**.



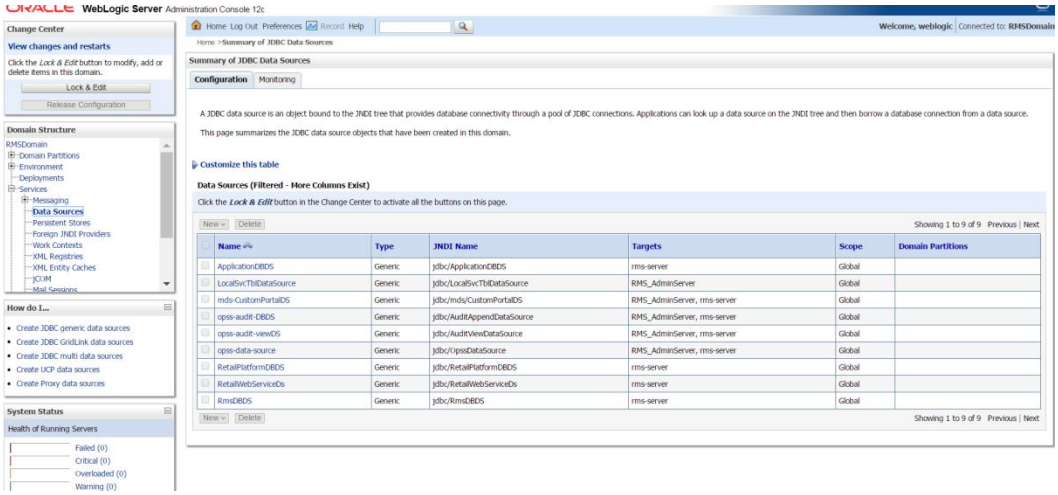
5. Click Test Configuration to test the DB connection and Click **Next**



6. Select Targets as Managed server and Admin Server, click **Next**.



7. Click on Activate Changes and verify mds-CustomPortalDS exists in the Data Sources.



Load LDIF Files in LDAP

> **Note:** In this section, the base DN "dn=us,dn=oracle,dn=com" is used as an example. Modify this value as per the organisation's ldap settings.

The OID (Oracle Internet Directory) must be set up in order to perform the configuration of OID Authenticator in WebLogic Domain.

There are four LDIF files provided in the application zip under STAGING_DIR/ rms/installer/mom/ldifs

- RGBU-oid-create-groups.ldif
- RGBU-oid-create-users.ldif
- RGBU-oid-delete-groups.ldif
- RGBU-oid-delete-users.ldif

> **Note:** You may use the existing users and existing groups if the enterprise users and groups are already available in the LDAP. The users provided in the LDIF files above may not be required to use the application. For more information, refer to the Retail Role Hierarchy section in the Implementing Functional Security of the *Oracle Retail Merchandising System Operation Guide*.

The steps given below can be used to import the Groups and Users into the LDAP using the LDIF files 'RGBU-oid-create-groups.ldif' and 'RGBU-oid-create-users.ldif'.

Note: If you are using the above LDIF files to set up the users and groups, you must update the 'RGBU-oid-create-user.ldif' LDIF file with your password for the 'userpassword' attribute for all the users mentioned in the RGBU-oid-create-user.ldif LDIF file. The changes must be done before importing the users LDIF file 'RGBU-oid-create-users.ldif' into the LDAP. Once the users are imported into the LDAP, remove the 'userpassword' attribute value from the LDIF file. Refer to the *Oracle Internet Directory Administration Guide* for OID password policies for setting up passwords.

User DN and Group DN values (example: dc=us,dc=oracle,dc=com) may need to be updated based on the DN values in your OID.

Once the LDIF files are updated for your configuration, the LDIF files can be loaded into LDAP using the ldapadd tool that is included in the OID installation. LDIF files can also be imported in other ways like ODSM.

For example to load RGBU-oid-create-users.ldif using ldapadd (this is done on the OID host)

```
export ORACLE_HOME=/u00/webadmin/products/wls_idm/ORACLE_IDM (this is the
ORACLE_HOME of your OID install)
export PATH=$ORACLE_HOME/bin:$PATH
$ORACLE_HOME/bin/ldapadd –v –c –h <OID_HOST> –p 3060 –w <ORCLADMIN PASSWORD> –D
"cn=orcladmin" –f RGBU-oid-create-users.ldif
```

The delete LDIF 'RGBU-oid-delete-groups.ldif' can be used as needed if you need to delete the groups created from the groups creation LDIF 'RGBU-oid-create-groups.ldif'.

The delete LDIF 'RGBU-oid-delete-users.ldif' can be used if you need to delete the users created from the users LDIF file 'RGBU-oid-create-users.ldif'.

Clustered Installations – Preinstallation Steps

Skip this section if you are not clustering the application server.

- Make sure that you are able to start and stop the managed servers that are part of the RMS Cluster from the WebLogic Admin Console.

There are no additional steps before running the installer for RMS.

Create Staging Directory for RMS Application Server Files

To create the staging directory for the RMS Installer, complete the following steps.

Note: The same installer can be used to install multiple RMS components. If you are installing any of the RMS components (Database, Batch, or Application) on the same server, they can use the same installer and this step does not need to be repeated.

1. Log into the application server as the user who owns WebLogic Installation files.
2. Create a staging directory for the RMS application distribution (rms19installer.zip).

   Example: /u00/webadmin/media/RMS

3. This location is referred as STAGING_DIR when installing application software.
4. Copy rms19installer.zip to staging directory and extract its contents.

   Example: unzip rms19installer.zip

5. This will create rms/installer subdirectory under STAGING_DIR.

Run the RMS Application Installation

Note: See Appendix: RMS Application Installer Screens for details about the RMS application screens and fields in the installer.

Note: On the installer screen, **"RMS Application Deployment Details"** the default value "Rms" only should be used.

1. Log on to your application server as a user with read and write access to the WebLogic files.
2. Change directories to STAGING_DIR/rms/installer.
3. Set and export the following environment variables.

Variable	Description	Example
JAVA_HOME	Location of a Java 1.8 JDK.	JAVA_HOME= /u00/webadmin/java/jdk1.8 export JAVA_HOME
NLS_LANG	Locale setting for Oracle database client.	NLS_LANG=AMERICAN_AMERICA.AL 32UTF8 export NLS_LANG
J2EE_DOMAIN_HOME	The location of the WebLogic domain (RMSDomain).	J2EE_DOMAIN_HOME=/u00/webadmin/ config/domains/wls_retailRMSDomain export J2EE_DOMAIN_HOME
J2EE_ORACLE_HOME	The location of the WebLogic installation.	J2EE_ORACLE_HOME=/u00/webadmin /products/wls_retail. export J2EE_ORACLE_HOME
DISPLAY	Address and port of X server on desktop system of user running install. Optional when running the application installer	DISPLAY=<IP address>:0.0 export DISPLAY

4. Run the install.sh script to start the installer.

Note: Below are the usage details for install.sh. The typical usage for GUI mode is no arguments.

./install.sh [text | silent]

5. Verify that the installer reports "SUCCESS" for the WLS J2EE Preinstall Check preinstall check. If it reports "FAILED," check for errors in the output under the "Checking environment for Application installation" section, and verify that your environment variables are set properly.
6. Check the Install Application checkbox and proceed with the installation.
7. After the installer is complete, you can check its log file in the "logs" directory: STAGING_DIR/rms/installer/logs/rms-install.<timestamp>.log. RETAIL_HOME/orpatch/logs/detail_log/{javaapp_*}

8. 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`

RMS Application – Post installation Steps

1. As part of 19.0.1 RMS installation, Flashback archive will be enabled on RMS tables.

   Run the below grant script for allowing other DB schema users to query archived historical data from RMS flashback enabled tables.

   a. STAGING_DIR/rms/installer/create_db/grant_flashback_to_user.sql.

   ```
   SQL> @grant_flashback_to_user.sql
   ```

   The following prompts will occur:

   Please enter the schema user name User name for which you need to allow query on archived historical data from RMS Flashback enabled tables.

   b. Run the script using SQL*Plus as RMS schema owner.

   Note: This is a Optional Step. Execute this grant script only to grant enable other schema users to query archived historical data.

Resolving Errors Encountered During Application Installation

If the application installer encounters any errors, it halts execution immediately. You can run the installer in silent mode so that you do not have to re-enter the settings for your environment. See Appendix: Installer Silent Mode in this document for instructions on silent mode.

See Appendix: Common Installation Errors in this document for a list of common installation errors.

Because the application installation is a full reinstall every time, any previous partial installations are overwritten by the successful installation.

Test the RMS Application

After the application installer completes you should have a working RMS application installation. To launch the application, open a web browser and go to **Error! Hyperlink reference not valid.**>: <httpport>/Rms/faces/RmsHome

Examples:

- **Error! Hyperlink reference not valid.**http://apphost:app-server-port/ Rms/faces/RmsHome. You should use a user/password that you built in the previous section of this install guide “Load LDIF files in LDAP”.

  The default, preloaded user supplied in the LDIF scripts for testing this installed application is RMS_ADMIN; the password is the password, which was created in the LDIF file RGBU-oid-create-users.ldif as part of loading LDIF files into the LDAP>.

Single Sign-On

Skip this section if RMS is not used within an Oracle Single Sign-On environment.

Note: This section assumes the Oracle WebLogic Server has already been registered with the Oracle Access Manager (OAM) via the oamreg tool. See the Oracle Single Sign-On (OAM using webgate) documentation for details.

If RMS is being used in an Oracle Single Sign-On environment, then the RMS root context must be protected. Modify the following files.

mod_wl_ohs.conf located in

DOMAIN_HOME/config/fmwconfig/components/OHS/instances/instanceName

LoadModule weblogic_module "${ORACLE_HOME}/ohs/modules/mod_wl_ohs.so"

<IfModule weblogic_module>

</IfModule>

```
<Location /Rms />
        WebLogicHost <weblogichostname>
        WebLogicPort <rmsserverport>
         WLCookieName RMSSESSIONID
        SetHandler weblogic-handler
</Location>
<Location /RetailAppsAdminConsole-RMS />
        WebLogicHost <weblogichostname>
        WebLogicPort <rmsserverport>
         WLCookieName RMSSESSIONID
        SetHandler weblogic-handler
</Location>

<Location /RmsReSTServices />
        WebLogicHost <weblogichostname>
        WebLogicPort <rmsserverport>
         WLCookieName RMSSESSIONID
        SetHandler weblogic-handler
</Location>
```

Adding Logout URI

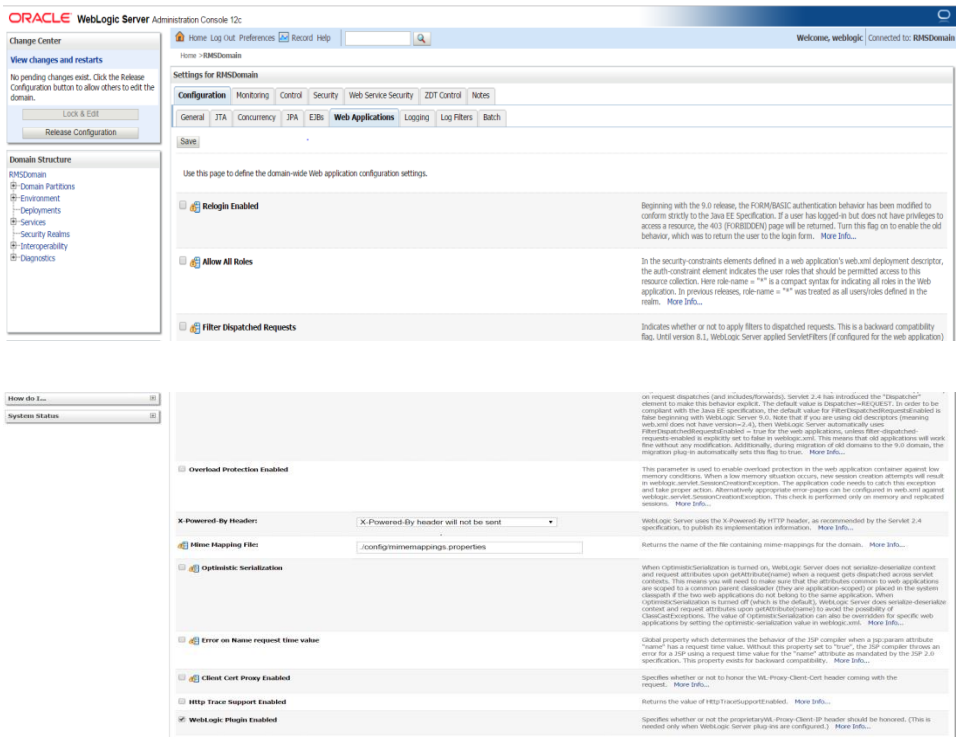
After verifying default authenticator's control flag set correctly as per the OAM documentation, and order of the providers are correct, follow the below steps to configure RMS Application SSO url logout using wlst tool

1. Navigate to < ORACLE_HOME>/oracle_common/common/bin and run wlst.sh

2. Connect RMS domain using admin credentials created during Weblogic domain creation and add OAM SSO Provider.

```
connect('<WEBLOGIC_ADMIN_USERNAME>',
'<<WEBLOGIC_ADMIN_PASSWORD>','t3://<APP_HOSTNAME>:<ADMIN_PORT>')
wls:/crmodsso/serverConfig>domainRuntime()
wls:/crmodsso/serverConfig>addOAMSSOProvider(loginuri="/${app.context}/adf
Authentication",logouturi="/oamsso/logout.html",
autologinuri="/obrar.cgi")
```

3. Login to Weblogic Admin Console and click on Lock & Edit
4. Enable “Weblogic Plugin Enabled” under RMS Domain→Web Applications Tab.



5. Save it and click on Activate Changes
6. Restart RMS Domain servers and verify Application url is logging out properly by displaying OAM page.

Clustered Installations – Post-Installation Steps

If you are installing the RMS application to a clustered environment, there are some extra steps you need to take to complete the installation. In these instructions, the application server node with the ORACLE_HOME you used for the RMS application installation is referred to as *master node*. All other nodes are referred to as *remote nodes*.

> **Note:** Do not copy the entire file from one node to another. Only copy the RMS entries modified in these files by the installer. There is node-specific information in this file that is different between ORACLE_HOME installations.

Copy runtime12.jar to all the remote nodes under domain lib location.
<WEBLOGIC_DOMAIN_HOME>/lib

RMS Reports Copied by the Application Installation

The application installation copies RMS report files to $RETAIL_HOME /reports. These files should be installed into BI Publisher as documented in the RMS Reports chapter of this document.

BDI Job Admin install

1. Create a managed server for the RMS batch admin otherwise consider that this will be installed on the managed server created for Rms application deployment.
   - The managed server should have JRF templates and oracle WSM libraries targeted to it.
   - A simple way to do this is to clone the RMS server and give it different port number.
2. Extract the contents of the BdiEdgeAppJobAdminPak19.0.0ForRms19.0.0_eng_ga.zip from the RMS release in a staging directory.

   Example:

   ```
   cd /u00/webadmin/BDIRMS_EXTRACTOR_INSTALL
   unzip *.zip
   ```

3. Update the /u00/webadmin/BDIRMS_EXTRACTOR_INSTALL /conf/deploymentenvinfo.json file. Update the data source and weblogic server information. See sample below: (NOTE: Managed server entry is same as Admin server in the example.)

   Blue: data source

   Yellow: WebLogic Admin server

   Green: Managed server

```
{
    "BdiJobAdminDeploymentEnvInfo": {

        "DataSourceDef":{
"JobAdminDataSource":{
                "dataSourceName":"RmsJobAdminDataSource",
                "dataSourceClass":"oracle.jdbc.pool.OracleDataSource",
                "dataSourceJndiName":"jdbc/RmsJobAdminDataSource",

"jdbcUrl":"jdbc:oracle:thin:@//<dbhost.example.com>:1522/pdborcl",
                "jdbcUserAlias":"rmsJobAdminDataSourceUserAlias",
                "jdbcUser":"GET_FROM_WALLET",
                "jdbcPassword":"GET_FROM_WALLET"
            },
            "BatchInfraDataSource":{
                "dataSourceName":"BatchInfraDataSource",
                "dataSourceClass":"oracle.jdbc.xa.client.OracleXADataSource",
                "dataSourceJndiName":"jdbc/BatchInfraDataSource",
                "jdbcUrl":"jdbc:oracle:thin:@//
<dbhost.example.com>:1522/pdborcl",
                "jdbcUserAlias":"batchInfraDataSourceUserAlias",
                "jdbcUser":"GET_FROM_WALLET",
                "jdbcPassword":"GET_FROM_WALLET"
            },
            "JobXmlDataSource":{
                "dataSourceName":"JobXmlDataSource",
                "dataSourceClass":"oracle.jdbc.xa.client.OracleXADataSource",
                "dataSourceJndiName":"jdbc/JobXmlDataSource",
                "jdbcUrl":"jdbc:oracle:thin:@//
<dbhost.example.com>:1522/pdborcl",
                "jdbcUserAlias":"jobXmlDataSourceUserAlias",
                "jdbcUser":"GET_FROM_WALLET",
                "jdbcPassword":"GET_FROM_WALLET"        },
        "MiddlewareServerDef":{
```

```
            "JobAdminAppServer": {
                "weblogicDomainName": " RMS_BATCH_DOMAIN ",
                "weblogicDomainHome":
"/u00/webadmin/config/domains/wls_retail/ RMS_BATCH_DOMAIN ",
                "weblogicDomainAdminServerUrl": "t3://APPHOST:7001",
                "weblogicDomainAdminServerProtocol": "t3",
                "weblogicDomainAdminServerHost": "APPHOST",
                "weblogicDomainAdminServerPort": "7001",
                "weblogicDomainAdminServerUserAlias":
"bdiAppServerAdminServerUserAlias",
                "weblogicDomainTargetManagedServerName": "BDI_EX_JOB_SERVER",

                "jobAdminUiUrl":"http://APPHOST:7001/rms-batch-job-admin",
                "jobAdminUiUserGroup":"RmsJobAdminGroup",
                "jobAdminUiUserAlias":"rmsJobAdminUiUserAlias",
                "jobAdminUiUser":"GET_FROM_WALLET",
                "jobAdminUiPassword":"GET_FROM_WALLET",

"jobOperatorUiUserGroup":"RmsJobOperatorGroup",
                "jobOperatorUiUserAlias":"rmsJobOperatorUiUserAlias",
                "jobOperatorUiUser":"GET_FROM_WALLET",
                "jobOperatorUiPassword":"GET_FROM_WALLET",

                "jobMonitorUiUserGroup":"RmsJobMonitorGroup",
                "jobMonitorUiUserAlias":"rmsJobMonitorUiUserAlias",
                "jobMonitorUiUser":"GET_FROM_WALLET",
                "jobMonitorUiPassword":"GET_FROM_WALLET"
            }
        },
        "JobAdminApplication":{
            "appName":"rms",
            "JobAdminAppUses":[
                "JobAdminDataSource",
                "JobAdminAppServer"
            ]
        }
    }
}
```

4. Compile and Deploy the RMS bdi application using admin deployer script.

   Example:

```
cd <root-directory>/rms-home/bin
./bdi-job-admin-deployer.sh -setup-credentials -deploy-job-admin-app

Output:
/rms-home/bin> ./bdi-job-admin-deployer.sh -setup-credentials -deploy-job-
admin-app
Extracting jars from jps-wallet-all.
log4j:WARN No appenders could be found for logger
(com.oracle.retail.integration.common.security.credential.CredentialStoreManag
er).
log4j:WARN Please initialize the log4j system properly.

Credential required for weblogicDomainAdminServerHost(msp00abx)
weblogicDomainAdminServerPort(7001):
Enter username for alias (bdiAppServerAdminServerUserAlias):weblogic
Enter Password: <weblogic-password>

Credential required for jobAdminUiUrl(http://APPHOST:7001/rms-batch-job-
admin):
Enter username for alias (rmsJobAdminUiUserAlias):rmsbatchadmin
Enter Password: <rms-batch-admin-password>

Credential required for jobOperatorUiUrl(http:// APPHOST:9010/rms-batch-job-
admin):
Enter username for alias (rmsJobOperatorUiUserAlias): rmsbatchoperator
Enter Password: <rms-batch-opr-password>

Credential required for jobMonitorUiUrl(http:// APPHOST:9010/rms-batch-job-
admin):
```

```
Enter username for alias (rmsJobMonitorUiUserAlias): rmsbatchmonitor
Enter Password: <rms-batch-mon-password>

Credential required for dataSource(jdbc/RmsJobAdminDataSource)
jdbcUrl(jdbc:oracle:thin:@//<dbhost.example.com>:1521/APPDBSID):
Enter username for alias (rmsAppDataSourceUserAlias):rms-bdi-int-schema
Enter Password: <password>
Credential required for BatchInfraDataSource
dataSource(jdbc/BatchInfraDataSource) jdbcUrl(jdbc:oracle:thin:@//
<dbhost.example.com>:1521/APPDBSID):
Enter username for alias (batchInfraDataSourceUserAlias):batchinfra_wls
Enter Password: <password>

Credential required for JobXmlDataSource dataSource(jdbc/JobXmlDataSource)
jdbcUrl(jdbc:oracle:thin:@// <dbhost.example.com>:1521/APPDBSID):
Enter username for alias (jobXmlDataSourceUserAlias): rms01app
Enter Password: <password>
```

5. When finished, bounce the WebLogic domain, and launch the application using the path (**Error! Hyperlink reference not valid.**).

5

Oracle Analytics Server Configuration for RMS Reports

RMS 15.0.3.1 reports supports OAS Publisher 5.5. RMS Reports are copied to RETAIL_HOME /reports during the application installation.

> **Note:** In the following sections, the Oracle OAS 5.5 installation steps are for demo purpose only. Refer to the *Installing and configuring Oracle Analytics Server 5.5.0* for more information.

OAS Server Component Installation Tasks

Oracle OAS Publisher is used as the main RMS, RWMS, REIM, and SIM reporting engine and can be used in conjunction with external printing solutions like label printing. This section describes the installation of Oracle OAS Publisher as a server application within WebLogic 12.2.1.4.0. One deployment of OAS Publisher can be used for any of the RMS, RWMS, REIM, and SIM reports.

When installing OAS Publisher, refer to the appropriate Fusion Middleware guides for the installation of the product in a WebLogic server environment.

Installation Process Overview

Installing the OAS Publisher server as a standalone web application in a WebLogic server involves the following tasks:

1. Install Oracle Analytics Server under an existing WebLogic Server (WLS) 12.2.1.4 infrastructure home.
2. Configure Oracle Analytics Server, create default OAS Domain and configure component “OAS Publisher” only.
3. Select the OAS Platform schema for update of the ORACLE 19c DB.
4. Configure ports and document and test the URL’s that are created.

The following post-installation tasks are involved once OAS Publisher has been installed:

5. Configure the OAS Publisher repository. Set security model, add users, assign roles, add reports, add printers, set repository path, set data source, etc.
6. Set up and copy the RMS OAS Publisher Report Templates produced for RMS.
7. Set up for the RMS application specific configuration files to integrate OAS Publisher with the RMS online app.

Install Oracle Analytics Server 5.5

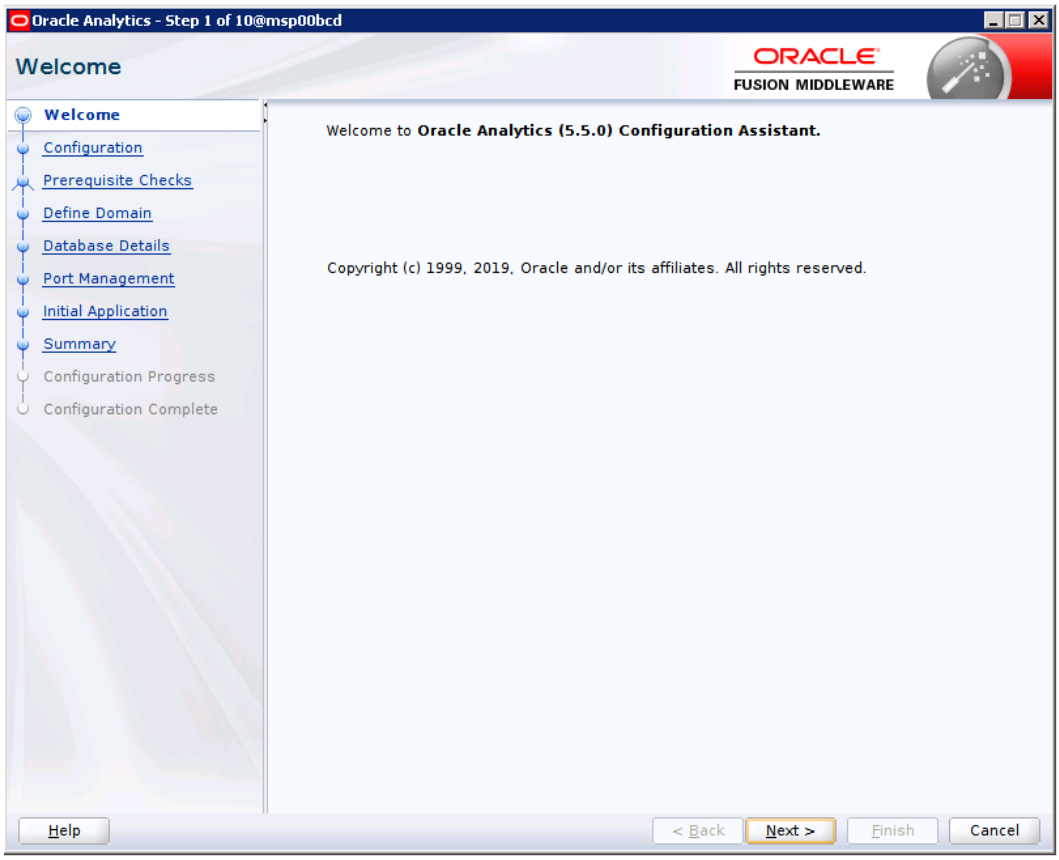
1. Install JDK 1.8 as per product certification

   For Example, /u01/product/fmw/wls_oas
2. Export your DISPLAY.

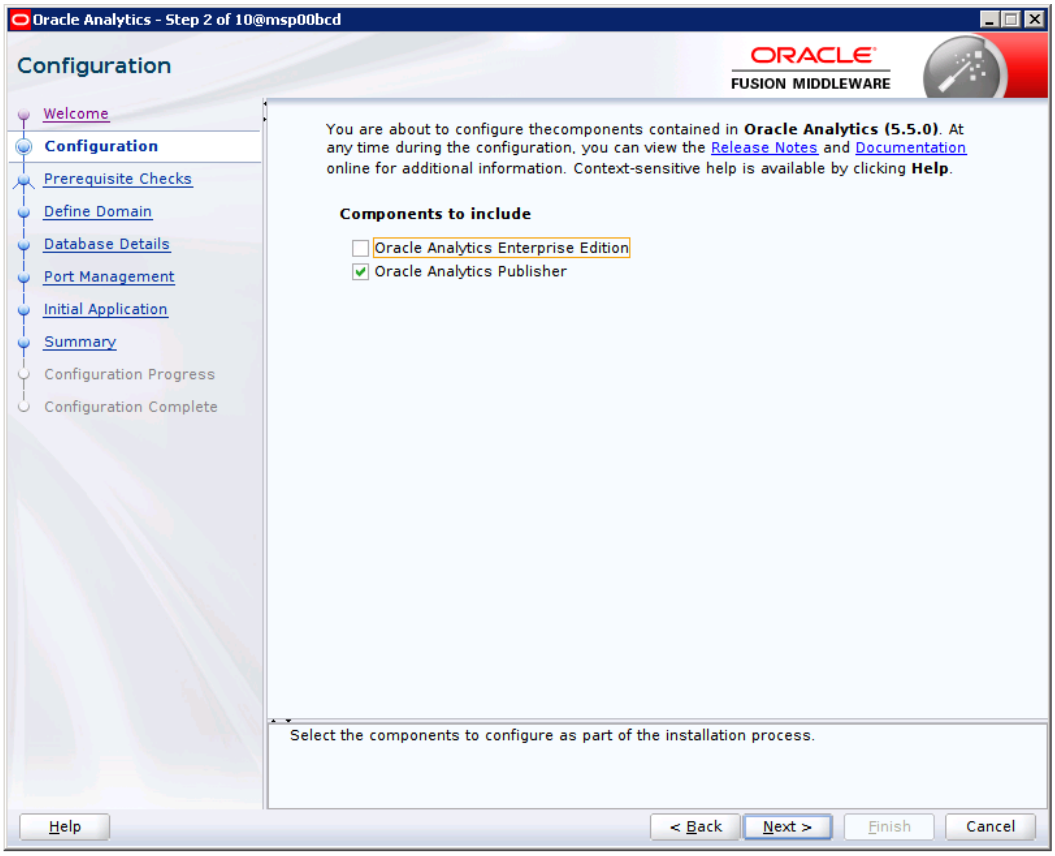
   Example: export DISPLAY=10.141.10.110:0.0

3. Install Oracle Fusion Middleware Infrastructure in OAS Product Home (fmw_12.2.1.4.0_infrastructure_generic.jar)
   JDK_HOME/bin/java -jar fmw_12.2.1.4.0_infrastructure.jar
4. Install Oracle Analytics Server 5.5
   JDK_HOME/bin/java -jar Oracle_Analytics_Server_5.5.0.jar
   For more details, refer *Installing and Configuring Oracle Analytics Server 5.5.0 F27232-04 Guide*
5. Launch Oracle Analytics configuration Assistant by executing
   - Go to $OAS HOME/bi/bin

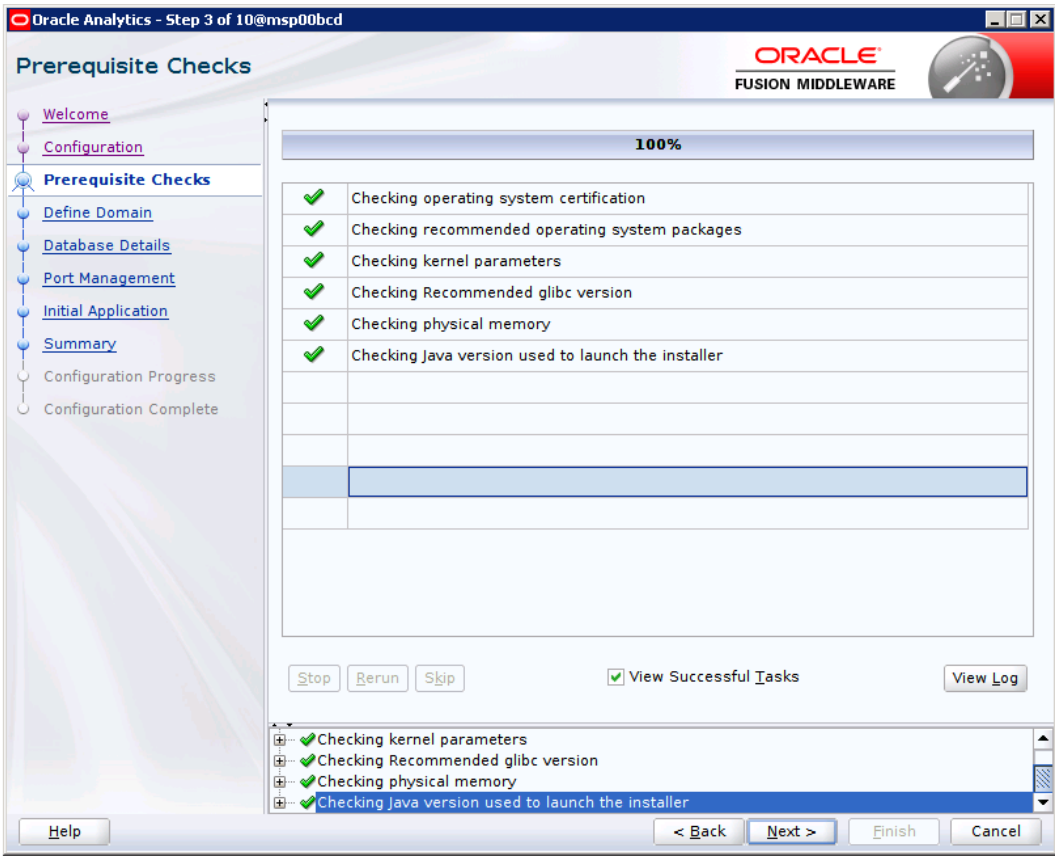
   Example: /u01/product/fmw/wls_oas/bi/bin
   ```
   Start configuring domain:  ./config.sh
   ```
6. Click **Next**.



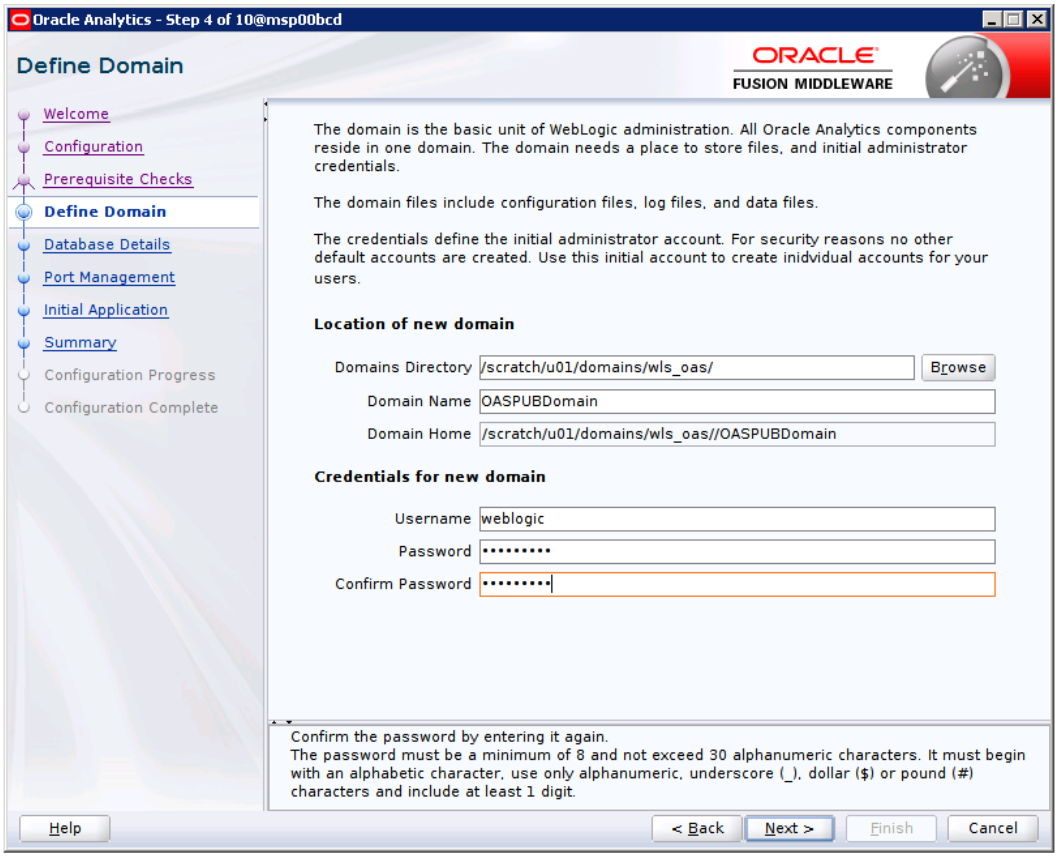
7. Select Oracle Analytics Publisher


Oracle Analytics - Step 2 of 10@msp00bcd
Configuration
ORACLE
FUSION MIDDLEWARE
Welcome
Configuration
Prerequisite Checks
Define Domain
Database Details
Port Management
Initial Application
Summary
Configuration Progress
Configuration Complete
You are about to configure thecomponents contained in Oracle Analytics (5.5.0). At any time during the configuration, you can view the Release Notes and Documentation online for additional information. Context-sensitive help is available by clicking Help.
Components to include
Oracle Analytics Enterprise Edition
Oracle Analytics Publisher
Select the components to configure as part of the installation process.
Help
< Back
Next >
Finish
Cancel

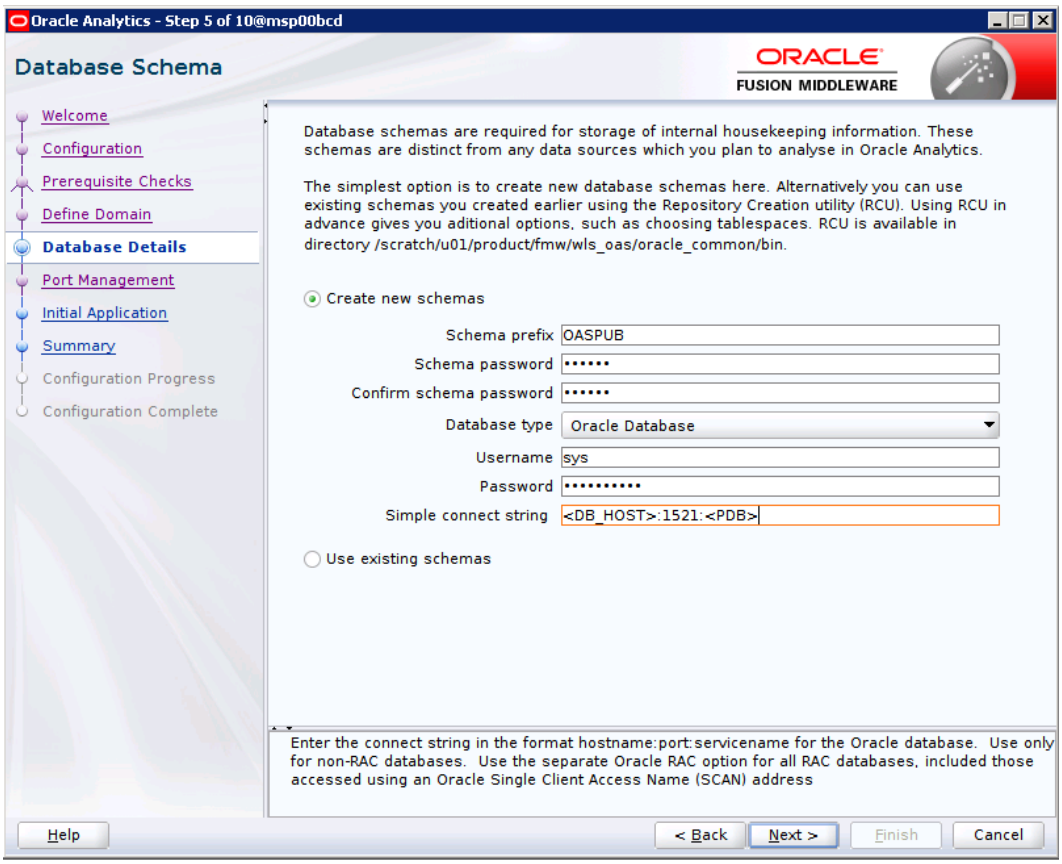
8. Click **Next**.



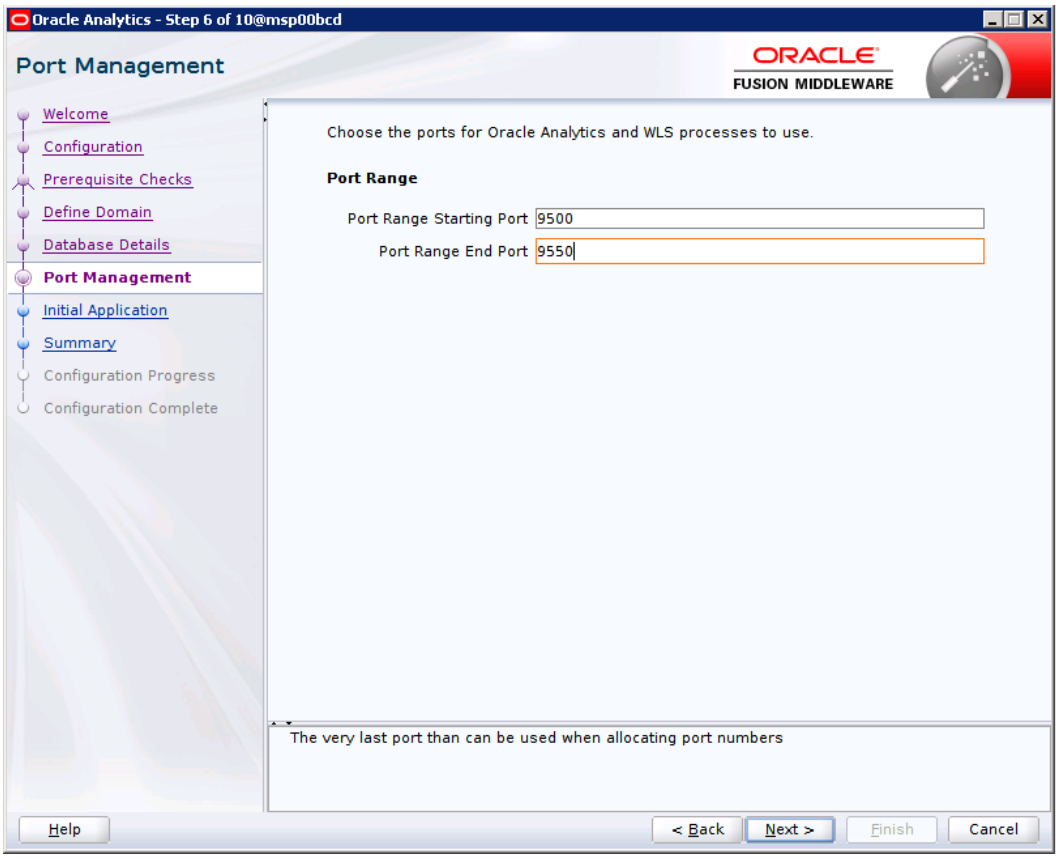
9. Provide Domain Directory and Name. Create Admin Username and Password



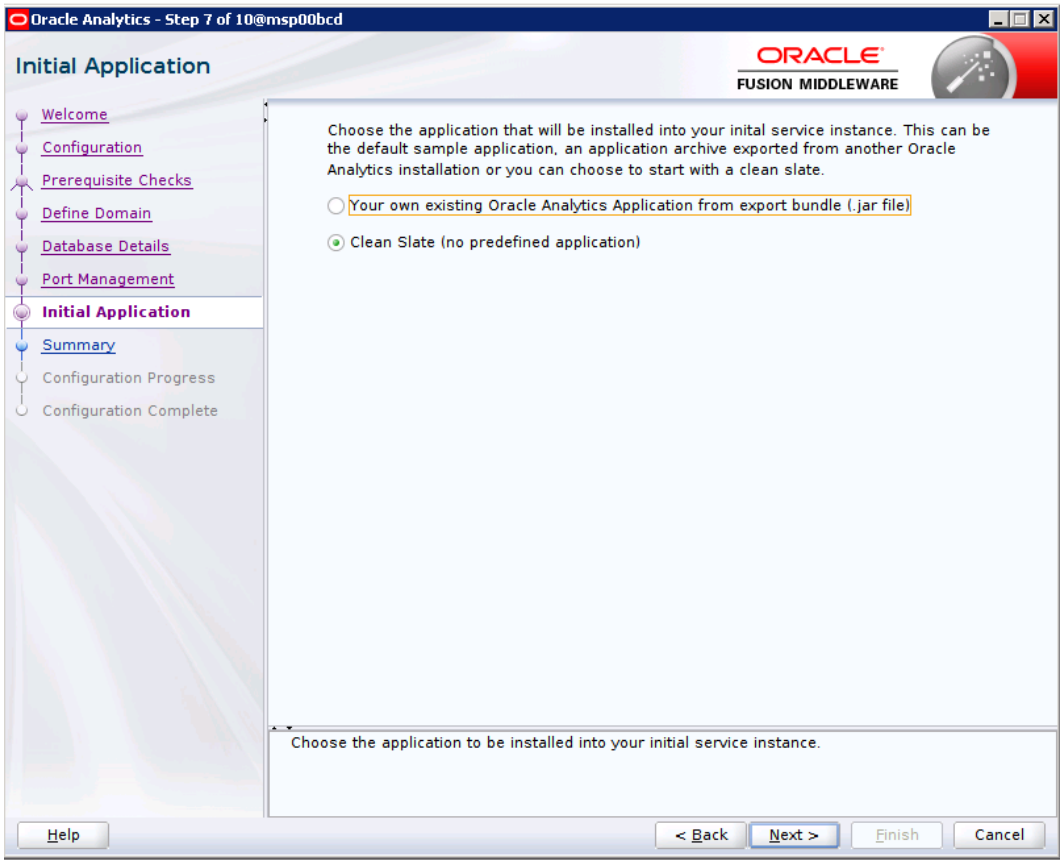
10. Select Create New schemas. Enter your Oracle Database information. Simple connect string like <DBHOST>:1521:<PDB>



