Contents

Preface............................................................................................................................................................... xv
  Audience......................................................................................................................................................... xv
  Downloading Oracle Communications Documentation.............................................................................. xv
  Document Revision History ......................................................................................................................... xv
  Documentation Accessibility ....................................................................................................................... xv

1 OSM System Administration Overview
  OSM System Administration Tasks ........................................................................................................... 1-1
  About OSM System Administration Programs ....................................................................................... 1-1
  Directory Placeholders Used in This Guide ............................................................................................... 1-2

2 Starting and Stopping OSM
  About Starting and Stopping OSM ............................................................................................................... 2-1
  Starting the OSM Server ............................................................................................................................. 2-1
  Verifying that OSM Has Started ............................................................................................................... 2-1
  Stopping OSM .......................................................................................................................................... 2-2
  Verifying that OSM Has Stopped ............................................................................................................... 2-2

3 Setting Up OSM Security
  About OSM Security .................................................................................................................................. 3-1
  About Storing Sensitive Data ...................................................................................................................... 3-1
  Adding Users to OSM ............................................................................................................................... 3-2
    Adding Users to Groups in the WebLogic Server Console ..................................................................... 3-2
    Creating Workgroups as Roles in Design Studio ............................................................................... 3-2
    Assigning Users to Workgroups in OSM Administrator .................................................................. 3-4
    Required Groups and Users ............................................................................................................... 3-4
  Using Secure Sockets Layer ...................................................................................................................... 3-5
    Changing Secure Sockets Layer Configuration in OSM .................................................................. 3-6
  Setting Up a Caching Realm ....................................................................................................................... 3-6
  Using the XML Import/Export Application to Administer Users and Workgroups .............................. 3-6
  Using the EncryptPasswords Utility .......................................................................................................... 3-8
    About the EncryptPasswords Utility .................................................................................................. 3-8
    Running the EncryptPasswords Utility ............................................................................................ 3-9
    Removing an Encrypted Password ...................................................................................................... 3-10
  Using WebLogic Authenticators with OSM ............................................................................................ 3-10
4 Securing Credentials Required to Access External Systems

About the Credential Store ................................................................................................. 4-1
How OSM Retrieves Credentials from the Credential Store ........................................... 4-2
Using the Credential Store in OSM ................................................................................. 4-2
  Configuring the Java Security Policy for the OSM Credential Store ............................. 4-3
  Verifying the Java Security Policy Configuration ......................................................... 4-3
  Configuring the Java Security Policy Using credStoreAdmin Command ..................... 4-4
Managing Credentials in the Credential Store ................................................................. 4-5
Developing Cartridges to Use the Credential Store ......................................................... 4-7
  Developing Automation Plug-ins to Use the Credential Store ...................................... 4-8
  Defining Data Providers in OSM Cartridges to Use the Credential Store ....................... 4-8
  Using the Credential Store with Custom Data Providers ............................................... 4-8
  Using the Credential Store with Built-In Data Providers ................................................. 4-9
Upgrading Existing Cartridge Code to Use the Credential Store .................................... 4-9
  Using Built-in SOAP Adapter as a Data Provider Class .................................................. 4-10
OSM Credential Store Commands ..................................................................................... 4-10
  Setting Build Properties for Credential Store Commands .............................................. 4-10
OSM Credential Store Command and API Reference Material ....................................... 4-12
  userAdmin Command ....................................................................................................... 4-13
  credStoreAdmin Command .............................................................................................. 4-17
  CredStore .......................................................................................................................... 4-22
  PasswordCredStore ............................................................................................................ 4-23
  CredStoreException ............................................................................................................ 4-23
  SoapAdapter ....................................................................................................................... 4-27
  ObjectelHPPTAdapter ........................................................................................................ 4-28
  ViewRuleContext ................................................................................................................ 4-30
  AutomationContext ......................................................................................................... 4-32

5 Changing GUI Application Appearance and Functionality

About Configuring the User Experience ............................................................................ 5-1
About Behaviors .................................................................................................................. 5-1

6 Configuring OSM with oms-config.xml

Working with the oms.ear File .......................................................................................... 6-1
  Unpacking the oms.ear File ............................................................................................... 6-1
  Packing the oms.ear File .................................................................................................... 6-2
  Undeploying and Redeploying the oms.ear File ............................................................... 6-3
Working with oms-config.xml ......................................................................................... 6-4
  Using an External oms-config.xml File ............................................................................ 6-4
  Using an oms-config.xml File Stored in the oms.ear File ................................................. 6-4
oms-config.xml Parameters ............................................................................................... 6-5
Sizing Range Partitions for Infrequent Maintenance ................................................................. 8-28
Online vs. Offline Maintenance .................................................................................................. 8-29
Managing Order Data ................................................................................................................ 8-30
Adding Partitions (Online or Offline) ....................................................................................... 8-30
Using Row-Based Order Purge ................................................................................................ 8-31
  Purging a Single Order by Order Id ..................................................................................... 8-33
  Purging Orders that Satisfy Given Criteria ......................................................................... 8-33
  Scheduling Order Purge ...................................................................................................... 8-34
  Stopping and Resuming an Order Purge .............................................................................. 8-34
Using Partition-Based Order Purge .......................................................................................... 8-34
  Differences Between Purging and Dropping Partitions ..................................................... 8-34
  Purging Partitions (Online or Offline) ................................................................................ 8-35
  Purging Entire Partitions That Do Not Contain Open Orders (Online or Offline) .............. 8-36
  Purging Partitions Partially (Offline Only) ......................................................................... 8-37
  Dropping Partitions (Offline Only) .................................................................................... 8-41
  Dropping Empty Partitions (Online or Offline) ................................................................. 8-42
Order Purge Policies .................................................................................................................. 8-42
  Purging Related Orders Independently ............................................................................ 8-42
Auditing and Monitoring Order Purges ................................................................................... 8-44
  Audit Tables ....................................................................................................................... 8-45
Managing Exchange Tables for Partition-Based Order Purge ................................................ 8-45
  About OSM Purge Tables ................................................................................................... 8-45
  About OSM Backup Tables ............................................................................................... 8-47
  Creating Exchange Tables (Online or Offline) ................................................................. 8-47
  Purging Exchange Tables (Online or Offline) .................................................................... 8-48
  Dropping Exchange Tables (Online or Offline) ............................................................... 8-48
Estimating Partition Disk Space (Online or Offline) ................................................................ 8-48
Managing Cartridges ................................................................................................................. 8-49
  Using Fast Undeploy ........................................................................................................ 8-50
  Purging Metadata of Undeployed Cartridges .................................................................. 8-50
Configuration Parameters ...................................................................................................... 8-50
  range_partition_size ........................................................................................................ 8-51
  subpartitions_number ....................................................................................................... 8-51
  default_xchg_capacity .................................................................................................... 8-51
  xchg_retained_orders_thres ............................................................................................ 8-52
  degree_of_parallelism ...................................................................................................... 8-52
  degree_of_parallelism_rebuild_indexes .......................................................................... 8-52
  degree_of_parallelism_rebuild_xchg_indexes ............................................................... 8-52
  purge_job_class .............................................................................................................. 8-52
  parallel_execute_chunk_size ........................................................................................... 8-53
  partition_auto_creation ................................................................................................. 8-53
  purge_policy_rebuild_unusable_indexes ....................................................................... 8-53
  purge_policy_purge_related_orders_independently ....................................................... 8-54
  purge_policy_consolidate_partitions .............................................................................. 8-54
  purge_policy_time_to_close_wait ................................................................................... 8-54
  purge_audit_retention_days ............................................................................................ 8-54
  deferred_segment_creation ............................................................................................... 8-55
About PL/SQL API .................................................................................................................... 8-55
DBMS Output ....................................................................................................................... 8-55
Specifying Purge Criteria .............................................................................................. 8-55
Parallel Execution ............................................................................................................. 8-57
Concurrency Restrictions ................................................................................................. 8-58
PL/SQL API Reference ....................................................................................................... 8-58
Setup and Tuning Procedures .......................................................................................... 8-58
  om_part_maintain.setup_xchg_tables (Online or Offline) .................................................. 8-59
  om_part_maintain.drop_xchg_tables (Online or Offline) .................................................... 8-59
  om_part_maintain.set_dop (Online or Offline) .................................................................. 8-60
  om_part_maintain.set_dop_rebuild_indexes (Online or Offline) ........................................ 8-60
  om_part_maintain.set_dop_rebuild_xchg_index (Online or Offline) ................................. 8-60
  om_part_maintain.purge_entire_partition (Online or Offline) .......................................... 8-67
  om_part_maintain.estimate_ptn_purged_space (Online or Offline) .................................... 8-68
  om_part_maintain.purge_xchg_bck_tables (Online or Offline) ........................................ 8-69
  om_part_maintain.purge_xchg_prg_tables (Online or Offline) ......................................... 8-69
  om_new_purge_pkg.delete_order (Online or Offline) ....................................................... 8-69
  om_new_purge_pkg.purge_orders (Online or Offline) ....................................................... 8-69
  om_new_purge_pkg.schedule_order_purge_job (Online or Offline) .................................... 8-71
  om_new_purge_pkg.select_orders (Online or Offline) ....................................................... 8-72
  om_new_purge_pkg.purge_selected_orders (Online or Offline) .......................................... 8-72
  om_new_purge_pkg.stop_purge (Online or Offline) ........................................................... 8-73
  om_new_purge_pkg.resume_purge (Online or Offline) ...................................................... 8-73
Maintenance Procedures and Functions ........................................................................... 8-60
  om_part_maintain.add_partition (Offline Only) ............................................................. 8-60
  om_part_maintain.add_partitions (Offline Only) ............................................................. 8-60
  om_part_maintain.drop_partitions (Offline Only) ............................................................ 8-61
  om_part_maintain.drop_empty_partitions (Online or Offline) .......................................... 8-62
  om_part_maintain.purge_partitions (Online or Offline) .................................................... 8-63
  om_part_maintain.drop_xchg_tables (Online or Offline) ................................................... 8-69
Advanced Procedures ....................................................................................................... 8-74
  om_part_maintain.backup_selected_ords (Offline) ............................................................ 8-74
  om_part_maintain.restore_ords (Offline) ....................................................................... 8-75
Troubleshooting Functions ............................................................................................... 8-75
  om_part_maintain.get_partitions (Online or Offline) ...................................................... 8-75
  om_part_maintain.is_equi_partitioned (Online or Offline) ............................................... 8-75
Recovery Procedures ....................................................................................................... 8-76
  om_part_maintain.rebuild_unusable_index (Offline or Offline) ....................................... 8-76
  om_part_maintain.purge_orphan_order_data (Online or Offline) ...................................... 8-76
  om_part_maintain.rebuild_index (Online or Offline) ....................................................... 8-77
  om_part_maintain.sys$undo_restore_table (Offline) ....................................................... 8-77
  om_part_maintain.sys$undo_restore_orders(Offline) .................................................. 8-78
Database Reference ......................................................................................................... 8-78
Database Views ............................................................................................................... 8-78
  OM_AUDIT_PURGE_ALL .............................................................................................. 8-78
  OM_AUDIT_PURGE_LATEST ....................................................................................... 8-80
Database Tables ................................................................................................................ 8-80
Troubleshooting and Error Handling ................................. 8-83
   Error Handling for add_partitions ................................ 8-84
   Error Handling for drop_partitions ............................... 8-85
   Error Handling for purge_partitions .............................. 8-85
   Troubleshooting ......................................................... 8-85
   Error Handling ......................................................... 8-87
   Error Handling for rebuild_unusable_indexes ................. 8-88
   Error Handling for setup_xchg_tables .......................... 8-89
Performance Tuning .......................................................... 8-89
   Tuning degree_of_parallelism ...................................... 8-89
   Tuning degree_of_parallelism_rebuild_indexes ............... 8-90
   Tuning degree_of_parallelism_rebuild_xchg_indexes ......... 8-90
   Tuning Parallel Job Execution ...................................... 8-90
      Tuning parallel_execute_chunk_size ......................... 8-90
   Tuning Row-Based Purge ............................................ 8-91
Database Terms .................................................................. 8-92
Managing Optimizer Statistics ............................................ 8-92
   Automated Optimizer Statistics Collection .................... 8-93
   Online vs. Offline ..................................................... 8-93
   Gathering Cartridge Metamodel Statistics ...................... 8-93
   Gathering Order Statistics .......................................... 8-93
      High Volatility Order Tables .................................... 8-93
      Medium Volatility Order Tables ............................... 8-94
      Low Volatility Order Tables ..................................... 8-94
      High, Low, or Medium Volatility Order Tables .......... 8-94
   Manual or Scheduled Maintenance of Volatile Order Tables 8-95
Impact of Statistics on Purging .......................................... 8-95
Gathering Statistics on Other OSM Tables ......................... 8-96
Gathering Fixed Object Statistics ..................................... 8-96
Gathering System Statistics ............................................ 8-97
Populating New Partition Statistics ................................. 8-97
      Using Statistics from Another Partition ....................... 8-97
      Using Statistics from a Statistics Table ....................... 8-98
      Using Statistics from Another System ....................... 8-98
   Configuration Parameter for Managing Optimizer Statistics 8-98
Optimizer Statistics Management Error Handling ............... 8-99
   Automated Copy Partition Statistics Jobs ..................... 8-99
Datapump Jobs ............................................................ 8-99
Optimizer Statistics Management Performance Tuning ........ 8-99
   Parallel Collection .................................................. 8-99
   Concurrent Collection .............................................. 8-100
   Incremental Statistics .............................................. 8-100
   Cursor Invalidations ................................................. 8-100
   Dynamic Statistics (or Dynamic Sampling) .................... 8-101
About OSM Attachment
- WebLogic Persistent Store ............................................................................ 9-2
- WebLogic Server Domain Directory ............................................................. 9-2

Setting OSM to Pause Processing of JMS Messages ........................................... 9-4

Oracle Database Backup Considerations .......................................................... 9-3
- RMAN Considerations ................................................................................... 9-3
- Oracle Flashback Technology Considerations ................................................ 9-3
- Mirroring Technology Considerations ............................................................ 9-3

Backing Up and Restoring the WebLogic Server Configuration ......................... 9-4
- Before You Back Up the WebLogic Server ..................................................... 9-4
- Setting OSM to Pause Processing of JMS Messages ....................................... 9-4
- WebLogic Server Domain Directory ............................................................. 9-5
- WebLogic Persistent Store ............................................................................ 9-5
- OSM Attachment Directory .......................................................................... 9-5
- External Deployments .................................................................................... 9-6
- Remote Managed Servers (Clustered Server Only) ......................................... 9-6

Restoring the WebLogic Server Configuration .................................................. 9-6

Restoring the WebLogic Server Files ................................................................ 9-6
# 10 Exporting and Importing OSM Schema Data

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>About Exporting and Importing OSM Schema Data</td>
<td>10-1</td>
</tr>
<tr>
<td>Exporting and Importing the OSM Model Data Only</td>
<td>10-1</td>
</tr>
<tr>
<td>Exporting OSM Model Data</td>
<td>10-2</td>
</tr>
<tr>
<td>Preparing the Target OSM Schema Before Import</td>
<td>10-4</td>
</tr>
<tr>
<td>Creating the Target OSM Schema</td>
<td>10-4</td>
</tr>
<tr>
<td>Adding Partitions</td>
<td>10-4</td>
</tr>
<tr>
<td>Importing OSM Model Data</td>
<td>10-4</td>
</tr>
<tr>
<td>Exporting and Importing the OSM Model and a Single Order</td>
<td>10-5</td>
</tr>
<tr>
<td>Exporting OSM Order Data</td>
<td>10-5</td>
</tr>
<tr>
<td>Preparing to Export Order Tables for a Single Order</td>
<td>10-5</td>
</tr>
<tr>
<td>Exporting Order Tables That Define an Order Sequence ID</td>
<td>10-6</td>
</tr>
<tr>
<td>Exporting the OSM Model Data</td>
<td>10-7</td>
</tr>
<tr>
<td>Importing the OSM Model and Order Data</td>
<td>10-7</td>
</tr>
<tr>
<td>Exporting and Importing a Range of Orders and the OSM Model</td>
<td>10-8</td>
</tr>
<tr>
<td>Exporting the OSM Order Data Range</td>
<td>10-8</td>
</tr>
<tr>
<td>Preparing to Export Order Tables for a Range of Orders</td>
<td>10-8</td>
</tr>
<tr>
<td>Exporting Order Tables That Define an Order Sequence ID</td>
<td>10-9</td>
</tr>
<tr>
<td>Exporting the OSM Model Data</td>
<td>10-10</td>
</tr>
<tr>
<td>Importing OSM Model and Order Data</td>
<td>10-10</td>
</tr>
<tr>
<td>Exporting and Importing a Range of OSM Orders Only</td>
<td>10-11</td>
</tr>
<tr>
<td>Exporting an Additional Range of Orders</td>
<td>10-11</td>
</tr>
<tr>
<td>Preparing to Export Order Tables for a Range of Orders</td>
<td>10-11</td>
</tr>
<tr>
<td>Exporting a Range of Orders from Order Tables That Define an Order Sequence ID</td>
<td>10-12</td>
</tr>
<tr>
<td>Importing an Additional Range of Orders</td>
<td>10-13</td>
</tr>
<tr>
<td>About Order Export Queries</td>
<td>10-14</td>
</tr>
<tr>
<td>Changing PAR File Parameters</td>
<td>10-15</td>
</tr>
<tr>
<td>About Import Parameters</td>
<td>10-17</td>
</tr>
<tr>
<td>Troubleshooting Export/Import</td>
<td>10-18</td>
</tr>
</tbody>
</table>