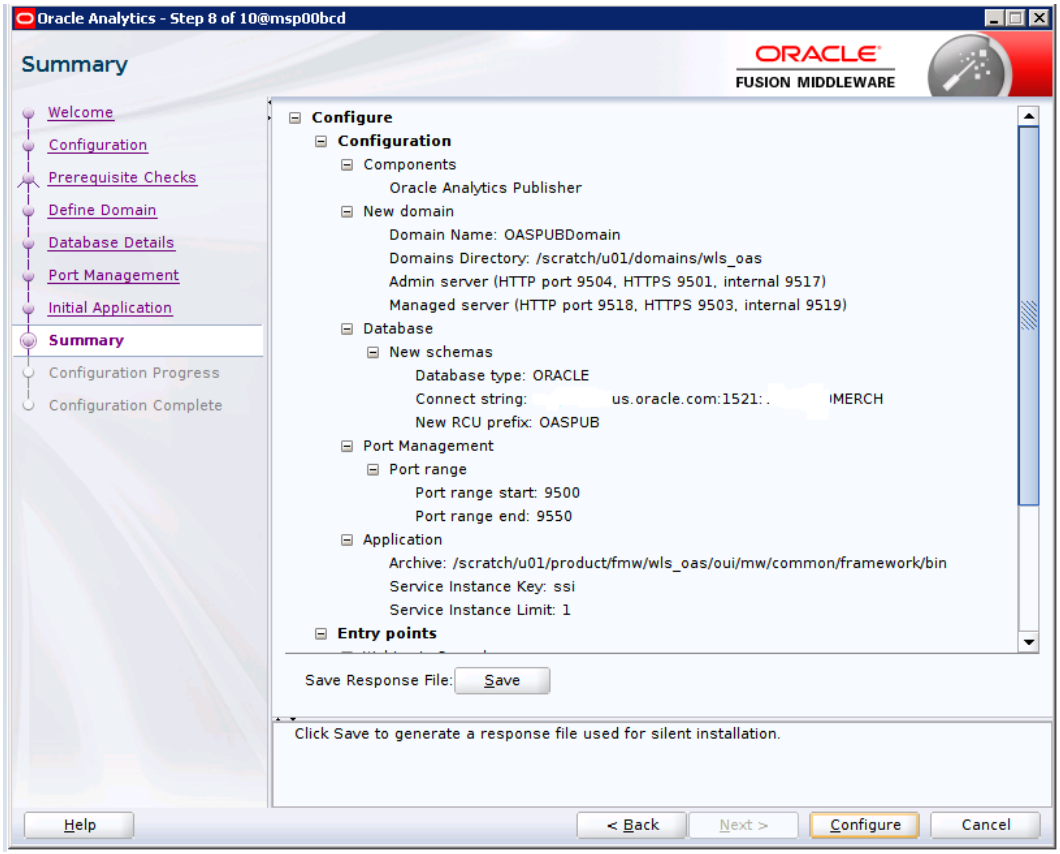
11. Choose Port Range



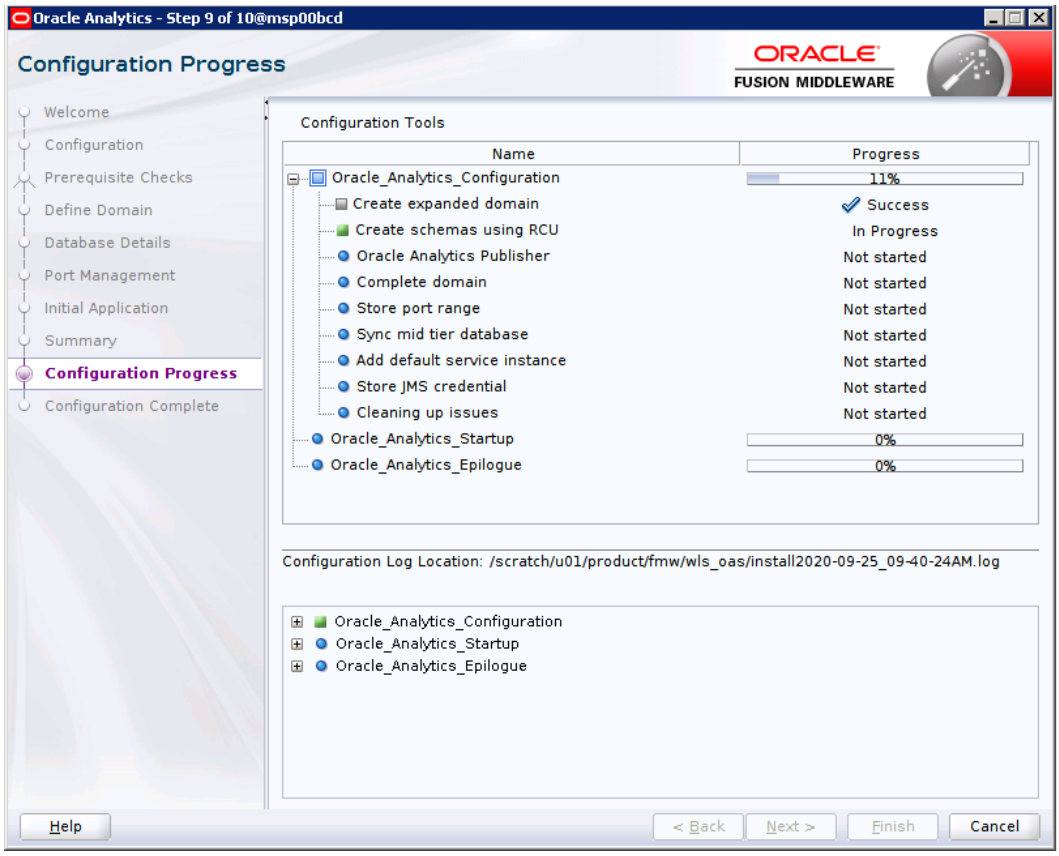
12. Select Clean Slate

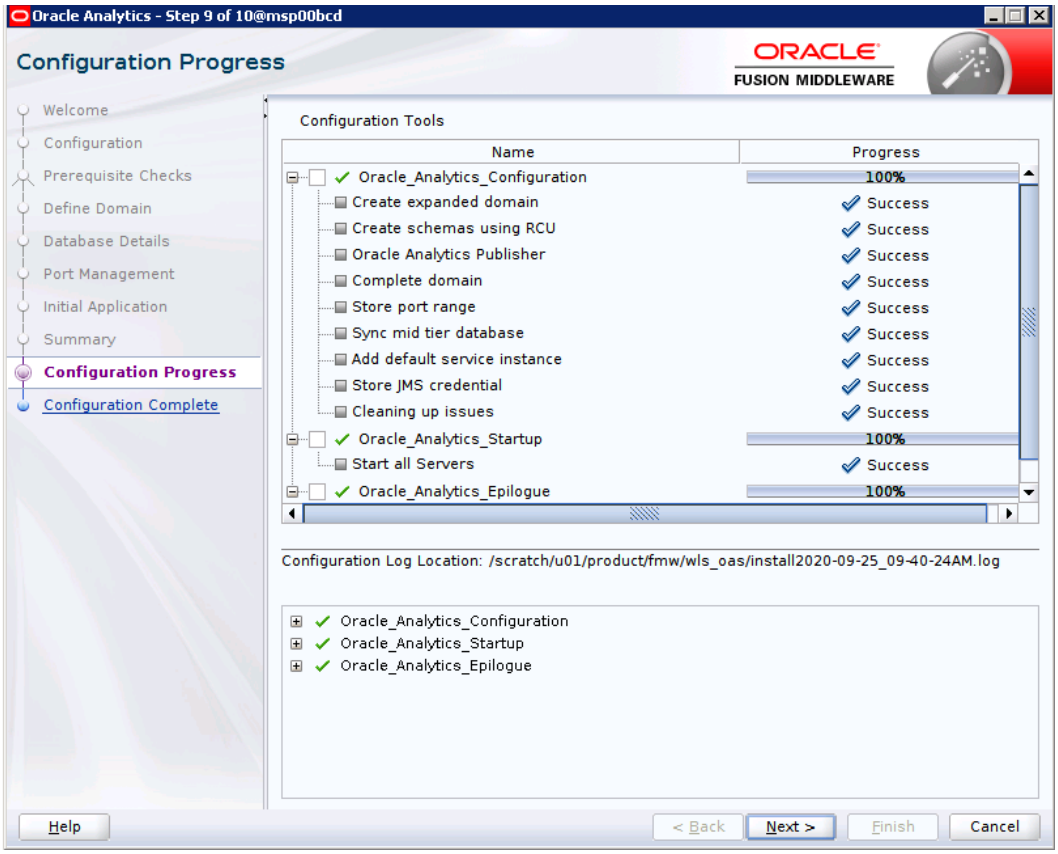


13. Click on Configure

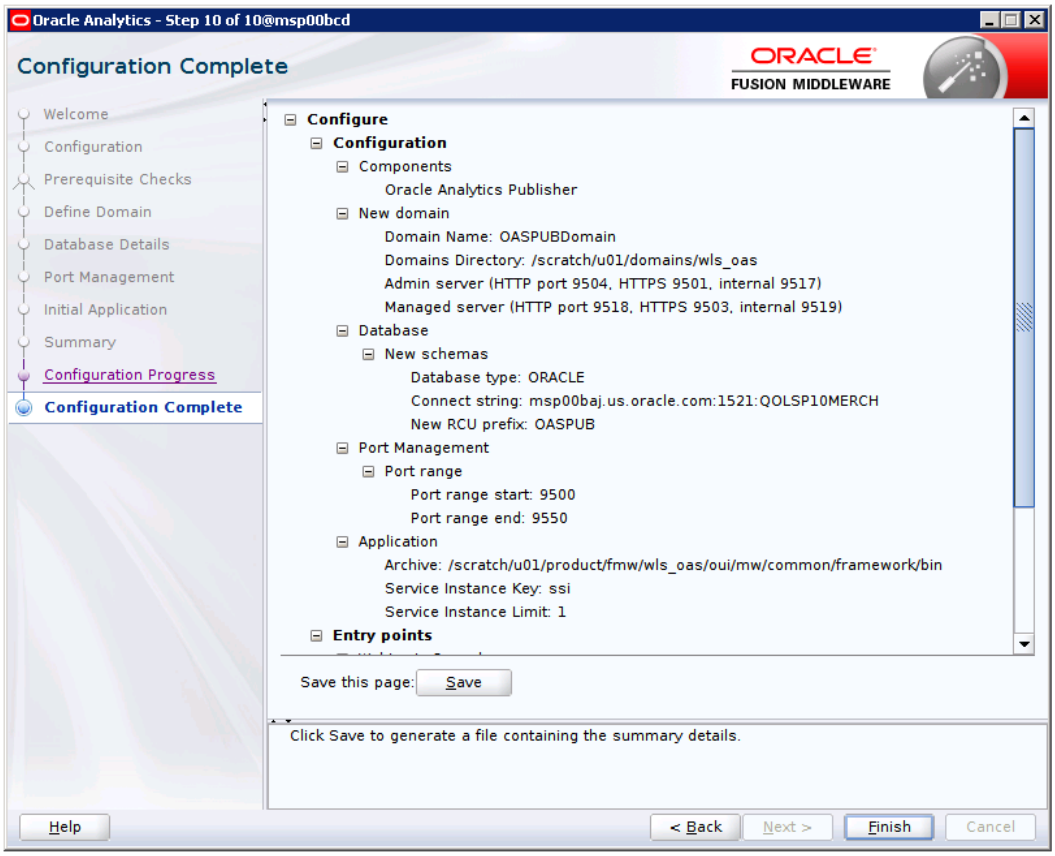


14. Click Next


Oracle Analytics - Step 9 of 10@msp00bcd
Configuration Progress
ORACLE
FUSION MIDDLEWARE
Welcome
Configuration
Prerequisite Checks
Define Domain
Database Details
Port Management
Initial Application
Summary
Configuration Progress
Configuration Complete
Configuration Tools
Name
Progress
Oracle_Analytics_Configuration
11%
Create expanded domain
Success
Create schemas using RCU
In Progress
Oracle Analytics Publisher
Not started
Complete domain
Not started
Store port range
Not started
Sync mid tier database
Not started
Add default service instance
Not started
Store JMS credential
Not started
Cleaning up issues
Not started
Oracle_Analytics_Startup
0%
Oracle_Analytics_Epilogue
0%
Configuration Log Location: /scratch/u01/product/fmw/wls_oas/install2020-09-25_09-40-24AM.log
Oracle_Analytics_Configuration
Oracle_Analytics_Startup
Oracle_Analytics_Epilogue
Help
< Back
Next >
Finish
Cancel

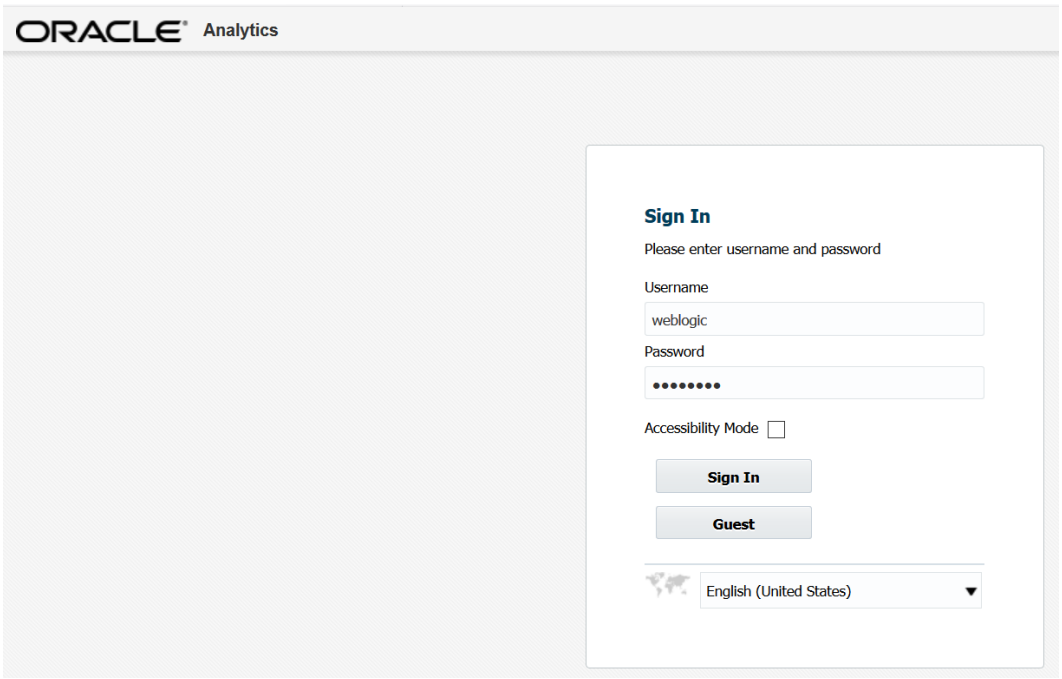


15. Click Finish

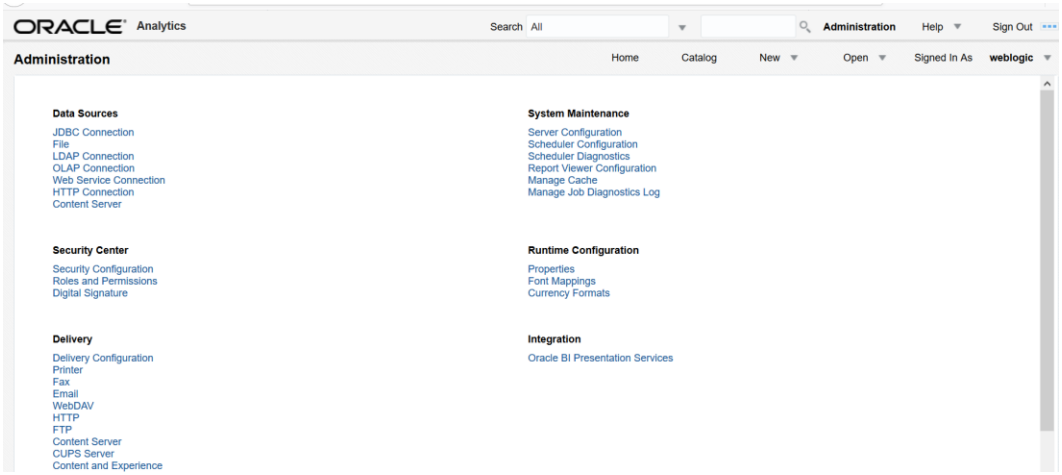


Post install steps for OAS5.5

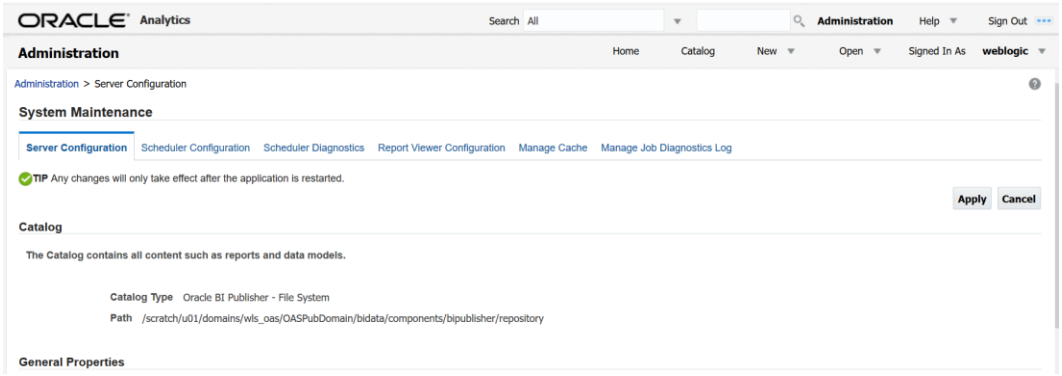
1. Test your OAS Publisher installation, Get the xmlpserver url from your Installation Screen and launch xmlpserver. Login with the credentials you entered in your Oracle AS configuration (weblogic / password). Example URL:http://[OAS_host]:[OAS server_port]/xmlpserver



2. After sign on, select “Administration”.



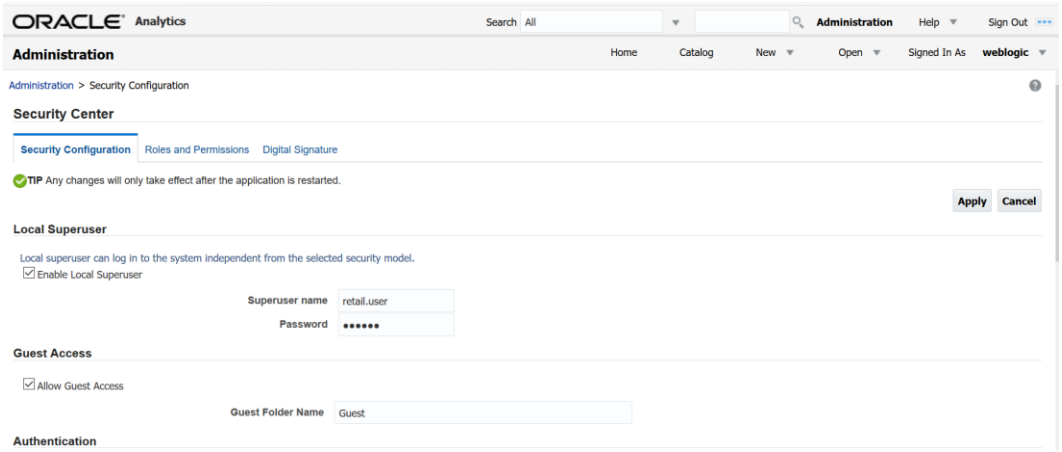
3. On the System Maintenance Section, click **Server Configuration**.



4. On this screen - In the Server Configuration Folder section, enter the path to your repository. On the Catalog section enter Catalog Type: Oracle AS Publisher – File System from the drop down menu.
   - This is the path you entered in the Configuration Section and Catalog Section:

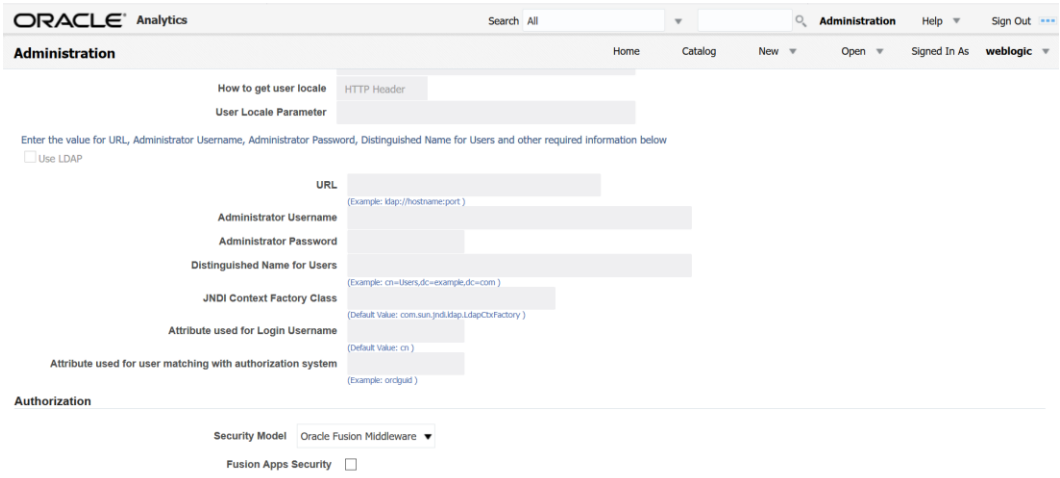
   `Example: $<OAS_DOMAIN_HOME>/config/bipublisher/repository`

   -
5. Click **Apply**.
6. Click Administration link at top of screen.
7. Click on the Security Configuration link under the Security Center to setup a super user and apply the BI Publisher security model.



8. Enable a Superuser by checking the “Enable Local Superuser” box and by entering name and password on the corresponding fields on this screen.
9. Mark “Allow Guest Access” check box. Enter “Guest” as Guest Folder Name.
10. Click **Apply**.

11. Scroll down the screen and locate the Authorization section:



12. Select Oracle Fusion Middleware from the Security Model list.
13. Click **Apply**.
    - Leave OAS Publisher up while completing the next section.

Installing the RMS OAS Publisher Templates

In this section we will outline how the RMS report templates are installed into the appropriate OAS server repositories.

Example: <OAS_DOMAIN_HOME>/config/bipublisher/repository

Report files are placed by the application installation in the directory - "`RETAIL_HOME /reports`" and have to be copied into a newly created directory within OAS Publisher repository Guest Reports directory.

1. Create the directory to hold the reports under <AS_REPOSITORY>

   ```
   mkdir <BI_REPOSITORY>/Reports/Guest/RMS
   ```

2. Change directory to the RETAIL_HOME /reports/RMS created during the application install. This directory contains subdirectories whose names reflect the names of report templates provided with RMS.
3. Copy each report directory into the directory created above

   For example,

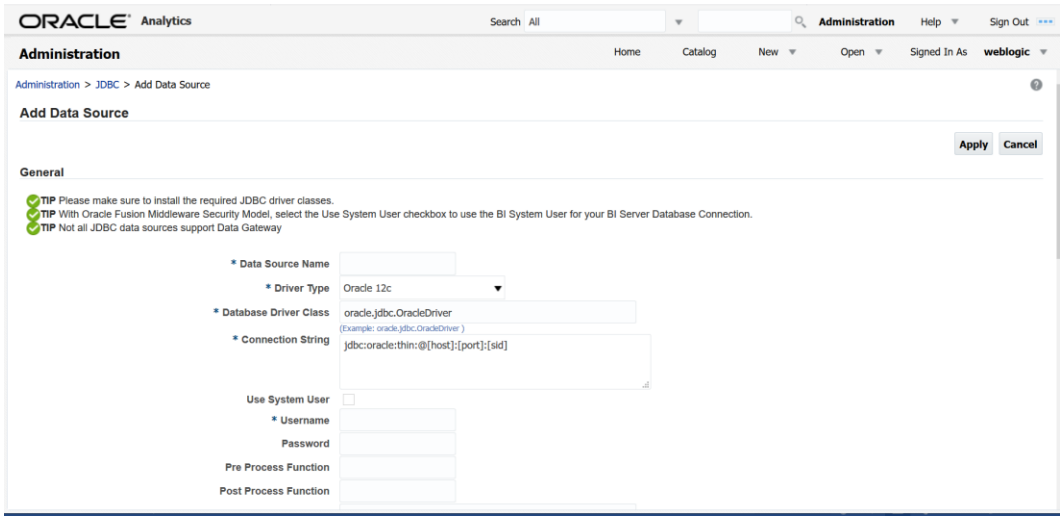
   ```
   cp -R *  <BI_REPOSITORY>/Reports/Guest/RMS
   ```

Configuring the RMS JDBC connection

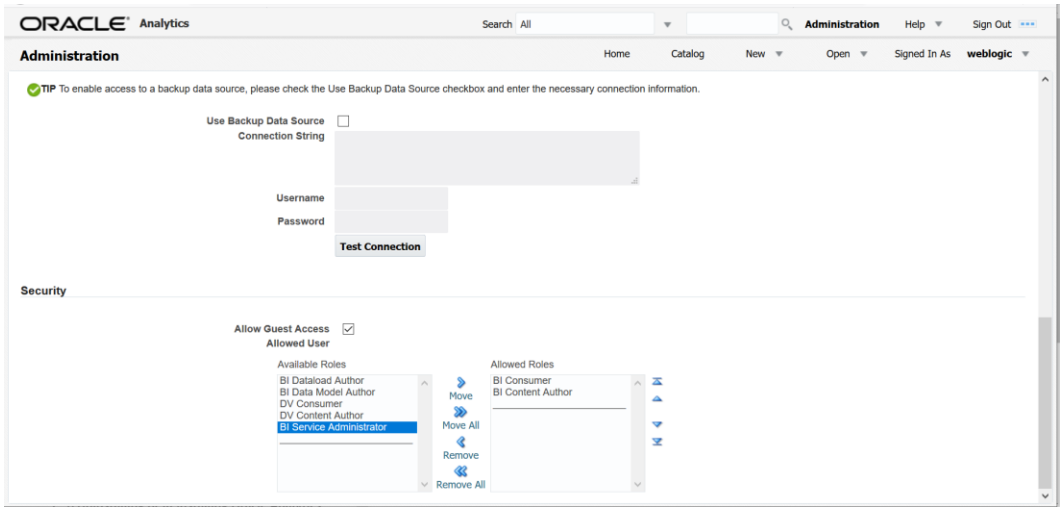
Follow the below steps to configure a JDBC connection for the RMS Data Source, which is required for RMS reports.

1. If not still logged into OASPublisher:
   - Login with the credentials you entered in your Oracle AS configuration. (weblogic / password)
2. If the server was restarted:
   - Login as the super user that was created in prior security setup steps.
3. Click the **Administration** link at top of screen
4. Select the JDBC Connection hyperlink in the Data Sources lists.

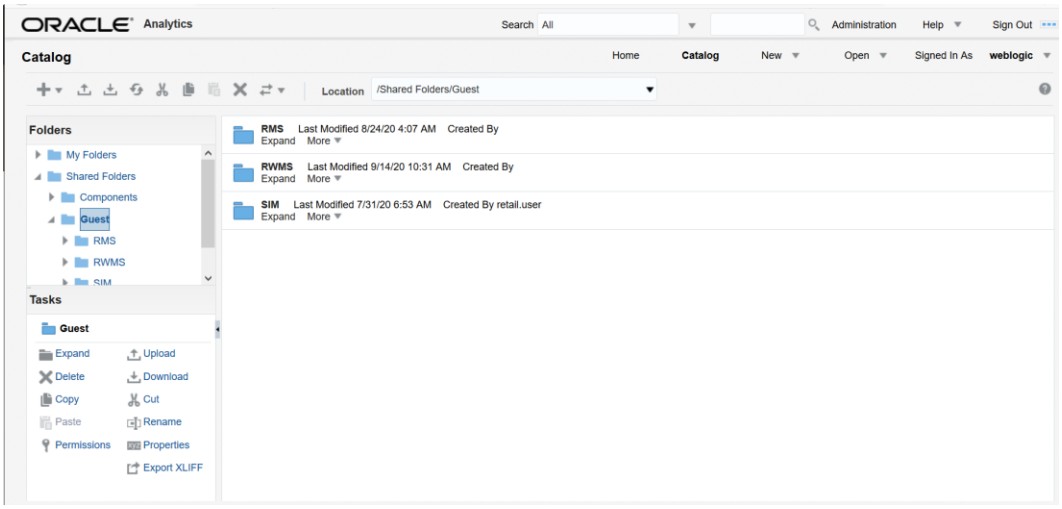
5. Click the **Add Data Source** button.



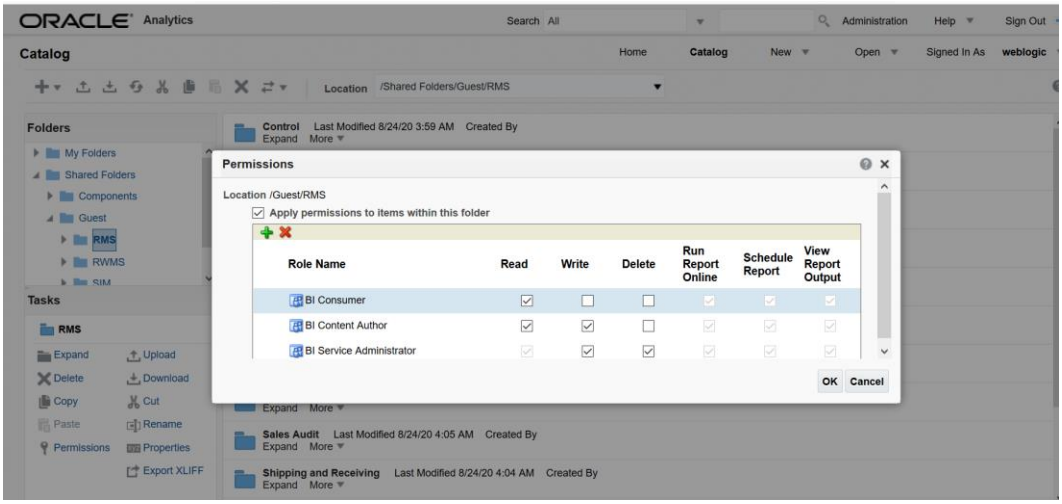
6. Enter the appropriate details for the RMS data source. Click Test Connection to test the connection on the screen once the data is entered.
   - Data Source Name: RMS
     - Must be RMS due to code dependencies.
   - Driver type is ORACLE 12c
   - Database driver class should be oracle.jdbc.OracleDriver.
   - Connection string is similar to this example:
     - Pluggable: jdbc:oracle:thin:@dbhostname:1521/servicename
     - Non- Pluggable ¡dbc:oracle:thin:@dbhostname:1521:SID
   - Enter the username and password for the RMS application user's data source. Click Test Connection to test the connection on the screen once the data is entered.
7. Scroll to the bottom of the screen and check the Allow Guest Access check box. Click **Apply**.



8. Click Catalog link at the top of the screen – and then click the Guest folder on the left so that it is highlighted.



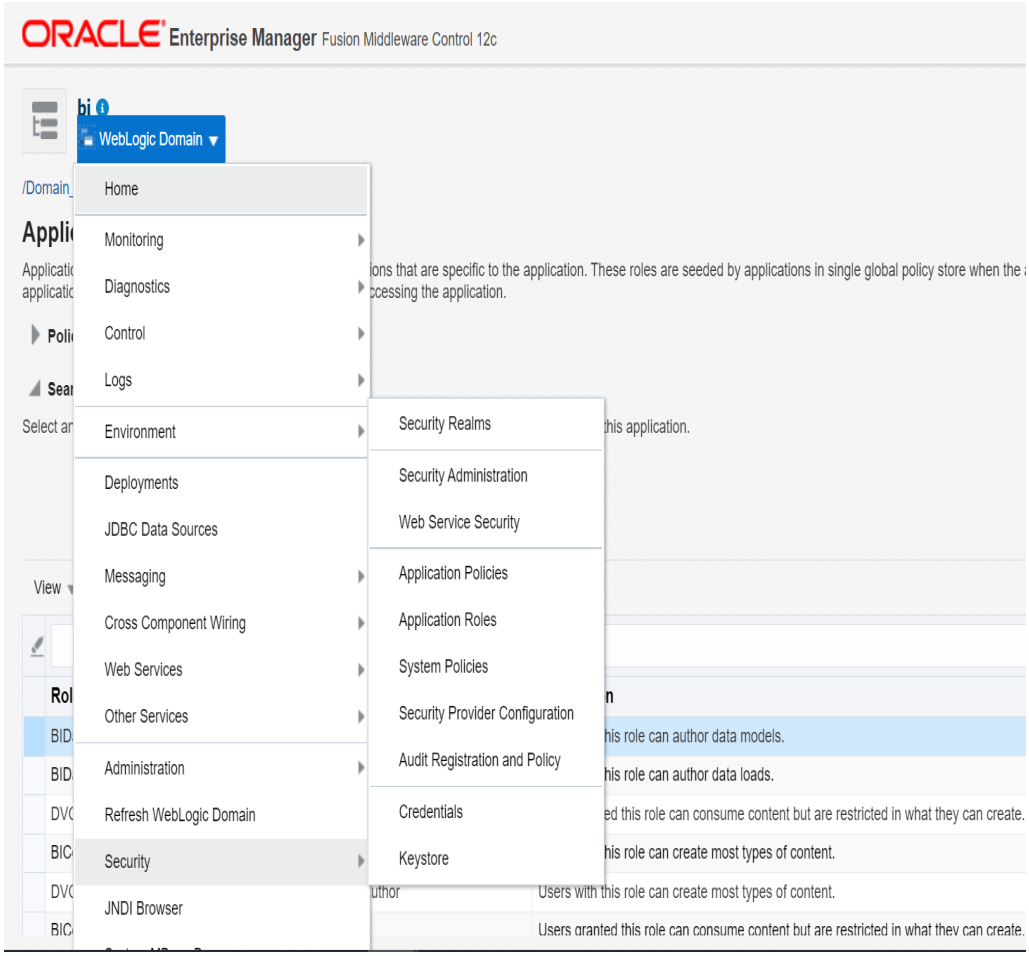
9. Click the Permissions link on the lower left of the screen.



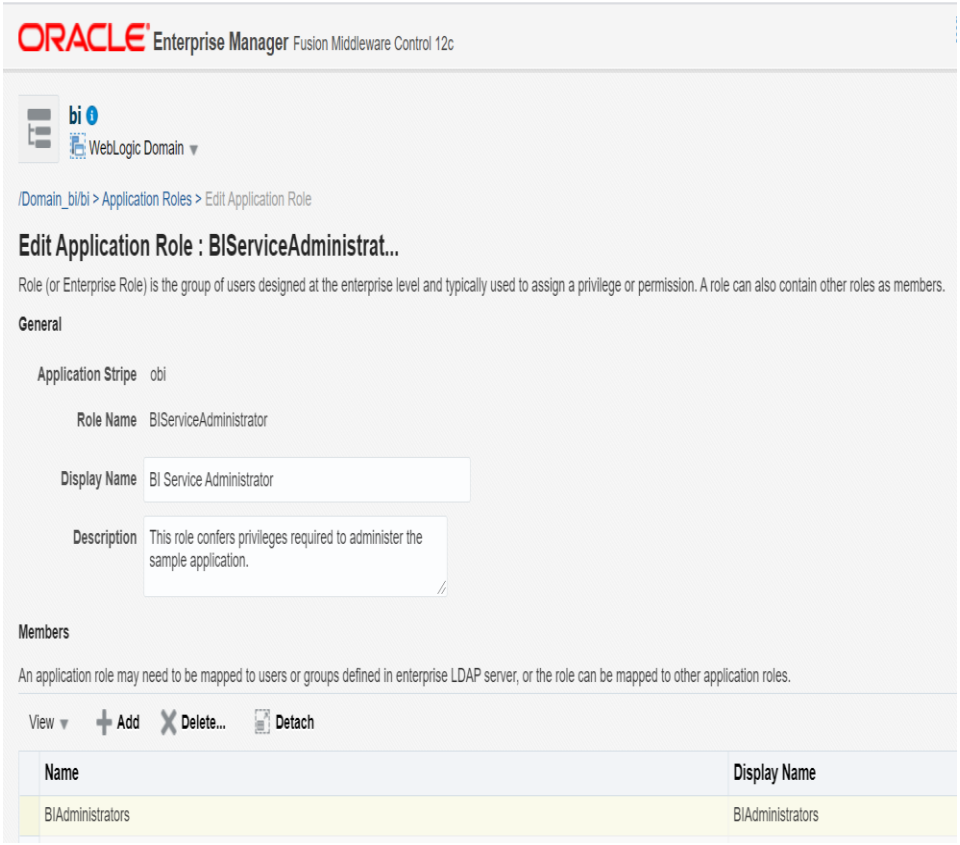
10. Click **OK**.
11. Restart OAS Publisher domain.
12. Verify application reports by logging through applications (For Example: RMS, RESA, SIM, etc...) and It may prompt again for login credentials to display reports. For SSO configured applications, reports should be displayed through application user logins if those users/groups added into BI Application roles through EM console.

Map Application LDAP Users/Groups to BI Application roles using EM console

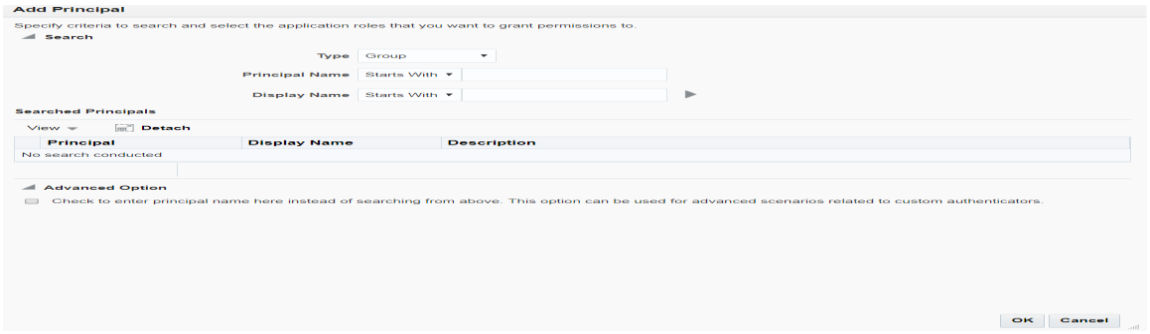
1. Login to EM console with Weblogic admin credentials and Navigate to Weblogic Domain→Security→Application Roles.



2. Select BI ServiceAdministrator role name and click **Edit**.
3. Click the **Add** button to add group members.

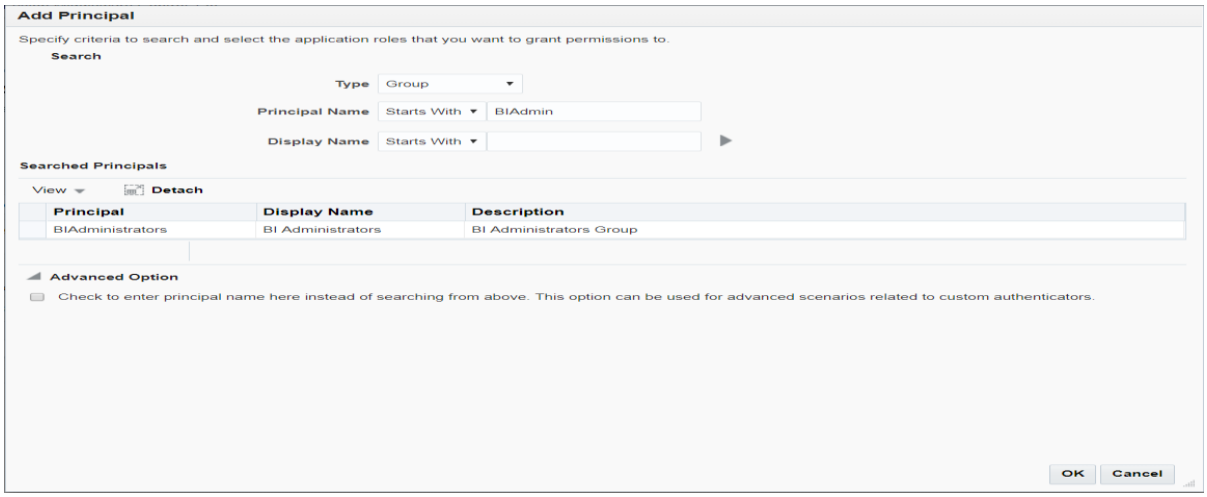


4. Select Type value as Group.



5. Search for BI Administrator group by providing Principal name and click the **Search** button.

> **Note:** BI Administrator group with application users like rms_admin, resa_admin, retail.user should be available in weblogic Security Realm.



6. Select “BIAdministrators” in Searched Principals.
7. Click **OK**.
8. Save the changes and click **OK**

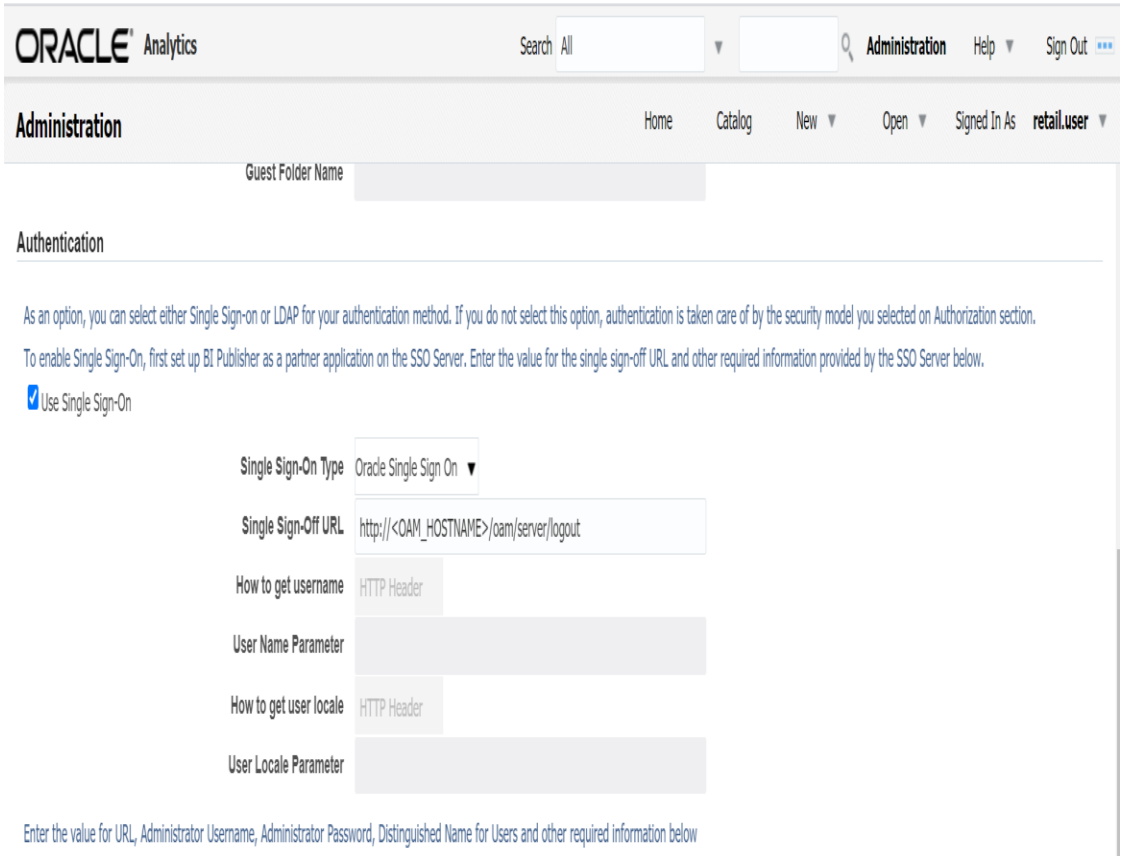
   Restart BIP domain using stop.sh and start.sh scripts available under <*OAS_DOMAIN*>/bitools/bin directory.

   ```
   ./stop.sh
   ./start.sh
   ```

Post Configurations for SSO setup

> **Note:** This procédure is only needed if OAS application setting up using Single Sign On (SSO) authentication. This can be skipped if SSO is not going to be used. The Oracle Access Manager must be configured and the Oracle http server (Webtier and webgate) must be registered into the Oracle Access Manager.

1. Login to xmlpserver and Navigate to Security Configuration link under the Security Center to configure SSO configurations like Single Sign-on Type, Sign-Off url, User Name Parameter(For example, OAM_REMOTE_USER) and Locale Language.



2. Restart OAS domain using stop.sh and start.sh scripts available under <*OAS_DOMAIN*>/bitools/bin directory.

```
./stop.sh
./start.sh
```

6

Data Access Schema Implementation – Optional

Data Access Schema

Data Access Schema (DAS) exposes a subset of core Merchandising/Sales Audit data to external applications via database replication. DAS allows these applications read only access Merchandising/Sales Audit data as they need it. The use of a separate schema insulates core Merchandising/Sales Audit processes from outside requests for information. DAS user must be created in a different database from Merchandising/Sales Audit.

Prepare DAS Database

To prepare the DAS database perform the following steps:

1. DAS Patches and updates are available from My Oracle Support.

   Download 19.0.1 DAS patch from My Oracle Support.
2. Log into the DAS database server as a user that can connect to the DAS database.
3. Create a staging directory for the DAS implementation.
4. Copy the DAS patch Das19.0.00.zip file from the RMS 19.0.1 release to the staging directory.
5. Change directories to DAS_STAGING_DIR and extract the Das19.0.00.zip file. This creates a Das19.0.00/ subdirectory under DAS_STAGING_DIR. The sql scripts under Das19.0.00/ will be used for setting up DAS schema.
6. Create DAS tablespaces:
   a. Modify DAS_STAGING_DIR/create_das_tablespaces.sql to use the correct size values. The DAS schema tablespaces require approximately 50% of the capacity for the corresponding Merchandising/Sales Audit tablespaces, so set these values to half the size allotted for each tablespace in Merchandising/Sales Audit.
   b. As SYSDBA in the PDB, run DAS_STAGING_DIR/create_das_tablespaces.sql. The following tablespaces are created:
      RETAIL_DATA
      RETAIL_INDEX
      LOB_DATA
      USERS
   c. If you hold the Advanced Security Options license, you can create the encrypted tablespaces with TDE tablespace encryption to protect PII data at rest.
      i. Follow the instructions in Appendix: Configure a Wallet for Tablespace Encryption to create a Wallet.
      ii. Modify the DAS_STAGING_DIR/create_encrypted_das_tablespaces_TDE.sql to use the correct size values. The DAS schema tablespaces require approximately 50% of the capacity for the corresponding Merchandising/Sales Audit tablespace, so set these values to half the size allotted for each tablespace in Merchandising/ Sales Audit.
      iii. As SYSDBA in the PDB, run DAS_STAGING_DIR/create_encrypted_das_tablespaces_TDE.sql. The following tablespaces are created:

ENCRYPTED_RETAIL_DATA
ENCRYPTED_RETAIL_INDEX

d. If you do not hold an Advanced Security Option license or do not wish to use TDE tablespace encryption, create the remaining tablespaces without encryption.

i. Modify the DAS_STAGING_DIR/create_encrypted_das_tablespaces_no_TDE.sql to use the correct size values. The DAS schema tablespaces require approximately 50% of the capacity for the corresponding Merchandising/Sales Audit tablespace, so set these values to half the size allotted for each tablespace in Merchandising/Sales Audit.

ii. As SYSDBA in the PDB, run the DAS_STAGING_DIR/create_encrypted_das_tablespaces_no_TDE.sql.
The following tablespaces are created:
ENCRYPTED_RETAIL_DATA
ENCRYPTED_RETAIL_INDEX

7. Create the DAS schema owner. Connect to the DAS PDB as SYSDBA and run the following:

a. Run the DAS_STAGING_DIR/create_das_user.sql. The following prompts will occur:

– Schema Name: The name of the DAS schema owner (for example; RMS19DAS).

– Password: Password for the DAS schema owner.

– Temp Tablespace: Temporary tablespace for the DAS schema owner.

```
SQL> @create_das_user.sql
Please enter the schema owner name: <schema_owner>
Please enter the password for the schema owner:
Please enter the temporary tablespace for the schema owner: TEMP
```

8. Create flashback tablespace and flashback archive. Grant flashback archive privileges to the DAS schema owner.

a. Modify DAS_STAGING_DIR/create_das_flashback_tablespace.sql to use the correct size values and data file path. Modify the flashback archive retention as desired.

b. As SYSDBA in the PDB, run DAS_STAGING_DIR/create_das_flashback_tablespace.sql. The script will prompt for the DAS schema owner name.
The following tablespaces are created:
FLASHBACK_DATA

9. Create DAS views and sequences:

a. As the new DAS schema owner, run DAS_STAGING_DIR/create_das_views.sql

The created views will be invalid because the dependent tables do not exist yet. After the tables are imported via Oracle Datapump, the views must be recompiled.

```
SQL> @create_das_views.sql
CREATING VIEW 'POVIEW_ORDLOC_VIEW'…
```

b. As the new DAS schema owner, run create_das_views_rpm.sql

```
SQL> @create_das_views_rpm.sql
CREATING VIEW 'DAS_WV_RPM_CLEARANCE'…
```

c. As the new DAS schema owner, run create_das_views_reim.sql

```
SQL> @create_das_views_reim.sql
CREATING VIEW 'DAS_WV_IM_AP_STAGE_DETAIL'…
```

Initialize DAS Tables

The Oracle Database datapump utility can be utilized to initialize objects in DAS. An export of objects taken from the main Merchandising/Sales Audit schema containing the subset of DAS objects should be used for DAS table creation as well as initial data seeding. Using a utility such as Oracle Database datapump for the export and import of DDL and corresponding data will guarantee that underlying table structure such as partitions line up exactly between Merchandising/Sales Audit and DAS. Replication issues are likely to arise if partitions are not aligned. The latest DAS data model is available on My Oracle Support.

Review and validate the following items in the DAS schema post initialization and prior to enabling replication:

1. Verify that all constraints are enabled and in a validated state. Run the DAS_STAGING_DIR/enable_das_primary_keys.sql script to generate and execute a SQL script that will enable and validate any disabled or non-validated primary keys.

   In SQLPLUS, as DAS schema owner on PDB,

   ```
   SQL> @enable_das_primary_keys.sql
   Please enter the DAS schema owner name: <schema_owner>
   …
   ```

2. Recompile the views.
3. The ITEM_LOC_SOH table has been altered to maintain flashback archive data. To enable users to query historical data an additional privilege must be granted.