# 11 Monitoring and Managing OSM

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>About Monitoring and Managing OSM</td>
<td>11-1</td>
</tr>
<tr>
<td>Accessing the WebLogic Server Administration Console</td>
<td>11-1</td>
</tr>
<tr>
<td>Using the WebLogic Console to Determine the Status of the OSM Application</td>
<td>11-2</td>
</tr>
<tr>
<td>Managing Log Files</td>
<td>11-2</td>
</tr>
<tr>
<td>Viewing Log Files Using WebLogic Server Administration Console</td>
<td>11-2</td>
</tr>
<tr>
<td>Configuring the Log View</td>
<td>11-3</td>
</tr>
<tr>
<td>Log Size and Rotation</td>
<td>11-3</td>
</tr>
<tr>
<td>Managing Error Message Volume and Logging Levels</td>
<td>11-3</td>
</tr>
<tr>
<td>Severity Levels</td>
<td>11-4</td>
</tr>
<tr>
<td>Displaying Current Log Levels</td>
<td>11-4</td>
</tr>
<tr>
<td>Configuring the Severity Levels</td>
<td>11-5</td>
</tr>
<tr>
<td>Monitoring and Analyzing Performance</td>
<td>11-9</td>
</tr>
<tr>
<td>Monitoring Performance Using WebLogic Server Administration Console</td>
<td>11-9</td>
</tr>
<tr>
<td>Monitoring the Managed Server</td>
<td>11-10</td>
</tr>
</tbody>
</table>
Analyzing Garbage Collection
Analyzing Class Loading
Coherence Datagram Testing
Monitoring WebLogic Server
Monitoring the Operating System
Gathering OSM Execution Statistics
Analyzing Heap Dumps
Analyzing Thread Dumps
Monitoring the Database
Managing Database Connections
Updating Database Schema Statistics
Using JMS Queues to Send Messages
Monitoring the Event Queue
Sending Data to External Systems Using Plug-Ins
About OSM and XA Support
Using Work Managers to Prioritize Work
Creating and Configuring Work Managers
Overriding the Internet Explorer Language in the OMS Web Clients
Evaluating System Configuration Compliance
Running the Compliance Tool
Cluster Considerations
Evaluating Compliance Results
About Metrics Data
Viewing Metrics Data
About DMS Metrics Tables
About WebLogic Metrics Tables
About Non-J2EE Metrics Tables
About Aggregated Metrics Tables
About Metric Rules Files
Analyzing Metric Data

12 Configuring Time Zone Settings
Configuring Time Zone Settings

13 Improving Performance
Guidelines for Performance Testing
Measuring Order Throughput
Performance Considerations
Performance Considerations for Solutions
Performance Considerations for Large Orders
Planning for OSM Performance Testing
Guidelines for a Performance Test Environment
Setting Goals for Testing
Selecting a Performance Test Environment
Establishing Warm-up Criteria
Synchronizing Time Across Servers
Maintenance and Prevention

Improving Performance

Analyzing Performance Data

14 Troubleshooting OSM

14-1 Diagnosing and Resolving Problems

14-2 Diagnosing Some Common Problems with OSM

14-3 Cannot Log in or Access Certain Functionality

14-4 System Appears Slow

14-5 Avoiding Timeout Issues in Design Studio

14-6 Error: "Login failed. Please try again."

14-7 No Users or Groups Are Displayed

14-8 Automation Plug-ins Are Not Getting Called

14-9 Too Many Open Files

14-10 Proxy Fails on a Clustered System

14-11 Orders Are Not Being Created on a Clustered System

14-12 Problems Displaying Gantt Charts on Solaris 5.10 Hosted Systems

14-13 OSM Fails to Process Orders Because of Metadata Errors

14-14 Error: "Not Backend Servers Available"

14-15 Quick Fix Button Active During Order Template Conflicts in Design Studio

14-16 Cannot Create New Orders on a New Cartridge Version

14-17 Getting Help with OSM Problems

14-18 Before You Contact Support

14-19 Reporting Problems
15 OSM Log Messages
OSM Catalog Messages ................................................................. 15-1
Automation Catalog Messages ...................................................... 15-21

16 Using the XML Import/Export Application
About the XML Import/Export Application ........................................ 16-1
About Using the XML Import/Export Application .............................. 16-1
About XML Import/Export Batch Scripts and Ant Commands .......... 16-3
   About XML Import/Export Ant Commands and Syntax .................. 16-3
   About XML Import/Export Batch Scripts and Syntax .................... 16-3
Configuring the XML Import/Export Environment Files .................. 16-4
   Configuring the build.properties File for Ant Commands ............. 16-4
   Configuring the config.bat Script for a Batch Scripts ................. 16-5
Configuring the config.xml File XML Import/Export Nodes and Elements 16-6
About Importing and Exporting Metadata ....................................... 16-8
   About Exporting Metadata ...................................................... 16-8
      About the Order of Exported Metadata ................................. 16-9
      About Export Layout Options ............................................. 16-10
      Keeping the ID Integrity in SQL Rules ................................. 16-11
      Configuring and Running an Export ..................................... 16-12
   About Importing Metadata ...................................................... 16-13
      Configuring and Running an Import ..................................... 16-14
      Sample Procedure for Adding a New Workgroup Definition (Role) 16-15
      Sample Procedure for Adding a Task to a Workgroup (Role) ....... 16-17
About Purging MetaData and Data .................................................. 16-19
   Undeploying Cartridges and Purging the Database Schema ............ 16-19
   About Purging Orders ............................................................ 16-21
      Purging Orders with the orderPurge.bat Script on Windows ..... 16-21
      Running Ant with the orderPurge.xml file On UNIX or Linux to Purge Orders 16-24
About Migrating Orders ............................................................... 16-27
   Configuring and Running an Order Migration ............................ 16-27
About Validating the Metadata Model and Data ............................. 16-29
   Configuring and Running an XML Document Validation ............... 16-30
   Validating an XML Document During the Import or Export Process 16-30
About Creating a Graphical Representation of the Metadata Model ... 16-31
   Configuring and Creating a Graphical Representation of a Metadata Model 16-31
   Viewing the Graphical Representation .................................... 16-31

A Tools for Performance Testing, Tuning, and Troubleshooting
WebLogic Server Administration Console ..................................... A-1
Java VisualVM ............................................................................ A-1
JMap ......................................................................................... A-1
JStack ....................................................................................... A-2
OSM Task Web Client .................................................................. A-2
Enterprise Manager .................................................................... A-2
SoapUI ....................................................................................... A-2
<table>
<thead>
<tr>
<th>Component</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Compliance Tool</td>
<td>A-3</td>
</tr>
<tr>
<td>Design Studio</td>
<td>A-3</td>
</tr>
<tr>
<td>Software Load Balancer</td>
<td>A-3</td>
</tr>
<tr>
<td>GCViewer</td>
<td>A-3</td>
</tr>
<tr>
<td>ThreadLogic</td>
<td>A-3</td>
</tr>
<tr>
<td>OSW Black Box</td>
<td>A-4</td>
</tr>
<tr>
<td>Remote Diagnostics Agent</td>
<td>A-5</td>
</tr>
<tr>
<td>Operating System RDA Report</td>
<td>A-5</td>
</tr>
<tr>
<td>WebLogic RDA Report</td>
<td>A-6</td>
</tr>
<tr>
<td>Database RDA Report</td>
<td>A-6</td>
</tr>
<tr>
<td>Oracle RAC Cluster RDA Report</td>
<td>A-6</td>
</tr>
<tr>
<td>OSM RDA Report</td>
<td>A-6</td>
</tr>
</tbody>
</table>
This document describes Oracle Communications Order and Service Management (OSM) system administration tasks.

**Audience**

This document is intended for system administrators, system integrators, Database Administrators (DBA), and other individuals who are responsible for managing OSM and ensuring that the software is operating in the manner required for your business. This guide assumes that users have a working knowledge of Windows XP Professional, Linux, Oracle 11, Oracle WebLogic Server and Java J2EE software.

**Downloading Oracle Communications Documentation**

OSM documentation and additional Oracle documentation (such as database and WebLogic Server documentation) is available from the Oracle Help Center Web site: [http://docs.oracle.com](http://docs.oracle.com)

**Document Revision History**

The following table lists the revision history for this guide.

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>E41518-04</td>
<td>December 2015</td>
<td>Added note stating that you cannot use the default DATA_PUMP_DIR directory when using impdp and expdp with a pluggable databases (PDB).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Added information about the new OSM compliance tool override file that you can use to change the default behavior of the compliance tool. See &quot;Evaluating System Configuration Compliance&quot; for more information.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Added backup and restore recommendation for mirroring technology. See &quot;Mirroring Technology Considerations&quot; for more information.</td>
</tr>
</tbody>
</table>

**Documentation Accessibility**

Access to Oracle Support

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

This chapter provides an overview of Oracle Communications Order and Service Management (OSM) system administration tasks.

OSM System Administration Tasks

As an OSM system administrator, you can perform the following tasks:

- Install and configure OSM.
- Monitor and manage OSM. See "Monitoring and Managing OSM" for more information.
- Read OSM log messages. See "OSM Log Messages" for more information.
- Troubleshoot OSM. See "Troubleshooting OSM" for more information.
- Configure the rule engine. See "Configuring the Rule Engine" for more information.
- Set up OSM security. See "Setting Up OSM Security" for more information.
- Partition the OSM Database schema. See "Managing the OSM Database Schema" for more information.
- Import, export, purge, and migrate data and metadata. See "Using the XML Import/Export Application" for more information.

About OSM System Administration Programs

Use the following programs for OSM system administration:

- Use the Oracle WebLogic Server Console to do the following:
  - Start and stop server components
  - Monitor system components
  - Read log files
  - Manage security
- Use the XML Import/Export application to do the following:
  - Export and import the metadata used for defining the order model
  - Purge orders from the database
  - Validate the OSM data model
  - Create a graphical representation of a data model
- Add or remove users from workgroups
- Purge data from the database
- Migrate data

**Note:** Although these functions can be done using the XML Import/Export application, Oracle Communications Design Studio is the preferred application for executing these functions.

### Directory Placeholders Used in This Guide

Table 1-1 shows the placeholders used in this guide to refer to directories related to the OSM application.

<table>
<thead>
<tr>
<th>Placeholder</th>
<th>Directory Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OSM_home</td>
<td>The directory into which OSM was installed. This directory contains the SDK directory (if the SDK was installed) and various installation-related files.</td>
</tr>
<tr>
<td>MW_home</td>
<td>The location where the Oracle Middleware product was installed. This directory contains the base directory for the WebLogic Server, a utils directory, and other files and directories.</td>
</tr>
<tr>
<td>WLS_home</td>
<td>The base directory for the WebLogic Server core files. It is located in the MW_home directory.</td>
</tr>
<tr>
<td>domain_home</td>
<td>The directory which contains the configuration for the domain into which OSM is installed. The default location is MW_home/user_projects/domains/domain_name but it is frequently set to some other directory at installation.</td>
</tr>
</tbody>
</table>
This chapter describes how to start and stop the Oracle Communications Order and Service Management (OSM) product.

About Starting and Stopping OSM

Because OSM resides on Oracle WebLogic servers, starting or stopping the appropriate WebLogic server also starts or stops OSM.

Note: If your OSM environment is in a WebLogic Server cluster, consult the Oracle WebLogic Server documentation for information about how to start and stop the necessary servers. If all of the servers in the cluster start properly, it means that OSM has started properly.

Starting the OSM Server

To start the OSM Server:

1. Change to the domain_home directory. This is the base directory for the WebLogic domain into which OSM was deployed.
2. Run the following command:
   
   nohup ./startWebLogic.sh &
   
   This starts the WebLogic server in the background and so that it will not stop if you close the terminal window.

Verifying that OSM Has Started

To verify that the OSM application has finished starting, you can use either the following method or the method described in "Using the WebLogic Console to Determine the Status of the OSM Application".

1. From the directory where you started OSM, run the following command:
   
   tail -f nohup.out
   
   Wait until the following text is displayed. It indicates that server startup is complete.
   
   <Server started in RUNNING mode>

2. Use Ctrl + c to halt the tail command and return to the command prompt.
Stopping OSM

To stop OSM:

1. Change to the domain_home/bin directory.
2. Run the following command:

   ./stopWebLogic.sh

The procedure above stops OSM by stopping the Administration server for the WebLogic Server. If the WebLogic Server does not shut down completely, you will not be able to start it again due to a port conflict. If the procedure above has completed, but some WebLogic Server processes are still running for the domain, you can use the kill command to stop them. See "Verifying that OSM Has Stopped" for information about verifying whether OSM and WebLogic have stopped completely.

If you cannot perform a graceful shutdown of OSM, for example, if you need to use the kill command or if there is a power failure on the host machine running the OSM WebLogic server instance, you must manually delete the files in the /tmp/vfs_cache_user folder (where user is the Linux or UNIX user account that you use to run the OSM WebLogic server) before you can restart OSM.

Verifying that OSM Has Stopped

To verify that OSM has stopped, do any of the following:

- Look at the text in the nohup.out file located in the directory where OSM was started (usually domain_home). One of the last messages displayed when the server shuts down is:

   <Order Management Webservice destroyed.>

- Try connecting to the WebLogic console. If you cannot, WebLogic is probably not running.

- Look at the process list for the user who started the server. If WebLogic is running, there will probably be at least one process with startWebLogic.sh in its description.

- Look in the user’s process list for a java process that was started out of the Java directory for WebLogic. Process descriptions vary from platform to platform, so look at the process list when you know OSM is running to see what the entries look like on your platform. You can later use this information to confirm that the WebLogic server has shut down completely.
This chapter describes how to set up security on your Oracle Communications Order and Service Management (OSM) system.

About OSM Security

You use the Oracle WebLogic Server Console to manage OSM security. When you manage OSM security, you can perform the following tasks:

- Add users to groups. See "Adding Users to OSM".
- Set up Secure Sockets Layer (SSL). See "Using Secure Sockets Layer".
- Integrate additional security measures; for example, LDAP servers, Microsoft domain servers, or Linux security.

**Note:** If you use an external security implementation such as LDAP, you should also use a caching realm to improve performance. See "Setting Up a Caching Realm" for more information.

- Manage workgroups. See *OSM Administrator Application User’s Guide* for more information.

For more information about WebLogic security realms, refer to the WebLogic Server Console documentation.

**Note:** OSM supports LDAP Version 2.

About Storing Sensitive Data

As a fulfillment system, OSM does not need a fixed data model, and so is not required or typically used to store sensitive data other than that used for OSM user authentication.

You can secure OSM user credentials as described in this chapter and in "Securing Credentials Required to Access External Systems", but if your implementation requires OSM to store or log other sensitive data that appears on orders, Oracle recommends that you encrypt the data outside of OSM. Because the encryption happens outside of OSM, you are responsible for developing and maintaining the encryption method.
Adding Users to OSM

To add a user to OSM:

■ Add users to groups in the WebLogic Server Administration Console.
■ Create workgroups as roles in Oracle Communications Design Studio.
■ Assign users to workgroups in the OSM Administrator.

Adding Users to Groups in the WebLogic Server Console

All security for OSM users and groups is managed through the WebLogic Server Console. See the Oracle WebLogic Server documentation for more information about creating and deleting users and groups.

To add users to groups:

1. Log in to the WebLogic Server Administration Console.
   You must be a WebLogic administrator.

2. In the Domain Structure tree, click Security Realms.
   The Summary of Security Realms page is displayed.

3. Click myrealm.
   The settings for the security realm are displayed.

4. Click the Users and Groups tab.
   A list of users that have been configured is displayed.

5. Click on a user.
   The user’s description, password, and group membership is displayed. Users are assigned to one or more parent groups that have different levels of access to WebLogic resources, depending on their roles and the tasks they can perform. Groups in the WebLogic security realm represent the roles.

6. On the page that displays the settings for the selected user, click the Groups tab.

7. Select a group or groups from the Available list, click the right arrow to move the selected group to the Chosen list. See "Required Groups and Users" for more information.

8. Click Save.

Creating Workgroups as Roles in Design Studio

In Design Studio, you create roles and assign permissions to give users in that role access to related functions in the Task and Order Management Web clients and the Web Service and XML APIs.

---

**Note:** You assign individual users to roles using the Administrator application. See OSM Administrator Application User’s Guide for more information.