   ```
   SQL> GRANT FLASHBACK ON ITEM_LOC_SOH TO <USER>;
   ```

7

Web Services Installation

Some Oracle Retail applications; <app> (for example, RMS) use Oracle Objects for the PL/SQL API's. The tool generates a Web Service Provider layer between the external clients and the <app> API's to provide the Web Service functionality, such as faults, logging, and security, as well as the conversion from xml payloads to Oracle Objects. The Retail Service Enabler (RSE) tool creates the appropriate Provider web service endpoints as well as templates for the PL/SQL APIs.

Set up Environment

To set up the environment, do the following:

1. Source the oraenv script to set up the Oracle environment variables (ORACLE_HOME, ORACLE_SID, PATH, and so on).

   Example: prompt$. oraenv
   ORACLE_SID = [] ? mydb
   prompt$

2. Verify the ORACLE_HOME and ORACLE_SID variables after running this script.

   Example: prompt$ echo $ORACLE_HOME
   /u00/oracle/product/mydbversion
   prompt$ echo $ORACLE_SID
   mydb

3. export TNS_ADMIN=/path/to/wallet/files/dir/
4. export UP=/@<Schema Owner Wallet Alias>

   Note: See "Appendix: Setting Up Password Stores with Oracle Wallet" for how to set up database wallet.

5. Verify that TNS is set up correctly by using the UP variable to successfully log in to the RMS 19 schema.

   Example: /u00/oracle> sqlplus $UP

Grant permissions to RMS Database Schema

1. Change directories to RETAIL_HOME/dbsql_rms/Cross_Pillar/webservice_objects/consumer/sql
2. Verify the contents of the following files. They should contain commands to run grants to your RMS schema owner.
   - DrillBackForwardUrlServiceConsumer_grant.sql
   - GlAccountValidationServiceConsumer_grant.sql

   Note: If necessary, change all occurrence of <USER> to RMS schema owner RMS19DEV in the files:

   ```
   dbms_java.grant_permission( '<USER>',
   'SYS:java.lang.RuntimePermission', 'setFactory', '' )
   ```

   to

```
dbms_java.grant_permission( 'RMS19DEV',
'SYS:java.lang.RuntimePermission', 'setFactory', '' )
```

Note: For Multitenant databases comment the line CONN / AS SYSDBA)

3. Run the above files as the database SYS user.
4. You do NOT create synonyms to each java object loaded as the synonyms were created in packages previously loaded pointing to the exposed java objects.

8

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 that have already been made to the environment. For example, the patching process uses a database manifest table to skip database change scripts that 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
Oracle Retail Price Management (RPM)	▪ Database scripts (included with RMS) ▪ Java Application ▪ Batch scripts

Product	Supported Technology
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 19.0.1 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, the 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 19.0.1, 16.0.2, 16.0, 15.0.2, and so on. Cumulative patches come with a standard Oracle Retail installer and 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 that consist of metadata about the 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 that 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 that 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 Wallet 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 that 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 that 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 that 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

Log File	Used For
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.

Argument	Valid For Actions	Description
-new	apply	Forces ORPatch to not attempt to restart a failed ORPatch session
-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, this 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 that 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 that 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
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/config/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 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 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 that 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.

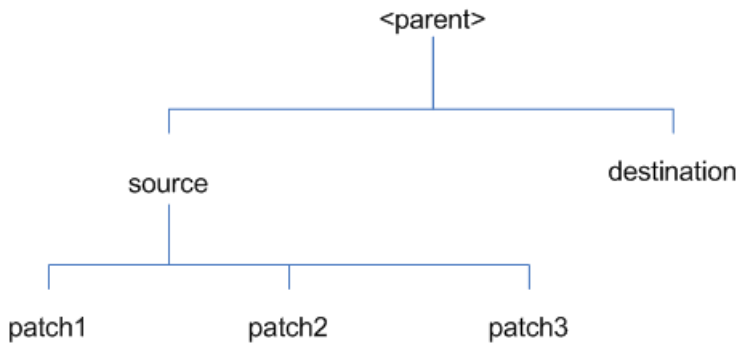
Source and Destination Directories

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
-d	Yes	Path to destination directory that will contain merged patch
-name	No	Name of 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 removes 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 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.

Application	Type	Description
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.
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 RETAIL_HOME and within database schemas creates 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 changes 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

Location	Installation Type
$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
$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 alias 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 that 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 that 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 that 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 that 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 that 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.
-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 that 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 that appear 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
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

Order	Action Name	Description
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
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 that 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.

Restart Phase Number	Phase Name	Description
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 that 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/ra_db/radm/database_change_scripts/000035_s12733240_w_party_per_d.sql.

```
ordbmreg –unregister –f
$RETAIL_HOME/dbsql_radm/ra_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 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_19_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_19_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.

A

Appendix: Oracle 19C Database Parameter File

```
##############################################################################
# Copyright (c) 2019 by Oracle Corporation
# Oracle 19.3.0.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='19.0.0.0'
*.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'
```

B

Appendix: Configure Listener for External Procedures

> **Note:** This example illustrates the listener configuration required for external procedures. It does not include environment specific settings that may be needed. Consult Oracle Net Services guides for additional information.

```
################################################################
#  File:  listener.ora
#  Desc:  Oracle Net8 listener file.
#  Notes: Modify <hostname>
################################################################

LISTENER =
  (DESCRIPTION_LIST =
    (DESCRIPTION =
      (PROTOCOL_STACK =
        (PRESENTATION = TTC)
        (SESSION = NS))
      (ADDRESS =
        (PROTOCOL = tcp)
        (HOST = <hostname>)
        (PORT = 1521))
      (ADDRESS =
        (PROTOCOL = IPC)
        (KEY = extproc_key))
    )
  )

SID_LIST_LISTENER =
  (SID_LIST =
    (SID_DESC =
      (PROGRAM = extproc)
      (SID_NAME = extproc_agent)
      (ENVS='EXTPROC_DLLS=ANY')
    )
)
```

Note: This example illustrates the configuration of net services names required for external procedures. It does not include environment specific settings that may be needed. Consult Oracle Net Services guides for additional information

```
#################################################################
# File: tnsnames.ora
# Desc: Net Services configuration file.
# Note: Change these values: <service_name>, <oracle_sid>, <hostname>,
#       <global_name>
#################################################################

EXTPROC_CONNECTION_DATA =
  (DESCRIPTION =
    (ADDRESS_LIST = (ADDRESS = (PROTOCOL = IPC)(Key = extproc_key)))
    (CONNECT_DATA = (SID = extproc_agent)))

EXTPROC_CONNECTION_DATA.world =
  (DESCRIPTION =
    (ADDRESS_LIST = (ADDRESS = (PROTOCOL = IPC)(Key = extproc_key)))
    (CONNECT_DATA = (SID = extproc_agent)))

< Connect_string> =
  (DESCRIPTION =
    (ADDRESS_LIST = (ADDRESS = (PROTOCOL = tcp)(host = <hostname>)(Port = 1521)))
    (CONNECT_DATA = (Service_Name = <Service_Name>) (GLOBAL_NAME =
<global_name>)))

<Connect_String>.world =
  (DESCRIPTION =
    (ADDRESS_LIST = (ADDRESS = (PROTOCOL = tcp)(host = <hostname>)(Port = 1521)))
    (CONNECT_DATA = (Service_Name = <Service_Name> >) (GLOBAL_NAME =
<global_name>)))

Example:
EXTPROC_CONNECTION_DATA =
  (DESCRIPTION =
    (ADDRESS_LIST = (ADDRESS = (PROTOCOL = IPC)(Key = extproc_key)))
    (CONNECT_DATA = (SID = extproc_agent)))

EXTPROC_CONNECTION_DATA.world =
  (DESCRIPTION =
    (ADDRESS_LIST = (ADDRESS = (PROTOCOL = IPC)(Key = extproc_key)))
    (CONNECT_DATA = (SID = extproc_agent)))

prod_db1 =
  (DESCRIPTION =
    (ADDRESS_LIST = (ADDRESS = (PROTOCOL = tcp)(host = server_01)(Port = 1521)))
    (CONNECT_DATA = (Service_Name = prod_db1) (GLOBAL_NAME = prod_db1.world)))

prod_db1.world =
  (DESCRIPTION =
    (ADDRESS_LIST = (ADDRESS = (PROTOCOL = tcp)(host = server_01)(Port = 1521)))
    (CONNECT_DATA = (Service_Name = prod_db1) (GLOBAL_NAME = prod_db1.world)))
```

C

Appendix: Tablespace Creation

Non-Encrypted Tablespace Creation

Standard RMS tablespaces are created using the create_rms_tablespaces.sql script located in STAGING_DIR/rms/installer/create_db.

1. Modify STAGING_DIR/rms/installer/create_db/create_rms_tablespaces.sql. The table below shows the default initial sizes.

TABLESPACE_NAME	Size
ENCRYPTED_RETAIL_INDEX	12G
ENCRYPTED_RETAIL_DATA	10G
RETAIL_INDEX	10G
RETAIL_DATA	8G
LOB_DATA	2G
USERS	2G
FLASHBACK_DATA	2G

2. Once this script has been modified, execute it in SQL*Plus as sys.
3. Review create_rms_tablespaces.log for errors and correct as needed.

Encrypted Tablespace Creation

If you do not have an Advanced Security Option license, create the encrypted_retail_data and encrypted_retail_index tablespaces as normal tablespaces.

1. Modify STAGING_DIR/rms/installer/create_db/create_encrypted_tablespaces_no_TDE.sql
2. Run the script using SQL*Plus as sys
3. Review Create_encrypted_retail_tablespaces_no_TDE.log for errors and correct as needed

With an Advanced Security license, tablespaces can be created in an encrypted format. The steps are:

Configure a Wallet

1. Create a sqlnet.ora in $TNS_ADMIN directory of the database server similar to the below entry:

```
ENCRYPTION_WALLET_LOCATION =
  (SOURCE = (METHOD = FILE)
    (METHOD_DATA =
      (DIRECTORY = /u00/oracle/admin/ORACLE_SID/wallet)))
```

2. Create the wallet directory:

```
mkdir –p /u00/oracle/admin/<ORACLE_SID>/wallet
```

3. As a user with the ‘alter system’ privilege, create the wallet as follows:

Non-container databases:

a. ADMINISTER KEY MANAGEMENT CREATE KEYSTORE '/u00/oracle/admin/dbName/wallet' IDENTIFIED BY "pwd#";
b. ADMINISTER KEY MANAGEMENT SET KEYSTORE OPEN IDENTIFIED BY "pwd#";
c. ADMINISTER KEY MANAGEMENT SET KEY IDENTIFIED BY "pwd#" WITH BACKUP;
d. ADMINISTER KEY MANAGEMENT CREATE AUTO_LOGIN KEYSTORE FROM KEYSTORE '/u00/oracle/admin/dbName/wallet' identified by pwd#;

Container databases:

a. ADMINISTER KEY MANAGEMENT CREATE KEYSTORE '/u00/oracle/admin/dbName/wallet' IDENTIFIED BY "pwd#";
b. ADMINISTER KEY MANAGEMENT CREATE AUTO_LOGIN KEYSTORE FROM KEYSTORE '/u00/oracle/admin/dbName/wallet' identified by "pwd#";
c. ADMINISTER KEY MANAGEMENT SET KEYSTORE OPEN IDENTIFIED BY "pwd#" Container=ALL;
d. ADMINISTER KEY MANAGEMENT SET KEY IDENTIFIED BY "pwd#" WITH BACKUP USING 'TDE_ENCRYPTION' Container=all;

2. Confirm if the wallet is created and open (the TDE master encryption key has been created and inserted automatically):

```
SQL>
select substr(wrl_type, 1, 10) wrl_type, substr(wrl_parameter, 1, 45) param,
substr(status, 1, 10) status, substr(wallet_type, 1, 15) w_type
from v$encryption_wallet;

WRL_TYPE   PARAM                                    STATUS     W_TYPE
---------- ---------------------------------------- ---------- ---------------
FILE       /u00/oracle/admin/ORACLE_SID/wallet  OPEN       AUTOLOGIN


An auto-open wallet is created. You are ready to create the encrypted
tablespaces as shown in the following section.
```

Encryption at Tablespace Level

Once the wallet is configured, determine an encryption algorithm to be used for the encrypted tablespace and then create them. The sample scripts use the default algorithm AES128:

1. Modify STAGING_DIR/rms/installer/create_db/create_encrypted_ tablespaces_TDE.sql.
2. Run the script using SQL*Plus as sys.
3. Review Create_encrypted_retail_tablespaces_TDE.log for errors and correct as needed.

   Once the tablespaces have been created, the RMS schema installation can be run.

> **Note:** After encryption at the tablespace level, it is crucial to backup the contents in the wallet directory; otherwise, if they are lost you will not be able to access the tablespaces.

D

Appendix: RMS RETL Instructions

This appendix summarizes the RETL program features utilized in the RMS Extractions (RMS ETL). More information about the RETL tool is available in the *Oracle Retail Extract, Transform, and Load Programmer's Guide*. More information about RMS ETL is available in the *Oracle Retail Merchandising System Operations Guide*.

Configuration: RETL

Before trying to configure and run RMS ETL, install RETL (version 13.2.9), which is required to run RMS ETL. For installation instructions, see Chapter 2 of the *Oracle Retail Extract, Transform, and Load Programmer's Guide.* Run the verify_retl script (included as part of the RETL installation) to ensure that RETL is working properly before proceeding.

RETL 13.2.9 creates a wallet under $RFX_HOME/etc/security, with the following files:

- cwallet.sso
- jazn-data.xml
- jps-config.xml
- README.txt

To set up RETL wallets, complete the following steps:

1. Set the following environment variables:
   - `ORACLE_SID=retaildb`
   - `RFX_HOME=/u00/rfx/rfx-16.0`
   - `RFX_TMP=/u00/rfx/rfx-16.0/tmp`
   - `JAVA_HOME=/usr/jdk1.864bit`
   - `LD_LIBRARY_PATH=$ORACLE_HOME/lib:$LD_LIBRARY_PATH`
   - `PATH=$RFX_HOME/bin:$JAVA_HOME/bin:$PATH`
2. Change directory to $RFX_HOME/bin.
3. Run setup-security-credential.sh as follows.
   a. Enter 1 to add a new database credential.
   b. Enter the dbuseralias (for example, retl_java_rms01user).
   c. Enter the database user name (for example, rms01user).
   d. Enter the database password.
   e. Re-enter the database password.
   f. Enter D to exit the setup script.
4. Update your RETL environment variable script to reflect the names of both the Oracle Networking wallet and the Java wallet.

   For example, to configure RETLforRPAS, modify the following entries in $RETAIL_HOME/RETLforRPAS/rfx/etc/rmse_rpas_config.env.
   - The RETL_WALLET_ALIAS should point to the Java wallet entry:
     `export RETL_WALLET_ALIAS="retl_java_rms01"`
   - The ORACLE_WALLET_ALIAS should point to the Oracle network wallet entry in $RETAIL_HOME/orpatch/rms_wallet:
     `export ORACLE_WALLET_ALIAS="retaildb rms01"`

> **Note:** See the section, Setting Up Wallets for Database User Accounts.

- The SQLPLUS_LOGON should use the ORACLE_WALLET_ALIAS:
  ```
  export SQLPLUS_LOGON="/@${ORACLE_WALLET_ALIAS}"
  ```
  Note: When connecting to a pluggable database, the JDBCCONN property in the .env file needs to be properly set. This requires <port>/<service_name> instead of <port>:<sid>. Below is an example:

```
export JDBCCONN="<PROPERTY name=\"jdbcconnectionstring\"
value=\"jdbc:oracle:thin:@msp28165.us.oracle.com:1521/retaildb\"/>"
```

5. To change a password later, run setup-security-credential.sh as follows.
   a. Enter 2 to update a database credential.
   b. Select the credential to update.
   c. Enter the database user to update or change.
   d. Enter the password of the database user.
6. Re-enter the password.
7. Note the following, which is how the setup-security-credential.sh script looks as it runs.

```
/u00/rfx/rfx-16.0/bin> ./setup-security-credential.sh

================================================
RETL Database Credentials Configuration Utility.
================================================


Please select one of the below option:

1) Add a new database credentials
2) Modify/Update existing database credentials
3) Delete existing database credentials

([1], [2], [3], [D]one): 1

Please enter the dbuseralias (This has to be unique for each database):
<oracle_sid>_<database user name>, i.e., retl_java_rms01

Please enter the database username: <database user name>, i.e., rms01


Please enter the database password (password text will not be displayed as it
is entered) :

Verify database password :

Created the credentials for dbuseralias ”retl_java_rms01” successfully


Please select one of the below option:

1) Add a new database credentials
2) Modify/Update existing database credentials
3) Delete existing database credentials

([1], [2], [3], [D]one): /u00/rfx/rfx-16.0/bin>
```

```
To run the RETL wallet, the /RETLforRPAS/rfx/etc/rmse_rpas_config.env file
needs to be edited with the following entries included:

##The folloiwng setting is for dbuseralias being replaced for connectstring
and dbuserid
export RETL_WALLET_ALIAS=" retl_java_rms01”
```

> **Note:** The following is an example of how to run a sample RETL script.

- To run a RETL script, set up your environment with the following run-time variables.

```
export RFX_HOME = i.e., /u00/rfx/rfx-16.0
export RFX_TMP = i.e., /u00/rfx/rfx-16.0/tmp
export TNS_ADMIN = i.e., $RETAIL_HOME/orpatch/rms_wallet
export
ALCHOME = i.e.,
/u00/webadmin/product/10.3.6/WLS/user_projects/domains/APPDomain/alloc14/rpas-
interfaces
export
RETAIL_HOME = i.e.,
/u00/webadmin/product/10.3.6/WLS/user_projects/domains/APPDomain/alloc14/rpas-
interfaces
export ORACLE_HOME = i.e., /u00/oracle/product/12.0.1
export JAVA_HOME = i.e., /usr/jdk1.864bit
export PATH =$ORACLE_HOME/bin:$JAVA_HOME/bin:$RFX_HOME/bin:$PATH
export
LD_LIBRARY_PATH = i.e.,
$RFX_HOME/lib:$ORACLE_HOME/lib:$RETAIL_HOME/oracle/lib/bin:/lib:/usr/lib:/usr/
dt/lib:/usr/openwin/lib
export TEMP_DIR = i.e., /home/alcbatch/rpas/tmp
export PATH = i.e., ${ORACLE_HOME}/bin:${JAVA_HOME}/bin:${PATH}

Go to $ALCHOME/log and $ALCHOME/error and delete all existing files.
Go to $ALCHOME/rfx/src and run the following command:

>alcl_plan.ksh plan_01.dat

To check for errors, run echo $?. If a 1 is returned, there are errors. If a 0
is returned, there were no errors.
```

E

Appendix: Oracle Trade Management System Expectations

This appendix describes the items expected by the Oracle Trade Management System.

Installation Scripts (elc_comp_post_htsupld.sql)

This script is for the RTM product only. This needs to be applied only after all static install scripts have been run, oga, tariff_treatment, quota_category, country_tariff_treatment and hts_headings scripts have all been run followed by running the htsupld.pc program. The last step is running this script. This script inserts the Expense and Assessment Cost Components. This script needs to be run once for each country of import that the client is using.

> **Note:** This script is expecting two parameters to be passed in (the user will be prompted for the parameters). The first parameter is country ID, this is the Import Country. The second parameter is Currency Code, this is the code of the currency that corresponds to the entered Import Country. Most likely this script will be run using the Base Country and the Primary Currency as defined in the System Variables form.

The inserted components include:

- MPFXX (Merchandise Processing Fee XX) – This component is used to store Merchandise Processing Fee. In place of the XX is the country code that is passed into the script. Therefore, if the Country is US, then there is one component created, MPFUS, with a description of Merchandise Processing Fee US. This leaves the client with the ability to create additional MPF components for each of the countries that they intend to import into. This component is inserted with a Component Rate of 100 percent. This rate should be modified to be the appropriate rate for the Import Country. This component is also set up as an Always Default, which means that it is defaulted to every Item/HTS combination.
- HMFXX (Harbor Maintenance Fee XX) – This component is used to store Harbor Maintenance Fee. In place of the XX will be the country code that is passed into the script. Therefore, if the Country is US, then there is one component created, HMFUS, with a description of Harbor Maintenance Fee US. This leaves the client with the ability to create additional HMF components for each of the countries that they intend to import into. This component is inserted with a Component Rate of 100 percent. This rate should be modified to be the appropriate rate for the Import Country.
- TDTYXX (Total Duty XX) – This component is used to store the total of the duty for each Item/HTS or Order/Item/HTS combination. It totals all duties, taxes, and fees within the Ordering dialog. This total is added together with the Total Expense and the Item's Cost to come up with the Total Estimated Landed Cost of the Item or Order/Item combination. This component should not be modified.
- VFDXX (Value For Duty XX) – This Computation Value Base (CVB) is used to store the value that duty should be calculated from. In place of the XX is the country code that is passed into the script. Therefore, if the Country is US, then there is one CVB created, VFDUS, with a description of Value for Duty US. This leaves the client with the ability to create additional VFD CVBs for each of the countries that they intend to

import into. Upon insert here, this CVB will only have one detail, which is ORDCST (Order Cost). If the client needs additional expenses (we are making the assumption that only Expense components will make up Value for Duty) to be used in the Value For Duty, they need to be added to VFDXX through SQL Plus. All automatically inserted Assessment components with a Calculation Basis of Value will have VFDXX as the CVB.

- VFDXXXX (XX% of Value For Duty XX) – This component is used to store a percent of the CVB, Value For Duty. This is used in the case when you have an Item that is classified with multiple HTS codes. For example, a button-down shirt may have one HTS code for the cotton material that is 75 percent of the cost, and a second HTS code for the buttons that make up the other 25 percent. The duty components associated with the first HTS code would be need to be calculated from 75 percent of the entire Value for Duty. To accomplish this, the associated components would use VFD75XX as their CVB instead of VFDXX. The detail component would be 'VFD75XX' and would have a Component Rate of 75 and a CVB of VFDXX, therefore, the component VFD75XX would be 75% of the Value for Duty. More generically speaking, VFDXXXX will be the only detail in an inserted CVB called VFDXXXX, where the first XX is replaced with the percentage. In place of the second XX will be the country code that is passed into the script. Therefore, if the Country is US, then there will be one component created, VFD25US, with a description of 25% of Value for Duty US. This leaves the client with the ability to create additional VFD components for each of the countries that they intend to import into. The script will insert VFD25XX, VFD50XX, and VFD75XX, these are meant to be used as a guide if the client needs additional components with different percentages. These components should not be modified.
- DTYXXXX (DTYXXXX) – These components are used to calculate duty for each HTS code. In place of the first XX is the HTS code's Duty Component Code concatenated with an A, B, or C as needed for duty calculation. In place of the second XX is the country code that is passed into the script. Therefore, if the Country is US, then there is one component created, DTYXXUS, with a description of DTYXXUS. This leaves the client with the ability to create additional components for each of the countries that they intend to import into. The Import Country for these components is the country code of the Base Country that is defined on the System Options table. This component is inserted with a Component Rate of 100 percent. This rate is overwritten with the appropriate Tariff Treatment rate upon calculation within the Item and Ordering dialogs. These components should not be modified.
- DUTYXX(DUTYXX) – This component is used as a sub-total. In place of the XX is the country code that is passed into the script. Therefore, if the Country is US, then there is one component created, DUTYUS, with a description of DUTYUS. This leaves the client with the ability to create additional components for each of the countries that they intend to import into. It contains the sum of all DTYXXXX components each HTS code. This component has a CVB called DUTYXX that contains every 'DTYXXXX' component as its details. This component should not be modified.
- XXXXXXX (XXXXXXX) – Fees and Taxes are created using a concatenation of information. The Component ID consists of the Fee or Tax Class Code concatenated with the Fee or Tax Component Code, and an A or B as needed for calculation, and then the import country. For example, there is an existing Fee Class Code (also referred to as Fee Type) which is 053, its Fee Component Code is 1, and importing into the US, so there is a component created that has an ID of 0531AUS. The descriptions are the same as the Component ID and can/should be modified to be clearer. Other than the description, these components should not be modified.
- ADXX (Anti-Dumping XX) – This component contains the Anti-Dumping charge for each Item/HTS code. In place of the XX is the country code that is passed into the script. Therefore, if the Country is US, then there is one component created, ADUS,

with a description of Anti-Dumping US. This leaves the client with the ability to create additional components for each of the countries that they intend to import into. This component should not be modified.

- CVDXX (Countervailing Duty XX) – This component contains the Countervailing Duty charge for each Item/HTS code. In place of the XX will be the country code that is passed into the script. Therefore, if the Country is US, then there is one component created, CVDUS, with a description of Countervailing Duty US. This component should not be modified.

HTS Upload / Mass Update

There are several installation scripts that must be run prior to HTS Upload to populate the following tables. These are one-time installations upon implementation of the product and must be maintained by the client.

- ELC_COMP
- QUOTA_CATEGORY (through the quota_category.sql script)
- OGA (through the oga.sql script)
- COUNTRY_TARIFF_TREATMENT (via the country_tariff_treatment.sql script)
- HTS_CHAPTER (via the hts_headings.sql script)
- TARIFF_TREATMENT (through the tariff_treatment.sql script)

After the initial load of the HTS data from executing the HTS Upload program. One additional install script must be run to populate the following tables with additional information:

- ELC_COMP, CVB_HEAD, CVB_DETAIL (through the elc_comp_post_htsupld.sql script)

The initial load of HTS information using a Customs provided tape and subsequent execution of the HTS Upload program will populate and update the following tables:

- HTS
- HTS_TARIFF_TREATMENT
- HTS_OGA
- HTS_FEE
- HTS_TAX
- HTS_TT_EXCLUSIONS

The following tables need to be populated by the client, but will be updated through the HTS Upload program.

- HTS_AD
- HTS_CVD
- HTS_REFERENCE

The following tables need to be populated and maintained by the client:

- HTS_CHAPTER_RESTRAINTS

Calculation of Merchandise Processing Fee

This particular cost component is the only Cost Component that is calculated with a Min/Max Range for each Customs Entry. This range is defined on the MPF_MIN_MAX table (note: this table does not have a corresponding form and needs to be populated by the client via SQL Plus. In order to process MPF the MPF_MIN_MAX table must be populated for the import country or else the calculation function errors out during processing.). If a client does not use Merchandise Processing Fee, but has a similar component, they can use the MPF_MIN_MAX table and the MPFXX component to accomplish the same result. They simply need to change the Component Description and Rate. Within the Customs Entry dialog, MPFXX is defaulted in along with all other assessments that are associated with each Order/Item combination. Once associated with the Entry, MPF is recalculated and checked to see if the value falls within the Min/Max Range. If not, the value is modified to be within the range and then allocated across all of the items on the Entry. Because this value is being calculated by the system, the user is not allowed to modify the rate or value of any MPF components within the Customs Entry dialog.

Unit of Measure Conversions

The internal process that calculates and distributes MPF charges on-line requires Unit of Measure (UOM) conversions in multiple instances. If a particular UOM conversion is missing the processing stops and a message will be displayed indicating that there is insufficient UOM information to continue. If this should occur, you must exit the dialog that generated the error add the missing conversion information and re-enter the dialog for the MPF charges to be processed.

Customs Entry Ref. Status

There are 4 possible CE Ref. Statuses for each Customs Entry. They are Worksheet, Send, Downloaded, and Confirmed. In general, when an Entry is created it is in Worksheet status. Once all of the necessary information has been added, the user is set the Status to Send, indicating that the Entry is ready to be sent to the Broker. That night in the nightly batch run, the Entry is downloaded to the Broker (cednld.pc). Once the download process is complete, the Status is automatically set to Downloaded; a user can never set the Status to this value manually. At that point once the user receives confirmation from the Broker, makes any necessary changes, and is sure that the information is correct, they can set the CE Ref. Status to ‘Confirmed’. From that point on the Status cannot be changed, however, most of the fields on the CE Header form remain editable. All information on the CE Shipment form is view only. In addition, all information on the CE Order/Item form is view only except for the Cleared Quantity, Cleared Quantity UOM, Apply button, and Comments fields. Finally, all information in the CE Charges form will be view only as well.

Since some clients may prefer not to download their Entries to a Broker, the user will have the ability to set the CE Ref. Status from Worksheet directly to Confirmed.Customs Entry Totals

The following describes customs entry totals.

- Total Duty contains the sum of the duty charges (any component beginning with DTY) for each item times the associated item's Manifest Item quantity, summed together for all items on the entry.
- Total Taxes contains the sum of the tax charges (any component beginning with a tax type (see attached document for a description of taxes)) for each item times the associated item's Manifest Item quantity, summed together for all items on the entry.
- Total Other contains the sum of all other charges (including fees) for each item times the associated item's Manifest Item quantity, summed together for all items on the entry.
- Total VFD contains the Value for Duty (which can be made up of order cost plus other dutiable expenses such as selling commission, royalties, etc.) times the associated item's Manifest Item quantity, summed together for all items on the entry.
- Total Est. Assessments contains the sum of the estimated duty/fees/taxes for each item, calculated from the Purchase Order/Item HTS Assessments, times the associated item's Manifest Item quantity, summed together for all items on the entry.
- Total Act. Assessments contain the sum of the Total Duty, Total Taxes, and Total Other values.

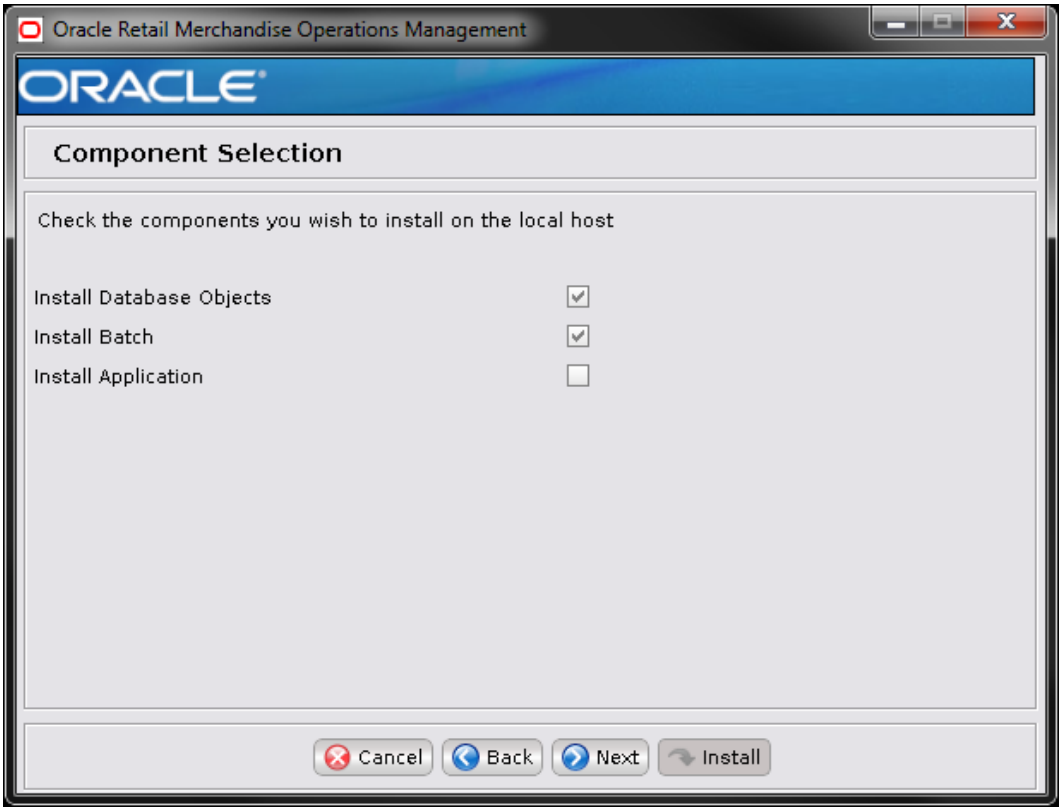
F

Appendix: RMS Database Schema and Batch Installation Screens

You need the following details about your environment for the installer to successfully create the RMS database schema and install the RMS batch programs. Depending on the options you select, you may not see some screens or fields.

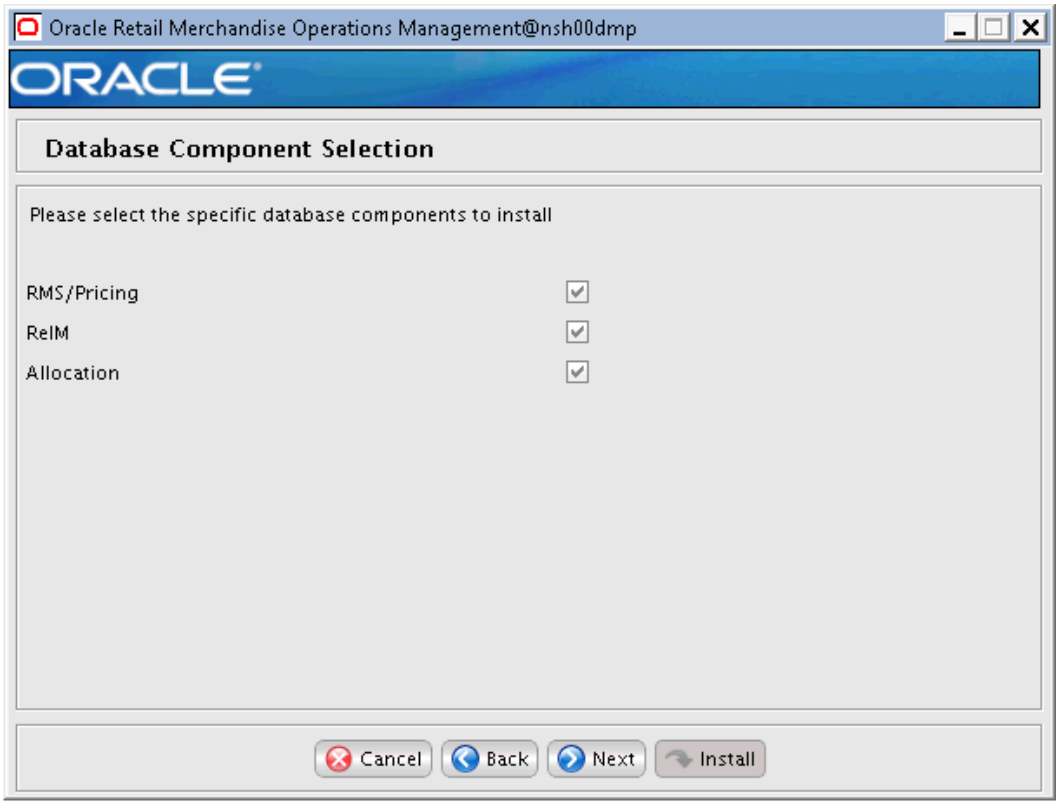
The RMS database schema installation also includes the option to install the database schema objects for the ReIM and Allocation products. The RPM database schema objects will be included with RMS.

Screen: Component Selection



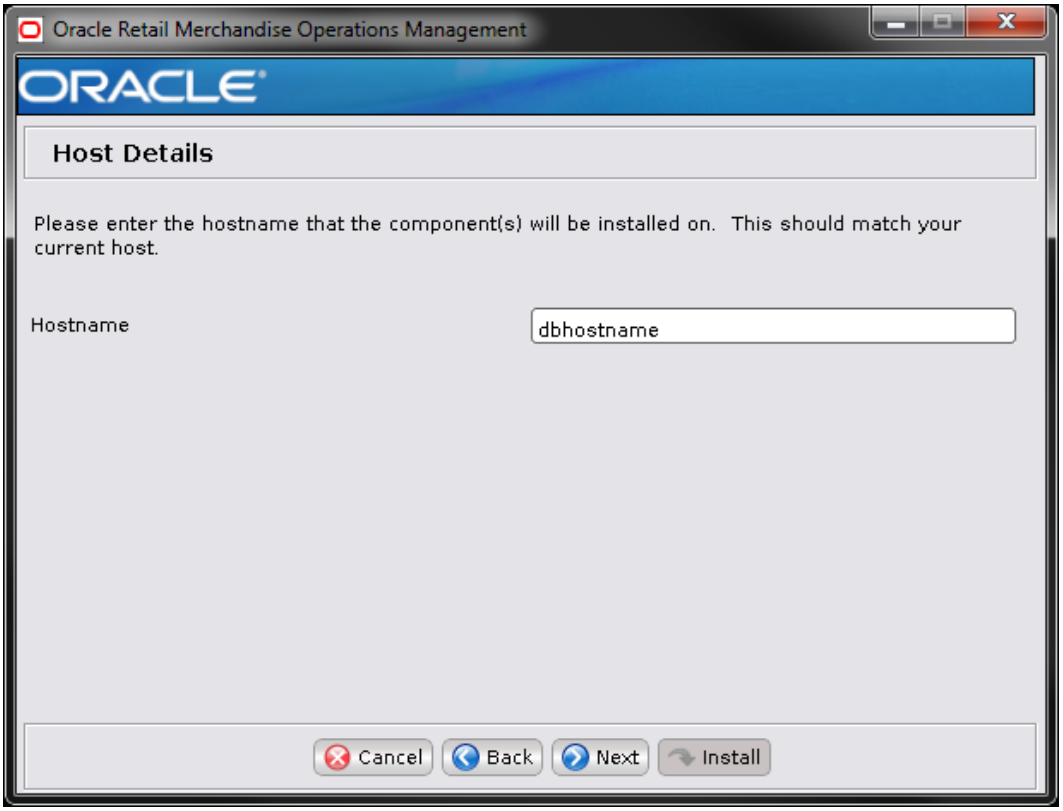
Field Title	Component Selection
Field Description	Select the RMS component(s) you would like to install. Multiple components may be selected. You will not be able to install a component if the preinstall check for that component has failed. Subsequent screens may or may not be displayed based on this choice.

Screen: Database Component Selection



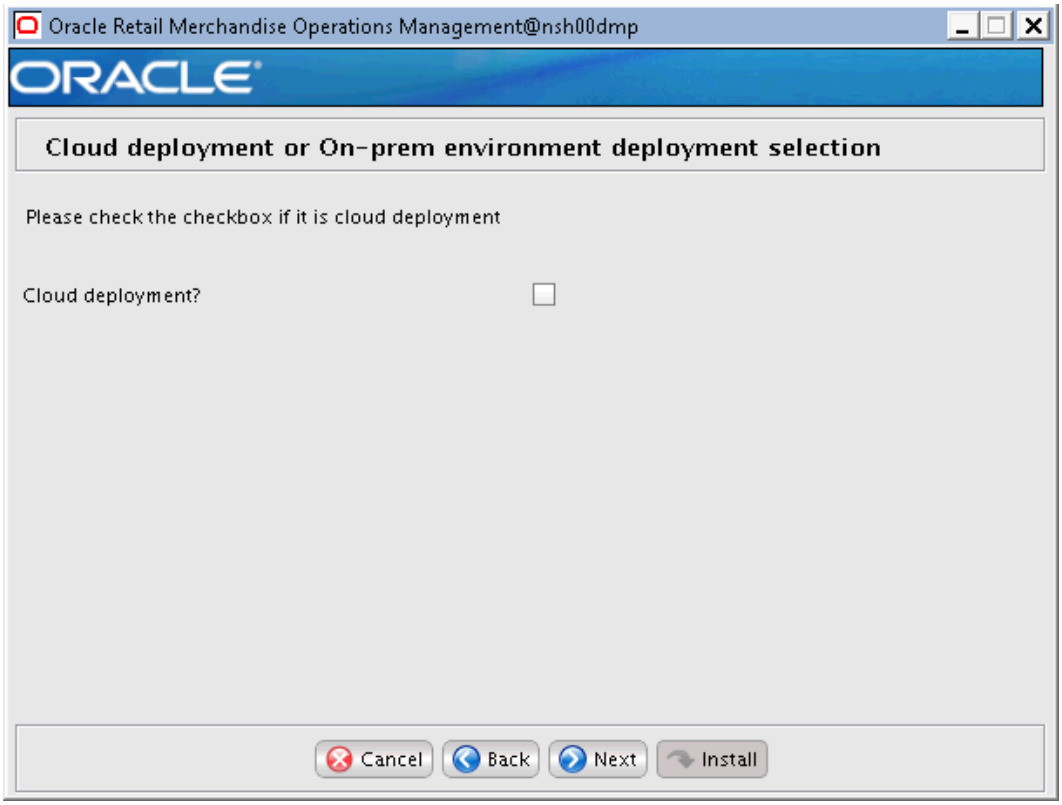
Field Title	Database Component Selection
Field Description	By default, the RMS database schema installer creates database objects for RMS/ReSA/RTM and Pricing. Optionally, the database objects for ReIM and Allocation may be installed at the same time or later. Subsequent screens may or may not be displayed based on this choice.

Screen: Host Details



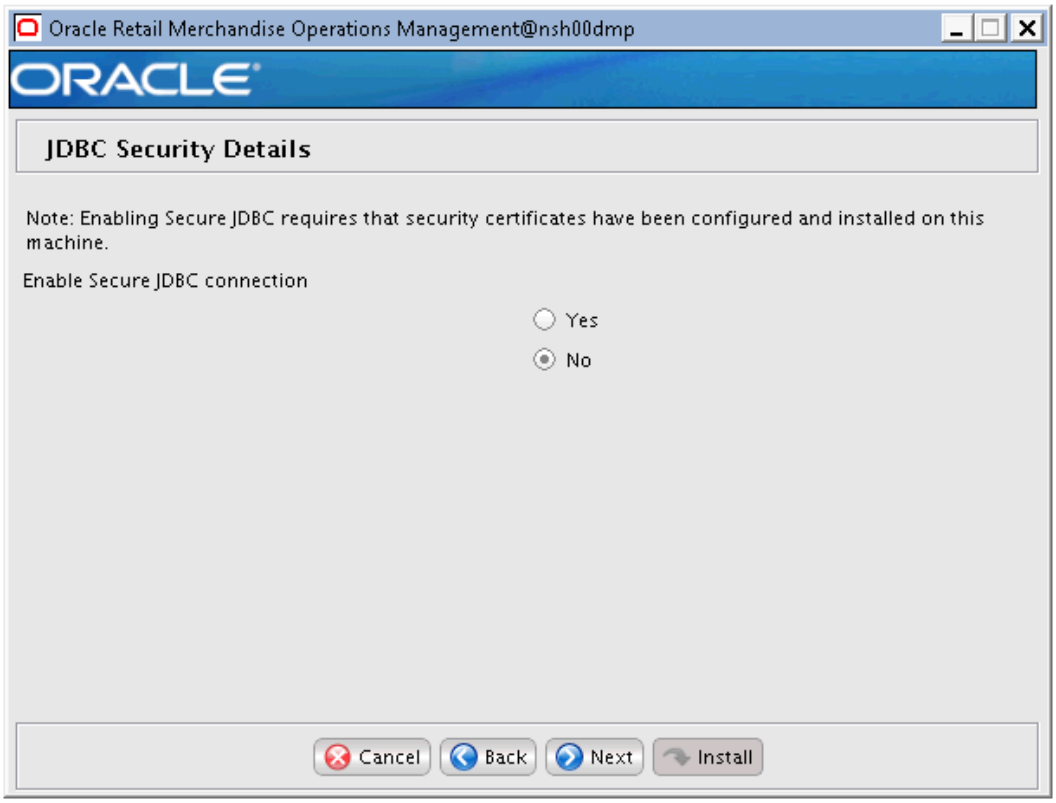
Field Title	Hostname
Field Description	Provide the hostname of the Oracle Database Server.
Example	Dbhostname

Screen: Cloud deployment or On-prem environment deployment selection



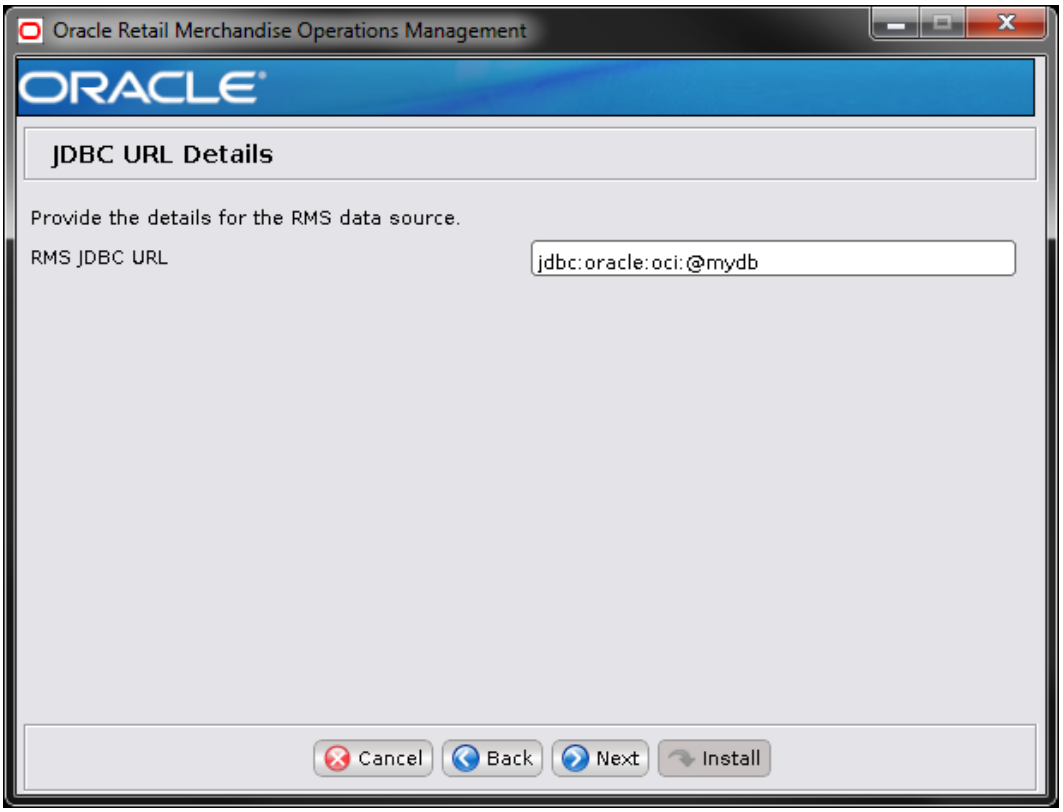
Field Title	Cloud deployment or On-prem environment deployment selection
Field Description	For on-prem installation this checkbox has to be unchecked.

Screen: JDBC Security Details



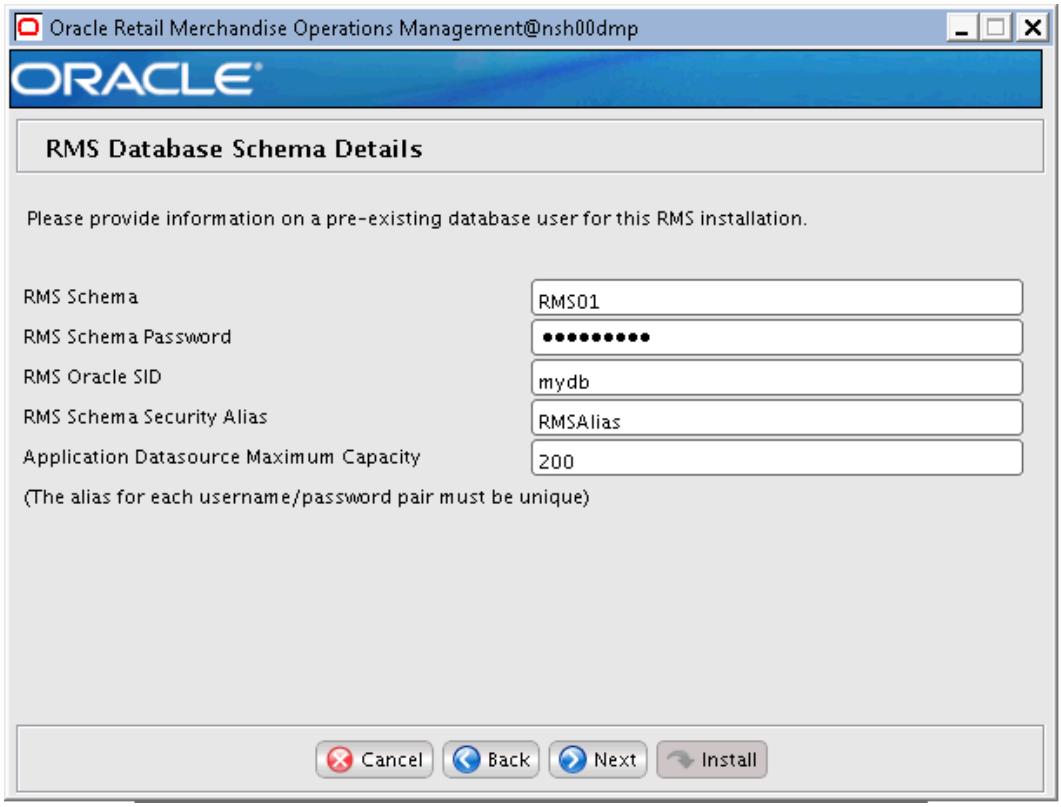
Field Title	Enable Secure JDBC connection
Field Description	Select Yes to use a secure jdbc connection during installation, otherwise choose No. A secure data base connection must already be set up if you want to use this option.

Screen: JDBC URL Details



Field Title	RMS JDBC URL
Field Description	URL used by the RMS application to access the RMS database schema. See Appendix: URL Reference for expected syntax.
Examples	For Non Secure JDBC Connection: jdbc:oracle:oci:@mydb or jdbc:oracle:thin:@dbhostname:1521/mydb For Secure JDBC Connection: jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS_LIST=(ADDRESS=(PROTOCOL=tcps)(HOST=dbhostname)(PORT=2484)))(CONNECT_DATA=(SERVICE_NAME=mydb)))

Screen: RMS Database Schema Details



Field Title	RMS schema
Field Description	Provide the RMS database user here. The installer logs into the database as this user to create the RMS schema and uses it to compile RMS batch. This user must already exist in the database when the RMS database schema installer is run.
Example	RMS01

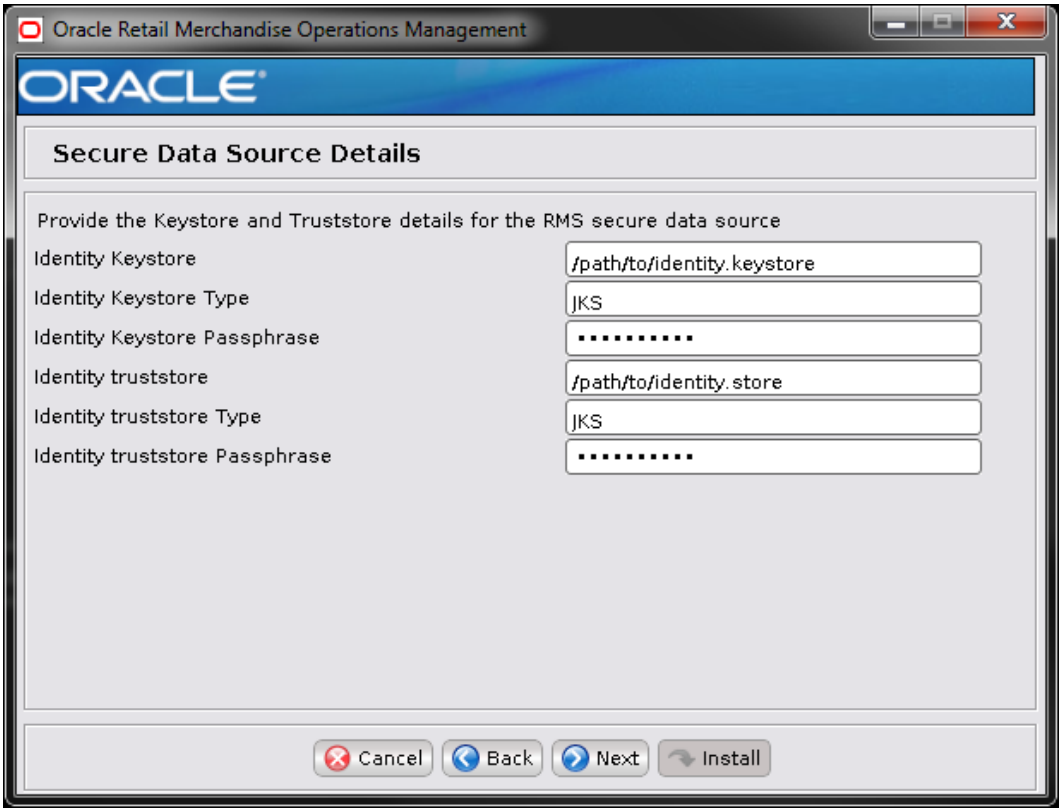
Field Title	RMS schema password
Field Description	Database password for the RMS Schema Owner.

Field Title	RMS Oracle SID
Field Description	Oracle system identifier for the database where RMS will be installed
Example	mydb

Field Title	RMS Schema Security Alias
Field Description	The alias to store the schema credentials.
Example	RMSAlias
Note	This alias must be unique. Do not use the same value for any other alias fields in the installer. If the same alias is used, entries in the wallet can override each other and cause problems with the application.

Field Title	Application Data Source Maximum Capacity
Field Description	Maximum number of database connections in connection pool for this data source in WebLogic Server
Example	200

Screen: Secure Data Source Details



Note: This screen appears only if you have enabled 'Secure JDBC' for RMS. Ignore this step in case you have not enabled 'Secure JDBC' for RMS.

Field Title	Identity Keystore
Field Description	Keystores ensure the secure storage and management of private keys and trusted certificate authorities (CAs). This screen lets you provide the keystore to be used for datasource connection These settings help you to manage the security of message transmissions. For further information, please refer *MOM security Guide.* Location or path where identity keystore file is stored.

Field Title	Identity Keystore Type
Field Description	Type of the identity keystore used. Example: jks

Field Title	Identity Keystore Passphrase
Field Description	The password to access the keystore mentioned above.

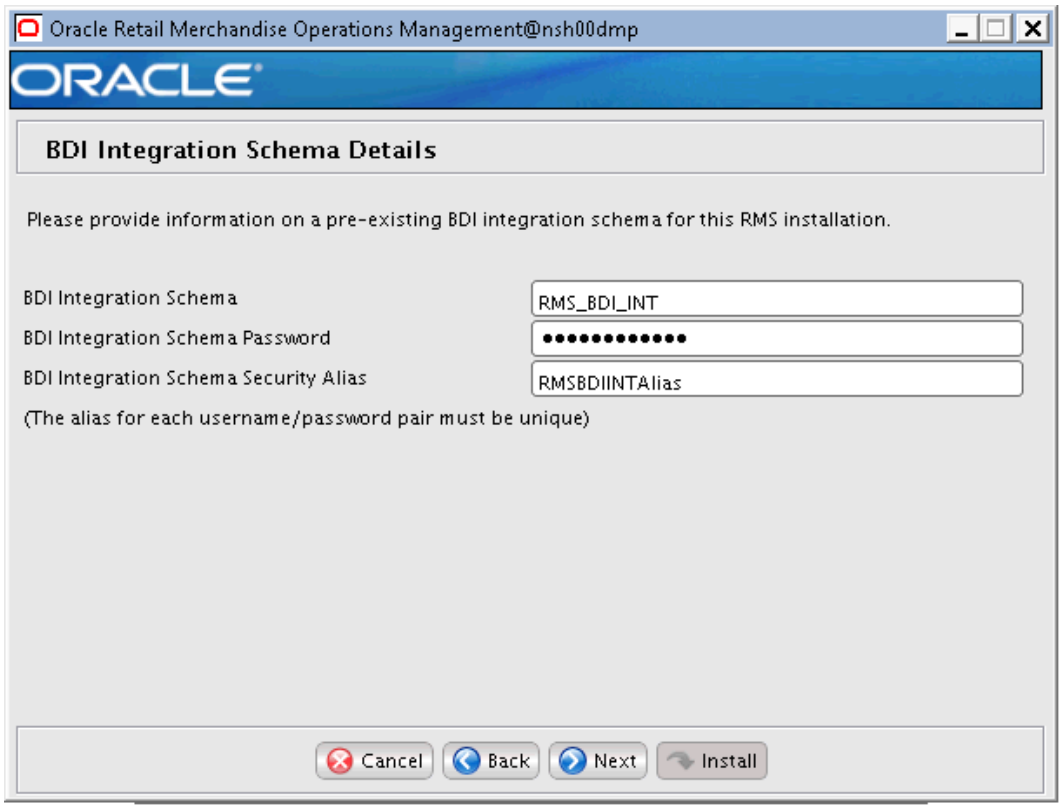
Field Title	Identity truststore
Field Description	Location or path where identity truststore file is stored.

Field Title	Identity truststore Type
Field Description	Type of the identity truststore used. Example: jks

Field Title	Identity truststore Passphrase
Field Description	The password to access the truststore mentioned above.

The installer validates the database settings provided, when you advance to the next screen.

Screen: BDI Integration Schema Details



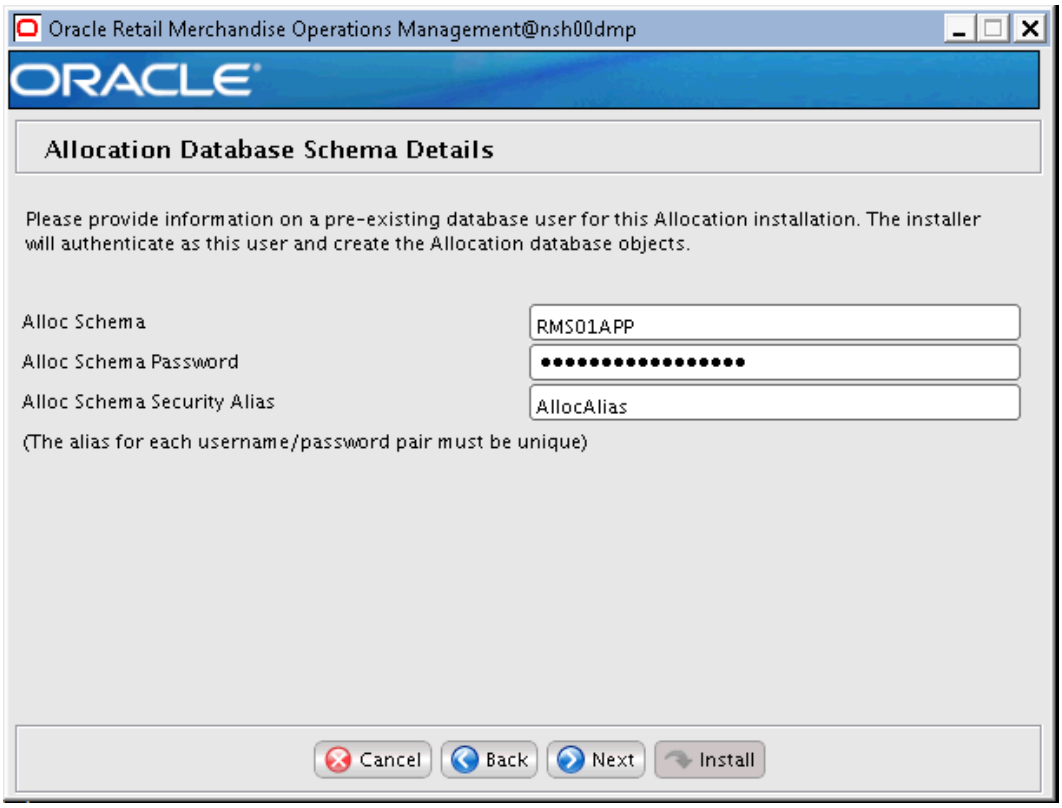
Field Title	BDI Integration Schema
Field Description	Provide the RMS BDI Integration database user here. The installer logs into the database as this user to create the RMS BDI Integration schema. This user must already exist in the database when the RMS database schema installer is run.
Example	RMS_BDI_INT

Field Title	BDI Integration Schema Password
Field Description	Database password for the RMS BDI Integration database user.

Field Title	BDI Integration Schema Security Alias
Field Description	The alias to store the schema credentials.
Example	RMSBDIIntAlias
Note	This alias must be unique. Do not use the same value for any other alias fields in the installer. If the same alias is used, entries in the wallet can override each other and cause problems with the application.

The installer validates the database settings provided, when you advance to the next screen.

Screen: Allocation Database Schema Details



Field Title	Alloc schema
Field Description	Provide the Allocation database user here. The installer logs into the database as this user to create the Allocation schema objects. This user must already exist in the database when the database schema installer is run.
Example	RMS01APP

Field Title	Alloc schema password
Field Description	Database password for the Allocation database user.

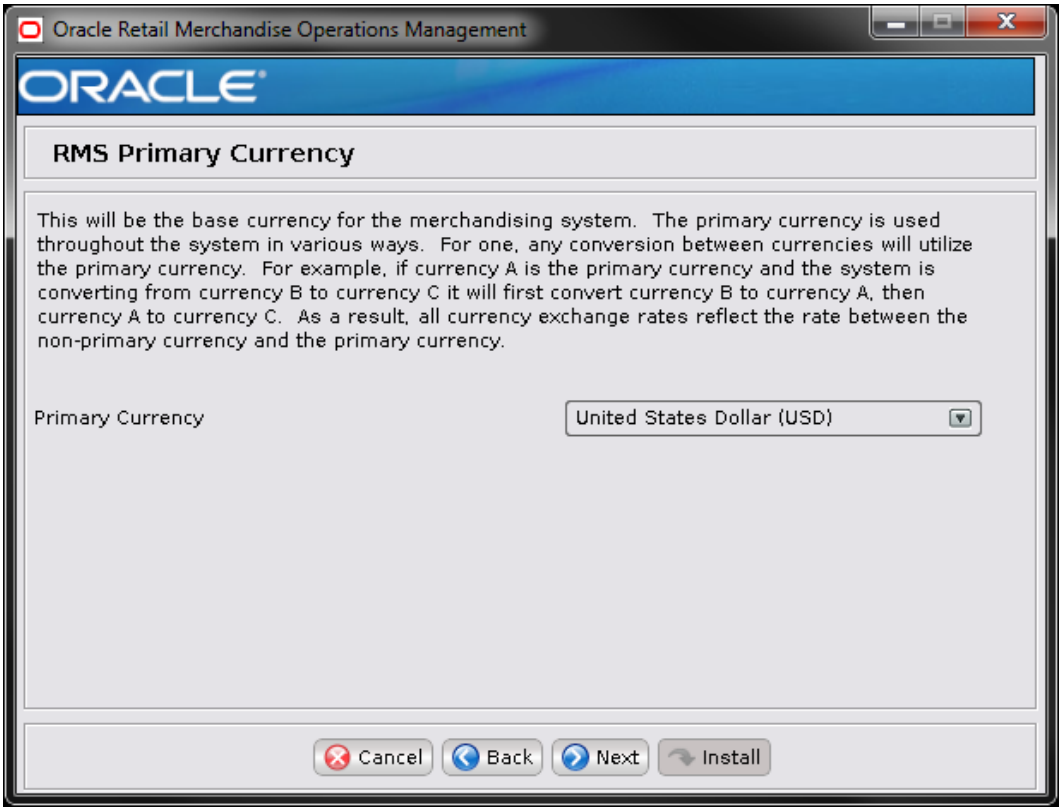
Field Title	Alloc Schema Security Alias
Field Description	The alias to store the schema credentials.
Example	dsAllocAlias
Note	This alias must be unique. Do not use the same value for any other alias fields in the installer. If the same alias is used, entries in the wallet can override each other and cause problems with the application.