---

Table 3–1 describes the client functions to which you provide access.
### Table 3–1  Client Permissions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
<th>Applies To</th>
</tr>
</thead>
</table>
| Create Versioned Orders   | Enables users to create orders for different versions of cartridges. If not granted this permission, users can only create orders for the default version of the cartridge.  
  This permission relates to:  
  - In the Task Web client: creating a new order  
  - In the Web Services API: using the CreateOrderBySpecification calls  
  - In the XML API: using the CreateOrder XML API call | Task Web client  
  Web Services API  
  XML API |
| Exception Processing      | Enables users to alter the flow of a process by applying exception statuses at any time throughout the process.  
  This permission relates to:  
  - In the Task Web client: raising exceptions on a task  
  - In the XML API: using the SetException call | Task Web client  
  XML API |
| Online Reports            | Enables users to view summarized reports on all orders and tasks on the system.  
  This permission relates to:  
  - In the Task Web client: using the reporting feature | Task Web client |
| Order Priority Modification| Enables users to modify the priority of an order or of a task in an order.  
  This permission relates to:  
  - In the Order Management Web client: setting the order priority  
  - In the Task Web client: setting the task priority  
  - In the Web Services API: changing the order priority using the UpdateOrder call  
  - In the XML API: changing the order or task priority using the UpdateOrder call | Order Management  
  Web client  
  Task Web client  
  Web Services API  
  XML API |
| Reference Number Modification | Enables users to modify the reference number of an order.  
  This permission relates to:  
  - In the Order Management Web client and Task Web client: Modifying the reference number of an order  
  - In the Web Services API and XML API: changing the reference number using the UpdateOrder call | Order Management  
  Web client  
  Task Web client  
  Web Services API  
  XML API |
In addition to granting Web client permissions, you can also grant permissions at the order level (by associating a role to an order type) and the task level. See the discussion about creating new roles in the Design Studio Modeling OSM Processes Help for more information. After you create a role, you must assign permissions to the role entities. See the description of the Role Editor Role tab in the Design Studio Modeling OSM Processes Help for more information about permissions for role entities.

Assigning Users to Workgroups in OSM Administrator
See the discussion about assigning users to a workgroup in OSM Administrator Application User’s Guide for more information.

Required Groups and Users
When you install OSM, required groups and users are created automatically. To use another security implementation such as LDAP, you must create the appropriate groups and manually add users to them.

Table 3–2 lists the groups and the members of each group.

### Table 3–2  Groups and Access Members

<table>
<thead>
<tr>
<th>Group Name</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>OMS_client</td>
<td>All users of the OSM Web Client, OSM XML API, or OSM Web Services API must belong to this group.</td>
</tr>
<tr>
<td>OMS_ws_api</td>
<td>All users of the OSM Web Services API must belong to this group.</td>
</tr>
<tr>
<td>OMS_xml_api</td>
<td>All users of the OSM XML API must belong to this group.</td>
</tr>
</tbody>
</table>
Using Secure Sockets Layer

OSM supports three levels of security for interactive users:

- **No use of SSL.** If no SSL support is chosen, the HTTP connection is made on the WebLogic server non-secure port.

- **SSL for all communications.** Complete SSL support is provided by the WebLogic server. An HTTPS connection is made to the SSL port for that server and all communication takes place using SSL on that port.

- **SSL during login only.** Provides a connection to the secure HTTP port. You can use SSL during authentication so user names and passwords are encrypted. After authentication is complete, a non-secure HTTP connection is used. This avoids the performance overhead of using SSL for all communication but gives a level of security for user and password protection. If you choose this option, extra configuration is required.

---

### Table 3–2 (Cont.) Groups and Access Members

<table>
<thead>
<tr>
<th>Group Name</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>OMS_ws_diag</td>
<td>All users of the OSM Web service diagnostics API must belong to this group</td>
</tr>
<tr>
<td>OSM_automation</td>
<td>All users configured to run OSM automation plug-ins must belong to this group.</td>
</tr>
<tr>
<td>OMS_workgroup_manager</td>
<td>Gives members the ability to create and modify workgroups using the OSM Administrator. They cannot add users to workgroups.</td>
</tr>
<tr>
<td>OMS_user_assigner</td>
<td>Gives members the ability to add users to workgroups from the OSM Administrator.</td>
</tr>
<tr>
<td>OMS_log_manager</td>
<td>Gives access to the log4jAdmin web page for reading log messages.</td>
</tr>
<tr>
<td>OMS_designer</td>
<td>You can do all process, order modeling, and system maintenance. You cannot create or modify workgroups, or assign users to workgroups.</td>
</tr>
<tr>
<td>OMS_cache_manager</td>
<td>Monitoring and control of order cache.</td>
</tr>
<tr>
<td>OSM_USER_Manager</td>
<td>All users of the OSM Administrator Application must belong to this group to manage OSM users and workgroups.</td>
</tr>
<tr>
<td>Administrators</td>
<td>Gives access to:</td>
</tr>
<tr>
<td></td>
<td>- WebLogic Server administrative user; that is, the alias that starts the WebLogic server</td>
</tr>
<tr>
<td></td>
<td>- WebLogic Server instance</td>
</tr>
<tr>
<td></td>
<td>- WebLogic Server Clusters</td>
</tr>
<tr>
<td></td>
<td>- Security parameters, including managing users, groups, and roles</td>
</tr>
<tr>
<td></td>
<td>- Deploying your applications</td>
</tr>
<tr>
<td></td>
<td>- Monitor server and application performance</td>
</tr>
<tr>
<td></td>
<td>- View server and domain log files</td>
</tr>
<tr>
<td></td>
<td>- View application deployment descriptors</td>
</tr>
<tr>
<td></td>
<td>- Edit selected run-time application deployment descriptor elements</td>
</tr>
</tbody>
</table>
Changing Secure Sockets Layer Configuration in OSM

You configure SSL for OSM using the XML parameter file \texttt{web.xml} which is stored in the \texttt{oms.war} file inside the \texttt{oms.ear} file. To edit the \texttt{web.xml} file, you must unpack the \texttt{oms.ear} file, edit the \texttt{web.xml} file, and then repack the \texttt{oms.ear} file and re-deploy it to the OSM server.

To change SSL configuration on the OSM server:

1. Locate and extract the \texttt{web.xml} file from the \texttt{oms.ear/oms.war} file.
2. Add or modify the parameter \texttt{secure_login} in the \texttt{web.xml} file.
3. Edit the parameter value as follows:
   - For no SSL support, enter 0.
   - For SSL support, enter 1.
4. Save the file.

Setting Up a Caching Realm

If you use an external security implementation such as LDAP, you should also use a caching realm to improve performance. A caching realm holds the results of security checks in memory so that subsequent checks are not required to communicate directly with an external security server. The default settings for caching realms are appropriate for small numbers of users in the external realm; however, they do not help if your external security implementation has large numbers of users.

To set up a caching realm:

1. Log in to the WebLogic Administration Console.
2. In the Domain Structure tree, select \texttt{Security Realms}.
   - The Summary of Security Realms page is displayed.
3. Select the realm from the table.
   - The Settings for myrealm window is displayed.
   - From this window, you can change the settings for your realm.

Using the XML Import/Export Application to Administer Users and Workgroups

The userAdmin command lets you add users to WebLogic groups and OSM workgroups using an XML document. The XML document contains the user information you want to add or configure based on the \texttt{UserAdmin.xsd} schema, which can be found in the \texttt{OSM_home/SDK/XMLImportExport/models} directory.

Administering users in this way allows you to manage users in volume instead of assigning them individually, and it permits the integration of OSM users into a larger, enterprise system administration application.

You must encrypt the passwords in the source XML document prior to running the userAdmin command. Otherwise, when you run userAdmin, you will trigger an error indicating that the EncryptPassword utility should be run. See “Using the EncryptPasswords Utility” for more information.

To run the userAdmin command, you should first modify the classpath in the \texttt{config.bat} file (if you are using the command-line script) or the
xml.project.class.path in the build.xml file (if you are using Ant), to ensure the following:

- Contains only one reference to the weblogic.jar file associated with the WebLogic installation to which the command is connecting. If there is no such reference, add it. If there is any reference pointing to the one under XMLImportExport/lib, change it.

- Contains only one reference to the ojdbc<X>.jar file associated with the WebLogic instance to which the command is connecting. If there is one pointing to the one under XMLImportExport/lib, change it.

- Contains a reference to XMLImportExport/classes/schemaTypes.jar. If there is no such reference, add it.

- Remove any references to XMLImportExport/lib/nls_charset12.jar and XMLImportExport/lib/classes12.zip.

To set up your environment and run the command:

1. Define an XML file with the user information to add, similar to the sample shown below. See UserAdmin.xsd for more details on the XML schema.

```
<userConfig xmlns="http://www.metasolv.com/Provisioning/UserConfig"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.metasolv.com/Provisioning/UserConfig
/u01/app/OSM/SDK/XMLImportExport/models/UserAdmin.xsd">
  <user name="demo">
    <description>A test user</description>
  </user>
  <clientGroup>
    <user>demo</user>
  </clientGroup>
  <designerGroup>
    <user>demo</user>
  </designerGroup>
  <logManagerGroup>
    <user>demo</user>
  </logManagerGroup>
  <userAssignerGroup>
    <user>demo</user>
  </userAssignerGroup>
  <workgroupManagerGroup>
    <user>demo</user>
  </workgroupManagerGroup>
  <workgroup name="demo">
    <user>demo</user>
  </workgroup>
  <workgroup name="everyone">
    <user>demo</user>
  </workgroup>
</userConfig>
```

2. Encrypt the passwords in your source XML file. See "Using the EncryptPasswords Utility".

3. If required, edit the config.xml file to modify your connection information to WebLogic. You must have the authority to administer users in WebLogic.

```
<j2eeAdminConnection>
  <user>weblogic</user>
  <password>password</password>
</j2eeAdminConnection>
```
Using the EncryptPasswords Utility

You use the EncryptPasswords utility to encrypt the user name and password credentials of XML Import/Export application users. For information on the EncryptPasswords utility, see the following:

- About the EncryptPasswords Utility
- Running the EncryptPasswords Utility

About the EncryptPasswords Utility

The EncryptPasswords utility (located in the OSM_home/SDK/XMLImportExport directory) is a password management utility that secures the credentials that the XML Import/Export application uses to access the OSM database, the XML API interface, and the WebLogic domain Administration Server. The EncryptPasswords utility encrypts the credentials, preventing their accidental exposure.

Running the EncryptPasswords utility script stores the user names and passwords of all XML Import/Export application users in encrypted format in the configuration file which the XML Import/Export application uses to provide these credentials (for example, OSM_home/SDK/XMLImportExport/config/config.xml). When the XML application runs, it reads the encrypted credentials from the configuration file and uses them to access the OSM database, the XML API interface, and the WebLogic domain Administration Server.

About the EncryptPasswords Utility

The EncryptPasswords utility is a command line tool that requires the following environment variables to be set:

- OSM_HOME: The directory where the OSM application is installed.
- SDK_DIR: The directory where the SDK is installed.
- XMLImportExport_DIR: The directory where the XML Import/Export application is installed.
- JAVA_HOME: The directory where the Java runtime environment is installed.
- PATH: The directory where the EncryptPasswords utility is installed.

Note: The <password> element in the config.xml file is optional. If you opt not to specify the password in the <password> element, you will be prompted to supply it when you run the script.

4. Run one of the following commands:

   - **UNIX and Linux:**
     ant userAdmin
   - **Windows:**
     userAdmin XMLModel XMLConfig

   for example:
   userAdmin data\users.xml config\config.xml

   where:
   XMLModel is the name of the XML model document containing the user information, and
   XMLConfig is the name of the configuration file.

5. In the OSM Administrator, use the refresh buttons to refresh the Users and Workgroups lists, or use ant Refresh in the CDK to refresh the metadata.

6. After the users have been created, you can assign functions and tasks to them using Design Studio.

Using the EncryptPasswords Utility

You use the EncryptPasswords utility to encrypt the user name and password credentials of XML Import/Export application users. For information on the EncryptPasswords utility, see the following:

- About the EncryptPasswords Utility
- Running the EncryptPasswords Utility

About the EncryptPasswords Utility

The EncryptPasswords utility (located in the OSM_home/SDK/XMLImportExport directory) is a password management utility that secures the credentials that the XML Import/Export application uses to access the OSM database, the XML API interface, and the WebLogic domain Administration Server. The EncryptPasswords utility encrypts the credentials, preventing their accidental exposure.

Running the EncryptPasswords utility script stores the user names and passwords of all XML Import/Export application users in encrypted format in the configuration file which the XML Import/Export application uses to provide these credentials (for example, OSM_home/SDK/XMLImportExport/config/config.xml). When the XML application runs, it reads the encrypted credentials from the configuration file and uses them to access the OSM database, the XML API interface, and the WebLogic domain Administration Server. The EncryptPasswords utility encrypts the credentials, preventing their accidental exposure.
Using the EncryptPasswords Utility

Import/Export application runs, it decrypts the passwords as part loading its configuration file.

The EncryptPasswords utility can be run only by a user who has write access to the XML files in which the credentials are stored.

When you install the XML Import/Export application, you can optionally provide passwords for the OSM database, the XML API interface, and the WebLogic domain Administration Server. To set and reset those passwords, you run the EncryptPasswords utility. See "Running the EncryptPasswords Utility" for information on running the utility.

The EncryptPasswords utility prompts you to enter the user name and password credentials of each XML Import/Export application user that requires access to the OSM database, the XML API interface, and the WebLogic domain Administration Server. The utility then encrypts the credentials and stores their encrypted form in the configuration file specified when the utility is run.

Ant build files for the EncryptPasswords utility are located in the following directories:

- OSM_home/SDK/CartridgeManagement/production
- OSM_home/SDK/CartridgeManagement/development

The Ant build files have targets corresponding to each of the batch files in the OSM_home/SDK/XMLImportExport directory that include the EncryptPasswords functionality.

Running the EncryptPasswords Utility

Run the EncryptPasswords utility script:

- As part of the initial setup of the XML Import/Export application
- Each time the user name or password credentials of an XML Import/Export application user changes

See "About the EncryptPasswords Utility" for information on how the EncryptPasswords utility works.

Note: To run the EncryptPasswords utility, you have write access to the XML files in which the XML Import/Export application user credentials are stored.

To run the EncryptPasswords utility, do one of the following:

- **UNIX and Linux:**
  Source the following command:
  for example:
  . EncryptPasswords.sh config/config.xml -dbUser /u01/admin/Oracle/Middleware

- **Windows:**
  Run the following command:
  EncryptPasswords XMLConfig [-dbUser] [-xmlapiUser] [-wlsUser]
for example:

```
EncryptPasswords config\config.xml -dbUser -xmlapiUser
```

where XMLConfig indicates the XML Import/Export application configuration XML file you are using (for example, `config.xml`) and the following optional parameters indicate which passwords should be encrypted:

- **-dbUser**: the OSM database
- **-xmlapiUser**: the XML API interface
- **-wlsUser**: the WebLogic domain Administration Server

When you set a user’s credentials, you specify only the systems that they use for the XML Import/Export application operations they perform. For example, if the user only imports or exports cartridges, you only need to specify the **-dbUser** flag.

### Removing an Encrypted Password

To remove a user name and password for a user that no longer requires credentials, open the XML file where the credentials are stored and remove them manually. If you do not remove them manually, the user name and password combination continues to exist in the XML file.

### Using WebLogic Authenticators with OSM

OSM supports using either the WebLogic Server’s embedded LDAP directory or another authenticator that is external to, and integrated with, the WebLogic server. The latter is referred to in this section as an external authenticator. This section provides information about how these directories work with OSM.

**Note:** If multiple authentication providers are configured in WebLogic, all the authentication providers (even if they are configured as optional in WebLogic) should be active. If any of the authentication provider is not active, an exception will be raised and the users will not be able to log in to OSM.

#### User-level Authenticator Support

OSM fully supports external authenticators and embedded LDAP directories at the user level. A user that exists in either the WebLogic server’s embedded LDAP directory or in an external authenticator receives the privileges of any group to which it is assigned in WebLogic. Also, users can be assigned to workgroups in the OSM Administrator Application and will receive the appropriate permissions for the workgroup.

#### Group-level Authenticator Support

OSM also supports external authenticators and embedded LDAP directories at the group level. A group that exists in either the WebLogic server’s embedded LDAP directory or in an external authenticator receives the privileges of any OSM group to which it is assigned in WebLogic. Also, groups can be assigned to workgroups in the OSM Administrator Application and will receive the appropriate permissions for the workgroup.

A child LDAP group will inherit the following from its parent LDAP group:
Permissions from OSM roles in Design Studio
Permissions from groups assigned in the WebLogic Administrator console
Tasks
Filters
Flexible headers

It is not possible to restrict a child group from inheriting from the parent group.

Note: If a user is assigned (directly or indirectly) to multiple groups which have different query tasks for the same order, it is not predictable which query task view the user will see when querying the order.

Authenticator User and Group Assignment Considerations

There are some considerations and best practices to take into account when assigning users and groups to OSM workgroups/roles.

- The first step in assigning permissions for external authenticator and embedded LDAP users and groups is to assign them to either the OMS_client or OSM_automation groups in WebLogic.

- Only users and groups which have been assigned directly to the OMS_client or OSM_automation groups in WebLogic will be available to assign to workgroups in the OSM Administration Application. For example, if UserA1 is a member of GroupA, and GroupA has been assigned to the OMS_client group in WebLogic, only GroupA will be displayed in the OSM Administrator Application. However, when GroupA is assigned to a workgroup, UserA1 (and all of the users in GroupA) will inherit the permissions of the workgroup.

- Both users and groups will be displayed the same in the OSM Administrator Application. There is no indication which elements are users and which are groups.