The installer validates the database settings provided, when you advance to the next screen.

Screen: RMS Primary Country

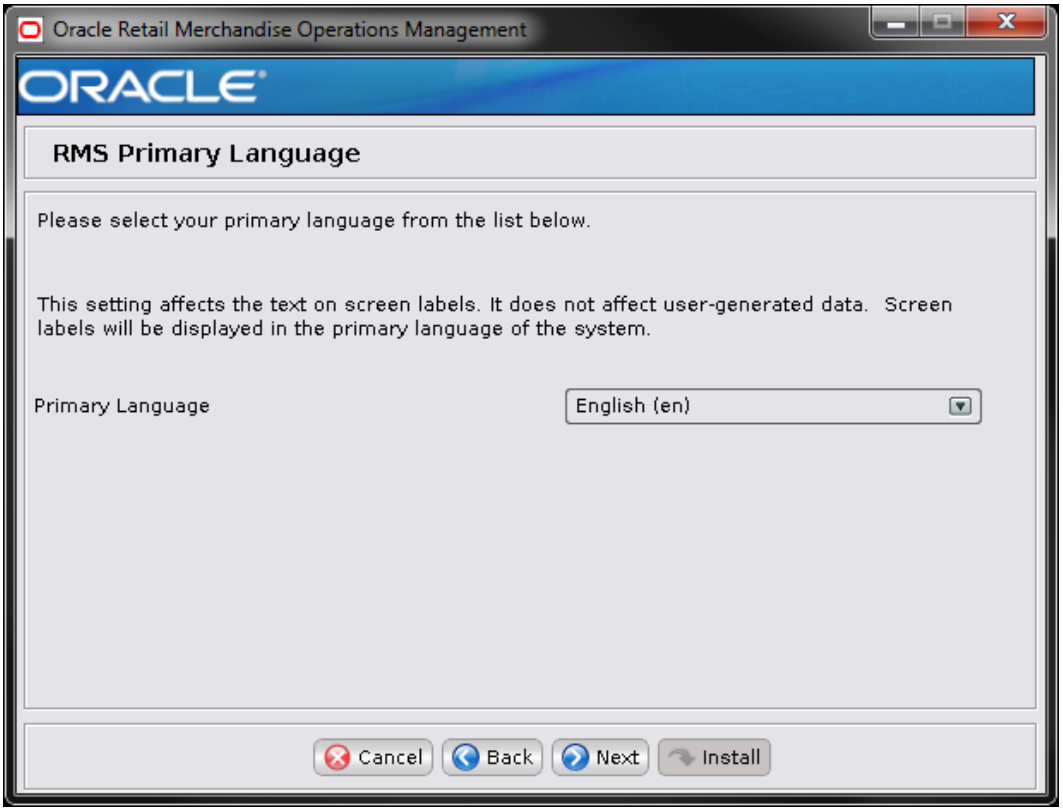
Field Title	Primary Country
Field Description	Choose your primary country from the dropdown list provided.
Example	UNITED STATES OF AMERICA (US)

Screen: RMS Primary Currency



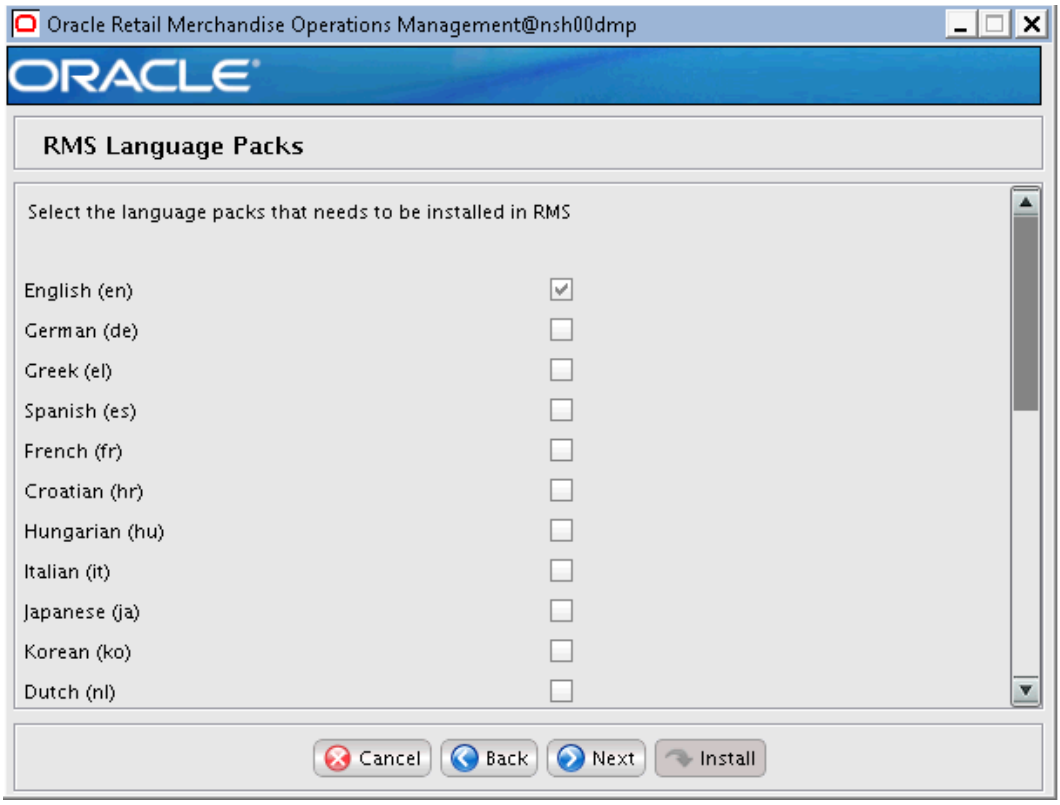
Field Title	Primary Currency
Field Description	Choose your primary currency from the dropdown list provided.
Example	United States Dollar (USD)

Screen: RMS Primary Language



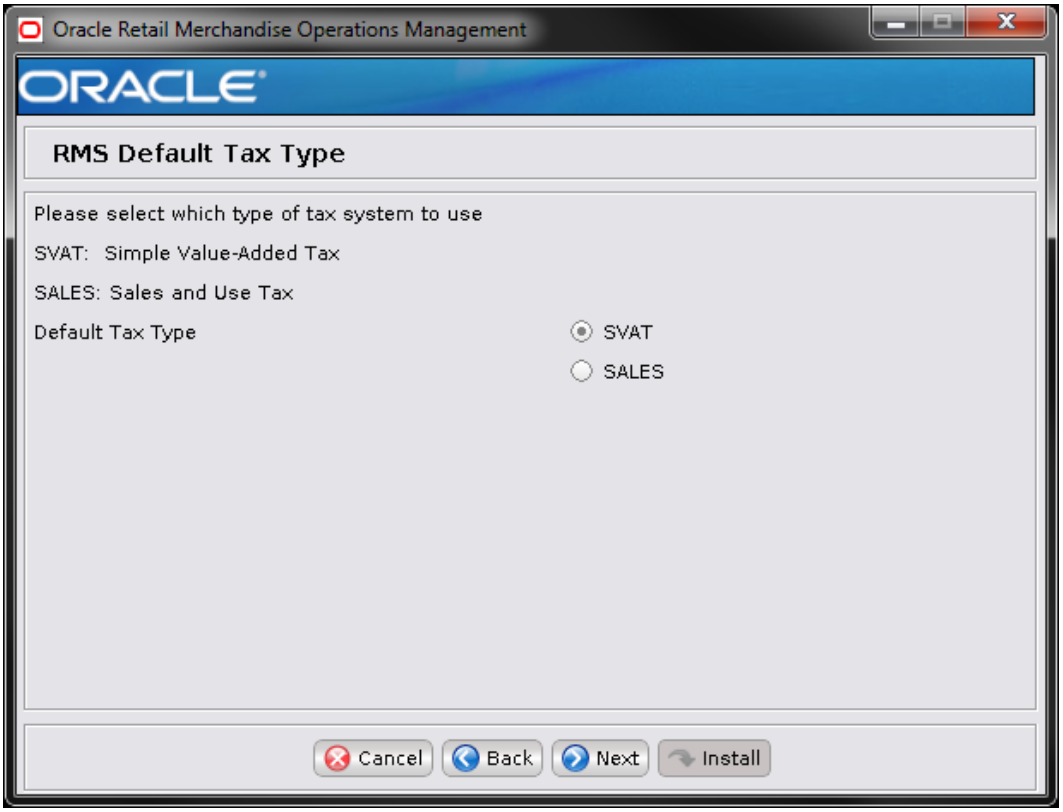
Field Title	Primary Language
Field Description	Choose your primary language from the dropdown list provided.
Example	English (en)

Screen: RMS Language Packs



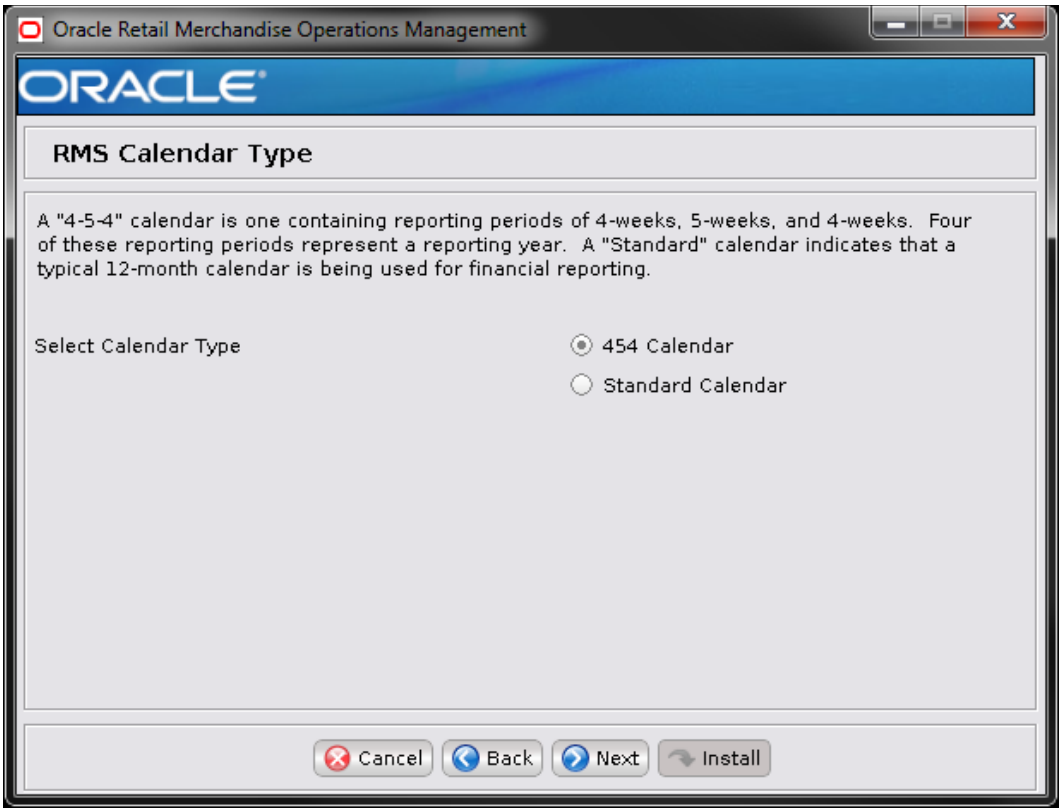
Field Title	Select the language packs that need to be installed in RMS
Field Description	Select the checkboxes to include the respective language packs in the installation.

Screen: RMS Default Tax Type



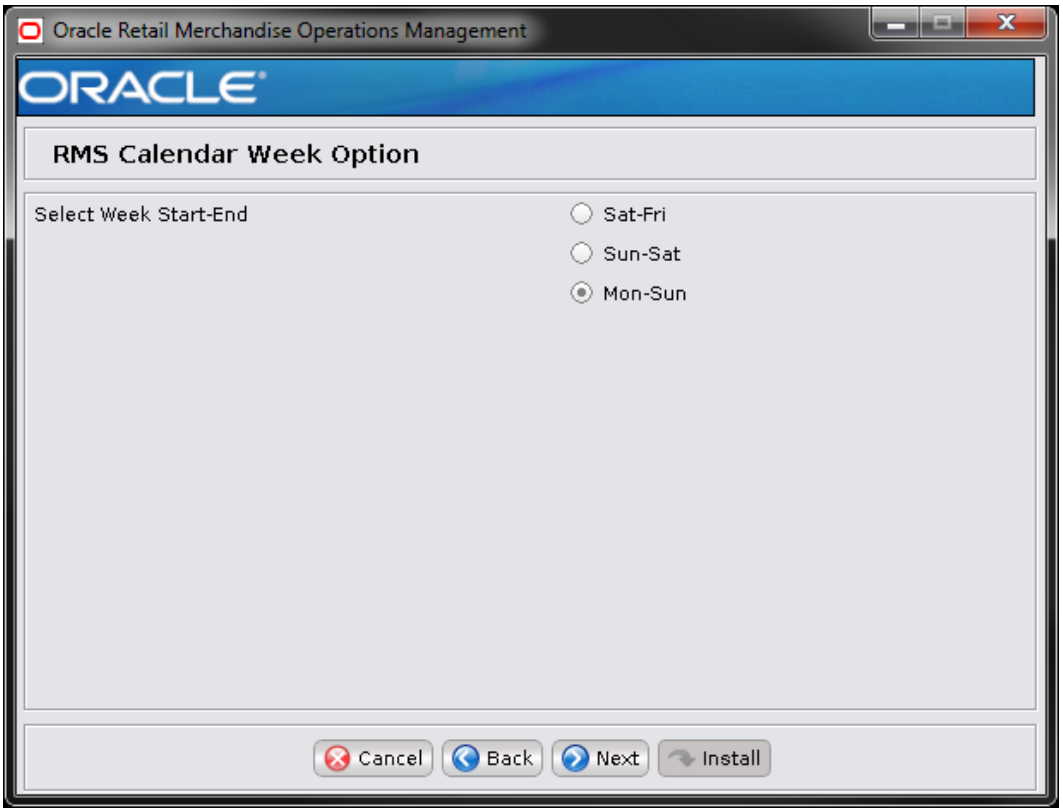
Field Title	Default Tax type
Field Description	Select the tax type that will be used with the system. SVAT: Simple Value-Added Tax (VAT information is configured in RMS) SALES: Sales and Use Tax If VAT is enabled, then select SVAT. For a configuration with only Sales Tax, Select SALES.

Screen: RMS Calendar Type



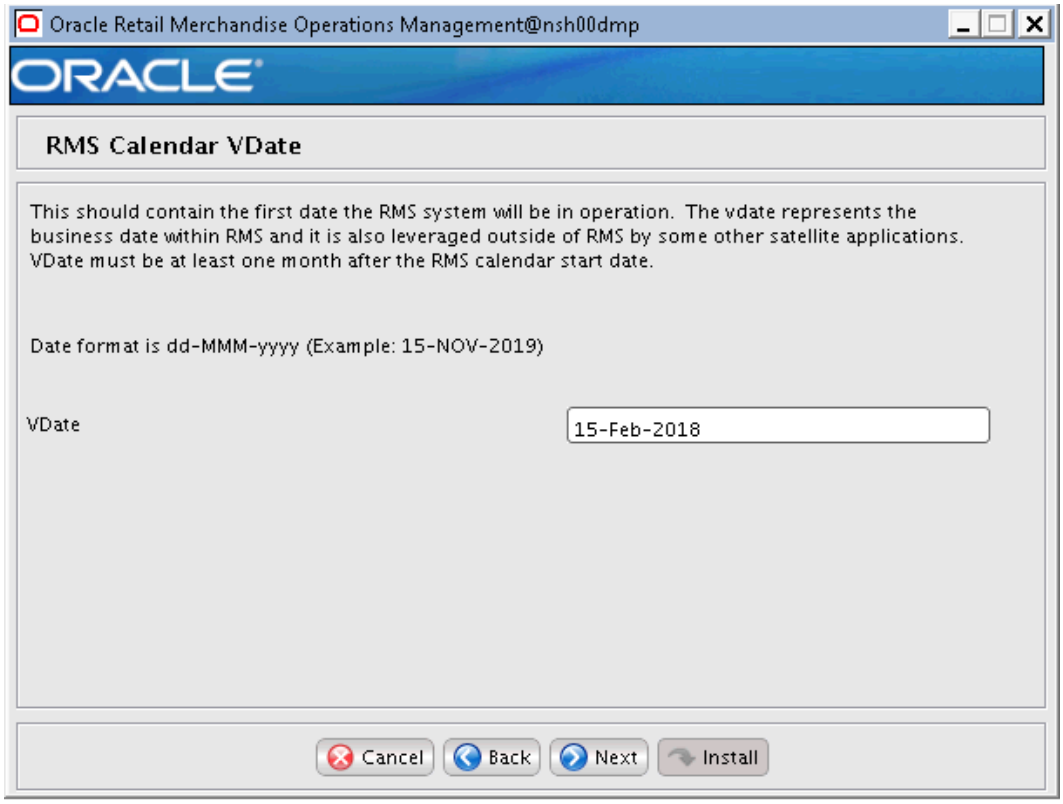
Field Title	Select Calendar Type
Field Description	Choose the type of calendar to use.

Screen: RMS Calendar Week Option



Field Title	Select Week Start-End
Field Description	Select the range that defines the first and last days of the week.

Screen: RMS Calendar VDate



Field Title	VDate
Field Description	Enter the first date the RMS System will be in operation. The format dd-MMM-yyyy must be used.
Example	15-NOV-2019

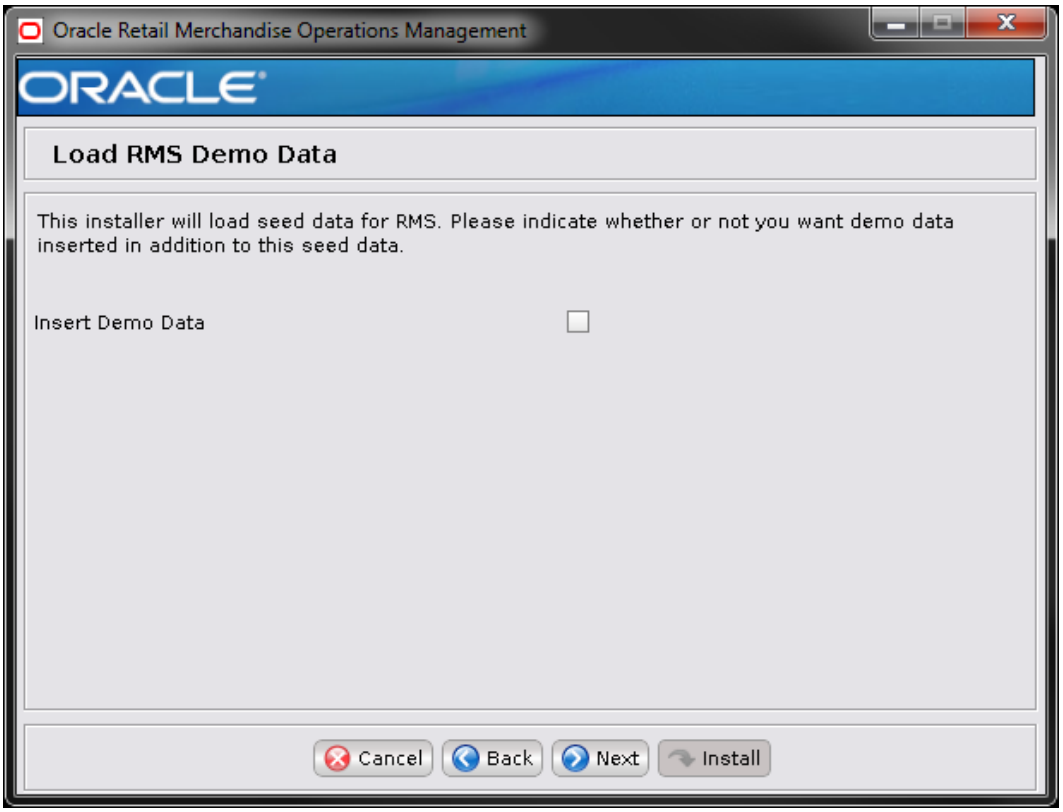
Screen: HTS Tracking Level

Field Title	HTS Tracking Level
Field Description	Select the basis for HTS tariffs and fees. The options are either the item's country of manufacturer or the item's country of sourcing.

Screen: Data Level Security

Field Title	Enable Data Level Security?
Field Description	Indicates if data level security is being utilized in the system.

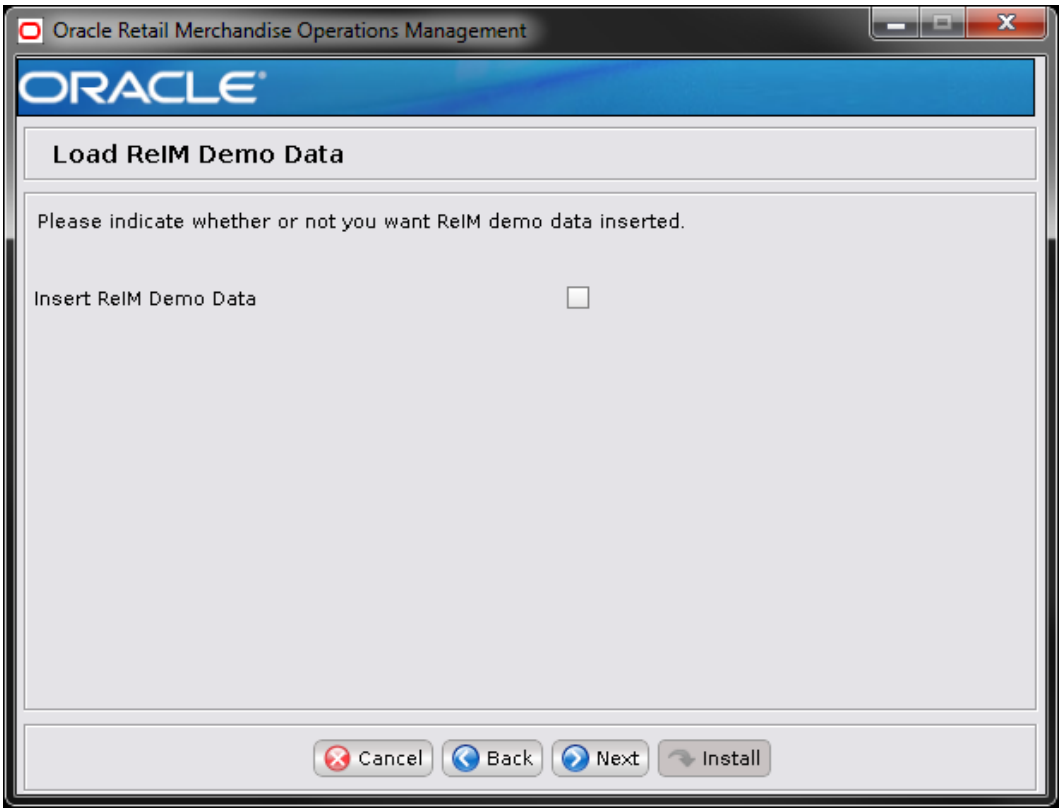
Screen: Load RMS Demo Data



Field Title	Insert Demo Data
Field Description	The Installer by default loads seed data for RMS. Check the box to insert RMS demo data in addition to seed data.
Note	Demo data should not be installed in production environments. See the warning screen below for more details.

Screen: Load ReIM Demo Data

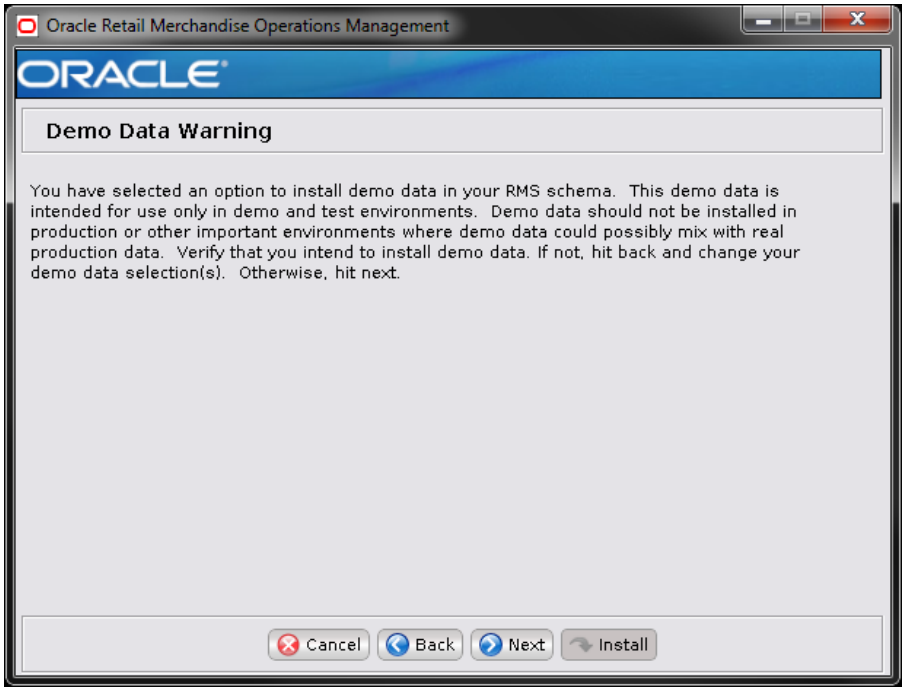
Note: The Load ReIM Demo Data screen is only shown if Insert Demo Data is checked in the previous screen and ReIM is selected in the beginning of the installation.



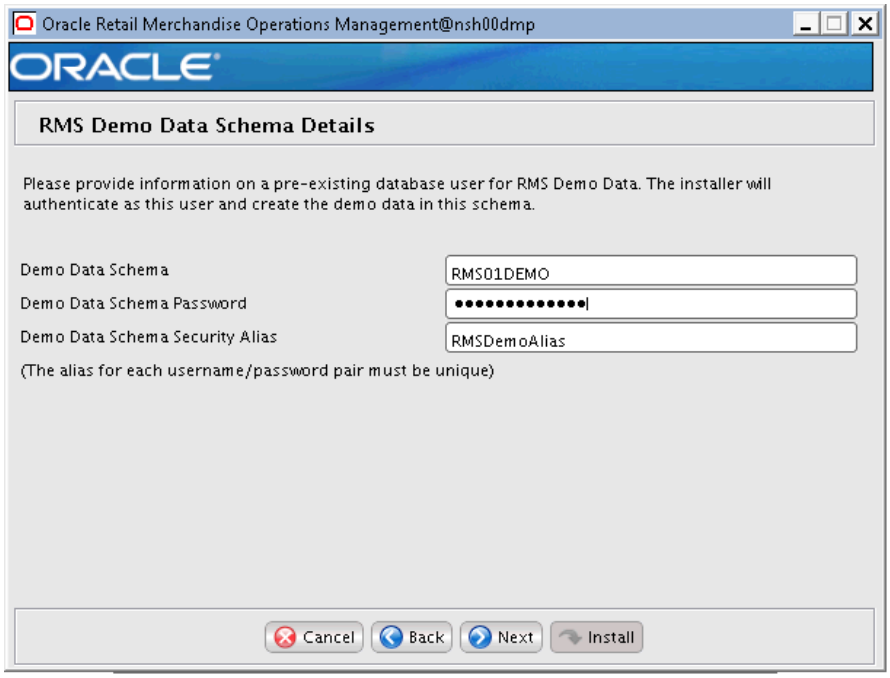
Field Title	Insert ReIM Demo Data
Field Description	Check the box to insert ReIM demo data.
Note	Demo data should not be installed in production environments. See the warning screen below for more details.

Screen: Demo Data Warning

Note: This screen is shown only if a Demo Data option is selected in the previous screens. Please read the Warning carefully.



Screen: RMS Demo Data Schema Details



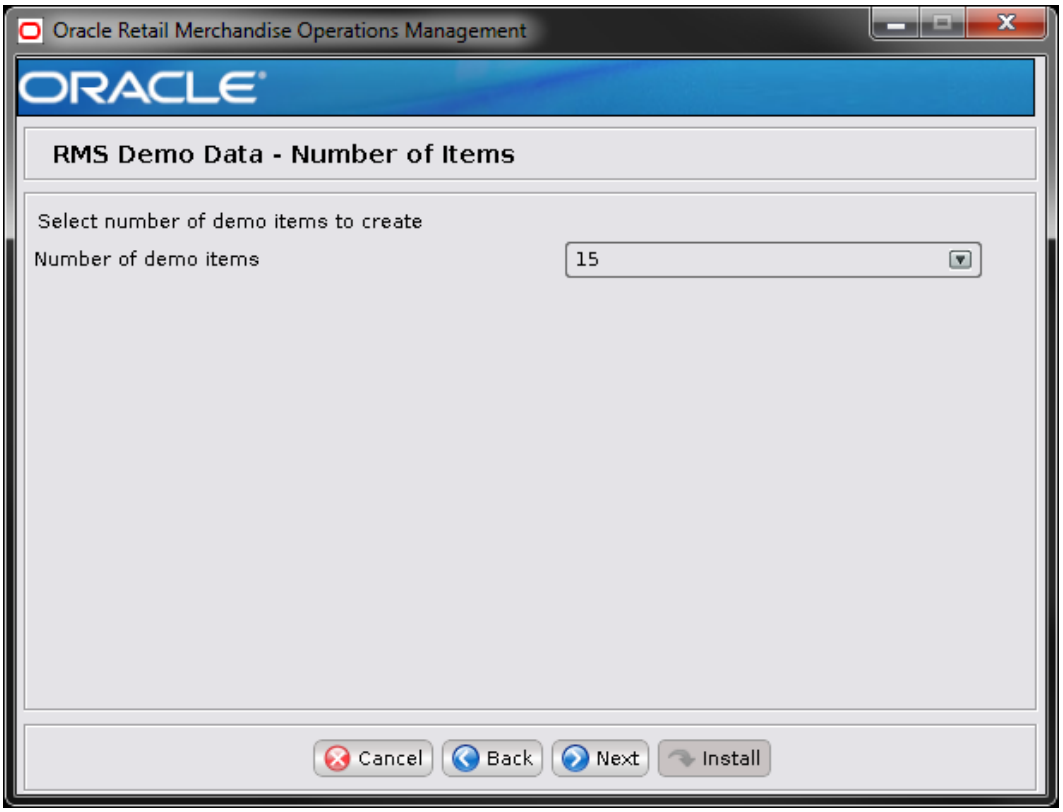
Field Title	Demo Data schema
Field Description	Schema that will be used to insert demo data into the RMS database.
Example	RMS01DEMO

Field Title	Demo Data schema password
Field Description	Password for the demo data schema.

Field Title	Demo Data Schema Security Alias
Field Description	The alias to store the schema credentials.
Example	RMSDemoAlias
Note	This alias must be unique. Do not use the same value for any other alias fields in the installer. If the same alias is used, entries in the wallet can override each other and cause problems with the application.

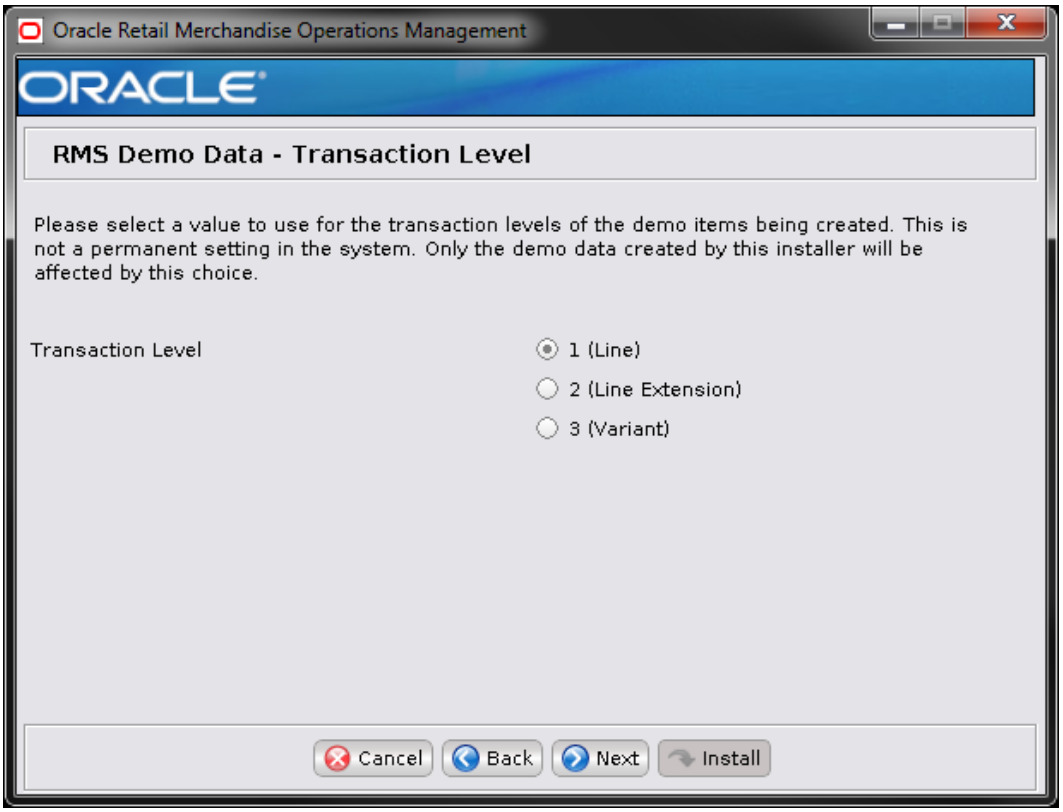
The installer validates the database settings provided, when you advance to the next screen.

Screen: RMS Demo Data – Number of Items



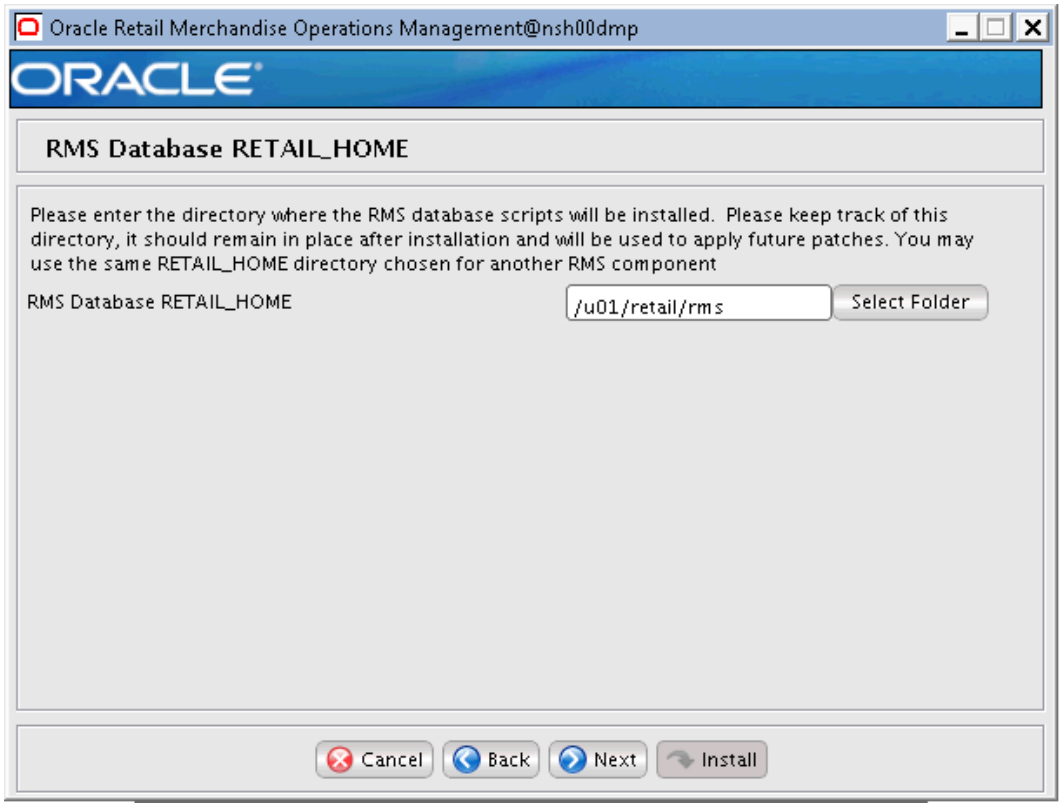
Field Title	Number of demo items
Field Description	The number of demo data items to create.
Example	15

Screen: RMS Demo Data – Transaction Level



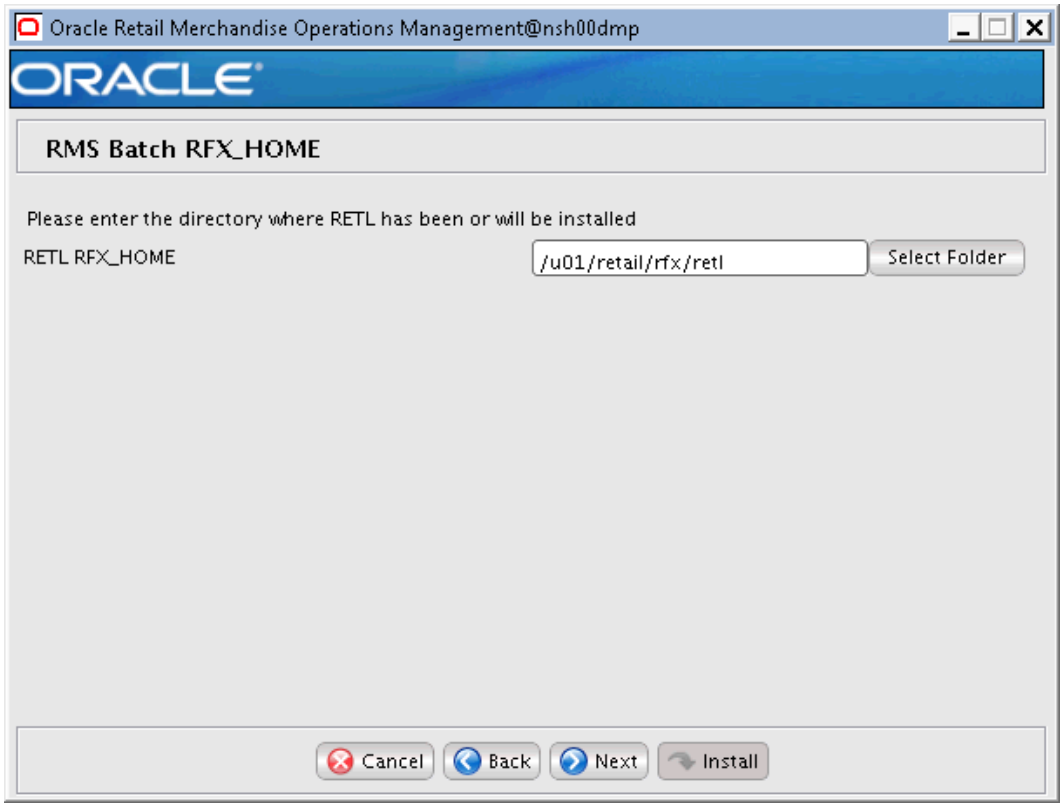
Field Title	Transaction Level
Field Description	Value to use for the transaction level of the demo items being created.
Example	1 (Line)

Screen: RMS Database RETAIL_HOME



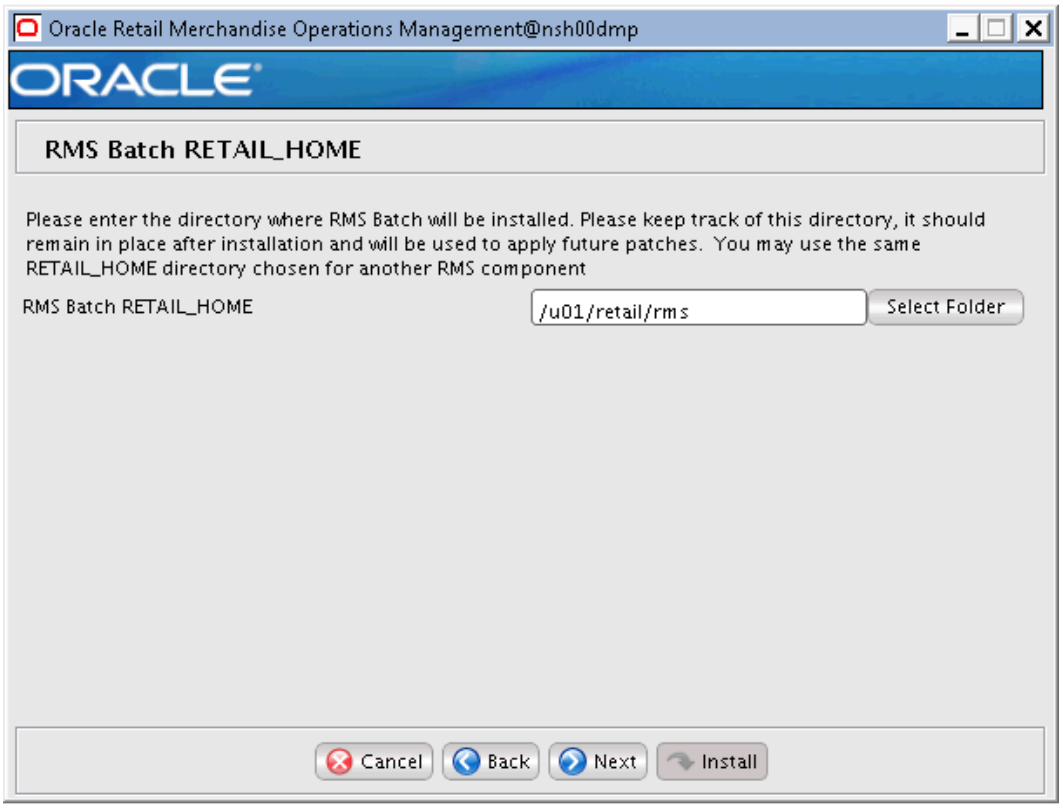
Field Title	RMS DB RETAIL_HOME
Field Description	The location where the RMS Database Files are stored by the installer. This location will be used during the subsequent patching of RMS, and will contain the ORPatch utility.
Example	/u01/retail/rms
Note	If you have selected an existing RETAIL_HOME, and it has been configured to run other components than the ones you have selected for this installation, those components will also be installed regardless of what you selected on the Component Selection screen.

Screen: RMS Batch RFX_HOME



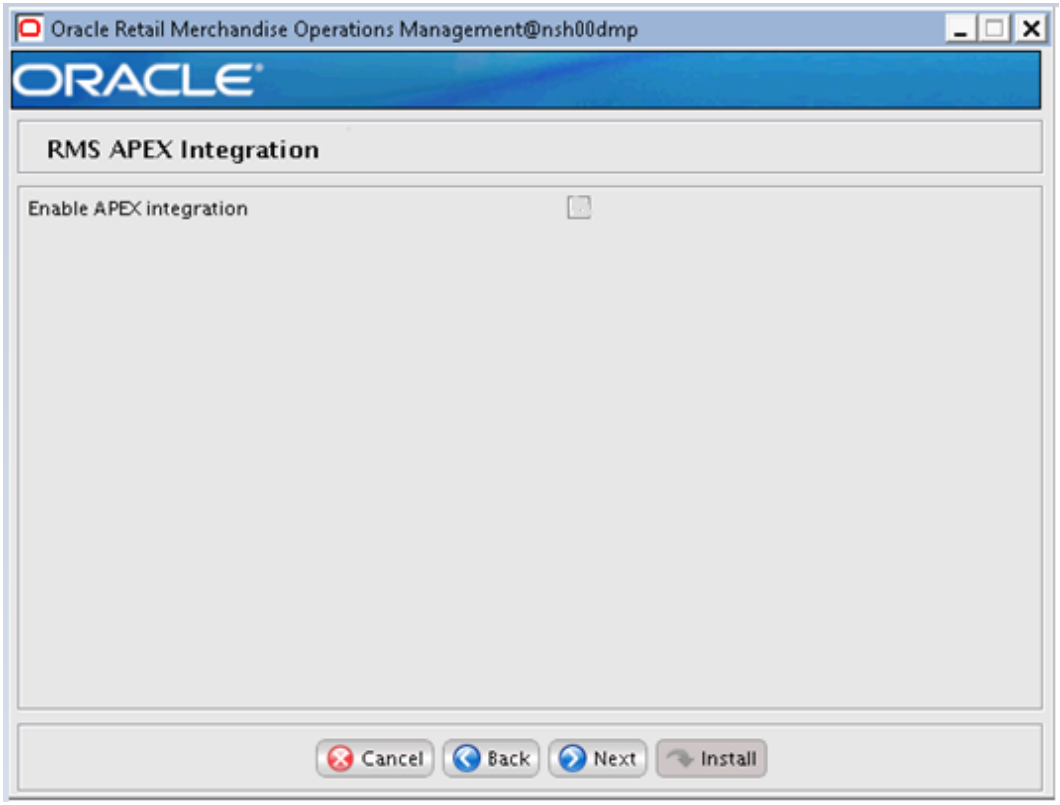
Field Title	RMS Batch FRX_HOME
Field Description	Enter the directory where RETL has been or will be installed
Example	/u01/retail/rfx/retl

Screen: RMS Batch RETAIL_HOME



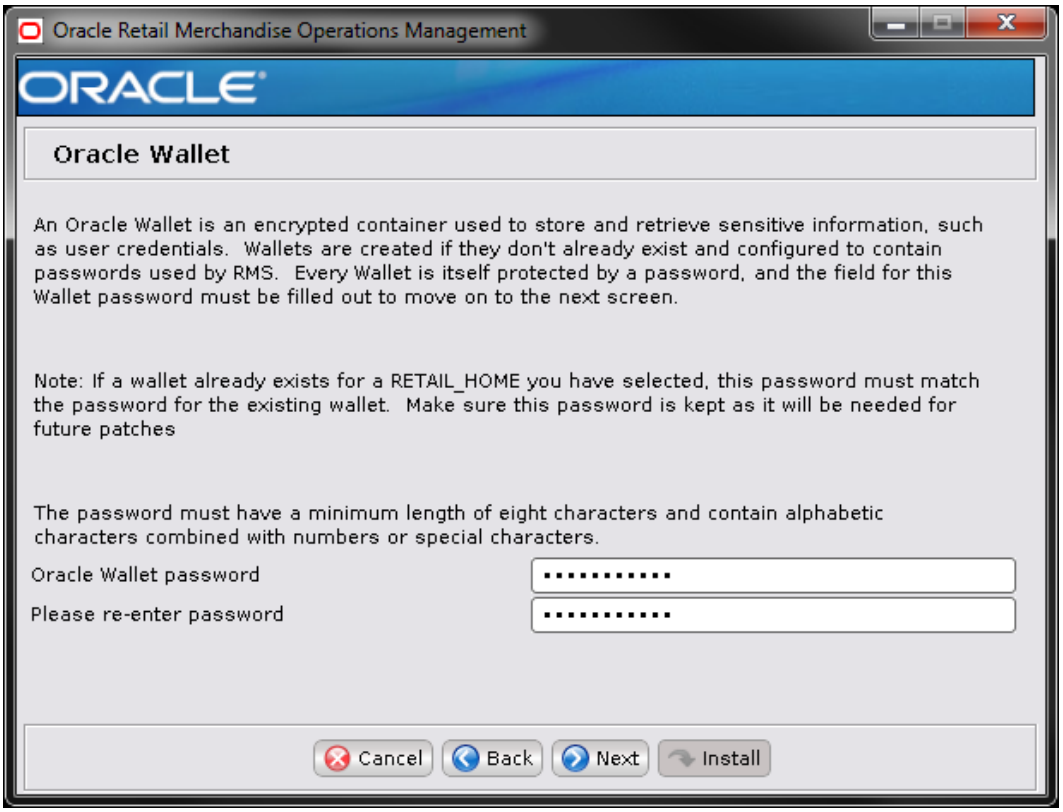
Field Title	RMS Batch RETAIL_HOME
Field Description	This is the Batch Installation Directory, the location where RMS Batch Files will be installed along with the ORPATCH utility. This can be the same RETAIL_HOME that was used for another component.
Example	/u01/retail/rms
Note	If you have selected an existing RETAIL_HOME, and it has been configured to run other components than the ones you have selected for this installation, those components will also be installed regardless of what you selected on the Component Selection screen.

Screen: RMS APEX Integration



Field Title	Enable Apex Integration
Field Description	Uncheck this option as Apex integration is not supported in 19.0.1

Screen: Oracle Wallet

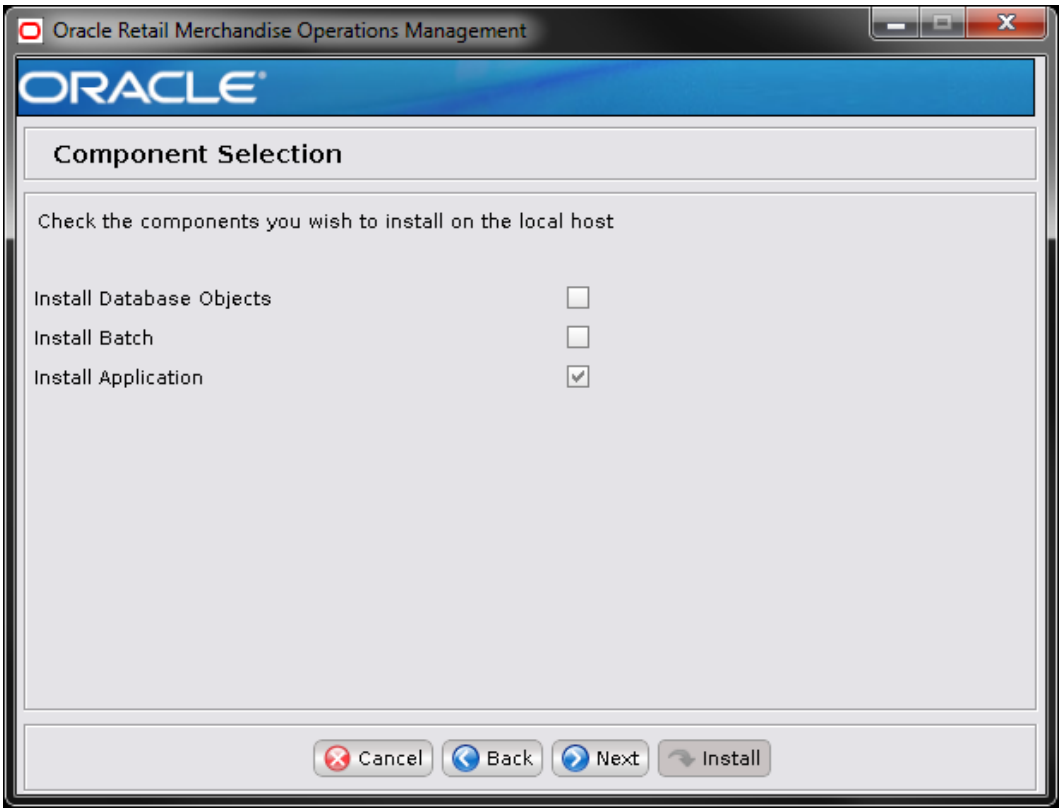


Field Title	Oracle Wallet
Oracle Wallet Password	This is the password for the wallet that will store the credentials used during the RMS installation. If you have selected an existing RETAIL_HOME in the previous screens, you will need to enter the password that was used for the wallet in that RETAIL_HOME.
Note	Make sure this password is kept, as it will be needed for future upgrades.

G

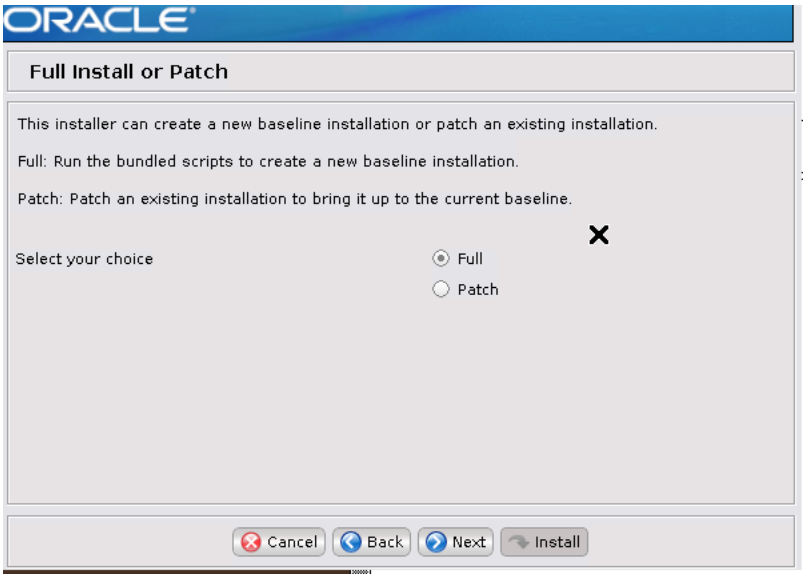
Appendix: RMS Application Installer Screens

Screen: Component Selection



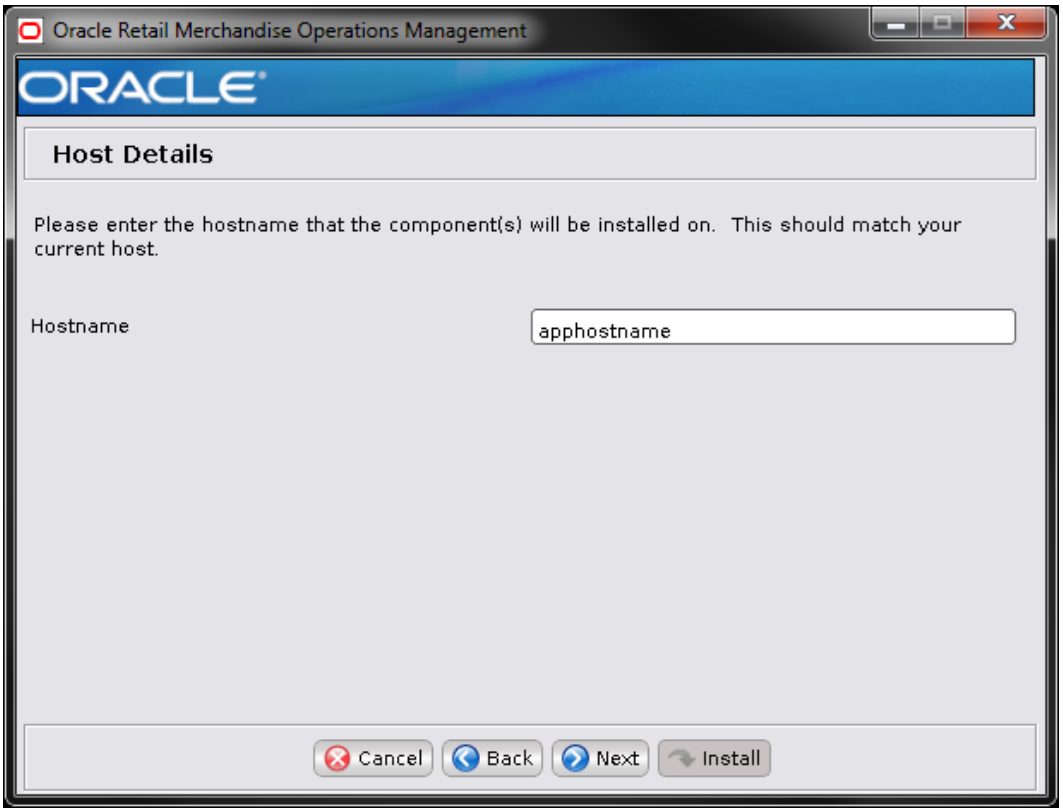
Field Title	Component Selection
Field Description	Select the RMS component(s) you would like to install. Multiple components may be selected. You will not be able to install a component if the preinstall check for that component has failed. Subsequent screens may or may not be displayed based on this choice.

Screen: Full install or Patch



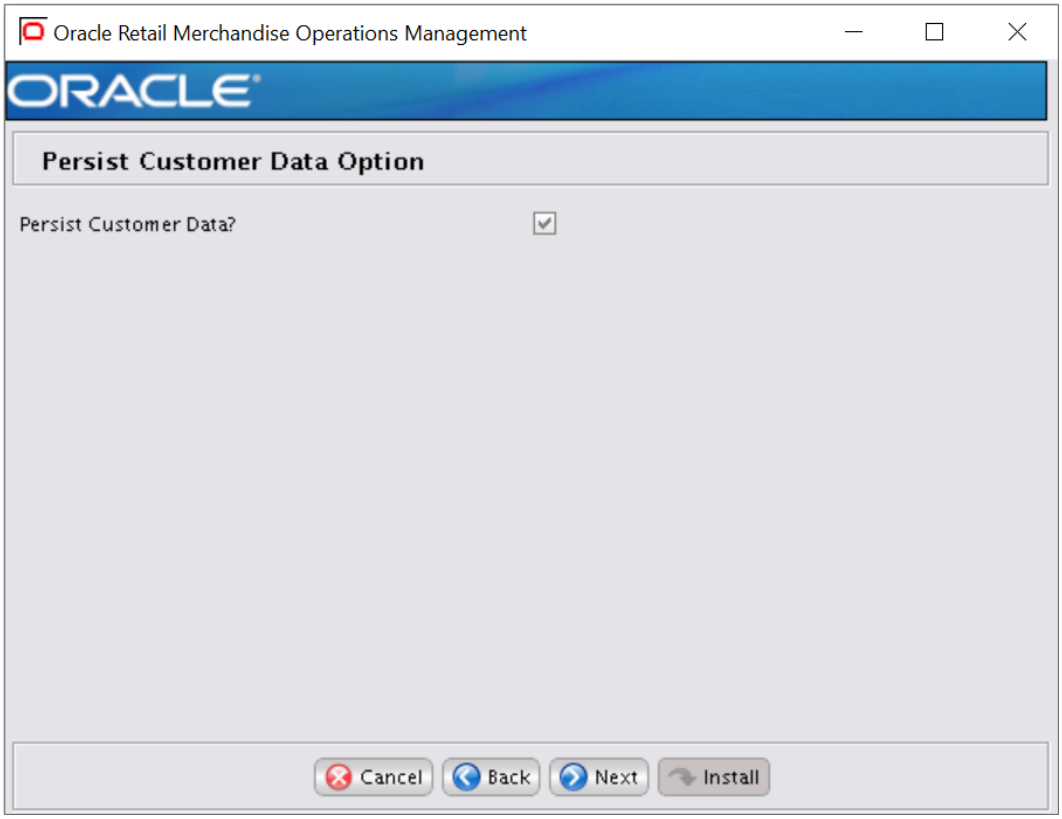
Field Title	Full install or Patch
Field Description	Gives an option to chose full or Patch Install
Example	Full

Screen: Host Details



Field Title	Hostname
Field Description	Provide the hostname where the RMS Application is being installed.
Example	apphostname

Screen: Persist Customer Data Option

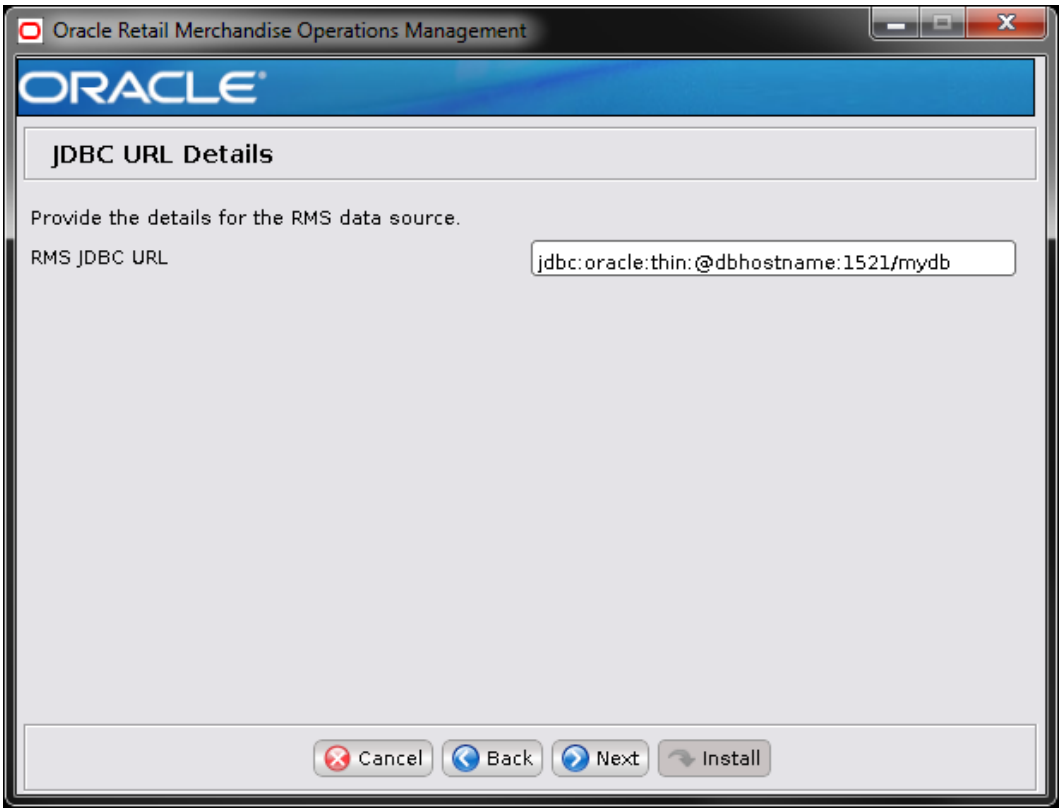


Field Title	Persist Customer Data?
Field Description	Check the checkbox if you would like to persist the customer PII in the database, else uncheck it. This is used for implementation of GDPR requirement

Screen: JDBC Security Details

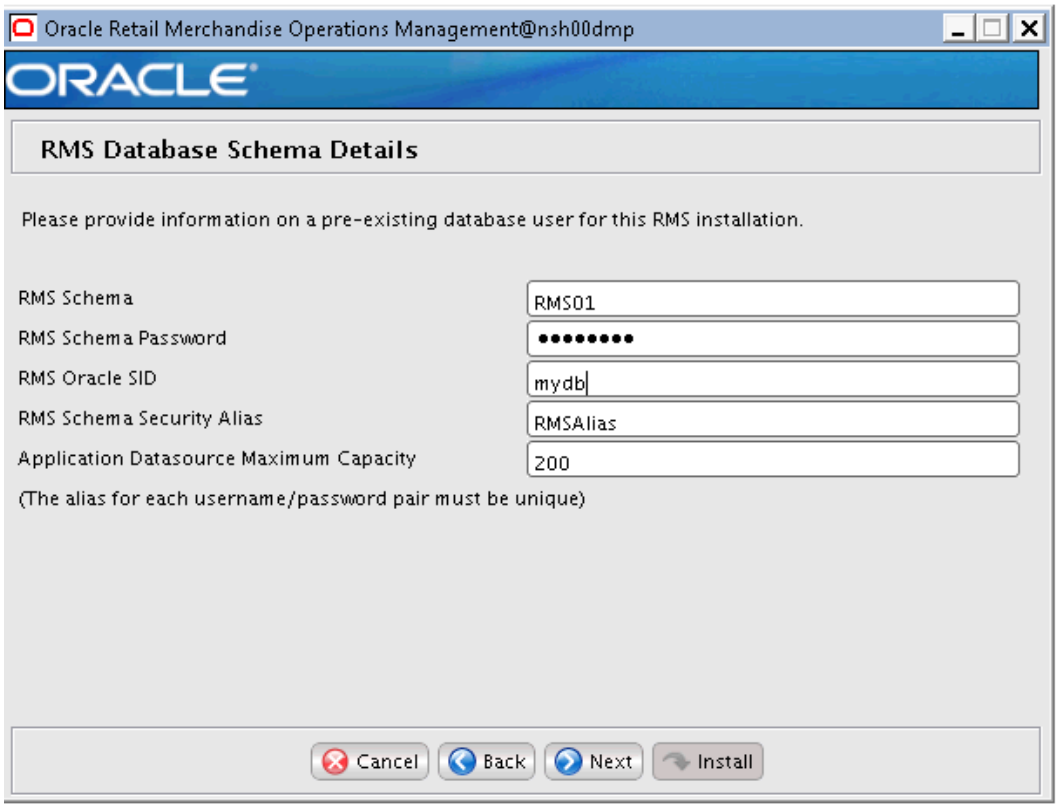
Field Title	Enable Secure JDBC connection
Field Description	Select Yes to create secured data sources in WebLogic, otherwise choose No. A secure data base connection must already be set up if you want to create a secure data source.

Screen: JDBC URL Details



Field Title	RMS JDBC URL
Field Description	URL used by the RMS application to access the RMS database schema. See Appendix: URL Reference for expected syntax.
Examples	For Non Secure JDBC Connection: jdbc:oracle:oci:@mydb or jdbc:oracle:thin:@dbhostname:1521/mydb For Secure JDBC Connection: jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS_LIST=(ADDRESS=(PROTOCOL=tcps)(HOST=dbhostname)(PORT=2484)))(CONNECT_DATA=(SERVICE_NAME=mydb)))

Screen: RMS Database Schema Details



Field Title	RMS schema
Field Description	Provide the RMS database user here. The installer logs into the database as this user to create the RMS schema and uses it to compile RMS batch. This user must already exist in the database when the RMS database schema installer is run.
Example	RMS01

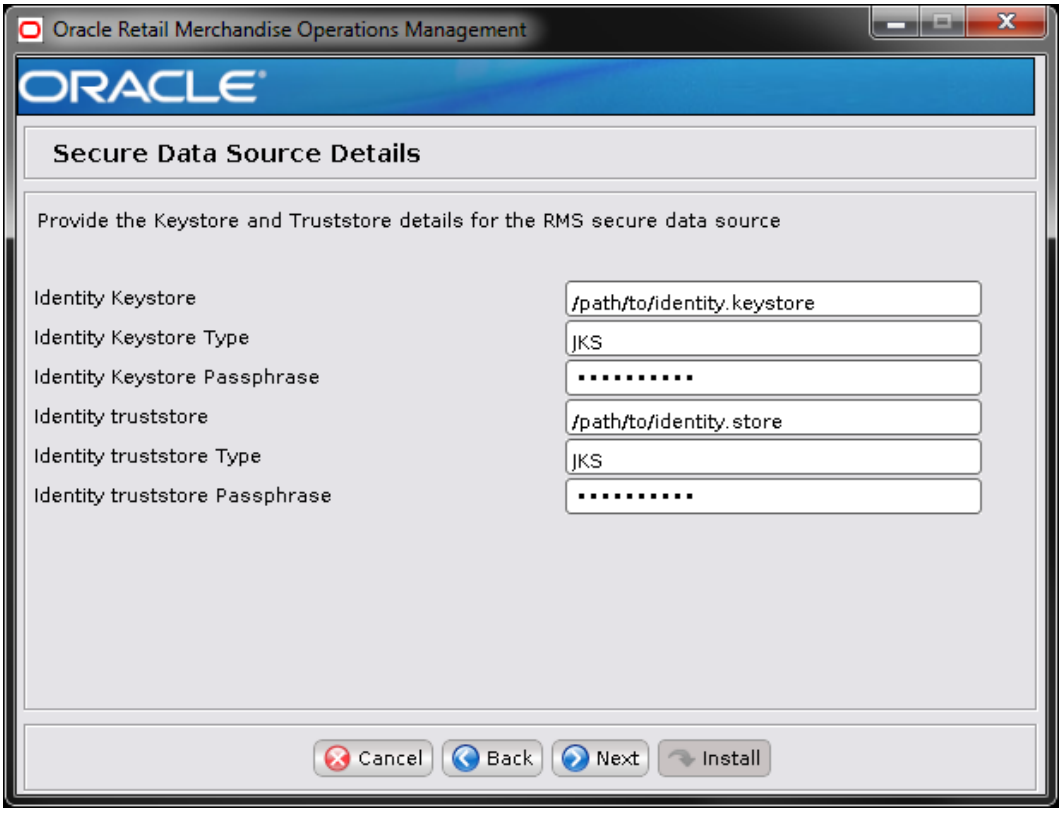
Field Title	RMS schema password
Field Description	Database password for the RMS Schema Owner.

Field Title	RMS Oracle SID
Field Description	Oracle system identifier for the database where RMS will be installed
Example	mydb

Field Title	RMS Schema Security Alias
Field Description	The alias to store the schema credentials.
Example	RMSAlias
Note	This alias must be unique. Do not use the same value for any other alias fields in the installer. If the same alias is used, entries in the wallet can override each other and cause problems with the application.

Field Title	Application Data Source Maximum Capacity
Field Description	Maximum number of database connections in connection pool for this data source in WebLogic Server
Example	200

Screen: Secure Data Source Details



Note: This screen appears only if you have enabled 'Secure JDBC' for RMS. Ignore this step in case you have not enabled 'Secure JDBC' for RMS.

Field Title	Identity Keystore
Field Description	Keystores ensure the secure storage and management of private keys and trusted certificate authorities (CAs). This screen lets you provide the keystore to be used for datasource connection These settings help you to manage the security of message transmissions. For further information, please refer *MOM security Guide.* Location or path where identity keystore file is stored.

Field Title	Identity Keystore Type
Field Description	Type of the identity keystore used. Example: jks

Field Title	Identity Keystore Passphrase
Field Description	The password to access the keystore mentioned above.

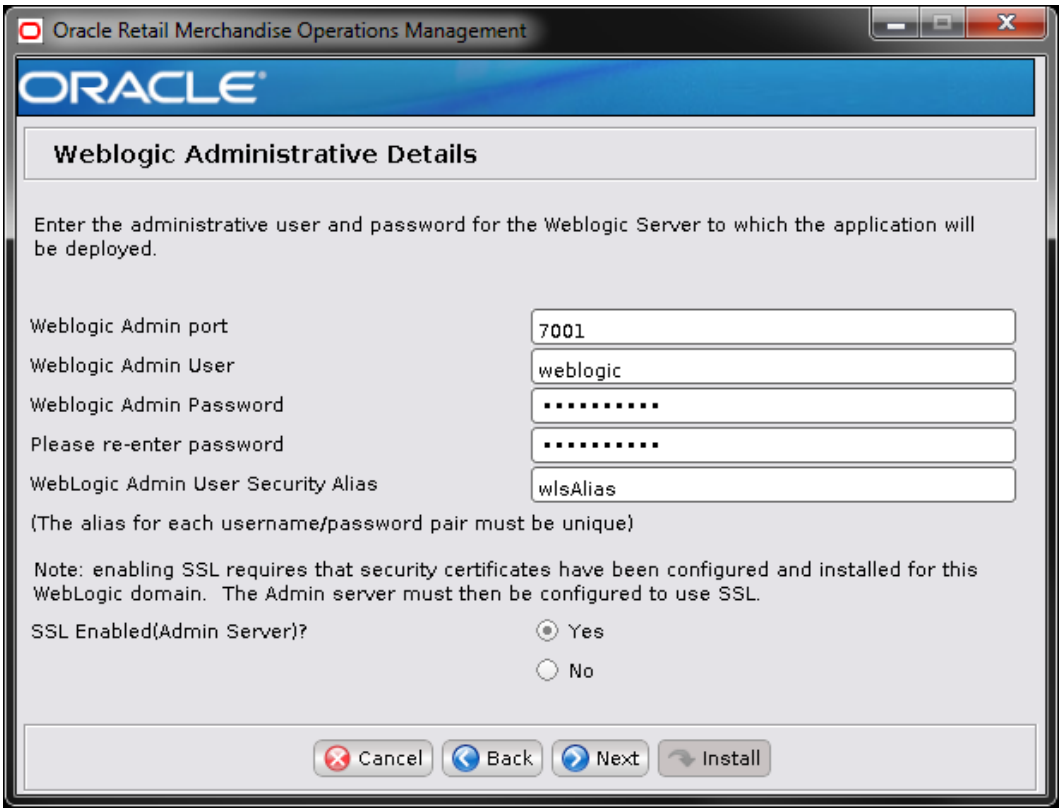
Field Title	Identity truststore
Field Description	Location or path where identity truststore file is stored.

Field Title	Identity truststore Type
Field Description	Type of the identity truststore used. Example: jks

Field Title	Identity truststore Passphrase
Field Description	The password to access the truststore mentioned above.

The database settings provided are validated by the installer when you advance to the next screen.

Screen: WebLogic Administrative Details



Field Title	Weblogic Admin port
Field Description	Listen port for the WebLogic Admin server.
Example	7001

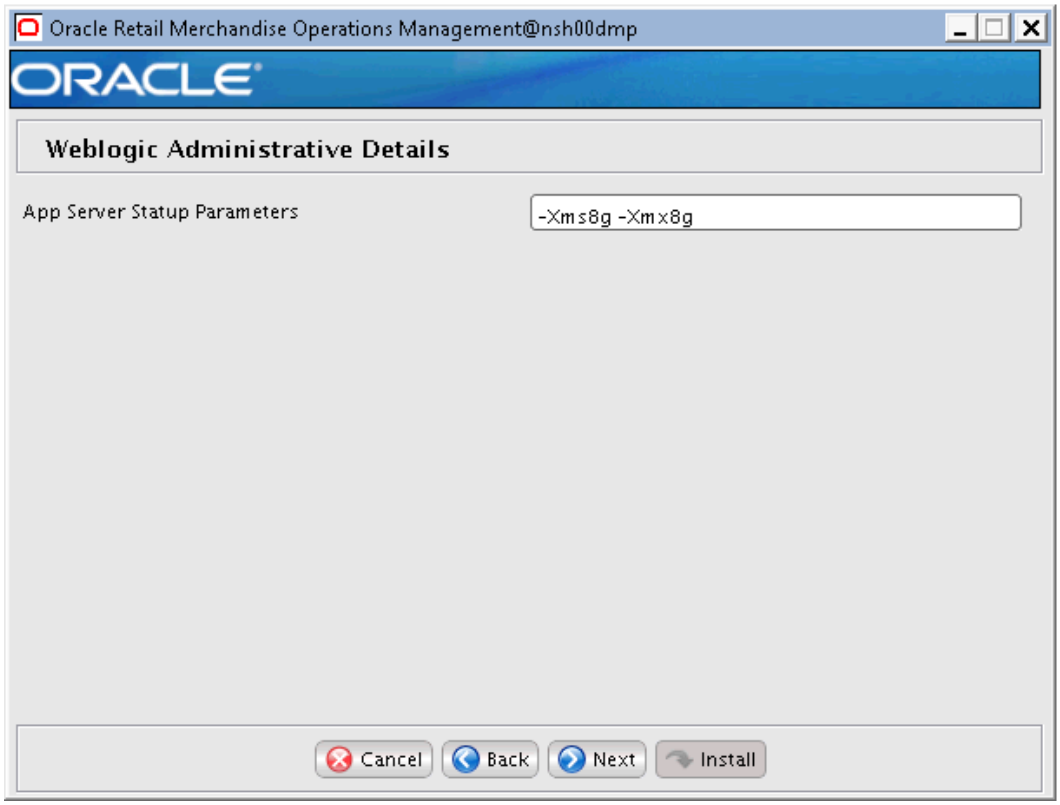
Field Title	Weblogic Admin User
Field Description	Username of the admin user for the WebLogic instance to which the RMS application will be deployed
Example	weblogic

Field Title	Weblogic Admin Password
Field Description	Password for the WebLogic admin user. You chose this password when you created the WebLogic instance.

Field Title	Weblogic Admin User Security Alias
Field Description	An alias for the WebLogic admin user.
Example	wlsAlias
Note	This alias must be unique. Do not use the same value for any other alias fields in the installer. If the same alias is used, entries in the wallet can override each other and cause problems with the application.

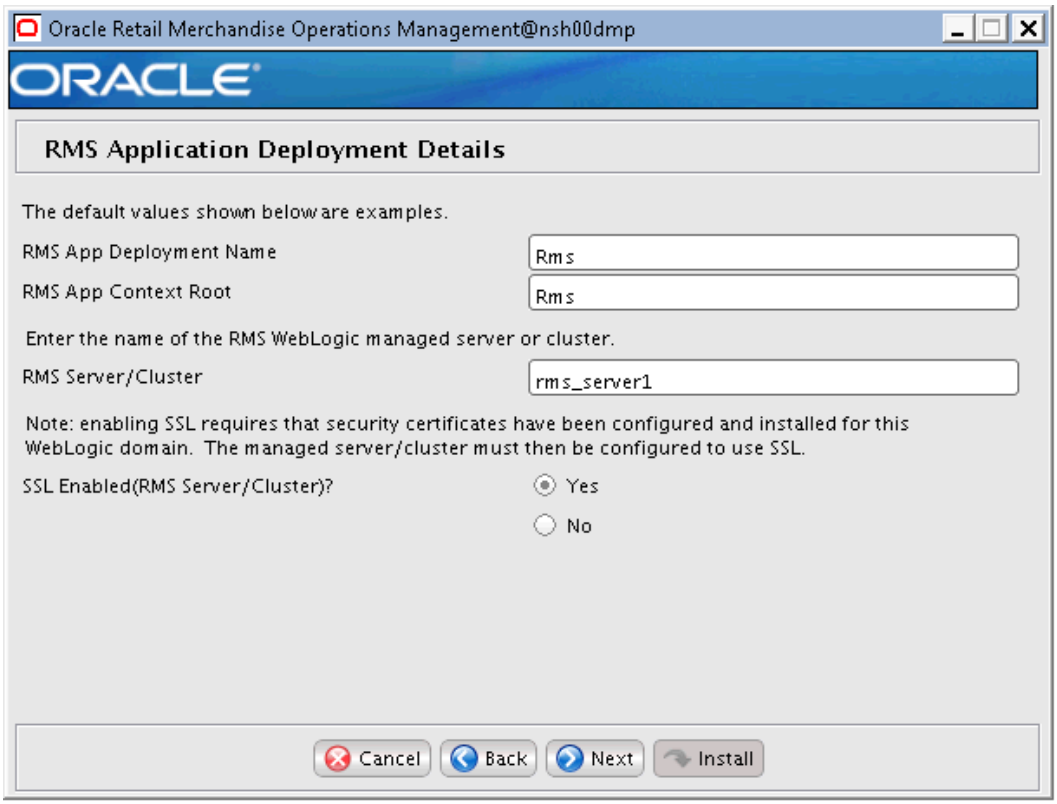
Field Title	SSL Enabled(Admin Server)?
Field Description	Choose Yes to install RMS using a WebLogic admin server configured to use SSL. In this case, SSL must be configured and the ports must be enabled for the admin server. Choose No to install using a WebLogic admin server configured without SSL. In this case the non-SSL ports must be enabled for the admin server.

Screen: WebLogic Administrative Details



Field Title	App Server Startup Parameters
Field Description	Provide initial and maximum heap size for the JVM of RMS WebLogic server during the server startup

Screen: RMS Application Deployment Details



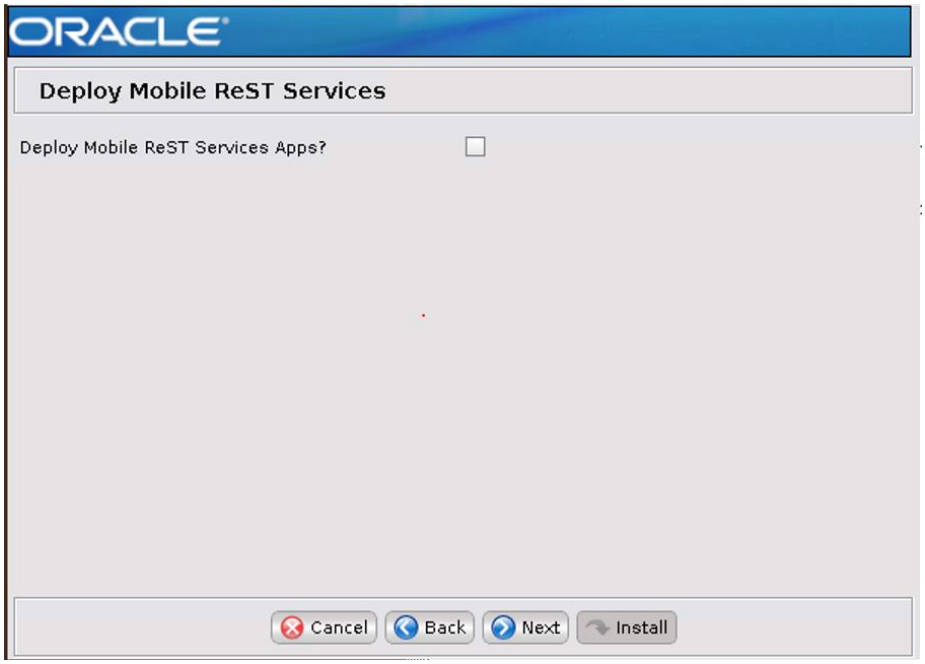
Field Title	RMS App Deployment Name
Field Description	Name by which this RMS application is identified in the application server.
Example	Rms

Field Title	RMS App Context Root
Field Description	Context root of the application, so that we can access the application with updated context root as follows. If the context root is set as Rms, then RMS url would look as follows https://<host>:<port>/Rms/faces/RmsHome
Example	Rms

Field Title	RMS server/cluster
Field Description	Name of the RMS WebLogic managed server or cluster.
Example	rms-server

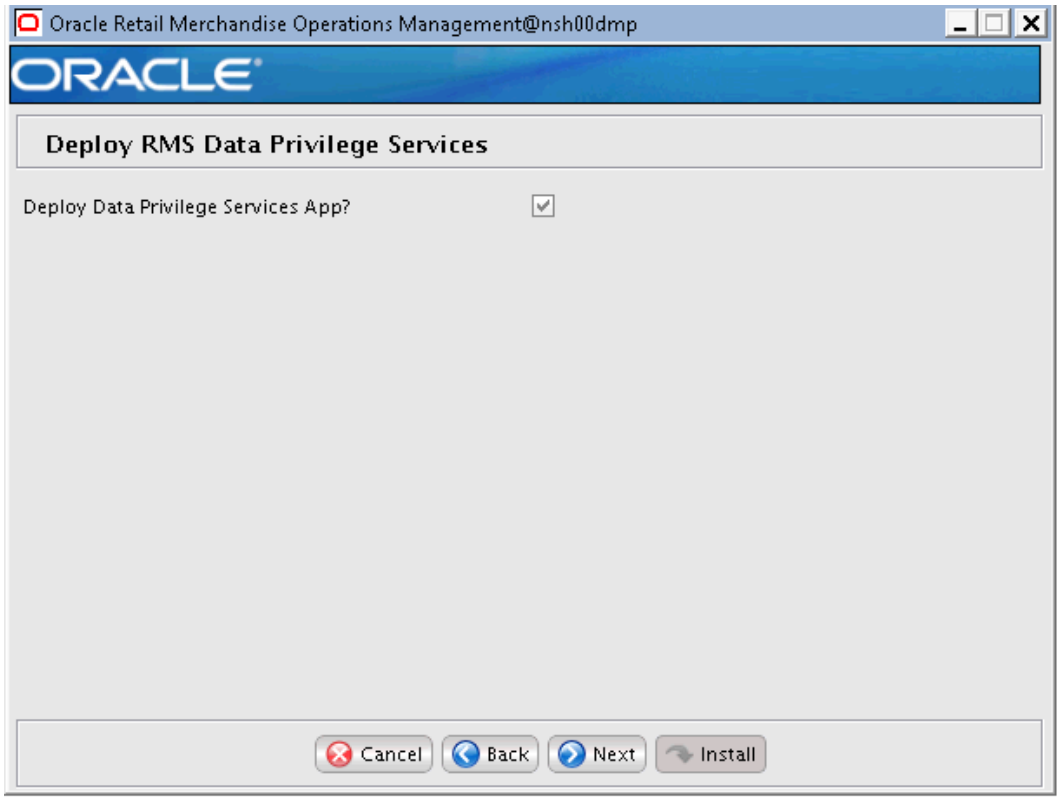
Field Title	SSL Enabled(RMS Server/Cluster)?
Field Description	Choose Yes to install RMS into a managed server/cluster configured to use SSL. In this case, SSL must be configured and the ports must be enabled for the managed server/cluster. Choose No to install into a managed server/cluster configured without SSL. In this case the non-SSL ports must be enabled for the managed server/cluster.

Screen: Deploy RMS ReST Services



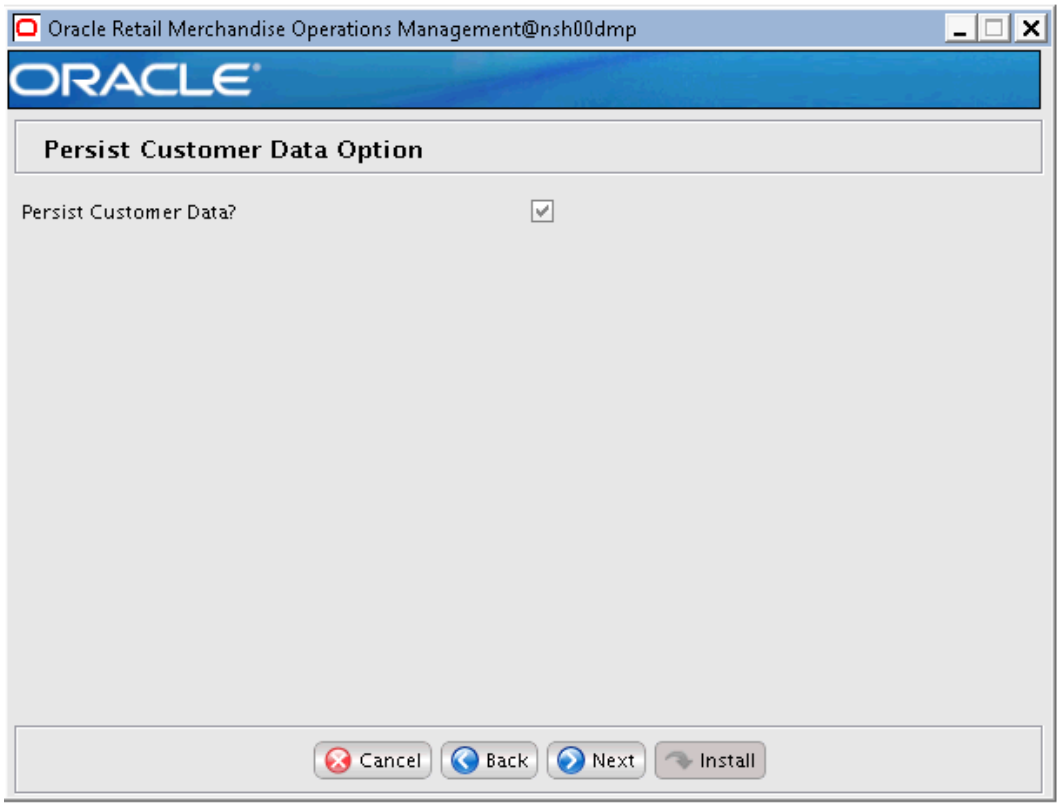
Field Title	Deploy Mobile ReST Services?
Field Description	Option to deploy RMS ReST Services

Screen: Deploy RMS Data Privilege Services



Field Title	Deploy Data Privilege Services App
Field Description	Option to deploy Data Privilege services to exercise the right to access/ forget GDPR standards on PII data.

Screen: Persist Customer Data Option

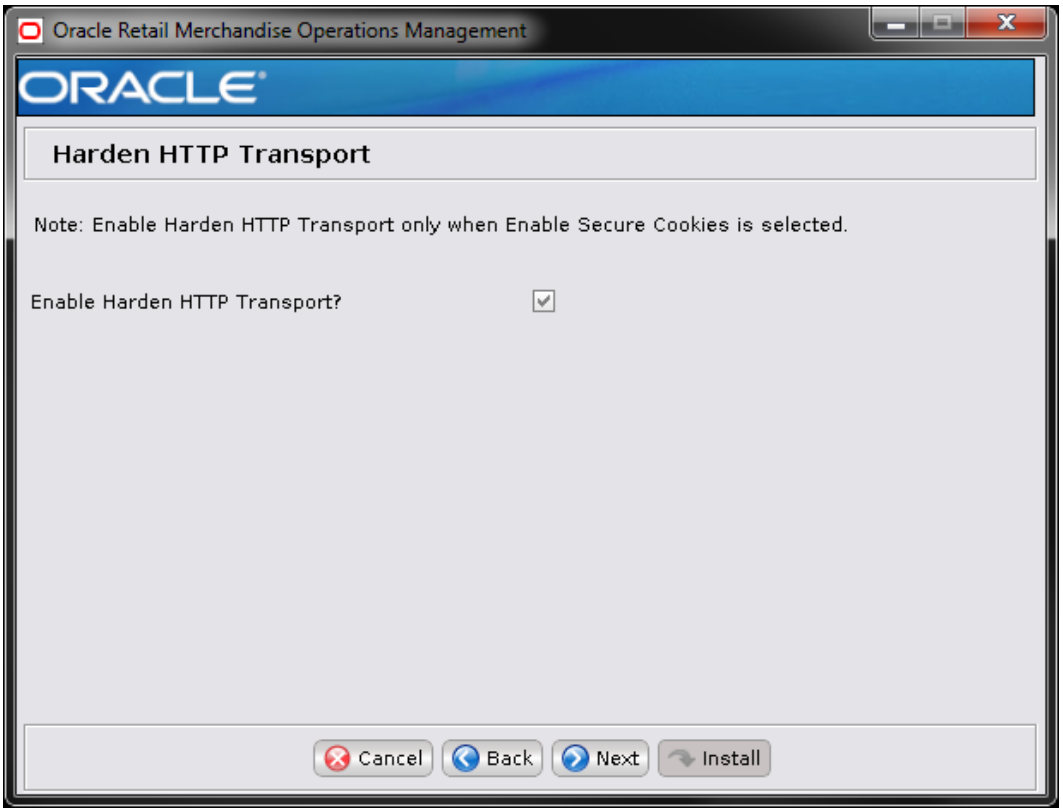


Field Title	Persist Customer Data?
Field Description	Option to persist customer PII data for GDPR

Screen: Enable Secure Cookies using JSESSIONID Flag

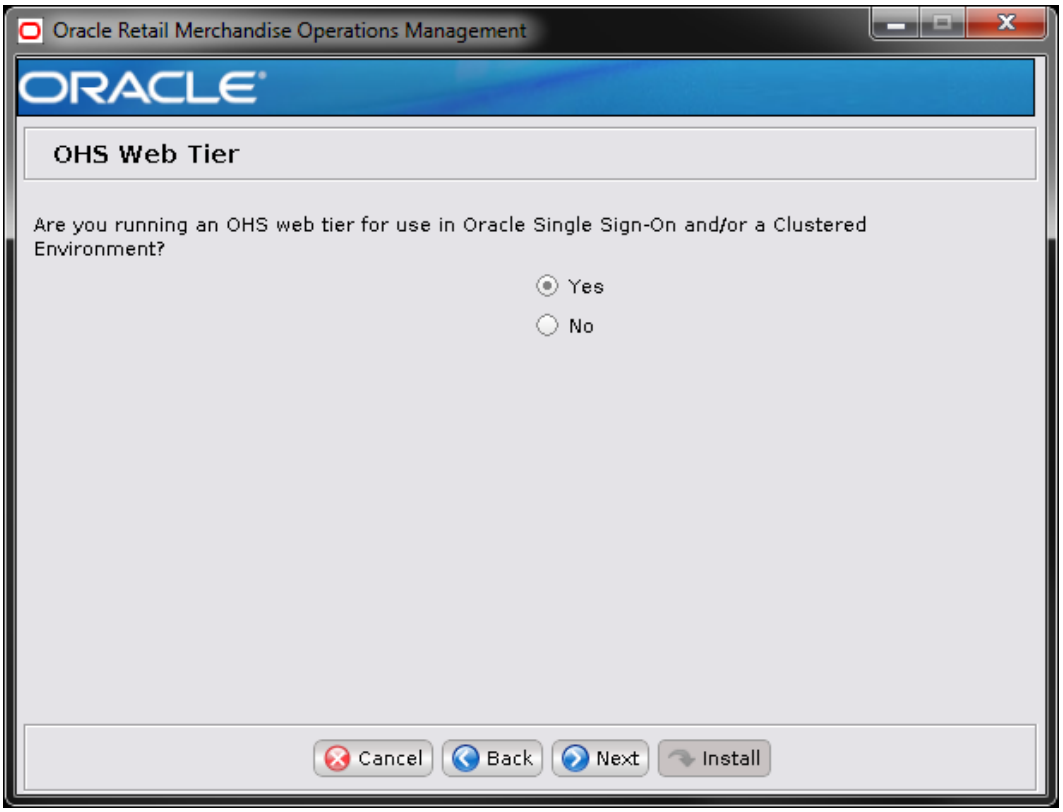
Field Title	Enable Secure Cookies using JSESSIONID Flag?
Field Description	Selecting “Yes” will enable secure cookies using JSESSIONID Flag. Selecting “No” will not enable secure cookies using JSESSIONID Flag.

Screen: Harden HTTP Transport



Field Title	Enable Harden HTTP Transport?
Field Description	Selecting “Yes” will enable Harden HTTP Transport. Selecting “No” will not enable Harden HTTP Transport.

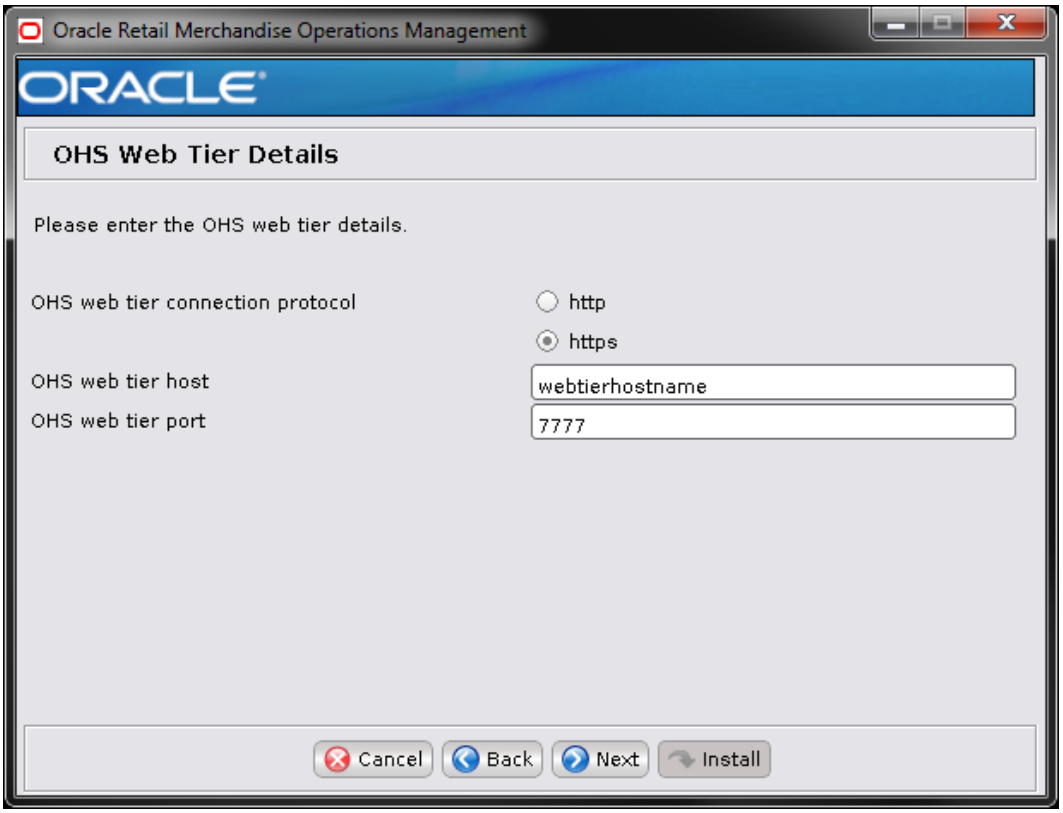
Screen: OHS Web Tier



Field Title	Are you running an OHS web tier for use in Oracle Single Sign-On and/or a Clustered Environment?
Field Description	Selecting the option 'Yes' will configure all the application URLs delivered by this installer with the OHS webtier hostname and port that will be entered in the next screen. Selecting option 'No' will result in application URLs having default hosts and ports.

Screen: OHS Web Tier Details

This screen appears only if you have selected Yes in the previous screen.

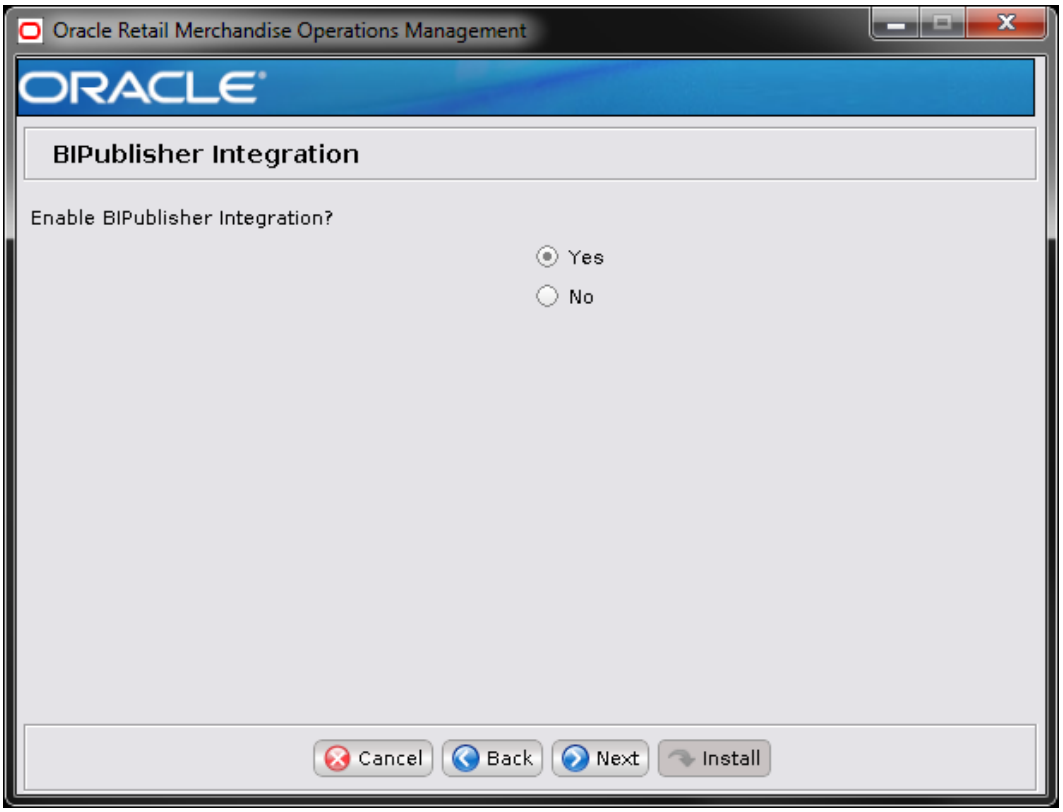


Field Title	OHS web tier connection protocol
Field Description	Connection protocol for OHS web tier – http or https

Field Title	OHS web tier host
Field Description	Host name for OHS web tier
Example	webtierhostname

Field Title	OHS web tier port
Field Description	Port number for OHS web tier
Example	7777

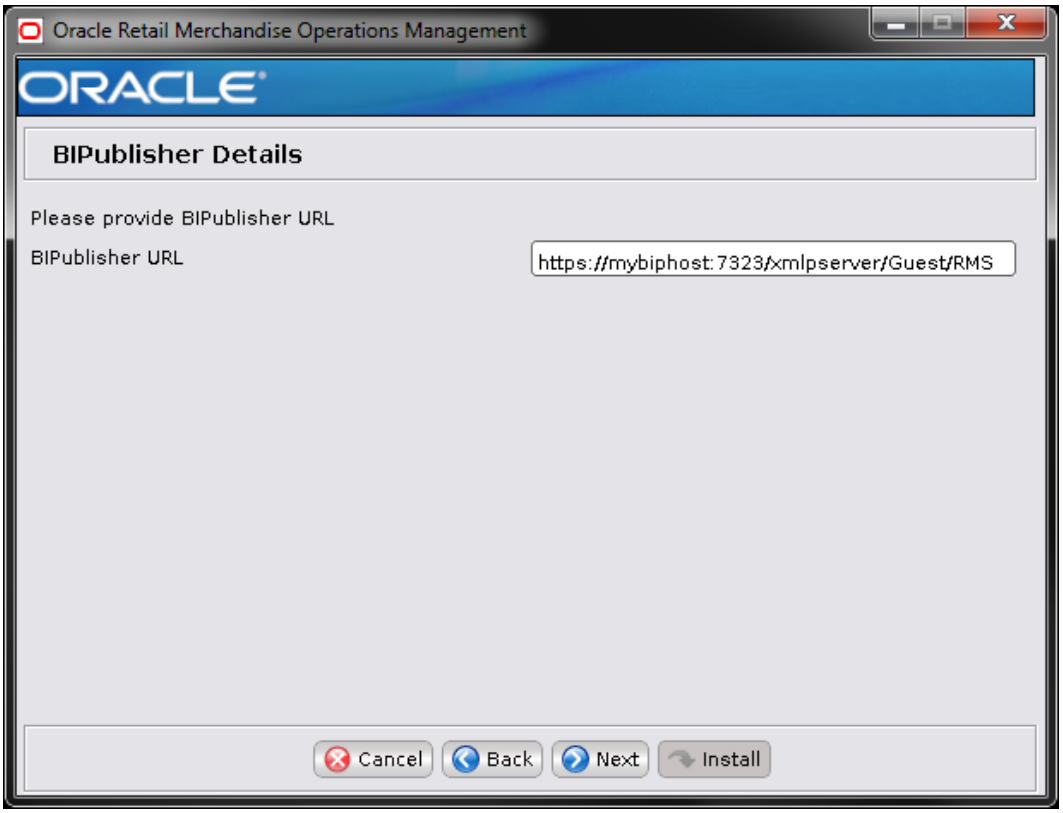
Screen: Enable BIPublisher Integration?



Field Title	Enable BIPublisher Integration?
Field Description	Selecting the option 'Yes' will configure RMS to use the BIPublisher url that will be entered in the next screen. Selecting option 'No' will result RMS not being configured to use BIPublisher.

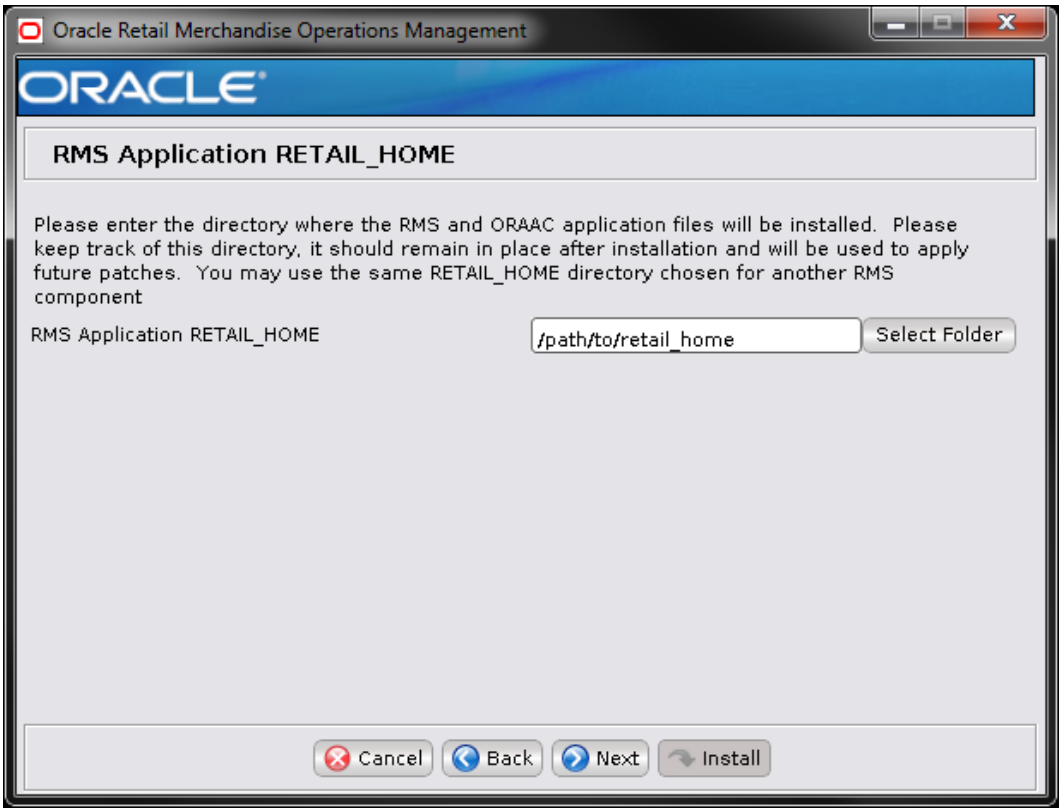
Screen: BIPublisher Details

This screen appears only if you have selected Yes in the previous screen.



Field Title	BIPublisher URL
Field Description	The url through which the RMS application will access RMS BIPublisher reports
Example	https://mybiphost:7323/xmlpserver/Guest/RMS

Screen: RMS Application RETAIL_HOME



Field Title	RMS Application RETAIL_HOME
Field Description	Retail Home is used to keep Orpatch related files, batches etc. by default. Please keep track of this directory, it should remain in place after installation and will be used to apply future patches.
Example	/path/to/retail_home
Note	If upgrading an existing RMS installation, please choose a RETAIL_HOME location different than one used previously to install an RMS 15.0.x or earlier application. If you have selected an existing RETAIL_HOME, and it has been configured to run other components than the ones you have selected for this installation, those components will also be installed regardless of what you selected on the Component Selection screen.

H

Appendix: RMS Analyze Tool

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. The installer 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. See the section "Analyzing the Impact of a Patch" in the chapter "RMS Patching Procedures" for more details.

Run the RMS Analyze Tool

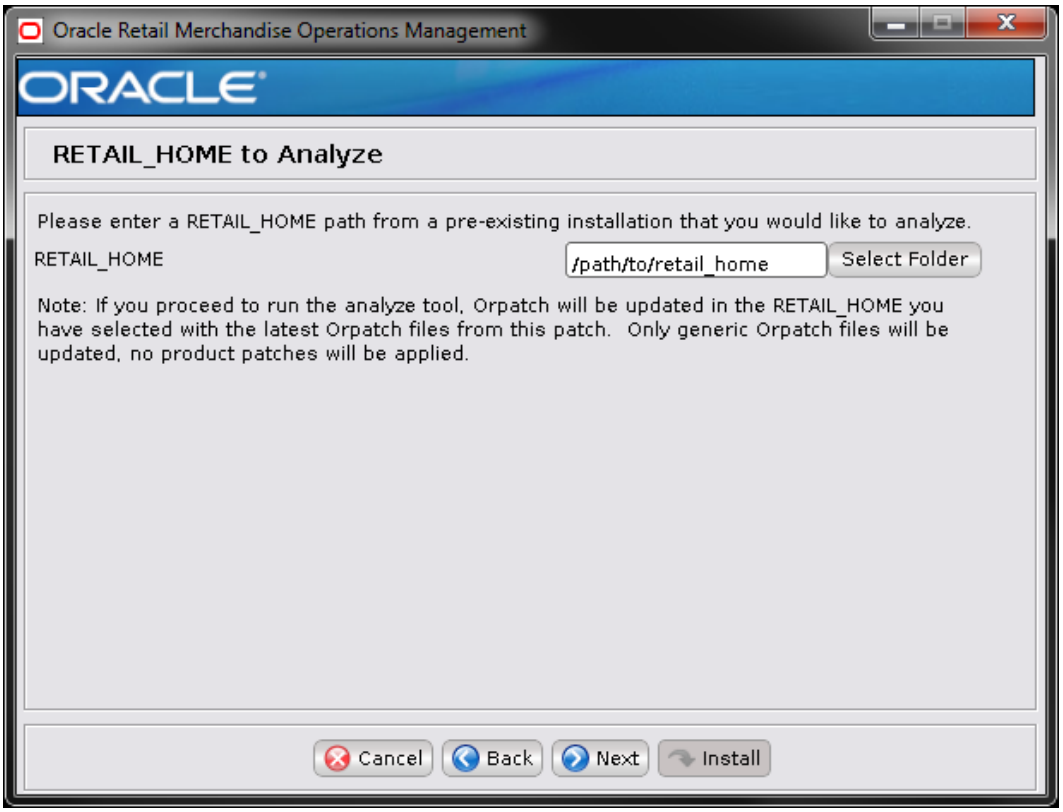
1. Log onto the server as a user with access to the RETAIL_HOME for the installation you want to analyze.
2. Change directories to STAGING_DIR/rms/installer. STAGING_DIR is the location where you extracted the installer.
3. Set and export the following environment variables.

Variable	Description	Example
JAVA_HOME	Location of a Java 1.8 JDK.	JAVA_HOME= /u00/webadmin/java/jdk1.8 export JAVA_HOME
DISPLAY	Address and port of X server on desktop system of user running install. Optional when running the Analyze tool	DISPLAY=<IP address>:0.0 export DISPLAY

4. 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.
5. Run the analyze.sh script to start the analyze tool.

> **Note:** Below are the usage details for analyze.sh. The typical usage for GUI mode is no arguments.
>
> ./analyze.sh [text | silent]

Screen: RETAIL_HOME to Analyze



Field Title	RETAIL_HOME
Field Description	The pre-existing location where RMS (database, batch, and/or application) was installed along with the ORPATCH utility. This location should contain directories with your installed files as well as the “orpatch” directory.
Example	/path/to/retail_home
Note	The ORPatch files in this RETAIL_HOME may need to be updated in order to be able to run the analysis. The Analyze tool will take care of this automatically.

1. After clicking “install”, the Analyze tool will generate a report of the files that will be patched if you apply this patch to the selected RETAIL_HOME. A high-level report can be found in the log file: STAGING_DIR/rms/installer/logs/rms-analyze.<timestamp>.log.

   The detailed list of patch files can be found in RETAIL_HOME/ orpatch/logs/detail_logs/analyze/details/

I

Appendix: Installer Silent Mode

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

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: install.sh silent

J

Appendix: URL Reference

This appendix provides URL reference information.

JDBC URL for a Database

Used by the Java application and by the installer to connect to the database.

Thick Client Syntax: jdbc:oracle:oci:@<sid>

<sid>: system identifier for the database

Example: jdbc:oracle:oci:@mysid

Thin Client Syntax: jdbc:oracle:thin:@<host>:<port>:<sid>

<host>: hostname of the database server

<port>: database listener port

<sid>: system identifier for the database

Example: jdbc:oracle:thin:@myhost:1521:mysid

K

Appendix: Common Installation Errors

This section provides some common errors encountered during installation of RMS.

Database 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.

Warning: Could Not 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.

Unresponsive Country and Currency Drop-Downs

Symptom

In GUI mode, when you click on the drop-down list selection for the primary country or currency, the list does not appear, and this message appears in the console window:

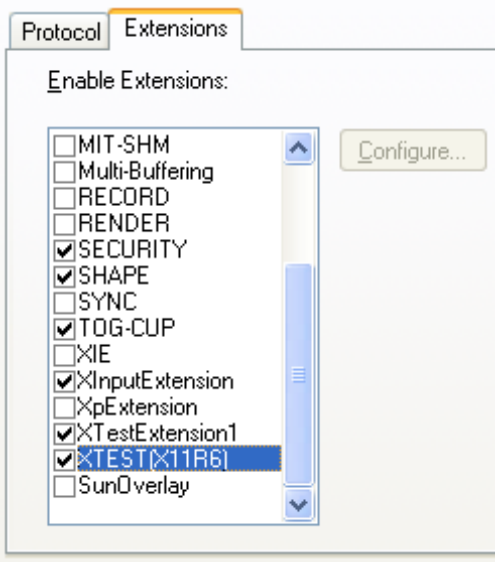
```
XTEST extension not installed on this X server: Error 0
```

Solution

To run the RMS installer in GUI mode you must have the XTEST extension enabled in your X server.

To Enabling XTEST in Exceed, do the following.

1. Open Xconfig to edit Exceed configuration.
2. Go to the X Server Protocol settings.
3. Click on the Extensions tab.
4. Make sure that the XTEST extension is selected, as shown.



5. Restart the X Server and re-run the RMS installer.

Could Not Execl Robot Child Process: Permission Denied

Symptom

When opening a drop-down list in GUI mode of the RMS installer, the installer freezes up and displays the following message in the console:

```
Couldn't execl robot child process: Permission denied
```

Solution

As the owner of the database ORACLE_HOME (i.e. **oracle**), grant execute permissions to the awt_robot* files under $ORACLE_HOME/jdk/jre/lib. The database schema installer uses $ORACLE_HOME/jdk for its JAVA_HOME.

Example (using SUN Solaris):

```
chmod a+x $ORACLE_HOME/jdk/jre/lib/sparc/awt_robot
chmod a+x $ORACLE_HOME/jdk/jre/lib/sparcv9/awt_robot
```

ConcurrentModificationException in Installer GUI

Symptom

In GUI mode, the errors tab shows the following error:

```
java.util.ConcurrentModificationException
            at
java.util.AbstractList$Itr.checkForComodification(AbstractList.java:448)
            at java.util.AbstractList$Itr.next(AbstractList.java:419)
… etc
```

Solution

You can ignore this error. It is related to third-party Java Swing code for rendering of the installer GUI and does not affect the retail product installation.

ORA-04031 (Unable to Allocate Memory) Error During Database Schema Installation

Symptom

When running the database schema installer you get the following error one or more times:

```
[ora:sqlplus] alter package
[ora:sqlplus] *
[ora:sqlplus] ERROR at line 1:
[ora:sqlplus] ORA-04031: unable to allocate 92120 bytes of shared memory ("shared
[ora:sqlplus] pool","unknown object","PL/SQL MPCODE","BAMIMA: Bam Buffer")
```

Solution

There was not enough available memory in the shared pool on the database at the time of compilation. There are several choices to get past this error:

- Log into the database and attempt to recompile invalid objects in the database schema. Subsequent attempts to compile the same object(s) can be successful.
- Have a DBA increase the shared pool size on the database and re-run the installer from scratch on a new schema user.

RIB Errors

At random times, the RIB will get certain errors such as GETNXT(?,?,?,?,?,?,?) and/or ORA-21700 object does not exist or is marked for delete. This is very confusing because you may research and find that the object exists and is valid.

You must re-initialize the reference to reference an existing object as follows.

1. Bring down the RIB in question.
2. Run /RIB_INSTALL_DIR>/InstallAndCompileAllRibOracleObjects.sql.
3. Run another object validate script (ex: inv_obj_comp.sql) to make sure objects are valid. (Some may have deal locked in the end of the previous step.)
4. Bring up the RIB in question.

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. If you 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.
```

It may mean that the installer is using the incorrect library path variables for the platform you are installing on. Open the file <STAGING_DIR>/rms/installer/common/preinstall.sh and make sure the variable use32bit is set to True if you are on a 32 bit platform, and False if you are on a 64 bit platform.

Multi-Threaded OCI Client Dumps Core after Reconnecting To Database

Symptom

If a multi-threaded Oracle client process that uses OCI to connect to a remote database loses connectivity with the database, it tries to reconnect and the client program continues to run. The program then dumps the core with the following stack trace, when Automatic Diagnostic Repository (ADR) is enabled.

```
skgfqio sdbgrfbibf_io_block_file dbgrfrbf_read_block_file dbgrmflrp_read_page
dbgrmblgmp_get_many_pages dbgrmmdrrmd_read_relation_meta_data
dbgrmmdora_open_record_access_full
dbgriporc_openrel_wcreate dbgrip_open_relation_access dbgrip_start_iterator
dbgrip_relation_iterator dbgruprac_read_adrctl...
```

Solution

Oracle Retail recommended you disable ADR (diag_adr_enabled=OFF, a sqlnet.ora parameter) while using multi-threaded OCI/OCCI application. diag_adr_enabled was

introduced in Oracle 11g as a new method of tracing ADR. This will dump additional trace details.

Disabling 'diag_adr_enabled' does not disturb any functionality. Therefore, it can safely be unset by doing diag_adr_enabled=off in sqlnet.ora. However, if you still want tracing, you can have following parameters/variables set in sqlnet.ora:

```
trace_level_server=16 -- for server side NET tracing
trace_level_client=16 -- for client side NET tracing
```

For additional information on how to set traditional tracing, see the My Oracle Support document, “SQL*Net, Net8, Oracle Net Services - Tracing and Logging at a Glance” (ID 219968.1).

Error Compiling Batch

Symptom

```
[exec] 14:41:21 10/26/2016 Executing make -f retek.mk  retek rms resa 2>&1
     [exec] 14:41:21 10/26/2016 ------- Start Pro*C Compilation Errors output ----
---
     [exec] make: Fatal error: Command failed for target `saoranumadd.o'
     [exec] 14:41:21 10/26/2016 -------  End  Pro*C Compilation Errors output ----
---
     [exec] 14:41:21 10/26/2016 ERROR: 1 errors while compiling, see
/u00/projects/rms/oracle/lib/src/libretek.log for full details
     [exec] 14:41:21 10/26/2016 Executing make -f retek.mk install 2>&1
     [exec] 14:41:24 10/26/2016 Command succeeded
     [exec] 14:41:24 10/26/2016 Errors while compiling libraries, attempting proc
compile
     [exec] 14:41:24 10/26/2016 Compiling Pro*C batch in
/u00/projects/rms/oracle/proc/src
     [exec] 14:41:24 10/26/2016 Moving /u00/projects/rms/oracle/proc/bin to
/u00/projects/rms/oracle/proc/bin-10262016-144124
     [exec] 14:41:24 10/26/2016 Executing make -f mts.mk  clobber 2>&1
     [exec] 14:41:25 10/26/2016 Command succeeded
     [exec] 14:41:25 10/26/2016 Executing { make -f mts.mk  depend || make -f
mts.mk  depend ; } 2>&1
     [exec] 14:41:26 10/26/2016 Command succeeded
     [exec] 14:41:26 10/26/2016 Executing make -f mts.mk
PRODUCT_PROCFLAGS=dynamic=ansi ditinsrt 2>&1
     [exec] 14:41:36 10/26/2016 ------- Start Pro*C Compilation Errors output ----
---
     [exec] make: Fatal error: Command failed for target `ditinsrt.o'
     [exec] make: Fatal error: Command failed for target `ditinsrt'
     [exec] 14:41:36 10/26/2016 -------  End  Pro*C Compilation Errors output ----
---
     [exec] 14:41:36 10/26/2016 ERROR: 2 errors while compiling, see
/u00/projects/rms/oracle/proc/src/srcditinsrt.log for full details
     [exec] 14:41:36 10/26/2016 Executing make  -f mts.mk  rms-ALL recs-ALL resa-
ALL rtm-ALL fif-ALL 2>&1
     [exec] 14:41:38 10/26/2016 ------- Start Pro*C Compilation Errors output ----
---
     [exec] make: Fatal error: Command failed for target `ang_prcqtydnld.o'
     [exec] make: Fatal error: Command failed for target `rms-ALL'
     [exec] 14:41:38 10/26/2016 -------  End  Pro*C Compilation Errors output ----
---
     [exec] 14:41:38 10/26/2016 ERROR: 2 errors while compiling, see
/u00/projects/rms/oracle/proc/src/srcall.log for full details
     [exec] 14:41:38 10/26/2016 Executing make -f mts.mk install 2>&1
     [exec] 14:41:38 10/26/2016 Command succeeded
     [exec] 14:41:43 10/26/2016 Errors while compiling Proc*C batch
     [exec] 14:41:43 10/26/2016 Failed to complete Post-action for action:
RMSBATCH
```

```
[exec] 14:41:43 10/26/2016 ORPatch session completed with errors
```

Solution

Ensure that the PATH variable is set to correct compiler for the respective operating systems.

L

Appendix: Single Sign-On for WebLogic

Single Sign-On (SSO) is a term for the ability to sign onto multiple Web applications via a single user ID/Password. There are many implementations of SSO. Oracle provides an implementation with Oracle Access Manager.

Most, if not all, SSO technologies use a session cookie to hold encrypted data passed to each application. The SSO infrastructure has the responsibility to validate these cookies and, possibly, update this information. The user is directed to log on only if the cookie is not present or has become invalid. These session cookies are restricted to a single browser session and are never written to a file.

Another facet of SSO is how these technologies redirect a user's Web browser to various servlets. The SSO implementation determines when and where these redirects occur and what the final screen shown to the user is.

Most SSO implementations are performed in an application's infrastructure and not in the application logic itself. Applications that leverage infrastructure-managed authentication (such as deployment specifying Basic or Form authentication) typically have little or no code changes when adapted to work in an SSO environment.

What Do I Need for Single Sign-On?

A Single Sign-On system involves the integration of several components, including Oracle Identity Management and Oracle Access Management. This includes the following components:

- An Oracle Internet Directory (OID) LDAP server, used to store user, role, security, and other information. OID uses an Oracle database as the back-end storage of this information.
- An Oracle Access Manager (OAM) 12c Release server and administrative console for implementing and configuring policies for single sign-on.
- A Policy Enforcement Agent such as Oracle Access Manager 12c (WebGate), used to authenticate the user and create the Single Sign-On cookies.
- Oracle Directory Services Manager (ODSM) application in Oracle Identity Management (12.2.1.4), used to administer users and group information. This information may also be loaded or modified via standard LDAP Data Interchange Format (LDIF) scripts.
- Additional administrative scripts for configuring the OAM system and registering HTTP servers.

Additional WebLogic managed servers will be needed to deploy the business applications leveraging the Single Sign-On technology.

Can Oracle Access Manager Work with Other SSO Implementations?

Yes, Oracle Access Manager has the ability to interoperate with many other SSO implementations, but some restrictions exist.

Oracle Single Sign-on Terms and Definitions

The following terms apply to single sign-on.

Authentication

Authentication is the process of establishing a user's identity. There are many types of authentication. The most common authentication process involves a user ID and password.

Dynamically Protected URLs

A Dynamically Protected URL is a URL whose implementing application is aware of the Oracle Access Manager environment. The application may allow a user limited access when the user has not been authenticated. Applications that implement dynamic protection typically display a Login link to provide user authentication and gain greater access to the application's resources.

Oracle Identity Management (OIM) and Oracle Access Manager (OAM)

Oracle Identity Management (OIM) 12c includes Oracle Internet Directory and ODSM. Oracle Access Manager (OAM) should be used for SSO using WebGate. Oracle Forms 12c contains Oracle HTTP server and other Retail Applications will use Oracle WebTier12c for HTTP Server.

MOD_WEBLOGIC

mod_WebLogic operates as a module within the HTTP server that allows requests to be proxied from the OracleHTTP server to the Oracle WebLogic server.

Oracle Access Manager 12c Agent (WebGate)

Oracle WebGates are policy enforcement agents, which reside with relying parties and delegate authentication and authorization tasks to OAM servers.

Oracle Internet Directory

Oracle Internet Directory (OID) is an LDAP-compliant directory service. It contains user ids, passwords, group membership, privileges, and other attributes for users who are authenticated using Oracle Access Manager.

Partner Application

A partner application is an application that delegates authentication to the Oracle Identity Management Infrastructure. One such partner application is the Oracle HTTP Server (OHS) supplied with Oracle Forms Server or WebTier12c Server if using other Retail Applications other than Oracle Forms Applications.

All partner applications must be registered with Oracle Access Manager (OAM) 12c. An output product of this registration is a configuration file the partner application uses to verify a user has been previously authenticated.

Statically Protected URLs

A URL is considered to be Statically Protected when an Oracle HTTP server is configured to limit access to this URL to only SSO authenticated users. Any unauthenticated attempt to access a Statically Protected URL results in the display of a login page or an error page to the user.

Servlets, static HTML pages, and JSP pages may be statically protected.

What Single Sign-On is not

Single Sign-On is NOT a user ID/password mapping technology.

However, some applications can store and retrieve user IDs and passwords for non-SSO applications within an OID LDAP server. An example of this is the Oracle Forms Web

Application framework, which maps Single Sign-On user IDs to a database logins on a per-application basis.

How Oracle Single Sign-On Works

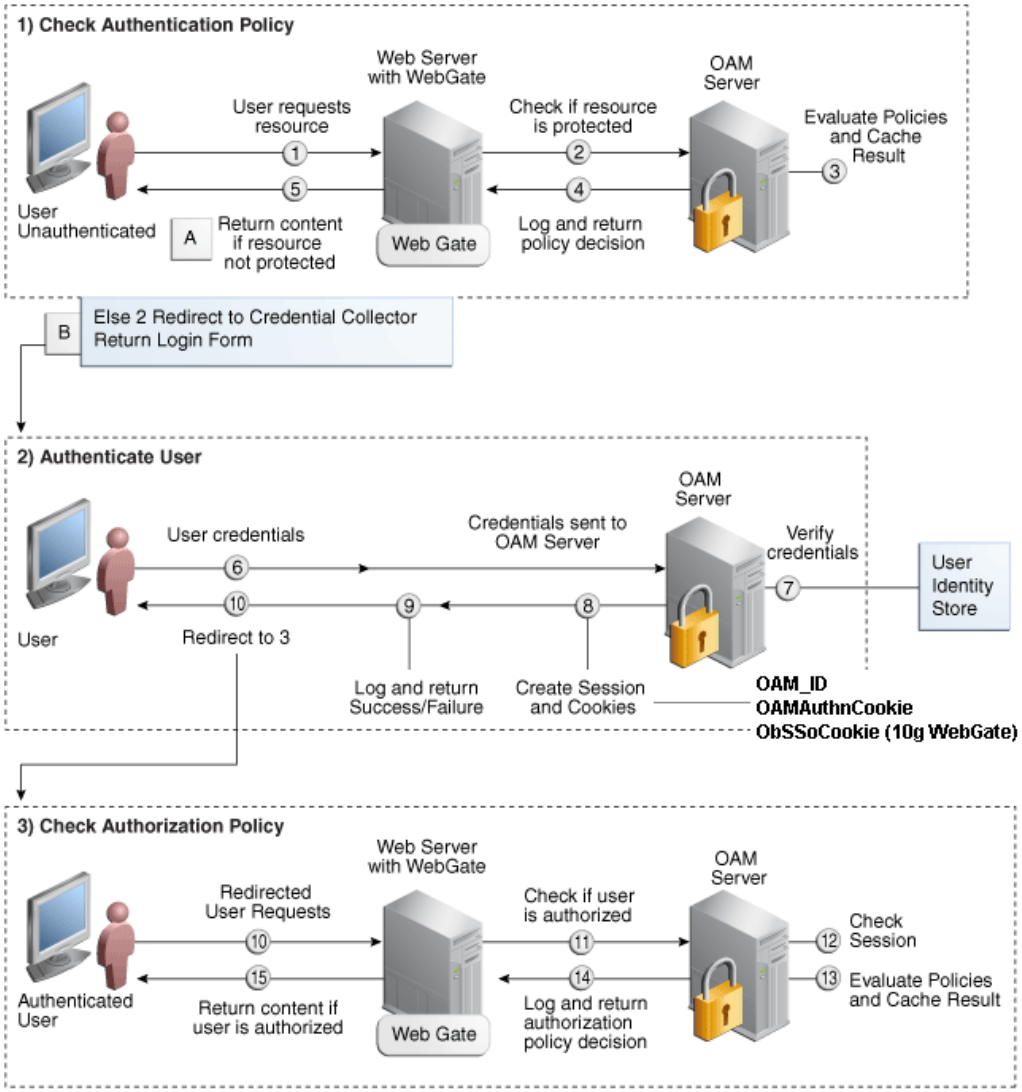
Oracle Access Manager involves several different components. These are:

- The Oracle Access Manager (OAM) server, which is responsible for the back-end authentication of the user.
- The Oracle Internet Directory LDAP server, which stores user IDs, passwords, and group (role) membership.
- The Oracle Access Manager Agent associated with the Web application, which verifies and controls browser redirection to the Oracle Access Manager server.
- If the Web application implements dynamic protection, then the Web application itself is involved with the OAM system.