- If new users are added to the authenticator, they will not be able to access OSM functionality until the OSM Metadata is refreshed. OSM Metadata can be refreshed by:
  - Clicking the refresh button from OSM Administrator tool
  - Executing the OSM ant target com.mslv.oms.deploytool.anttasks.OMSRefresh
  - Starting a Weblogic Managed Server that is running OSM
  - Touching an externally configured oms-config.xml file

- Oracle recommends that you assign automation users to the OSM_automation WebLogic group directly, rather than as members of a group. If you decide to assign automation users using a group instead of individually, you must ensure that you do not remove from the WebLogic group any users that are specified in the 'Run As' property of an automation plug-in.
Securing Credentials Required to Access External Systems

Oracle Communications Order and Service Management (OSM) applications, such as OSM Web clients and OSM cartridge applications, often are required to provide credential information to gain access and log in to external systems. The credential information must be secure and cannot be hard coded in OSM code. This chapter describes how to secure credentials for accessing external systems by using a credential store, through the Fusion Middleware Credential Store Framework (CSF).

About the Credential Store

A credential store is a central repository you can use to store credentials such as user names and passwords. OSM can use a credential store for the credentials needed to log in to and send requests to external systems (meaning any resources or systems that are out of the scope of your OSM cartridge-owned code and resources). These credentials can be used by OSM and OSM applications.

The type of credential store that OSM uses is offered through the Oracle Platform Security Services (PSS) and is part of the CSF security service available to WebLogic Server. CSF offers both file-based and LDAP-based credential stores. OSM uses the file-based credential store by default. Oracle recommends using LDAP-based stores in a production environment.

The CSF is a set of APIs that applications can use to create, read, update, and manage credentials securely. OSM includes wrapper APIs to the CSF APIs. Use the OSM credential store APIs when developing your OSM cartridges so your OSM applications can access and read credentials from the credential store in a secure form. For example, use the OSM credential store APIs when you define data provider classes in your cartridges which must access Web Service and database systems with credentials. See "Developing Cartridges to Use the Credential Store" and "OSM Credential Store Command and API Reference Material" for more information on using the credential store in your cartridge development.

Oracle recommends you to use the credential store as a repository for credentials and use the OSM credential store APIs in OSM-related code to access the repository in a secure form. Oracle strongly recommends you do not hard code user names and passwords in cartridge code and that you update any current cartridges that have hard coded credentials to use the OSM credential store APIs. See "Developing Cartridges to Use the Credential Store" for more information on security options and recommendations.

See the Oracle Fusion Middleware Application Security Guide Oracle Technology Network Web site at:
How OSM Retrieves Credentials from the Credential Store

The credential store in the WebLogic domain for OSM applications contains credentials which are stored using a credential store map and key names. OSM applications, such as OSM Web clients and OSM cartridge applications, retrieve credentials from the credential store based on the credential store map and key names. Automation plug-ins are used to call OSM credential store APIs to retrieve the credentials. OSM applications use the credentials to gain access to external systems.

OSM credential store APIs are used inside automation plug-ins to retrieve credentials and to gain access to external systems. OSM cartridge code can call the credStoreAdmin command (an XML Import/Export application command) to create and configure the required entries, such as map name, user name, and password, for OSM applications in the credential store. See "credStoreAdmin Command" for information on this utility.

OSM plug-in users in a cartridge application, such as automation plug-ins or external instance adapters access the OSM credential store map and read the credentials.

The following steps summarize how an OSM automation plug-in retrieves credentials from the credential store for an automation task that requires the credentials to access an external system:

1. The automation plug-in script uses the getCredential or getOsmCredentialPassword method of the AutomationContext API.
2. WebLogic Server checks the JPS (Java Platform Security) policy and confirms the automation plug-in has access permissions from the OSM domain to the credential store.
3. The automation plug-in user accesses the correct credential store map and reads the credentials required to access the external system.
   If the credentials are not in the store, the API fails with an exception and the automation task fails.
4. If the credentials are in the store, the user name and password variables in the plug-in script will be set with values retrieved from the credential store.
5. The message is sent to the external system with the credential information.
6. The automation task completes.

Using the Credential Store in OSM

OSM uses the credential store offered through CSF (see "About the Credential Store"). You can use the credential store to store the user name and password information that OSM and its applications require to log in to and send a request to external systems.

See "About the Credential Store" for a description of the credential store OSM uses by default.

See "How OSM Retrieves Credentials from the Credential Store" for information on how OSM retrieves the credentials from the credential store.

See "Managing Credentials in the Credential Store" for information on adding credentials to the credential store and managing credentials.
See "Developing Cartridges to Use the Credential Store" for information on developing OSM cartridges to use the credential store.

See "Upgrading Existing Cartridge Code to Use the Credential Store" to update your existing cartridge code to use the credential store.

**Configuring the Java Security Policy for the OSM Credential Store**

For OSM and its plug-in applications to read data from the credential store, the WebLogic server must be configured to grant them Java permissions to access the credential store map. In PSS, credential store access permission is configured through the JPS policy. The default JPS policy instance is in the system-jazn-data.xml file in ${domain.home}/config/fmwconfig.

When you configure the JPS policy, you specify the code source directory of the OSM application (oms); this grants credential store access permission to all JAR files in the code source directory.

The default credential store map name for OSM applications is osm. For OSM plug-in users to access the default map, configure the JPS policy using map name (oms). You can create your own credential store map in the credential store. For OSM plug-in users in cartridge applications to access your credential store map and read the credentials, configure the JPS policy using your own map name.

In a clustered environment, use the administration server URL and not the managed server when you configure the credential store and the JPS policy.

To configure the JPS policy, you can use either of the following methods:

- The credStoreAdmin command. See "Configuring the Java Security Policy Using credStoreAdmin Command" to configure the JPS policy for the OSM credential store map using this command. See "credStoreAdmin Command" for a description of this command.

- The Fusion Middleware WebLogic Scripting Tool grantPermission command. See the Oracle Fusion Middleware Security Guide for information on managing policies with the WebLogic Scripting Tool commands.

- The Fusion Middleware Control GUI. Credential store and JPS policy MBeans can be configured from this GUI. See the Oracle Fusion Middleware Application Security Guide at:
  

  for information.

**Verifying the Java Security Policy Configuration**

To verify the JPS policy is configured for your credential store:

Open the ${domain.home}/config/fmwconfig/system-jazn-data.xml file and verify that it contains the following elements in bold text:

```xml
<grant>
  <grantee>
  <codesource>
```

**Note:** This JPS policy entry shows credential store access from the OSM application (oms) to the default OSM credential store map osm in the credential store.
Using the Credential Store in OSM

Do not update the JPS policy configuration manually by editing system-jazn-data.xml directly. Use the methods described in "Configuring the Java Security Policy for the OSM Credential Store" to configure the JPS policy.

**Configuring the Java Security Policy Using credStoreAdmin Command**

To configure the JPS policy for the OSM credential store map using the credStoreAdmin command batch script (recommended method):

---

**Note:** This procedure assumes you have:

- Installed OSM.
- Installed OSM SDK.

---

1. Create and edit the configuration file for credStoreAdmin command (config.xml). See "Creating the Configuration File for the credStoreAdmin Command".
2. Create and edit the XML data file for credStoreAdmin command (credential.xml). See "Creating the XML Data File for the credStoreAdmin Command". The JPS policy and credentials are configured based on the data in the credential.xml file.
3. From the XML Import/Export application directory (OSM_home/SDK/XMLImportExport), set the following required environment variables for the credStoreAdmin command in config.bat (Windows) or config.sh (Unix):
   - JAVA_HOME
   - APP_ROOT: Point to the XMLImportExport directory in the OSM SDK installation.
   - MIDDLEWARE_HOME: Point to the Fusion Middleware installation directory.
4. Set the required properties in the build.properties file. See "Setting Build Properties for Credential Store Commands".
5. Run the credStoreAdmin batch script as follows:
   ```bash
   credStoreAdmin credential.xml config/config.xml
   ```
6. When prompted, enter the password for WebLogic user.
7. Verify the JPS policy is configured for your credential store as described in "Verifying the Java Security Policy Configuration".
You can also configure the JPS policy for the OSM credential store map using the `credStoreAdmin` command Ant script.

To configure the JPS policy for the OSM credential store map using the `credStoreAdmin` command Ant script:

---

**Note:** This procedure assumes you have:
- Installed OSM.
- Installed OSM SDK.

1. Create and edit the configuration file for the `credStoreAdmin` command (`config.xml`). See "Creating the Configuration File for the `credStoreAdmin` Command".

2. Create and edit the XML data file for the `credStoreAdmin` command (`credential.xml`). See "Creating the XML Data File for the `credStoreAdmin` Command".

3. From the XML Import/Export application directory (`OSM_home/SDK/XMLImportExport`), set required properties in the `build.properties` file as described in "Setting Build Properties for Credential Store Commands".

4. Run the `credStoreAdmin` Ant script and provide the WebLogic administrator user password in the command line as follows:

   ```
   > ant -Dwls_admin_password=password credStoreAdmin
   ```

5. Verify the JPS policy is configured for your credential store as described in "Verifying the Java Security Policy Configuration".