About SSO Login Processing with OAM Agents

1. The user requests a resource.
2. Webgate forwards the request to OAM for policy evaluation
3. OAM:
   a. Checks for the existence of an SSO cookie.
   b. Checks policies to determine if the resource is protected and if so, how?
4. OAM Server logs and returns the decision
5. Webgate responds as follows:
   - **Unprotected Resource:** Resource is served to the user
   - **Protected Resource:**
     Resource is redirected to the credential collector.
     The login form is served based on the authentication policy.
     Authentication processing begins
6. User sends credentials
7. OAM verifies credentials
8. OAM starts the session and creates the following host-based cookies:
   - **One per partner:** OAMAuthnCookie set by 12c WebGates using authentication token received from the OAM Server after successful authentication.
     Note: A valid cookie is required for a session.
   - One for OAM Server: OAM_ID
9. OAM logs Success of Failure.
10. Credential collector redirects to WebGate and authorization processing begins.
11. WebGate prompts OAM to look up policies, compare them to the user's identity, and determine the user's level of authorization.
12. OAM logs policy decision and checks the session cookie.
13. OAM Server evaluates authorization policies and cache the result.
14. OAM Server logs and returns decisions
15. WebGate responds as follows:
    - If the authorization policy allows access, the desired content or applications are served to the user.
    - If the authorization policy denies access, the user is redirected to another URL determined by the administrator.

SSO Login Processing with OAM Agents



Installation Overview

Installing an Oracle Retail supported Single Sign-On installation using OAM12c 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 12c has to be installed and configured.
3. Additional midtier instances (such as Oracle Forms 12c) 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 Guide12c.*

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 OID12c.

ODSM

Oracle Directory Services Manager (ODSM) is a Web-based application used in OID12c and 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.

M

Appendix: Setting Up Password Stores with wallets/credential stores

As part of an application installation, administrators must set up password stores for user accounts using wallets/credential stores. Some password stores must be installed on the application database side. While the installer handles much of this process, the administrators must perform some additional steps.

Password stores for the application and application server user accounts must also be installed; however, the installer takes care of this entire process.

ORACLE Retail Merchandising applications now have 3 different types of password stores. They are database wallets, java wallets, and database credential stores. Background and how to administer them below are explained in this appendix

About Database Password Stores and Oracle Wallet

Oracle databases have allowed other users on the server to see passwords in case database connect strings (username/password@db) were passed to programs. In the past, users could navigate to `ps -ef|grep <username>` to see the password if the password was supplied in the command line when calling a program.

To make passwords more secure, Oracle Retail has implemented the Oracle Software Security Assurance (OSSA) program. Sensitive information such as user credentials now must be encrypted and stored in a secure location. This location is called password stores or wallets. These password stores are secure software containers that store the encrypted user credentials.

Users can retrieve the credentials using aliases that were set up when encrypting and storing the user credentials in the password store. For example, if `username/password@db` is entered in the command line argument and the alias is called `db_username`, the argument to a program is as follows:

```
sqlplus /@db_username
```

This would connect to the database as it did previously, but it would hide the password from any system user.

After this is configured, as in the example above, the application installation and the other relevant scripts are no longer needed to use embedded usernames and passwords. This reduces any security risks that may exist because usernames and passwords are no longer exposed.

When the installation starts, all the necessary user credentials are retrieved from the Oracle Wallet based on the alias name associated with the user credentials.

There are three different types of password stores. One type explain in the next section is for database connect strings used in program arguments (such as `sqlplus /@db_username`). The others are for Java application installation and application use.

Setting Up Password Stores for Database User Accounts

After the database is installed and the default database user accounts are set up, administrators must set up a password store using the Oracle wallet. This involves assigning an alias for the username and associated password for each database user

account. The alias is used later during the application installation. This password store must be created on the system where the application server and database client are installed.

This section describes the steps you must take to set up a wallet and the aliases for the database user accounts. For more information on configuring authentication and password stores, see the *Oracle Database Security Guide*.

> **Note:** In this section, `<wallet_location>` is a placeholder text for illustration purposes. Before running the command, ensure that you specify the path to the location where you want to create and store the wallet.

To set up a password store for the database user accounts, perform the following steps:

1. Create a wallet using the following command:

   ```
   mkstore -wrl <wallet_location> -create
   ```

   After you run the command, a prompt appears. Enter a password for the Oracle Wallet in the prompt.

   > **Note:** The `mkstore` utility is included in the Oracle Database Client installation.

   The wallet is created with the auto-login feature enabled. This feature enables the database client to access the wallet contents without using the password. For more information, refer to the *Oracle Database Advanced Security Administrator's Guide.*

2. Create the database connection credentials in the wallet using the following command:

   ```
   mkstore -wrl <wallet_location> -createCredential <alias-name> <database-user-name>
   ```

   After you run the command, a prompt appears. Enter the password associated with the database user account in the prompt.

3. Repeat Step 2 for all the database user accounts.

4. Update the sqlnet.ora file to include the following statements:

   ```
   WALLET_LOCATION = (SOURCE = (METHOD = FILE) (METHOD_DATA = (DIRECTORY = <wallet_location>)))
   SQLNET.WALLET_OVERRIDE = TRUE
   SSL_CLIENT_AUTHENTICATION = FALSE
   ```

5. Update the tnsnames.ora file to include the following entry for each alias name to be set up.

   ```
   <alias-name> =
       (DESCRIPTION =
        (ADDRESS_LIST =
               (ADDRESS = (PROTOCOL = TCP) (HOST = <host>) (PORT = <port>))
           )
           (CONNECT_DATA =
               (SERVICE_NAME = <service>)
            )
        )
   ```

   In the previous example, `<alias-name>`, `<host>`, `<port>`, and `<service>` are placeholder text for illustration purposes. Ensure that you replace these with the relevant values.

Setting up Wallets for Database User Accounts

The following examples show how to set up wallets for database user accounts for the following applications:

- For RMS, RWMS, RPM Batch using sqlplus or sqlldr, RETL, RMS and RWMS

For RMS, RWMS, RPM Batch using sqlplus or sqlldr, RETL, RMS, and RWMS

To set up wallets for database user accounts, do the following.

1. Create a new directory called wallet under your folder structure.

```
cd /projects/rms19/dev/
mkdir .wallet
```

> **Note:** The default permissions of the wallet allow only the owner to use it, ensuring the connection information is protected. If you want other users to be able to use the connection, you must adjust permissions appropriately to ensure only authorized users have access to the wallet.

2. Create a sqlnet.ora in the wallet directory with the following content.

```
WALLET_LOCATION =    (SOURCE =      (METHOD = FILE)      (METHOD_DATA
=       (DIRECTORY =  /projects/rms19/dev/.wallet)) )
SQLNET.WALLET_OVERRIDE=TRUE
SSL_CLIENT_AUTHENTICATION=FALSE
```

> **Note**: WALLET_LOCATION must be on line 1 in the file.

3. Setup a tnsnames.ora in the wallet directory. This tnsnames.ora includes the standard tnsnames.ora file. Then, add two custom tns_alias entries that are only for use with the wallet. For example, `sqlplus /@dvols29_rms01user`.

```
ifile = /u00/oracle/product/12.1.0.2/network/admin/tnsnames.ora

Examples for a NON pluggable db:
dvols29_rms01user =
  (DESCRIPTION = (ADDRESS_LIST = (ADDRESS = (PROTOCOL = tcp)
  (host = xxxxxx.us.oracle.com) (Port = 1521)))
    (CONNECT_DATA = (SID = <sid_name> (GLOBAL_NAME = <sid_name>)))

dvols29_rms01user.world =
  (DESCRIPTION = (ADDRESS_LIST = (ADDRESS = (PROTOCOL = tcp)
  (host = xxxxxx.us.oracle.com) (Port = 1521)))
    (CONNECT_DATA = (SID = <sid_name>) (GLOBAL_NAME = <sid_name>)))

Examples for a pluggable db:
dvols29_rms01user =
  (DESCRIPTION = (ADDRESS_LIST = (ADDRESS = (PROTOCOL = tcp)
  (host = xxxxxx.us.oracle.com) (Port = 1521)))
    (CONNECT_DATA = (SERVICE_NAME = <pluggable db name>)))

dvols29_rms01user.world =
  (DESCRIPTION = (ADDRESS_LIST = (ADDRESS = (PROTOCOL = tcp)
  (host = xxxxxx.us.oracle.com) (Port = 1521)))
    (CONNECT_DATA = (SERVICE_NAME = <pluggable db name>)))
```

> **Note**: It is important to not just copy the tnsnames.ora file because it can quickly become out of date. The ifile clause (shown above) is key.

4. Create the wallet files. These are empty initially.
   a. Ensure you are in the intended location.
      ```
      $ pwd
      /projects/rms19/dev/.wallet
      ```
   b. Create the wallet files.
      ```
      $ mkstore -wrl . -create
      ```
   c. Enter the wallet password you want to use. It is recommended that you use the same password as the UNIX user you are creating the wallet on.
   d. Enter the password again.

      Two wallet files are created from the above command:
      - ewallet.p12
      - cwallet.sso
5. Create the wallet entry that associates the user name and password to the custom tns alias that was setup in the wallet's tnsnames.ora file.
   ```
   mkstore –wrl . –createCredential <tns_alias> <username> <password>
   ```

Example: `mkstore –wrl . –createCredential dvols29_rms01user rms01user passwd`

6. Test the connectivity. The ORACLE_HOME used with the wallet must be the same version or higher than what the wallet was created with.
   ```
   $ export TNS_ADMIN=/projects/rms19/dev/.wallet /* This is very import to use
   wallet to point at the alternate tnsnames.ora created in this example */

   $ sqlplus /@dvols29_rms01user

   SQL*Plus: Release 12

   Connected to:
   Oracle Database 12g

   SQL> show user
   USER is "rms01user"
   ```

   Running batch programs or shell scripts would be similar:

   ```
   Ex: dtesys /@dvols29_rms01user
   script.sh /@dvols29_rms01user

   Set the UP unix variable to help with some compiles :

   export UP=/@dvols29_rms01user
   for use in RMS batch compiles, and RMS, RWMS forms compiles.
   ```

As shown in the example above, users can ensure that passwords remain invisible.

Additional Database Wallet Commands

The following is a list of additional database wallet commands.

- Delete a credential on wallet
  ```
  mkstore –wrl . –deleteCredential dvols29_rms01user
  ```
- Change the password for a credential on wallet
  ```
  mkstore –wrl . –modifyCredential dvols29_rms01user rms01user passwd
  ```
- List the wallet credential entries

```
mkstore –wrl . –list
```

This command returns values such as the following.

```
oracle.security.client.connect_string1
oracle.security.client.user1
oracle.security.client.password1
```

- View the details of a wallet entry

```
mkstore –wrl . –viewEntry oracle.security.client.connect_string1
```

Returns the value of the entry:

```
dvols29_rms01user
mkstore –wrl . –viewEntry oracle.security.client.user1
```

Returns the value of the entry:

```
rms01user

mkstore –wrl . –viewEntry oracle.security.client.password1
```

Returns the value of the entry:

```
Passwd
```

Setting up RETL Wallets

RETL creates a wallet under $RFX_HOME/etc/security, with the following files:

- cwallet.sso
- jazn-data.xml
- jps-config.xml
- README.txt

To set up RETL wallets, perform the following steps:

1. Set the following environment variables:
   - `ORACLE_SID=<retaildb>`
   - `RFX_HOME=/u00/rfx/rfx-13`
   - `RFX_TMP=/u00/rfx/rfx-13/tmp`
   - `JAVA_HOME=/usr/jdk1.6.0_12.64bit`
   - `LD_LIBRARY_PATH=$ORACLE_HOME`
   - `PATH=$RFX_HOME/bin:$JAVA_HOME/bin:$PATH`
2. Change directory to $RFX_HOME/bin.
3. Run setup-security-credential.sh.
   - Enter 1 to add a new database credential.
   - Enter the dbuseralias. For example, `retl_java_rms01user`.
   - Enter the database user name. For example, `rms01user`.
   - Enter the database password.
   - Re-enter the database password.
   - Enter D to exit the setup script.
4. Update your RETL environment variable script to reflect the names of both the Oracle Networking wallet and the Java wallet.

   For example, to configure RETLforRPAS, modify the following entries in `$RETAIL_HOME/RETLforRPAS/rfx/etc/rmse_rpas_config.env`.
   - The RETL_WALLET_ALIAS should point to the Java wallet entry:
     - `export RETL_WALLET_ALIAS="retl_java_rms01user"`
   - The ORACLE_WALLET_ALIAS should point to the Oracle network wallet entry:

  - export ORACLE_WALLET_ALIAS="dvols29_rms01user"
- The SQLPLUS_LOGON should use the ORACLE_WALLET_ALIAS:
  - export SQLPLUS_LOGON="/@${ORACLE_WALLET_ALIAS}"

5. To change a password later, run setup-security-credential.sh.
   - Enter 2 to update a database credential.
   - Select the credential to update.
   - Enter the database user to update or change.
   - Enter the password of the database user.
   - Re-enter the password.

For Java Applications (SIM, ReIM, RPM, RIB, AIP, Alloc, ReSA, RETL)

For Java applications, consider the following:

- For database user accounts, ensure that you set up the same alias names between the password stores (database wallet and Java wallet). You can provide the alias name during the installer process.
- Document all aliases that you have set up. During the application installation, you must enter the alias names for the application installer to connect to the database and application server.
- Passwords are not used to update entries in Java wallets. Entries in Java wallets are stored in partitions, or application-level keys. In each retail application that has been installed, the wallet is located in <WEBLOGIC_DOMAIN_HOME>/retail/<appname>/config Example: /u00/webadmin/config/domains/wls_retail/RPMDomain/retail/rpm/config
- Application installers should create the Java wallets for you, but it is good to know how this works for future use and understanding.
- Scripts are located in <WEBLOGIC_DOMAIN_HOME>/retail/<appname>/retail-public-security-api/bin for administering wallet entries.
- Example:
- /u00/webadmin/config/domains/wls_retail/RPMDomain/retail/rpm/retail-public-security-api/bin
- In this directory is a script to help you update each alias entry without having to remember the wallet details. For example, if you set the RPM database alias to rms01user, you will find a script called update-RMS01USER.sh.

  Note: These scripts are available only with applications installed by way of an installer.

- Two main scripts are related to this script in the folder for more generic wallet operations: dump_credentials.sh and save_credential.sh.
- If you have not installed the application yet, you can unzip the application zip file and view these scripts in <app>/application/retail-public-security-api/bin.
- Example:
- /u00/webadmin/rpm/application/rpm/Build/orpatch/deploy/retail-public-security-api/bin

update-<ALIAS>.sh

update-<ALIAS>.sh updates the wallet entry for this alias. You can use this script to change the user name and password for this alias. Because the application refers only to the alias, no changes are needed in application properties files.

Usage:

```
update-<username>.sh <myuser>
```

Example:

```
/u00/webadmin/config/domains/wls_retail/RPMDomain/retail/rpm/retail-public-
security-api/bin> ./update-RMS01USER.sh
usage: update-RMS01USER.sh <username>
<username>: the username to update into this alias.
Example: update-RMS01USER.sh myuser
Note: this script will ask you for the password for the username that you pass in.
/u00/webadmin/config/domains/wls_retail/RPMDomain/retail/rpm/retail-public-
security-api/bin>
```

dump_credentials.sh

dump_credentials.sh is used to retrieve information from wallet. For each entry found in the wallet, the wallet partition, the alias, and the user name are displayed. Note that the password is not displayed. If the value of an entry is uncertain, run save_credential.sh to resave the entry with a known password.

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

Example:

```
dump_credentials.sh location:
/u00/webadmin/config/domains/wls_retail/RPMDomain/retail/rpm/config

Retail Public Security API Utility
==========================================
```

Below are the credentials found in the wallet at the location/u00/webadmin/config/domains/wls_retail/RPMDomain/retail/rpm/config

```
==========================================

Application level key partition name:rpm
User Name Alias:WLS-ALIAS User Name:weblogic
User Name Alias:RETAIL-ALIAS User Name:retail.user
User Name Alias:LDAP-ALIAS User Name:RETAIL.USER
User Name Alias:RMS-ALIAS User Name:rms19mock
User Name Alias:REIMBAT-ALIAS User Name:rpmbat
```

save_credential.sh

save_credential.sh is used to update the information in wallet. If you are unsure about the information that is currently in the wallet, use dump_credentials.sh as indicated above.

```
save_credential.sh -a <alias> -u <user> -p <partition name>  -l <path of the
wallet file location where credentials are stored>
```

Example:

```
/u00/webadmin/mock19_testing/rpm19/application/retail-public-security-api/bin>
save_credential.sh -l wallet_test -a myalias -p mypartition -u myuser

==========================================
Retail Public Security API Utility
==========================================

Enter password:
Verify password:
```

> **Note:** -p in the above command is for partition name. You must specify the proper partition name used in application code for each Java application.
>
> save_credential.sh and dump_credentials.sh scripts are the same for all applications. If using save_credential.sh to add a wallet entry or to update a wallet entry, bounce the application/managed server so that your changes are visible to the application. In addition, save a backup copy of your cwallet.sso file in a location outside of the deployment path, because redeployment or reinstallation of the application will wipe the wallet entries you made after installation of the application. To restore your wallet entries after a redeployment/reinstallation, copy the backed up cwallet.sso file over the cwallet.sso file. Then bounce the application/managed server.

Usage

```
==========================================
Retail Public Security API Utility
==========================================
usage: save_credential.sh -au[plh]
E.g. save_credential.sh -a rms-alias -u rms_user -p rib-rms -l ./
 -a,--userNameAlias <arg>              alias for which the credentials
needs to be stored
 -h,--help                             usage information
 -l,--locationofWalletDir <arg>        location where the wallet file is
created.If not specified, it creates the wallet under secure-credential-wallet
directory which is already present under the retail-public-security-api/
directory.
 -p,--appLevelKeyPartitionName <arg>   application level key partition name
 -u,--userName <arg>                   username to be stored in secure
credential wallet for specified alias*
```

How does the Wallet Relate to the Application?

The ORACLE Retail Java applications have the wallet alias information you create in an <app-name>.properties file. Below is the reim.properties file. Note the database information and the user are presented as well. The property called datasource.credential.alias=RMS-ALIAS uses the ORACLE wallet with the argument of

RMS-ALIAS at the csm.wallet.path and csm.wallet.partition.name = rpm to retrieve the password for application use.

Reim.properties code sample:

```
datasource.url=jdbc:oracle:thin:@xxxxxxx.us.oracle.com:1521:pkols07
datasource.schema.owner=rms19mock
datasource.credential.alias=RMS-ALIAS
# ===================================================================
# ossa related Configuration
#
# These settings are for ossa configuration to store credentials.
# ===================================================================

csm.wallet.path=/u00/webadmin/config/domains/wls_retail/RPMDomain/retail/rpm/confi
g
csm.wallet.partition.name=rpm
```

How does the Wallet Relate to Java Batch Program use?

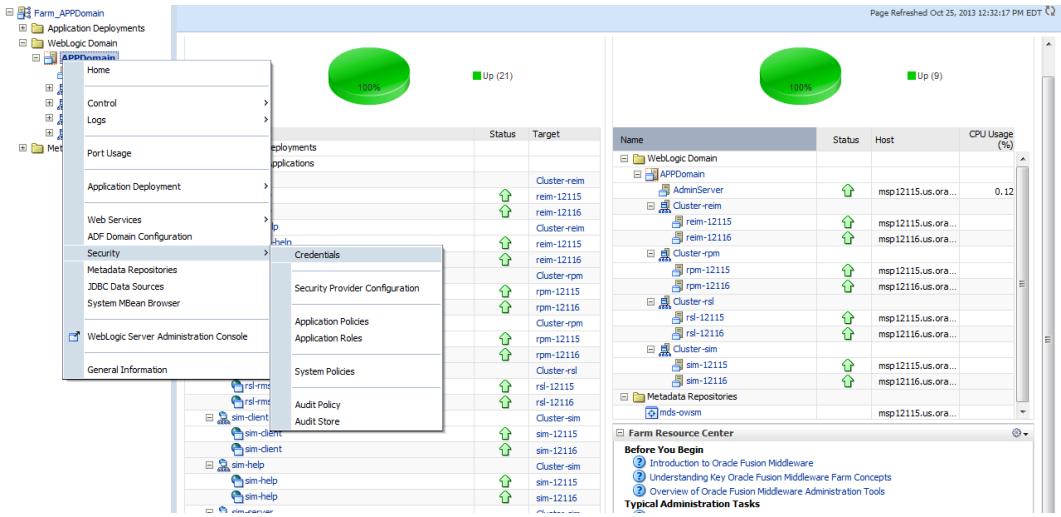
Some of the ORACLE Retail Java batch applications have an alias to use when running Java batch programs. For example, alias REIMBAT-ALIAS maps through the wallet to dbuser RMS01APP, already on the database. To run a ReIM batch program the format would be: reimbatchpgmname REIMBAT-ALIAS <other arguments as needed by the program in question>

Database Credential Store Administration

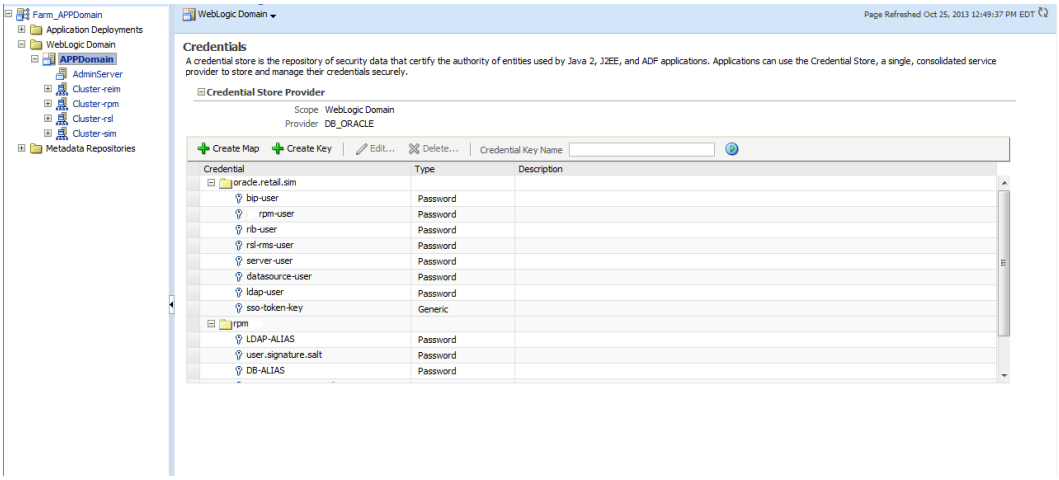
The following section describes a domain level database credential store. This is used in RPM login processing, SIM login processing, RWMS login processing, RESA login processing and Allocation login processing and policy information for application permission. Setting up the database credential store is addressed in the RPM, SIM, RESA, RWMS, and Alloc install guides.

The following sections show an example of how to administer the password stores thru ORACLE Enterprise Manger Fusion Middleware Control, a later section will show how to do this thru WLST scripts.

1. The first step is to use your link to Oracle Enterprise Manager Fusion Middleware Control for the domain in question. Locate your domain on the left side of the screen and do a right mouse click on the domain and select **Security** > **Credentials**

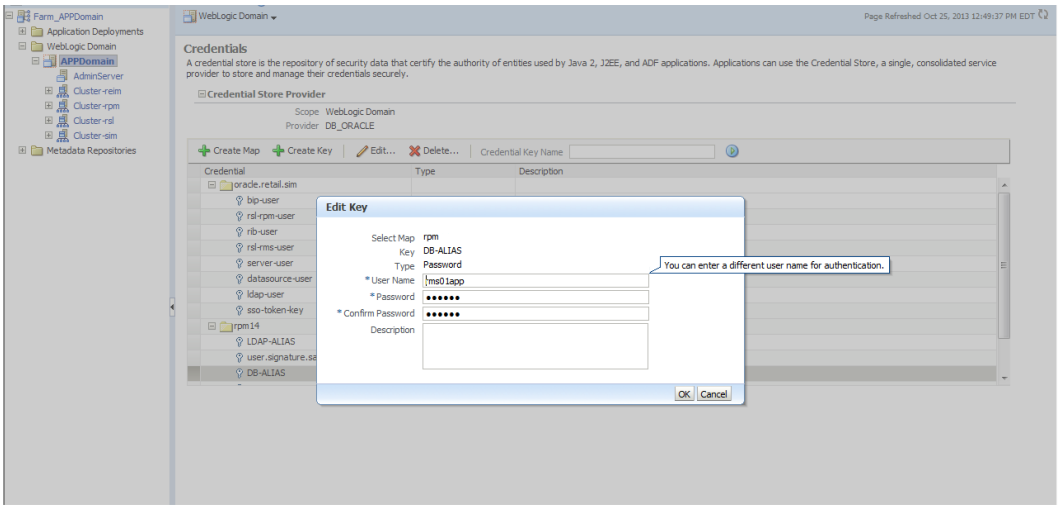


2. Click on Credentials and you will get a screen similar to the following. The following screen is expanded to make it make more sense. From here you can administer credentials.



The Create Map add above is to create a new map with keys under it. A map would usually be an application such as rpm. The keys will usually represent alias to various users (database user, WebLogic user, LDAP user, etc). The application installer should add the maps so you should not often have to add a map.

Creation of the main keys for an application will also be built by the application installer. You will not be adding keys often as the installer puts the keys out and the keys talk to the application. You may be using EDIT on a key to see what user the key/alias points to and possibly change/reset its password. To edit a key/alias, highlight the key/alias in question and push the edit icon nearer the top of the page. You will then get a screen as follows:



The screen above shows the map (rpm) that came from the application installer, the key (DB-ALIAS) that came from the application installer (some of the keys/alias are selected by the person who did the application install, some are hard coded by the application installer in question), the type (in this case password), and the user name and password. This is where you would check to see that the user name is correct and reset the password if needed. REMEMBER, a change to an item like a database password WILL make you come into this and also change the password. Otherwise your application will NOT work correctly.

Managing Credentials with WSLT/OPSS Scripts

This procedure is optional as you can administer the credential store through the Oracle enterprise manager associated with the domain of your application install for ReIM, RPM, SIM, RESA, or Allocation.

An Oracle Platform Security Scripts (OPSS) script is a WLST script, in the context of the Oracle WebLogic Server. An online script is a script that requires a connection to a running server. Unless otherwise stated, scripts listed in this section are online scripts and operate on a database credential store. There are a few scripts that are offline, that is, they do not require a server to be running to operate.

Read-only scripts can be performed only by users in the following WebLogic groups: Monitor, Operator, Configurator, or Admin. Read-write scripts can be performed only by users in the following WebLogic groups: Admin or Configurator. All WLST scripts are available out-of-the-box with the installation of the Oracle WebLogic Server.

WLST scripts can be run in interactive mode or in script mode. In interactive mode, you enter the script at a command-line prompt and view the response immediately after. In script mode, you write scripts in a text file (with a py file name extension) and run it without requiring input, much like the directives in a shell script.

The weakness with the WLST/OPSS scripts is that you have to already know your map name and key name. In many cases, you do not know or remember that. The database credential store way through enterprise manager is a better way to find your map and key names easily when you do not already know them. A way in a command line mode to find the map name and alias is to run orapki. An example of orapki is as follows:

/u00/webadmin/product/wls_apps/oracle_common/bin> ./orapki wallet display –wallet /u00/webadmin/product/wls_apps/user_projects/domains/APPDomain/config/fmwconfig

(where the path above is the domain location of the wallet)

Output of orapki is below. This shows map name of rpm and each alias in the wallet:

Requested Certificates:

User Certificates:

Oracle Secret Store entries:

rpm@#3#@DB-ALIAS

rpm@#3#@LDAP-ALIAS

rpm@#3#@RETAIL.USER

rpm@#3#@user.signature.salt

rpm@#3#@user.signature.secretkey

rpm@#3#@WEBLOGIC-ALIAS

rpm@#3#@WLS-ALIAS

Trusted Certificates:

Subject: OU=Class 1 Public Primary Certification Authority,O=VeriSign\, Inc.,C=US

OPSS provides the following scripts on all supported platforms to administer credentials (all scripts are online, unless otherwise stated. You need the map name and the key name to run the scripts below

- listCred
- updateCred
- createCred
- deleteCred
- modifyBootStrapCredential
- addBootStrapCredential

listCred

The script `listCred` returns the list of attribute values of a credential in the credential store with given map name and key name. This script lists the data encapsulated in credentials of type password only.

Script Mode Syntax

```
listCred.py -map mapName -key keyName
```

Interactive Mode Syntax

```
listCred(map="mapName", key="keyName")
```

The meanings of the arguments (all required) are as follows:

- `map` specifies a map name (folder).
- `key` specifies a key name.

Examples of Use:

The following invocation returns all the information (such as user name, password, and description) in the credential with map name `myMap` and key name `myKey`:

```
listCred.py -map myMap -key myKey
```

The following example shows how to run this command and similar credential commands with WLST:

```
/u00/webadmin/product/wls_apps/oracle_common/common/bin>
sh wlst.sh

Initializing WebLogic Scripting Tool (WLST)...

Welcome to WebLogic Server Administration Scripting Shell


wls:/offline> connect('weblogic','password123','xxxxxx.us.oracle.com:17001')
Connecting to t3://xxxxxx.us.oracle.com:17001 with userid weblogic ...
Successfully connected to Admin Server 'AdminServer' that belongs to domain
'APPDomain'.

wls:/APPDomain/serverConfig> listCred(map="rpm",key="DB-ALIAS")
Already in Domain Runtime Tree

[Name : rms01app, Description : null, expiry Date : null]
PASSWORD:retail
*The above means for map rpm in APPDomain, alias DB-ALIAS points to database user
rms01app with a password of retail
```

updateCred

The script `updateCred` modifies the type, user name, and password of a credential in the credential store with given map name and key name. This script updates the data encapsulated in credentials of type password only. Only the interactive mode is supported.

Interactive Mode Syntax

```
updateCred(map="mapName", key="keyName", user="userName", password="passW",
[desc="description"])
```

The meanings of the arguments (optional arguments are enclosed by square brackets) are as follows:

- `map` specifies a map name (folder) in the credential store.
- `key` specifies a key name.
- `user` specifies the credential user name.
- `password` specifies the credential password.
- `desc` specifies a string describing the credential.

Example of Use:

The following invocation updates the user name, password, and description of the password credential with map name `myMap` and key name `myKey`:

```
updateCred(map="myMap", key="myKey", user="myUsr", password="myPassw")
```

createCred

The script `createCred` creates a credential in the credential store with a given map name, key name, user name and password. This script can create a credential of type password only. Only the interactive mode is supported.

Interactive Mode Syntax

```
createCred(map="mapName", key="keyName", user="userName", password="passW",
[desc="description"])
```

The meanings of the arguments (optional arguments are enclosed by square brackets) are as follows:

- `map` specifies the map name (folder) of the credential.
- `key` specifies the key name of the credential.
- `user` specifies the credential user name.
- `password` specifies the credential password.
- `desc` specifies a string describing the credential.

Example of Use:

The following invocation creates a password credential with the specified data:

```
createCred(map="myMap", key="myKey", user="myUsr", password="myPassw")
```

deleteCred

The script `deleteCred` removes a credential with given map name and key name from the credential store.

Script Mode Syntax

```
deleteCred.py -map mapName -key keyName
```

Interactive Mode Syntax

```
deleteCred(map="mapName",key="keyName")
```

The meanings of the arguments (all required) are as follows:

- `map` specifies a map name (folder).
- `key` specifies a key name.

Example of Use:

The following invocation removes the credential with map name `myMap` and key name `myKey`:

```
deleteCred.py -map myMap -key myKey
```

modifyBootStrapCredential

The offline script `modifyBootStrapCredential` modifies the bootstrap credentials configured in the default jps context, and it is typically used in the following scenario: suppose that the policy and credential stores are LDAP-based, and the credentials to access the LDAP store (stored in the LDAP server) are changed. Then this script can be used to seed those changes into the bootstrap credential store.

This script is available in interactive mode only.

Interactive Mode Syntax

```
modifyBootStrapCredential(jpsConfigFile="pathName", username="usrName",
password="usrPass")
```

The meanings of the arguments (all required) are as follows:

- `jpsConfigFile` specifies the location of the file `jps-config.xml` relative to the location where the script is run. Example location: /u00/webadmin/product/wls_apps/user_projects/domains/APPDomain/config/fmwconfig. Example location of the bootstrap wallet is /u00/webadmin/product/wls_apps/user_projects/domains/APPDomain/config/fmwconfig/bootstrap

- username specifies the distinguished name of the user in the LDAP store.
- password specifies the password of the user.

Example of Use:

Suppose that in the LDAP store, the password of the user with distinguished name cn=orcladmin has been changed to <password>, and that the configuration file jps-config.xml is located in the current directory. Then the following invocation changes the password in the bootstrap credential store to <password>:

```
modifyBootStrapCredential(jpsConfigFile='./jps-config.xml',
username='cn=orcladmin', password='<password>')
```

Any output regarding the audit service can be disregarded.

addBootStrapCredential

The offline script addBootStrapCredential adds a password credential with given map, key, user name, and user password to the bootstrap credentials configured in the default jps context of a jps configuration file.

Classloaders contain a hierarchy with parent classloaders and child classloaders. The relationship between parent and child classloaders is analogous to the object relationship of super classes and subclasses. The bootstrap classloader is the root of the Java classloader hierarchy. The Java virtual machine (JVM) creates the bootstrap classloader, which loads the Java development kit (JDK) internal classes and java.* packages included in the JVM. (For example, the bootstrap classloader loads java.lang.String.)

This script is available in interactive mode only.

Interactive Mode Syntax

```
addBootStrapCredential(jpsConfigFile="pathName", map="mapName", key="keyName",
username="usrName", password="usrPass")
```

The meanings of the arguments (all required) are as follows:

- jpsConfigFile specifies the location of the file jps-config.xml relative to the location where the script is run. Example location: /u00/webadmin/product/wls_apps/user_projects/domains/APPDomain/config/fmwconfig
- map specifies the map of the credential to add.
- key specifies the key of the credential to add.
- username specifies the name of the user in the credential to add.
- password specifies the password of the user in the credential to add.

Example of Use:

The following invocation adds a credential to the bootstrap credential store:

```
addBootStrapCredential(jpsConfigFile='./jps-config.xml', map='myMapName',
key='myKeyName', username='myUser', password ='myPass')
```

Quick Guide for Retail Password Stores (db wallet, java wallet, DB credential stores)

Retail app	Wallet type	Wallet loc	Wallet partition	Alias name	User name	Use	Create by	Alias Example	Notes
RMS batch	DB	<RMS batch install dir (RETAIL_HOME)>/.wallet	n/a	<Database SID>_<Database schema owner>	<rms schema owner>	Compile, execution	Installer	n/a	Alias hard-coded by installer
RMWS forms	DB	<forms install dir>/base/.wallet	n/a	<Database SID>_<Database schema owner>	<rwms schema owner>	Compile forms, execute batch	Installer	n/a	Alias hard-coded by installer
RPM batch plsql and sqlldr	DB	<RPM batch install dir>/.wallet	n/a	<rms schema owner alias>	<rms schema owner>	Execute batch	Manual	rms-alias	RPM plsql and sqlldr batches
RWMS auto-login	JAVA	<forms install dir>/base/.javawallet							
			<RWMS Installation name>	<RWMS database user alias>	<RWMS schema owner>	RWMS forms app to avoid dblogin screen	Installer	rwms19inst	
			<RWMS Installation name>	BI_ALIAS	<BI Publisher administrative user>	RWMS forms app to connect to BI Publisher	Installer	n/a	Alias hard-coded by installer
AIP app	JAVA	<weblogic domain home>/retail/<deployed aip app name>/config							Each alias must be unique
			aip	<AIP weblogic user alias>	<AIP weblogic user name>	App use	Installer	aip-weblogic-alias	

Retail app	Wallet type	Wallet loc	Wallet partition	Alias name	User name	Use	Create by	Alias Example	Notes
			aip	<AIP database schema user alias>	<AIP database schema user name>	App use	Installer	aip01user-alias	
			aip	<rib-aip weblogic user alias>	<rib-aip weblogic user name>	App use	Installer	rib-aip-weblogic-alias	
RPM app	DB credential store		Map=rpm or what you called the app at install time.	Many for app use					<weblogic domain home>/config/fmwconfig/jps-config.xml has info on the credential store. This directory also has the domain cwallet.sso file.
RPM app	JAVA	<weblogic domain home>/retail/<deployed rpm app name>/config							Each alias must be unique
			rpm	<rpm weblogic user alias>	<rpm weblogic user name>	App use	Installer	rpm-weblogic-alias	
			rpm	<rpm batch user name> is the alias. Yes, here alias name = user name	<rpm batch user name>	App, batch use	Installer	RETAIL.USER	
	JAVA	<retail_home>/orpatch/config/javaapp_rpm							Each alias must be unique

Retail app	Wallet type	Wallet loc	Wallet partition	Alias name	User name	Use	Create by	Alias Example	Notes
			retail_installer	<rpm weblogic user alias>	<rpm weblogic user name>	App use	Installer	weblogic-alias	
			retail_installer	<rms schema user alias>	<rms schema user name>	App, batch use	Installer	rms01user-alias	
			retail_installer	<reim batch user alias>	<reim batch user name>	App, batch use	Installer	reimbat-alias	
			retail_installer	<LDAP-ALIAS>	cn=rpm.admin,cn=Users,dc=us,dc=oracle,dc=com	LDAP user use	Installer	LDAP_ALIAS	
ReIM app	JAVA	<weblogic domain home>/retail/<deployed reim app name>/config							Each alias must be unique
			<installed app name, ex: reim>	<reim weblogic user alias>	<reim weblogic user name>	App use	Installer	weblogic-alias	
			<installed app name, ex: reim>	<rms schema user alias>	<rms schema user name>	App, batch use	Installer	rms01user-alias	
			<installed app name, ex: reim>	<reim webservice validation user alias>	<reim webservice validation user name>	App use	Installer	reimwebservice-alias	
			<installed app name, ex: reim>	<reim batch user alias>	<reim batch user name>	App, batch use	Installer	reimbat-alias	

Retail app	Wallet type	Wallet loc	Wallet partition	Alias name	User name	Use	Create by	Alias Example	Notes
			<installed app name, ex: reim>	<LDAP-ALIAS>	cn=REIM.ADMIN,cn=Users,dc=us,dc=oracle,dc=com	LDAP user use	Installer	LDAP_ALIAS	
	JAVA	<retail_home>/orpatch/config/javaapp_reim							Each alias must be unique
			retail_installer	<reim weblogic user alias>	<reim weblogic user name>	App use	Installer	weblogic-alias	
			retail_installer	<rms schema user alias>	<rms schema user name>	App, batch use	Installer	rms01user-alias	
			retail_installer	<reim webservice validation user alias>	<reim webservice validation user name>	App use	Installer	reimwebservice-alias	
			retail_installer	<reim batch user alias>	<reim batch user name>	App, batch use	Installer	reimbat-alias	
			retail_installer	<LDAP-ALIAS>	cn=REIM.ADMIN,cn=Users,dc=us,dc=oracle,dc=com	LDAP user use	Installer	LDAP_ALIAS	

Retail app	Wallet type	Wallet loc	Wallet partition	Alias name	User name	Use	Create by	Alias Example	Notes
RESA app	DB credential store		Map=resaor what you called the app at install time	Many for login and policies					<weblogic domain home>/config/fmwconfig/jps-config.xml has info on the credential store. This directory also has the domain cwallet.sso file. The bootstrap directory under this directory has bootstrap cwallet.sso file.
RESA app	JAVA	<weblogic domain home>/retail/<deployed resa app name>/config							Each alias must be unique
			<installed app name>	<resa weblogic user alias>	<resa weblogic user name>	App use	Installer	wlsalias	
			<installed app name>	<resa schema db user alias>	<rmsdb schema user name>	App use	Installer	Resadb-alias	
			<installed app name>	<resa schema user alias>	<rmsdb schema user name>>	App use	Installer	resa-alias	
	JAVA	<retail_home>/orpatch/config/javaapp_resa							Each alias must be unique

Retail app	Wallet type	Wallet loc	Wallet partition	Alias name	User name	Use	Create by	Alias Example	Notes
			retail_installer	<resa weblogic user alias>	<resa weblogic user name>	App use	Installer	wlsalias	
			retail_installer	<resa schema db user alias>	<rmsdb schema user name>	App use	Installer	Resadb-alias	
	JAVA	<retail_ home>/orpatch/config/java app_rasrm							Each alias must be unique
			retail_installer	<alloc weblogic user alias>	<alloc weblogic user name>	App use	Installer	weblogic-alias	
Alloc app	DB credential store		Map=alloc or what you called the app at install time	Many for login and policies					<weblogic domain home>/config/fmwconfig/jps-config.xml has info on the credential store. This directory also has the domain cwallet.sso file. The bootstrap directory under this directory has bootstrap cwallet.sso file.
Alloc app	JAVA	<weblogic domain home>/retail/config							Each alias must be unique
			<installed app name>	<alloc weblogic user alias>	<alloc weblogic user name>	App use	Installer	weblogic-alias	

Retail app	Wallet type	Wallet loc	Wallet partition	Alias name	User name	Use	Create by	Alias Example	Notes
			<installed app name>	<rms schema user alias>	<rms schema user name>	App use	Installer	dsallocAlias	
			<installed app name>	<alloc batch user alias>	<SYSTEM_ ADMINIST RATOR>	Batch use	Installer	alloc14	
	JAVA	<retail_ home>/orpatch/config/java app_alloc							Each alias must be unique
			retail_installe r	<alloc weblogic user alias>	<alloc weblogic user name>	App use	Installer	weblogic-alias	
			retail_installe r	<rms schema user alias>	<rms schema user name>	App use	Installer	dsallocAlias	
			retail_installe r	<alloc batch user alias>	<SYSTEM_ ADMINIST RATOR>	Batch use	Installer	alloc14	
	JAVA	<retail_ home>/orpatch/config/java app_rasrm							Each alias must be unique
			retail_installe r	<alloc weblogic user alias>	<alloc weblogic user name>	App use	Installer	weblogic-alias	
SIM app	DB credenti al store		Map=oracle. retail.sim	Aliases required for SIM app use					<weblogic domain home>/config/fmwco nfig/jps-config.xml has info on the credential store. This directory also has the domain cwallet.sso file.

Retail app	Wallet type	Wallet loc	Wallet partition	Alias name	User name	Use	Create by	Alias Example	Notes
	JAVA	<weblogic domain home>/retail/<deployed sim app name>/batch/resources/conf	oracle.retail.sim	<sim batch user alias>	<sim batch user name>	App use	Installer	BATCH-ALIAS	
	JAVA	<weblogic domain home>/retail/<deployed sim app name>/wireless/resources/conf	oracle.retail.sim	<sim wireless user alias>	<sim wireless user name>	App use	Installer	WIRELESS-ALIAS	
RETL	JAVA	<RETL home>/etc/security	n/a	<target application user alias>	<target application db userid>	App use	Manual	retl_java_rms01user	User may vary depending on RETL flow's target application
RETL	DB	<RETL home>/.wallet	n/a	<target application user alias>	<target application db userid>	App use	Manual	<db>_<user>	User may vary depending on RETL flow's target application
RIB	JAVA	<RIBHOME DIR>/deployment-home/conf/security							<app> is one of aip, rfm, rms, rpm, sim, rwms, tafr
JMS			jms<1-5>	<jms user alias> for jms<1-5>	<jms user name> for jms<1-5>	Integration use	Installer	jms-alias	
WebLogic			rib-<app>-app-server-instance	<rib-app weblogic user alias>	<rib-app weblogic user name>	Integration use	Installer	weblogic-alias	
Admin GUI			rib-<app>#web-app-user-alias	<rib-app admin gui user alias>	<rib-app admin gui user name>	Integration use	Installer	admin-gui-alias	
Application			rib-<app>#user-alias	<app weblogic user alias>	<app weblogic user name>	Integration use	Installer	app-user-alias	Valid only for aip, rpm, sim

Retail app	Wallet type	Wallet loc	Wallet partition	Alias name	User name	Use	Create by	Alias Example	Notes
DB			rib-<app>#app-db-user-alias	<rib-app database schema user alias>	<rib-app database schema user name>	Integra-tion use	Installer	db-user-alias	Valid only for rfm, rms, rwms, tafr
Error Hospital			rib-<app>#hosp-user-alias	<rib-app error hospital database schema user alias>	<rib-app error hospital database schema user name>	Integra-tion use	Installer	hosp-user-alias	
RFI	Java	<RFI-HOME>/retail-financial-integration-solution/service-based-integration/conf/security							
			<installed app name>	rfiAppServerAdminServerUserAlias	<rfi weblogic user name>	App use	Installer	rfiAppServerAdminServerUserAlias	
			<installed app name>	rfiAdminUiUserAlias	<ORFI admin user>	App use	Installer	rfiAdminUiUserAlias	
			<installed app name>	rfiDataSourceUserAlias	<ORFI schema user name>	App use	Installer	rfiDataSourceUserAlias	
			<installed app name>	ebsDataSourceUserAlias	<EBS schema user name>	App use	Installer	ebsDataSourceUserAlias	
			<installed app name>	smtpMailFromAddressAlias	<From email address>	App use	Installer	smtpMailFromAddressAlias	

Appendix: Creating User Synonyms

Please refer to $RETAIL_HOME/orpatch/utilities/create_synonyms_one_user.sql.

```
-- ------------------------------------------
-- Copyright (C) 2013,2014, Oracle and/or its affiliates. All rights reserved.
-- ------------------------------------------
-- This script creates synonyms in one schema (the synonym schema) to all objects
-- in another schema (the owning schema)
-- Arguments: synonym_schema owning_schema
set serveroutput on size unlimited
set escape on

declare
    synonym_schema            varchar2(30);
    owning_schema          varchar2(30);
    run_schema             varchar2(30);
    missing_object         varchar2(130);
    prefix1                varchar2(128);
    prefix2                varchar2(128);

    cursor c_get_missing_object (ownerschema in varchar2,synschema in varchar2) is
        (select object_name
        from   dba_objects
        where  owner = upper(ownerschema)
            and object_type in ('TABLE', 'VIEW', 'CLUSTER', 'FUNCTION', 'PACKAGE',
'PROCEDURE', 'SEQUENCE', 'TYPE')
            and object_name not like 'DBC_%'
            and object_name not like 'BIN$%'
                union
        select synonym_name from dba_synonyms
        where
        owner = ownerschema
        and table_name in ('ARI_INTERFACE_SQL','RMS_NOTIFICATION_REC') and
synonym_name in ('ARI_INTERFACE_SQL','RMS_NOTIFICATION_REC'))
        MINUS
        select object_name
        from   dba_objects
        where  owner = upper(synschema)
        order by 1;

begin
    synonym_schema := sys.dbms_assert.schema_name(upper('&1'));
    owning_schema := sys.dbms_assert.schema_name(upper('&2'));
    run_schema := sys.dbms_assert.schema_name('&_USER');

    IF synonym_schema <> run_schema THEN
        prefix1:=sys.dbms_assert.enquote_name(synonym_schema,FALSE)||'.';
    ELSE
        prefix1:='';
    END IF;

    IF owning_schema <> run_schema THEN
        prefix2:=sys.dbms_assert.enquote_name(owning_schema,FALSE)||'.';
    ELSE
        prefix2:='';
    END IF;
```

```
    open c_get_missing_object(owning_schema,synonym_schema);
    LOOP
        fetch c_get_missing_object into missing_object;
        --When at end of objects, exit
        if c_get_missing_object%NOTFOUND then
            exit;
        end if;

        missing_object:=sys.dbms_assert.enquote_name(missing_object,FALSE);

        BEGIN
            execute immediate 'CREATE SYNONYM '||prefix1||missing_object||' FOR
'||prefix2||missing_object;
            dbms_output.put_line('Created synonym '||prefix1||missing_object||'
pointing to '||prefix2||missing_object);
        EXCEPTION
        WHEN OTHERS THEN
            dbms_output.put_line('Create synonym FAILED '||missing_object||'
'||SQLCODE||' - '||SQLERRM);
        END;
    END LOOP;
    close c_get_missing_object;
EXCEPTION
    WHEN OTHERS THEN
        raise;
end;
/
```

O

Appendix: Manual Batch Compilation

To manually recompile batch, please use the ORCompile utility.

This is only possible after installer has been run and configured Oracle Retail Patch Assistant.

- Set RETAIL_HOME environment variable
- $RETAIL_HOME/orpatch/bin/orcompile –a RMS –t BATCH

Usage:

orcompile -a <app> -t <type>

Potential Apps and Types:

ALLOC => DB-ALC,DB-RMS

REIM => DB

RMS => BATCH,DB,DB-DEMO

P

Appendix: Configure a Wallet for Tablespace Encryption

Configure a Wallet for Tablespace Encryption

1. Create sqlnet.ora in $TNS_ADMIN of the database server, similar to the below entry:
   ```
   ENCRYPTION_WALLET_LOCATION = (SOURCE = (METHOD = FILE) (METHOD_DATA =
   (DIRECTORY = /u00/oracle/admin/<ORACLE_SID>/wallet)))
   ```
2. Create the wallet directory (replace the path with your value for DIRECTORY):
   ```
   mkdir –p /u00/oracle/admin/<ORACLE_SID>/wallet
   ```
   a. As a user with the ALTER SYSTEM privilege, create the wallet as follows (replace the path with your value for DIRECTORY):

Non-container database:

```
ADMINISTER KEY MANAGEMENT CREATE KEYSTORE '/u00/oracle/admin/<ORACLE_SID>/wallet'
IDENTIFIED BY "pwd#";
ADMINISTER KEY MANAGEMENT SET KEYSTORE OPEN IDENTIFIED BY "pwd#";
ADMINISTER KEY MANAGEMENT SET KEY IDENTIFIED BY "pwd#" WITH BACKUP;
ADMINISTER KEY MANAGEMENT CREATE AUTO_LOGIN KEYSTORE FROM KEYSTORE
'/u00/oracle/admin/<ORACLE_SID>/wallet' identified by pwd#;
```

Container database:

```
ADMINISTER KEY MANAGEMENT CREATE KEYSTORE '/u00/oracle/admin/<ORACLE_SID>/wallet'
IDENTIFIED BY "pwd#";
ADMINISTER KEY MANAGEMENT CREATE AUTO_LOGIN KEYSTORE FROM KEYSTORE
'/u00/oracle/admin/<ORACLE_SID>/wallet' identified by "pwd#";
ADMINISTER KEY MANAGEMENT SET KEYSTORE OPEN IDENTIFIED BY "pwd#" Container=ALL;
ADMINISTER KEY MANAGEMENT SET KEY IDENTIFIED BY "pwd#" WITH BACKUP USING
'TDE_ENCRYPTION' Container=all;
```

Confirm that the wallet is created and open (the TDE master encryption key has been created and inserted automatically).
SQL>
select substr(wrl_type, 1, 10) wrl_type, substr(wrl_parameter, 1, 45) param, substr(status, 1, 10) status, substr(wallet_type, 1, 15) w_type from v$encryption_wallet;

WRL_TYPE PARAM STATUS W_TYPE
---------- ---------- ---------- ---------- ---------- ---------- ---------- ----------
FILE /u00/oracle/admin/<ORACLE_SID>/wallet OPEN AUTOLOGIN

An auto-open wallet is created. You are ready to create the encrypted tablespaces as shown.

Q

Appendix – Pre-installation of Retail Infrastructure in WebLogic

Oracle Retail applications are primarily deployed in the Oracle WebLogic server as the Middleware tier. Java and forms based applications rely on Middleware infrastructure for complete security a part from application specific security features.

This chapter describes the pre-installation steps for security setup of Oracle Retail Infrastructure in WebLogic.

- JDK Hardening for Use with Retail Applications
- Pre-installation - Steps for Secured Setup of Oracle Retail Infrastructure in WebLogic
- Certificate Authority
- Obtaining an SSL Certificate and Setting up a Keystore
- Creating a WebLogic Domain
- Configuring the Application Server for SSL
- Enforcing Stronger Encryption in WebLogic
- Securing Nodemanager with SSL Certificates
- Using Secured Lightweight Directory Access Protocol (LDAP)
- Connecting from Forms Application to Secured Database
- Enabling Access to Secured Database from Forms Oracle Home - Optional

JDK Hardening for Use with Retail Applications

See the following sections on JDK hardening for use with Retail applications

- Upgrading JDK to use Java Cryptography extension
- Disabling weak SSL protocols and obsolete ciphers in JDK

Upgrading JDK to Use Java Cryptography Extension

You need to install the unlimited encryption Java Cryptography Extension (JCE) policy if you want to use the strongest Cipher suite (256 bit encryption) AES_256 (TLS_RSA_WITH_AES_256_CBC_SHA). It is dependent on the Java Development Kit (JDK) version.

Using the following URL, download and install the JCE Unlimited Strength Jurisdiction Policy Files that correspond to the version of your JDK

http://www.oracle.com/technetwork/java/javase/downloads/index.html

For JDK 8 download from the following URL:

https://www.oracle.com/technetwork/java/javase/downloads/jce8-download-2133166.html

and replace the files in the JDK/jre/lib/security directory.

Pre-installation – Steps for Secured Setup of Oracle Retail Infrastructure in WebLogic

Secured Socket Layer (SSL) protocol allows client-server applications to communicate across a network in a secured channel. Client and server should both decide to use SSL

to communicate secured information like user credentials or any other secured information.

The WebLogic Server supports SSL on a dedicated listen port. Oracle Forms is configured to use SSL as well. To establish an SSL connection, a Web browser connects to the WebLogic Server by supplying the SSL port and the Hypertext Transfer Protocol (HTTPS) in the connection URL.

For example: https://myserver:7002

The Retail Merchandising System (RMS) setup is supported in WebLogic in secured mode. For enterprise deployment, it is recommended to use SSL certificates signed by certificate authorities.

> **Note:** You need to obtain a separate signed SSL certificate for each host where the application is being deployed.

The Security Guide focuses on securing Oracle Retail Applications in a single node setup and not on applications deployed in clusters.

Certificate Authority

Certificate Authority or certification Authority (CA) is an organization which provides digital certificates to entities and acts as a trusted third party. Certificates issues by the commercial CAs are automatically trusted by most of the web browsers, devices and applications. It is recommended to have certificates obtained from a trusted CA or commercial CAs to ensure better security.

Obtaining an SSL Certificate and Setting up a Keystore

> **Note:** SSL certificates are used to contain public keys. With each public key there is an associated private key. It is critically important to protect access to the private key. Otherwise, the SSL messages may be decrypted by anyone intercepting the communications.

Perform the following steps to obtain an SSL certificate and set up a Keystore:

1. Obtain an identity (private key and digital certificate) and trust (certificate of trusted certificate authority) for the WebLogic Server.
2. Use the digital certificate, private key and trusted CA certificate provided by the WebLogic Server Kit, the CertGen utility, Sun Microsystem's keytool utility, or a reputed vendor such as Entrust or Verisign to perform the following:
    a. Set appropriate JAVA_HOME and PATH to java, as shown in the following example:

    ```
    export JAVA_HOME=/u00/webadmin/product/jdk
    export PATH=$JAVA_HOME/bin:$PATH
    ```

    b. Create a new keystore

    ```
    Keytool -genkey -keyalg RSA -keysize 2048 -keystore<keystore> -
    alias<alias>
    ```

    For example:

    ```
    keytool -genkey -keyalg RSA -keysize 2048 -keystore hostname.keystore
    -alias hostname
    ```

    c. Generate the signing request.

    ```
    keytool -certreq -keyalg RSA -file <certificate request file> -
    keystore
    <keystore> -alias <alias>
    ```

    For example:

```
keytool -certreq -keyalg RSA -file hostname.csr -keystore hostname.keystore -alias
hostname
```

d. Submit the certificate request to CA

3. Store the identity and trust.

   Private keys and trusted CA certificates which specify identity and trust are stored in a keystore.

   In the following examples the same keystore to store all certificates are used

   a. Import the root certificate into the keystore as shown in the following example:

   ```
   keytool -import -trustcacerts –alias verisignclass3g3ca -file Primary.pem -
   keystore hostname.keystore
   ```

   A root certificate is either an unsigned public key certificate or a self-signed certificate that identifies the Root CA.

   b. Import the intermediary certificate (if required into the keystore as shown in the following example.

   ```
   keytool -import -trustcacerts -alias oracleclass3g3ca -file Secondary.pem
   -keystore hostname.keystore
   ```

   c. Import the received signed certificate for this requeste into the keystore as shown in the following example:

   ```
   keytool -import -trustcacerts -alias hostname -file cert.cer –keystore
   hostname.keystore
   ```

Creating a Weblogic Domain

WebLogic domain is created for Oracle Retail Applications as part of the installation. Different domains are created in different hosts for different applications in situations where applications are being managed by different users or deployed on different hosts. Once the domains are created, you need to enable the SSL ports if not done already.

1. Perform the following steps to enable the SSL:
2. Log in to WebLogic console using Administrator user. For example, weblogic.
3. Navigate to <Domain> > Environment > Servers > < Servername> > Configuration > General tab.
4. Click **Lock & Edit**.
5. Select **SSL Listen Port Enabled** and assign the port number.
6. Click **Save** and **Activate Changes**.
7. Restart SSL to enable the changes.

Figure 1 Restarting the Admin Server

Configuring the Application Server for SSL

Perform the following steps to configure the Application Server for SSL:

1. Configure the identity and trust keystores for WebLogic Server in the WebLogic Server Administration Console.
    a. In the Change Center of the Administration Console, click **Lock & Edit**.
    b. In the left pane of the console, expand Environment and select **Servers.**
    c. Click the name of the server for which you want to configure the identity and trust keystores as shown in the following example:

    ```
    WLS_RMS is for RMS server
    ```

    d. Select **Configuration,** then **Keystores.**

    The following options are available:

    - **Demo Identity and Demo Trust** - The demonstration identity and trust keystores, located in the BEA_HOME\server\lib directory and the Java Development Kit (JDK) cacerts keystore, are configured by default. You need to use for development purpose only.

    - **Custom Identity and Java Standard Trust** - A keystore you create and the trusted CAs defined in the cacerts file in the JAVA_HOME\jre\lib\security directory.

    - **Custom Identity and Custom Trust [Recommended]** - An Identity and trust keystores you create.

    - **Custom Identity and Command Line Trust**: An identity keystore you create and command-line arguments that specify the location of the trust key.

    e. Select **Custom Identity** and **Custom Trust**.
    f. In the Identity section, define the following attributes for the identity keystore:

    - **Custom Identity Keystore** - This is the fully qualified path to the identity keystore.

    - Custom Identity Keystore Type - This is the type of the keystore. Generally, this attribute is Java KeyStore (JKS); if it is left blank, it defaults to JKS.

    - Custom Identity Keystore Passphrase - This is the password you must enter when reading or writing to the keystore. This attribute is optional or required depending on the type of keystore. All keystores require the passphrase in order to write to the keystore. However, some keystores do not require the passphrase to read from the keystore. WebLogic Server only reads from the keystore so whether or not you define this property depends on the requirements of the keystore.

    g. In the **Trust** section, define properties for the trust keystore.

    If you choose Java Standard Trust as your keystore, specify the password defined when creating the keystore.

h. Confirm the password.

If you choose **Custom Trust [Recommended]** define the following attributes:

- **Custom Trust Keystore** - This is the fully qualified path to the trust keystore.

- **Custom Trust Keystore Type** - This is the type of the keystore. Generally, this attribute is JKS; if it is left blank, it defaults to JKS.

- **Custom Trust Keystore Passphrase** - This is the password that you need to enter when reading or writing to the keystore. This attribute is optional or required depending on the type of keystore. All keystores require the passphrase in order to write to the keystore. However, some keystores do not require the passphrase to read from the keystore. WebLogic Server only reads from the keystore, so whether or not you define this property depends on the requirements of the keystore.

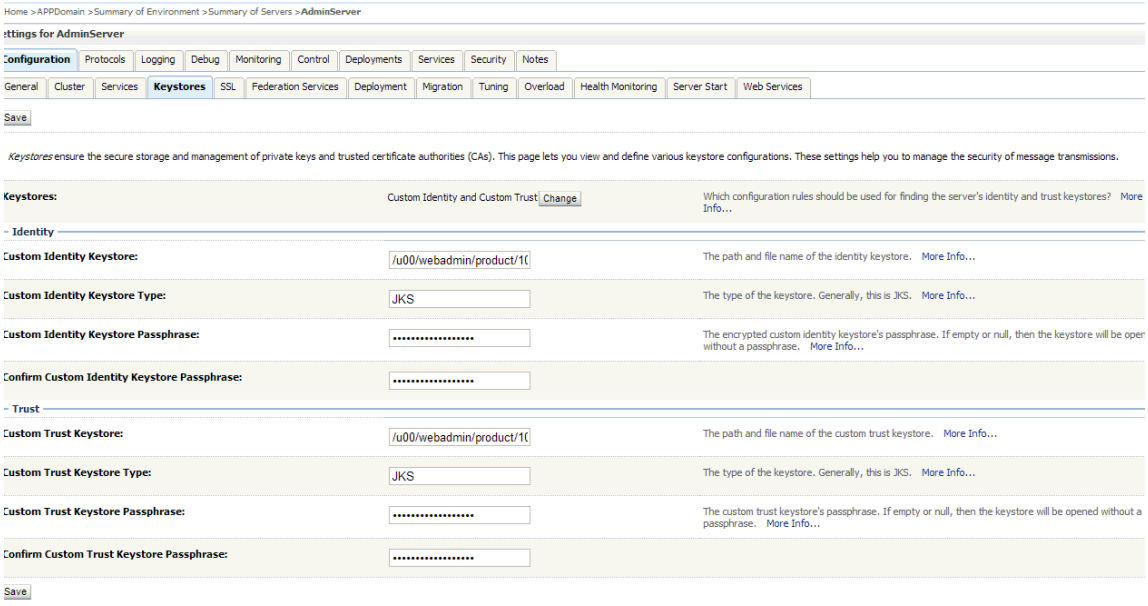
i. Click Save.

j. To activate these changes in the changes, in the change Center of the Administration Console, click Activate Changes.

Note: Not all changes take effect immediately, some require a restart.

Figure 2 shows how to configure the application server for SSL

Figure 2 Configuring the Identity and Trust Keystores for WebLogic Server



For more information on configuring Keystores, see the *Administration Console Online Help*.

2. Set SSL Configuration Options for the private key alias and password in the WebLogic Server Administration Console.

a. In the Change Center of the Administration Console, click **Lock & Edit**.

b. In the left pane of the Console, expand **Environment** and select **Servers**.

c. Click the name of the server for which you want to configure the identity and trust keystores.

d. Select **Configuration**, then select **SSL**.

e. In the Identity and Trust Locations, the Keystore is displayed by default.

f. In the **Private Key Alias**, type the string alias that is used to store and retrieve the server's private key.

g. In the **Private Key Passphrase**, provide the keystore attribute that defines the passphrase used to retrieve the server's private key.

h. Save the changes.

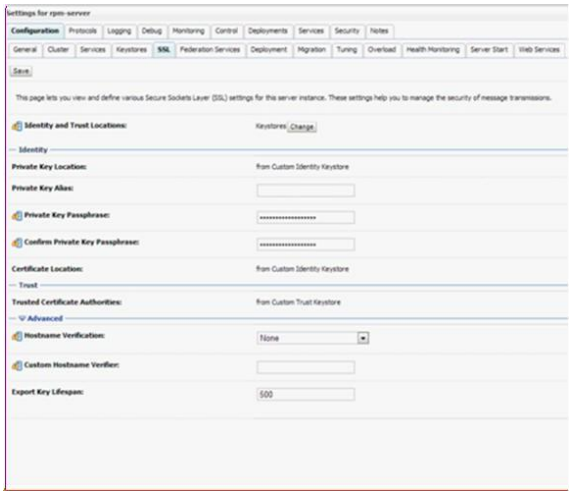
i. Click **Advanced** section of SSL tab.

j. In the Hostname Verification, select None.

   This specifies to ignore the installed implementation of the WebLogic.security.SSL.HostnameVerifier interface (this interface is generally used when this server is acting as a client to another application server).

k. **11.** Save the change

Figure 3 Configuring SSL



For more information on configuring SSL, see the section *Configure SSL* in the *Administration Console Online Help*.

All the server SSL attributes are dynamic; when modified through the Console. They cause the corresponding SSL server or channel SSL server to restart and use the new settings for new connections. Old connections will continue to run with the old configuration. You must reboot WebLogic Server to ensure that all the SSL connections exist according to the specified configuration.

Use the **Restart SSL** button on the **Control**: Start/Stop page to restart the SSL server when changes are made to the keystore files. You have to apply the same for subsequent connections without rebooting WebLogic Server.

Upon restart you can see the following similar entries in the log:

```
< Jan **, 20** 5:18:27 AM CDT> <Notice> <WebLogicServer> <BEA-000365> <Server state changed to
RESUMING>
```

```
< Jan **, 20** 5:18:27 AM CDT> <Notice> <Server> <BEA-002613> <Channel "DefaultSecure" is now
ing on 10.141.15.214:57002 for protocols iiops, t3s, ldaps, https.>
<Jan **, 20** 5:18:27 AM CDT> <Notice> <Server> <BEA-002613> <Channel "DefaultSecure[1]" is now
ing on 127.0.0.1:57002 for protocols iiops, t3s, ldaps, https.>
< Jan **, 20** 5:18:27 AM CDT> <Notice> <WebLogicServer> <BEA-000329> <Started WebLogic Admin
Server "AdminServer" for domain "APPDomain" running in Production Mode>
< Jan **, 20** 5:18:27 AM CDT> <Notice> <WebLogicServer> <BEA-000365> <Server state changed to
RUNNING>
< Jan **, 20** 5:18:27 AM CDT> <Notice> <WebLogicServer> <BEA-000360> <Server started in RUNNING
mode>
```

Note: For complete security of the Weblogic Server, it is recommended to secure both Administration as well as the Managed Server where the application is being deployed. You can choose to disable the non-SSL ports (HTTP. It is recommended to secure the Node Manager.

The steps to secure the Node Manager is provided in the following section.

Configuring WebLogic Scripts if Admin Server is Secured

Perform the following steps to configure the WebLogic scripts if Admin Server is secured:

1. Update the WebLogic startup/shutdown scripts with secured port and protocol to start/stop services.
2. Backup and update the following files in <DOMAIN_HOME>/bin with correct Admin server urls:

   startManagedWebLogic.sh: echo "$1 managedserver1 http://apphost1:7001"

   stopManagedWebLogic.sh: echo "ADMIN_URL defaults to t3://apphost1:7001 if not set as an environment variable or the second command-line parameter."

   stopManagedWebLogic.sh: echo "$1 managedserver1 t3://apphost1:7001

   WebLogic

   stopManagedWebLogic.sh: ADMIN_URL="t3://apphost1:7001"

   stopWebLogic.sh: ADMIN_URL="t3://apphost1:7001"
3. Change the URLs as follows:

   `t3s://apphost1:7002 https://apphost1:7002`

Adding Certificate to the JDK Keystore for Installer

You will need the Oracle Retail Application installer to run Java. In situations where Administration Server is secured using signed certificate, the Java keystore through which the installer is launched must have the certificate installed.

In case the installer is being run using JDK deployed at location /u00/webadmin/product/jdk, follow the steps as shown in the example below.

Adding certificate to the JDK keystore for Installer

```
apphost1:_apps] /u00/webadmin/ssl> keytool -import -trustcacerts -alias apphost1 -file
/u00/webadmin/ssl/apphost1.cer -keystore
/u00/webadmin/product/jdk/jre/lib/security/cacerts
Enter keystore password: Certificate was added to keystore apphost1:[_apps]
/u00/webadmin/ssl>
```

Enforcing Stronger Encryption in WebLogic

It is recommended to use a stronger encryption protocol in your production environment.

See the following sections to enable the latest SSL and cipher suites.

SSL protocol version configuration

In a production environment, Oracle recommends Transport Layer Security (TLS) Version 1.2 for sending and receiving messages in an SSL connection.

- Set the **WebLogic.security.SSL.minimumProtocolVersion=protocol** system property as an option in the command line that starts WebLogic Server.

This system property accepts one of the following values for protocol:

Figure 4 Values for Protocol of System Property

Value	Description
SSLv3	Specifies SSL V3.0 as the minimum protocol version enabled in SSL connections.
TLSv1	Specifies TLS V1.0 as the minimum protocol version enabled in SSL connections.
TLSv*x.y*	Specifies TLS V*x.y* as the minimum protocol version enabled in SSL connections, where: • *x* is an integer between 1 and 9, inclusive • *y* is an integer between 0 and 9, inclusive For example, TLSv1.2.

- Set the following property in startup parameters in WebLogic Managed server for enabling the higher protocol:

DWebLogic.security.SSL.minimumProtocolVersion=TLSv1.2 -Dhttps.protocols=TLSv1.2

Note: In case protocol is set for Managed servers, the same should be set for the Administration server. Ensure that all the managed servers are down when making changes to the Administration server for setting up the protocol. It is recommended to set the properties in the Administration Server and then Managed Server.

Enabling Cipher inWebLogic SSL Configuration

Configure the <ciphersuite> element in the <ssl> element in the <DOMAIN_HOME>\server\config\config.xml file in order to enable the specific Cipher Suite to use as follows:

> Note: You need to ensure that the tag <ciphersuite> is added immediately after tab <enabled>.

```
<ssl>
<name>examplesServer</name>
<enabled>true</enabled>
<ciphersuite>TLS_RSA_WITH_AES_256_CBC_SHA</ciphersuite>
<-port>17002</-port>
...
</ssl>
```

Securing Nodemanager with SSL Certificates

Perform the following steps for securing the Nodemanager with SSL certificates:

1. Navigate to weblogic 12c domain, the location is **<DOMAIN_HOME>/nodemanager**) and take a backup of nodemanager.properties.
2. Add the following similar entries to nodemanager.properties:

   KeyStores=CustomIdentityAndCustomTrust
   CustomIdentityKeyStoreFileName=/u00/webadmin/ssl/hostname.keystore
   CustomIdentityKeyStorePassPhrase=[password to keystore, this will get encrypted]

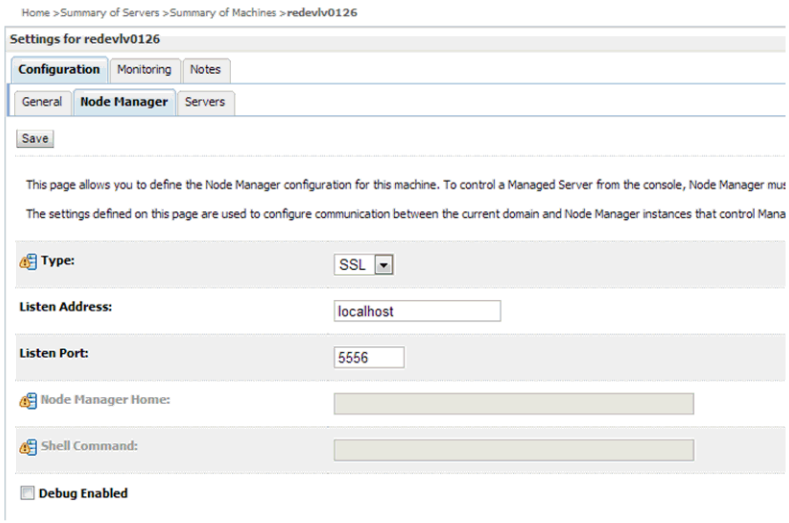
   CustomIdentityAlias=hostname

   CustomIdentityPrivateKeyPassPhrase=[password to keystore, this will get encrypted]

   CustomTrustKeyStoreFileName=/u00/webadmin/ssl/hostname.keystore

   SecureListener=true
3. Log in to **WebLogic console**, navigate to **Environment**, and then **Machines**.
4. Select the nodemanager created already and navigate to **Node Manager** tab.
5. In the Change Center, click **Lock & Edit**.
6. In the **Type** field, select **SSL** from the list.
7. Click Save and Activate.

Figure 5 Securing the Nodemanager



8. You need to bounce the entire WebLogic Domain for changes to take effect, after activating the changes.
9. You need to verify if the nodemanager is reachable in **Monitoring** tab after restart.

Using Secured Lightweight Directory Access Protocol (LDAP)

The Application can communicate with LDAP server on a secured port. It is recommended to use the secured LDAP server to protect user names and passwords from being sent in clear text on the network.

For information on Configuring Secure Sockets Layer (SSL), see the *Oracle Fusion Middleware Administration Guide*.

It is important to import the certificates used in LDAP server into the Java Runtime Environment (JRE) of the WebLogic server for SSL handshake, in case the secure LDAP is used for authentication.

For example:

1. Set JAVA_HOME and PATH to the JDK being used by WebLogic Domain.
2. Backup the JAVA_HOME/jre/lib/security/cacerts
   /u00/webadmin/product/jdk/jre/lib/security> cp -rp cacerts cacerts_ORIG
3. Import the Root and Intermediary (if required) certificates into the java keystore.
   /u00/webadmin/product/jdk/jre/lib/security> keytool -import -trustcacerts
   -alias digicertroot -file ~/ssl/Primary.pem -keystore cacerts/u00/webadmin/product/jdk/jre/lib/security> keytool -import -trustcacerts-alias digicertinter -file ~/ssl/Secondary.pem -keystore cacerts
4. Import the User certificate from LDAP server into the java keystore.
   /u00/webadmin/product/jdk/jre/lib/security> keytool -import -trustcacerts
   -alias hostname -file ~/ssl/cert.cer -keystore cacerts

Note: The default password of the JDK keystore is **changeit**

The deployed application should be able to communicate with LDAP on SSL port after successful SSL Handshake.

Advanced Infrastructure Security

Depending upon your security need for your production environment, infrastructure where Oracle Retail applications are deployed can be secured.

Ensure the following to secure complete protection of environment:

- Securing the WebLogic Server Host
- Securing Network Connections
- Securing your Database
- Securing the WebLogic Security Service
- - Securing Applications

For more information on Ensuring the Security of Your Production Environment, see https://docs.oracle.com/middleware/12214/wls/SECMG/toc.htm *Guide*.

R

Appendix – Post Installation of Retail Infrastructure in Database

Oracle Retail applications use the Oracle database as the backend data store for applications. In order to ensure complete environment security the database should be secured.

This chapter describes the post installation steps for secured setup of Retail infrastructure in the Database.

- The following topics are covered in this chapter:
- Configuring SSL Connections for Database Communications
- Configuring the Password Stores for Database User Accounts
- Configuring the Database Password Policies
- Configuring SSL Connection for Oracle Data Integrator (ODI)
- Creating an Encrypted Tablespace in Oracle 19c Container Database
- Additional Information

Configuring SSL Connections for Database Communications

Secure Sockets Layer (SSL) is the standard protocol for secure communications, providing mechanisms for data integrity and encryption. This can protect the messages sent and received by the database to applications or other clients, supporting secure authentication and messaging. Configuring SSL for databases requires configuration on both the server and clients, which include application servers.

This section covers the steps for securing Oracle Retail Application Clusters (RAC) database. Similar steps can be followed for single node installations also.

Configuring SSL on the Database Server

The following steps are one way to configure SSL communications on the database server:

1. Obtain an identity (private key and digital certificate) and trust (certificates of trusted certificate authorities) for the database server from a Certificate Authority.
2. Create a folder containing the wallet for storing the certificate information. For Real Application Cluster (RAC) systems, this directory can be shared by all nodes in the cluster for easier maintenance.

   mkdir-p/oracle/secure_wallet
3. Create a wallet in the path. For example

   ```
   orapki wallet create -wallet /oracle/secure_wallet -auto_login
   ```
4. Import each trust chain certificate into the wallet as shown in the following example:

   ```
   orapki wallet add -wallet /oracle/secure_wallet -trusted_cert -cert <trust chain certificate>
   ```
5. Import the user certificate into the wallet, as shown in the following example:

   ```
   orapki wallet add -wallet /oracle/secure_wallet -user_cert -cert <certificate file location
   ```
6. .Update the listener.ora by adding a TCPS protocol end-point first in the list of end points.

```
LISTENER1= (DESCRIPTION=
(ADDRESS=(PROTOCOL=tcps)(HOST=<dbserver>)(PORT=2484))
(ADDRESS=(PROTOCOL=tcp)(HOST=<dbserver>)(PORT=1521)))
```

7. Update the listener.ora by adding the wallet location and disabling SSL authentication.

   ```
   WALLET_LOCATION = (SOURCE= (METHOD=File) (METHOD_DATA=
   (DIRECTORY=wallet_location))) SSL_CLIENT_AUTHENTICATION=FALSE
   ```

8. Update the sqlnet.ora with the same wallet location information and disabling SSL authentication.

   ```
   WALLET_LOCATION = (SOURCE= (METHOD=File) (METHOD_DATA=
   (DIRECTORY=wallet_location))) SSL_CLIENT_AUTHENTICATION=FALSE
   ```

9. Update the tnsnames.ora to configure a database alias using TCPS protocol for Connections.

   ```
   <dbname>_secure= (DESCRIPTION= (ADDRESS_LIST=
   (ADDRESS=(PROTOCOL=TCPS)(HOST=<dbserver>)(PORT=2484)))
   (CONNECT_DATA=(SERVICE_NAME=<dbname>)))
   ```

10. Restart the database listener to pick up listener.ora changes.
11. Verify the connections are successful to the new >dbname>_secure alias.
12. At this point either the new secure alias can be used to connect to the database, or the regular alias can be modified to use TCPS protocol.
13. Export the identity certificate so that it can be imported on the client systems

    ```
    orapki wallet export -wallet /oracle/secure_wallet -dn <full dn of identity
    certificate> -cert <filename_to_create>
    ```

Configuring SSL on an Oracle Database Client

The following steps are one way to configure SSL communications on the database client:

1. Create a folder containing the wallet for storing the certificate information.

   ```
   Mkdir-p / oracle/secure_wallet
   ```

2. Create a wallet in the path. For example:

   ```
   orapki wallet create -wallet /oracle/secure_wallet -auto_login
   ```

3. Import each trust chain certificate into the wallet as shown in the following example:

   ```
   orapki wallet add -wallet /oracle/secure_wallet -trusted_cert -cert <trust
   chain certificate
   ```

4. Import the identity certificate into the wallet, as shown in the following example:

   ```
   orapki wallet add -wallet /oracle/secure_wallet -trusted_cert –cert
   <certificate file location>
   ```

   Note: on the client the identity certificate is imported as a trusted certificate, whereas on the server it is imported as a user certificate.

5. Update the sqlnet.ora with the wallet location information and disabling SSL authentication.

   ```
   WALLET_LOCATION = (SOURCE= (METHOD=File) (METHOD_DATA=
   (DIRECTORY=wallet_location))) SSL_CLIENT_AUTHENTICATION=FALSE
   ```

6. Update the tnsnames.ora to configure a database alias using TCPS protocol for connections.

   ```
   <dbname>_secure= (DESCRIPTION=
   (ADDRESS_LIST= (ADDRESS=(PROTOCOL=TCPS)(HOST=<dbserver>)(PORT=2484)))
   (CONNECT_DATA=(SERVICE_NAME=<dbname>)))
   ```

7. Verify the connections are successful to the new (dbname)_secure alias.
8. At this point either the new secure alias can be used to connect to the database, or the regular alias can be modified to use TCPS protocol.

Configuring SSL on a Java Database Connectivity (JDBC) Thin Client

The following steps are one way to configure SSL communications for a Java Database

1. Create a folder containing the keystore with the certificate information.
   ```
   mkdir-p /oracle/secure_jdbc
   ```
2. Create a keystore in the path. For example,
   ```
   keytool -genkey -alias jdbcwallet -keyalg RSA -keystore /oracle/secure_
   jdbc/truststore.jks -keysize 2048
   ```
3. Import the database certificate into the trust store as shown in the following example:
   ```
   keytool -import -alias db_cert -keystore /oracle/secure_jdbc/truststore.jks
   -file <db certificate file>
   ```
4. JDBC clients can use the following URL format for JDBC connections:
   ```
   jdbc:oracle:thin:@(DESCRIPTION= (ADDRESS= (PROTOCOL=tcps) (HOST=<dbserver>)
   (PORT=2484))  (CONNECT_DATA=  (SERVICE_NAME=<dbname>)))
   ```

   Note: The <dbname> would be replaced with the service name in case of a multitenant database.

5. You need to set the properties as shown in the table, either as system properties or as JDBC connection properties.

Table 1 Setting the Properties

Property	Value
Javax.net.ssl.trustStore	Path and file name of truststore. for Example, /oracle/secure_jdbc/truststore.jks
Javax.net.ssl.trustStoreType	Jks
Javax.net.ssl.trustStorePasword	password for trust store

Configuring the Password Stores for Database User Accounts

Wallets can be used to protect sensitive information, including usernames and passwords for database connections. The Oracle Database client libraries have built-in support for retrieving credential information when connecting to databases. Oracle Retail applications utilize this functionality for non-interactive jobs such as batch programs so that they are able to connect to the database without exposing user and password information to other users on the same system.

For information on configuring wallets for database access, see the Appendix Setting Up Password Stores with Oracle Wallet in the product installation guide.

Configuring the Database Password Policies

Oracle Database includes robust functionality to enforce policies related to passwords such as minimum length, complexity, when it expires, number of invalid attempts, and so on. Oracle Retail recommends these policies are used to strengthen passwords and lock out accounts after failed attempts.

For example, to modify the default user profile to lock accounts after five failed login attempts, run the following commands as a database administrator:

1. Query the current settings of the default profile select resource_name,limit,resource_type from dba_profiles where profile='DEFAULT';
2. Alter the profile, if failed_login_attempts is set to unlimited: alter profile default limit FAILED_LOGIN_ATTEMPTS 5;

> **Note:** Many other profile settings are available for increased security. For more information, see the Oracle Database Security Guide.

Creating an Encrypted Tablespace in Oracle 19c Container Database

The retail tablespaces can be encrypted in container databases using the following method:

1. Update the SQLNET.ORA file with the following encryption details:
   a. Configure the sqlnet.ora file for a software keystore location

   ```
   ENCRYPTION_WALLET_LOCATION= (SOURCE=
   (METHOD=FILE) (METHOD_DATA= (DIRECTORY=path_to_keystore)))
   ```

   b. Restart the listener
2. Set up the Tablespace Encryption in the container database.
   a. Create Software Keystores as follows:

   ```
   SQL> ADMINISTER KEY MANAGEMENT CREATE KEYSTORE
   '/u03/wallet_cdb' IDENTIFIED BY "val1ue#";
   Kystore altered
   ```

   b. Create an Auto-Login Software Keystore as follows:

   ```
   SQL> ADMINISTER KEY MANAGEMENT CREATE AUTO_LOGIN KEYSTORE FROM KEYSTORE
   '/u03/wallet_cdb' identified by "val1ue#'; Keystore altered.
   ```

   > **Note:** the auto-login software keystore can be opened from different computers from the computer where this keystore resides. However, the [local] auto-login software keystore can only be opened from the computer on which it was created.

   c. Open the software Keystore as follows:

   ```
   SQL> ADMINISTER KEY MANAGEMENT SET KEYSTORE OPEN IDENTIFIED BY
   "val1ue#" Container=ALL;
   Keystore altered.
   ```

   d. Set the Software TDE Master Encryption key as follows:

   ```
   SQL> ADMINISTER KEY MANAGEMENT SET KEY IDENTIFIED BY "val1ue#" WITH
   BACKUP USING 'TDE_ENCRYPTION' Container=all;
   Keystore altered
   ```

   > Note: One can set the Encryption Key only for a particular PDB if required, by specifying the Container =<PDB>

   e. Create the ENCRYPTED TABLESPACE in PDB as follows:

   SQL>conn sys / xxxxx@orcl as sysdba Connected

   SQL> create tablespace test datafile '+DATA1' size 100m ENCRYPTION DEFAULT STORAGE (ENCRYPT);

   Tablespace created.

f. Verify the Encryption:

```
SQL> select * from v$encryption_wallet
```

WRL TYPE	WRL PARAMETER	STATUS	WALLET TYPE	WALLET OR	FULLY BAC	CON id
File	/u03/wallet_cdb	OPEN	PASSWORD	SINGLE	NO	0

3. For more information on Configuring Transparent Data Encryption (TDE) see https://docs.oracle.com/en/database/oracle/oracle-database/19/asoag/asopart1.html
4. Other information may be useful during maintenance activity.
   a. Close the encryption wallet as follows.

```
SQL> ADMINISTER KEY MANAGEMENT SET KEYSTORE Close
IDENTIFIED BY "va11ue#" Container=ALL
```

Additional Information

For more information on the subjects covered in this section as well as information on other options that are available to strengthen database security, see the *Oracle Database Security guide 19c* https://docs.oracle.com/en/database/oracle/oracle-database/19/dbseg/index.html.

The Oracle Advanced Security Option provides industry standards-based solutions to solve enterprise computing security problems, including data encryption and strong authentication. Some of the capabilities discussed in this guide require licensing the Advanced Security Option.

For more information, see the *Oracle Database Advanced Security Administrator's Guide 19c* https://docs.oracle.com/en/database/oracle/oracle-database/19/asoag/index.html

Appendix – Post Installation of Retail Infrastructure in WebLogic

This chapter describes the post installation steps for secured setup of Oracle Retail infrastructure in WebLogic.

The following topics are covered in this chapter:

- Retail Application Specific Post installation Steps for Security
- Batch Set Up for SSL Communication
- Oracle Business Intelligence (BI) Publisher - Disable Guest User - Optional
- RMS - Forms Timeout Setting - Optional
- Asynchronous Task JMS Queue Security
- Hardening Use of Headers and Transport Layer Security

Retail Application Specific Post installation Steps for Security

See the following sections for steps to improve security after an Oracle Retail Application has been installed.

Batch Set Up for SSL Communication

Java batch programs communicate with Java applications deployed in WebLogic.

The communication needs to have SSL handshake with the deployed application. You need to import the SSL Certificates into the JAVA_ HOME/jdk/jre/lib/security/cacerts keystore for successful running of the application batches.

Example: Importing certificates into JDK keystore

```
/u00/webadmin/product/jdk/jre/lib/security> cp -rp cacerts cacerts_ORIG
/u00/webadmin/product/jdk/jre/lib/security> keytool -import -trustcacerts -alias
digicertroot -file ~/ssl/Primary.pem -keystore cacerts
/u00/webadmin/product/jdk/jre/lib/security> keytool -import -trustcacerts -alias
digicertinter -file ~/ssl/Secondary.pem -keystore cacerts
/u00/webadmin/product/jdk/jre/lib/security> keytool -import -trustcacerts -alias
hostname -file ~/ssl/cert.cer -keystore cacerts
/u00/webadmin/product/jdk/jre/lib/security> keytool -import -trustcacerts -alias
hostname -file ~/ssl/cert.cer -keystore cacerts
```

Note: The default password of JDK keystore is **changeit**

T

Appendix-Using Self Signed Certificates

Self-signed certificates can be used for development environment for securing applications. The generic steps to be followed for creating self signed certificates and configuring for use for Oracle Retail application deployment are covered in the subsequent sections.

The following topics are covered in this chapter:

- Creating a Keystore through the Keytool in Fusion Middleware (FMW) 12c
- Exporting the Certificate from the Identity Keystore into a File
- Importing the Certificate Exported into trust.keystore
- Configuring WebLogic
- Configuring Nodemanager
- Importing Self Signed Root Certificate into Java Virtual Machine (JVM) Trust Store
- Disabling Hostname Verification
- Converting PKCS7 Certificate to x.509 Certificate

Creating a Keystore through the Keytool in Fusion Middleware (FMW) 12c

Perform the following steps to create a keystore through the keytool in Fusion Middleware (FMW) 12c:

1. Create a directory for storing the keystores.

   ```
   $ mkdir ssl
   ```

2. Run the following to set the environment:

   ```
   $ cd $MIDDLEWARE_HOME/user_projects/domains/<domain>/bin
   $ . ./setDomainEnv.sh
   ```

Example:

```
apphost2:[_apps] /u00/webadmin/product/12c/WLS/user_
projects/domains/APPDomain/bin> . ./setDomainEnv.sh apphost2:[_apps]
/u00/webadmin/product/12c/WLS/user_ projects/domains/APPDomain>
```

3. Create a keystore and private key, by executing the following command:

   ```
   keytool -genkey -alias <alias> -keyalg RSA -keysize 2048 -dname <dn> -keypass
   <password> -keystore <keystore> -storepass <password> -validity 365
   ```

Example:

```
apphost2:[_apps] /u00/webadmin/ssl> keytool -genkey -alias apphost2
-keyalg RSA -keysize 2048 -dname "CN=<Server Name>,OU=<Organization Unit>,
O=<Organization>,L=<City>,ST=<State>,C=<Country>" -keypass <kpass> -keystore
/u00/webadmin/ssl/apphost2.keystore -storepass <spass> -validity 365

apphost2:[_apps] /u00/webadmin/ssl> ls -ltra total 12
drwxr-xr-x 18 webadmin dba 4096 Apr 4 05:31 ..
-rw-r--r--  1 webadmin dba 2261 Apr 4 05:46 apphost2.keystore drwxr-xr-x  2
webadmin dba 4096 Apr 4 05:46 . apphost2:[_apps] /u00/webadmin/ssl>
```

Exporting the Certificate from the Identity Keystore into a File

Perform the following steps to export the certificate from the identity keystore into a file (for example, pubkey.cer):

1. Run the following command:

```
$ keytool -export -alias selfsignedcert -file pubkey.cer -keystore
identity.jks
-storepass <password>
```

Example:

```
apphost2:[_apps] /u00/webadmin/ssl> keytool -export -alias apphost2 -file
/u00/webadmin/ssl/pubkey.cer -keystore /u00/webadmin/ssl/apphost2.keystore
-storepass <spass>
Certificate stored in file </u00/webadmin/ssl/ropubkey.cerot.cer>
apphost2:[_apps] /u00/webadmin/ssl> ls -l total 8
-rw-r--r-- 1 webadmin dba 2261 Apr 4 05:46 apphost2.keystore
-rw-r--r-- 1 webadmin dba 906 Apr 4 06:40 pubkey.cer apphost2:[_apps]
/u00/webadmin/ssl>
```

Importing the Certificate Exported into trust.keystore

Perform the following steps to import the certificate you exported into trust.keystore

1. Run the following command

```
$ keytool -import -alias selfsignedcert -trustcacerts -file pubkey.cer -
keystore trust.keystore -storepass <password>
```

Example:

```
apphost2:[_apps] /u00/webadmin/ssl> keytool -import -alias apphost2
-trustcacerts -file pubkey.cer -keystore trust.keystore -storepass <spass>
Owner: CN=apphost2, OU=<Organization Unit>, O=<company>,L=<city>,ST=<state or
province>, C=<country>
Issuer: CN=apphost2, OU=<Organization Unit>, O=<company>,L=<city>,ST=<state or
province>, C=<country>
Serial number: 515d4bfb
Valid from: Thu Apr 04 05:46:35 EDT 2013 until: Fri Apr 04 05:46:35 EDT 2014
Certificate fingerprints:
MD5: AB:FA:18:2B:BC:FF:1B:67:E7:69:07:2B:DB:E4:C6:D9
SHA1: 2E:98:D4:4B:E0:E7:B6:73:55:4E:5A:BE:C1:9F:EA:9B:71:18:60:BB

SHA256:2E:98:D4:4B:E0:E7:B6:73:55:4E:5A:BE:C1:9F:EA:9B:71:18:60:BB Signature
algorithm name: SHA256withRSA
Version: 3
Trust this certificate? [no]: yes Certificate was added to keystore
apphost2:[10.3.6_apps] /u00/webadmin/ssl>
```

Configuring WebLogic

You need to enable SSL for WebLogic server's Admin and managed servers by following the steps as provided in Configuring the Application Server for SSL section.

Configuring Nodemanager

You need to secure the Node manager by following the steps in Securing Nodemanager with SSL Certificates section.

Importing Self Signed Root Certificate into Java Virtual Machine (JMM) Trust Store

In order for the Java Virtual Machine (JVM) to trust in your newly created certificate, import your custom certificates into your JVM trust store.

Perform the following steps to import the root certificate into JVM Trust Store:

1. Ensure that JAVA_HOME has been already set up.

2. Run the following command:

```
$keytool -import -trustcacerts -file rootCer.cer -alias selfsignedcert -
keystore cacerts
```

Example

```
apphost2:[_apps] /u00/webadmin/product/jdk/jre/lib/security> keytool -import -
trustcacerts -file
/u00/webadmin/ssl/root.cer -alias apphost2 -keystore
/u00/webadmin/product/ jdk /jre/lib/security/cacerts -storepass
[spass default is changeit]
Owner: CN=apphost2, OU=<Organization Unit>, O=<company>,L=<city>,ST=<state or province>,
C=<country>"
Issuer: CN=apphost2, OU=<Organization Unit>, O=<company>,L=<city>,ST=<state or province>,
C=<country>"
Serial number: 515d4bfb
Valid from: Thu Apr 04 05:46:35 EDT 2013 until: Fri Apr 04 05:46:35 EDT 2014
Certificate fingerprints:
    MD5:  AB:FA:18:2B:BC:FF:1B:67:E7:69:07:2B:DB:E4:C6:D9
    SHA1: 2E:98:D4:4B:E0:E7:B6:73:55:4E:5A:BE:C1:9F:EA:9B:71:18:60:BB
    SHA256:DA:8B:72:24:DB:C2:B5:26:50:30:8F:8E:15:A5:34:56:DD:5D:18:28:11:17:40:6A:B2:69:16:E
    5:B8:26:5D:25
    Signature algorithm name: SHA256withRSA
    Version: 3
Trust this certificate? [no]: yes Certificate was added to keystore apphost2:[_apps]
/u00/webadmin/product/ jdk /jre/lib/security>
```

Converting PKCS7 Certificate to X.509 Certificate

Certificate authorities provide signed certificates of different formats. However, not all formats of certificates can be imported to Java based keystores. Hence the certificates need to be converted to usable form. Java based Keystores supports x.509 format of certificate.

The following example demonstrates converting certificate PKCS 7 to x.509 format:

1. Copy the PKCS 7 certificate file to a Windows desktop.
2. Rename the file and provide .p7b extension.
3. Open the .p7b file.
4. Click the plus (+) symbol.
5. Click the Certificates directory.

An Intermediary certificate if provided by CA for trust.

> **Note:** If an Extended Validation certificate is being converted you should see three files. The End Entity certificate and the two EV intermediate CAs.

6. Right click on your certificate file.
7. Select All Tasks > Export.
8. Click **Next**.
9. Select Base-64 encoded X.509 (.cer) > click Next.
10. Browse to a location to store the file.
11. Enter a File name.
12. For example, MyCert. The .cer extension is added automatically.
13. Click **Save**.
14. Click **Next**.
15. Click **Save**.

The Certificate can now be imported into Java based keystores.

Example:

```
apphost1:[_apps] /u00/webadmin/ssl> keytool -import -trustcacerts -alias apphost1
-file /u00/webadmin/ssl/cert-x509.cer -keystore
/u00/webadmin/product/jdk/jre/lib/security/cacerts Enter keystore password:
[default is changeit] Certificate was added to keystore apphost1:[_apps]
/u00/webadmin/ssl>
```