You can call the `credStoreAdmin` target in another Ant script. For an example of calling the `credStoreAdmin` Ant script in your own Ant script, see "Calling `credStoreAdmin` Target in Another Ant Script".

---

**Managing Credentials in the Credential Store**

You must provision the credential store with credentials before OSM applications can use them. Provision the credential store by adding the user names and passwords required for OSM and external systems. You must also update passwords in the credential store every time you change user passwords through the WebLogic Server console.

Store all OSM users in the default OSM map, `osm`, and use a default value in key name `osmUser_+username`.

To manage credentials, such as adding, updating, or deleting credentials for OSM and external systems in the credential store, do either of the following:

- Use Oracle WebLogic Scripting Tool commands provided by WebLogic Server. See Oracle Fusion Middleware Security Guide at:


  for information on managing credentials in the credential store by using WebLogic Scripting Tool commands.
Managing Credentials in the Credential Store

- Use the credStoreAdmin command in the XML Import/Export application. See "credStoreAdmin Command" for information on this command.

To manage credentials in the credential store using credStoreAdmin command:

1. From the XML Import/Export application directory (OSM_home/SDK/XMLImportExport), set required properties in the build.properties file as described in "Setting Build Properties for Credential Store Commands".

2. From the XML Import/Export application directory (OSM_home/SDK/XMLImportExport), run the script credStoreAdmin:

   ```bash
   >credStoreAdmin credential.xml config/config.xml
   ``

   where credential.xml is the XML data file that includes the map, key, user name, and password values. See "Creating the XML Data File for the credStoreAdmin Command".

   The following is an example of the XML data file:

   ```xml
   <?xml version="1.0" encoding="UTF-8"?>
   <ns2:CredentialConfig
       xmlns:ns2="http://www.metasolv.com/Provisioning/CredentialConfig"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:schemaLocation=".../XMLImportExport/models/CredStoreAdmin.xsd">
       <ns2:credential operation="create">
           <ns2:mapname>osm</ns2:mapname>
           <ns2:keyname>osmUser_osm</ns2:keyname>
           <ns2:user>osm</ns2:user>
           <ns2:password/>
       </ns2:credential>
       <ns2:credential operation="create">
           <ns2:mapname>osmlf</ns2:mapname>
           <ns2:keyname>osmUser_osmlf</ns2:keyname>
           <ns2:user>osmlf</ns2:user>
           <ns2:password/>
       </ns2:credential>
   </ns2:CredentialConfig>
   ```

   where config.xml is the configuration file that contains the data for WebLogic Server. See "Creating the Configuration File for the credStoreAdmin Command".

   The following is an example of the config.xml file:

   ```xml
   <j2eeAdminConnection>
     <user>weblogic</user>
     <password/>
     <hostname>localhost</hostname>
     <port>7001</port>
   </j2eeAdminConnection>
   ```

   To manage credentials in the credential store using the userAdmin command:

   1. From the XML Import/Export application directory (OSM_home/SDK/XMLImportExport), set required properties in the build.properties file as described in "Setting Build Properties for Credential Store Commands".
2. Create and edit the configuration file for the userAdmin command (config.xml). See "Creating the Configuration File for the userAdmin Command".

3. Create and edit the XML data file for the userAdmin command (user.xml). See "Creating the XML Data File for the userAdmin Command".

   It is recommended that you do not enter password values in user.xml. By leaving password values empty, you can enter passwords interactively at run time.

4. From the XML Import/Export application directory (OSM_home/SDK/XMLImportExport), do either of the following:
   ■ (Recommended method) Run the batch script userAdmin and input password information when prompted:
     ```shell
     >userAdmin user.xml config/config.xml
     ```
   ■ Run the Ant script userAdmin and provide password information in the command line:
     ```shell
     >ant -Dwls.admin.password=password -Dxmlie.root.dbPassword=password userAdmin
     ```

If the Ant target userAdmin is called by an Ant script that is running in your Oracle Communications Design Studio workspace, interactive mode is not supported and passwords must be entered in the XML data file. In this case, delete the XML data file immediately after use. For an example of how to call the userAdmin command Ant script in another Ant script, see "Calling UserAdmin Target in Another Ant Script".

### Developing Cartridges to Use the Credential Store

When your OSM cartridges require data from external systems, trigger actions at external systems, or provide data to external systems, credential information may be required by the external system. The external system can be any resource or system that is out of the scope of your OSM cartridge-owned codes and resources.

When you develop OSM cartridges, Oracle recommends you use the credential store to allow plug-in code to access credential information in a secured way. You can use the OSM credential store APIs for code that requires credential retrieval.

OSM uses the credential store offered through WebLogic Server; however, you are not required to use this credential store to secure credentials. You can use other methods of securing credentials. Oracle strongly recommends you do not hard code user credential information in OSM code such as in plug-in script files and cartridge model description files. Passing and storing passwords in plain text poses a security risk. Follow proper security guidelines to develop OSM cartridges to protect data over communication channels. Oracle recommends using SSL communication between OSM and an external system, particularly for Web Services of external systems.

The following are examples of external systems used in OSM cartridges that may require credential information:

- OSM Web Service
- Databases
- JMS queues and topics (excepts JMS queues deployed by the cartridge)
- Web Services of any system

To develop your OSM cartridges to use the credential store, see the following:
Use "AutomationContext" in your automation plug-in code to retrieve credentials from the credential store. See "Developing Automation Plug-ins to Use the Credential Store" for more information.

Use the operation APIs in "ViewRuleContext" in XQuery scripts to access credentials stored in the credential store.

Use "PasswordCredStore" in your JAVA classes to retrieve user name and password from the credential store.

Use the attributes for credential store in "SoapAdapter" to retrieve credentials from the credential store when sending a SOAP request using HTTP/HTTPS.

Use the attributes for credential store in "ObjectelHPPTAdapter" to retrieve credentials from the credential store when sending a request to Objectel. See "Defining Data Providers in OSM Cartridges to Use the Credential Store" for more information.

See "OSM Credential Store Command and API Reference Material" for a description of the OSM credential store APIs.

See "About the Credential Store" for information about the credential store.

Developing Automation Plug-ins to Use the Credential Store

Some OSM credential store APIs can be used in custom automation plug-in Java classes. Use these APIs when you define custom automation plug-in classes to access an external system with credentials. You can also call OSM credential store APIs from your automation context classes. The XQuery plug-in code for an automation or activation task can use credential store APIs to retrieve credentials from the credential store. See "AutomationContext" for example code of developing automation plug-ins in OSM cartridges to retrieve credentials from the credential store.

External instance adapters and automation plug-in classes (XQuerySender and XSLTSender) provided by Oracle to send messages and requests to external systems support the OSM credential store APIs.

Defining Data Providers in OSM Cartridges to Use the Credential Store

A data provider class in your cartridge where credential information is required by an external system must be set up to read the required credentials from the credential store. For information on setting up your data provider classes to be able to read from the credential store, see the following:

- Using the Credential Store with Custom Data Providers
- Using the Credential Store with Built-In Data Providers

Using the Credential Store with Custom Data Providers

When you add a data provider class in your cartridge where credential information is required by an external system, the data provider class can call OSM credential store APIs to read the required credentials from the credential store. Your data provider class must implement method retrieveInstance() of interface ExternalInstanceAdapter.

To read the required credential from the credential store when you define your own data provider class (provider type is "Custom"), use the following APIs in the retrieveInstance() method in your Java class:
Note: This example assumes you are using your own map. If you use the default map (osm) and key names for the OSM application, you can use simpler code:

```java
String password = context.getOsmCredentialPassword(username)
```

```java
public Element retrieveInstance(final ViewRuleContext context, final Map<String, Object> parameters) throws Exception {
    // DoCustomLogic

    String mapname = getStringParam(parameters, "para.mapname", null);
    String keyname = getStringParam(parameters, "para.keyname", null);
    String username = "";
    String password = "";

    if (mapname != null && keyname != null) {
        try {
            String credential = context.getCredential(mapname, keyname);
            int index = credential.indexOf("");
            username = credential.substring(0, index-1);
            password = credential.substring(index);
        } catch (Exception e) {
            // DoCredStoreExceptionHandling
        }
    }
    // DoAuthenticationWithUsernamePassword
    // DoCustomerRequest
    // DoResponseHandling
}
```

Using the Credential Store with Built-In Data Providers

Oracle provides pre-defined or built-in data provider classes "SoapAdapter" and "ObjectelHPPTAdapter" which contain the code required for using the credential store. To use the credential store when you use these built-in adapters, add the input parameters required for the credential store in your data provider.

If you use the SoapAdapter, add the input parameters:

- Name of parameter "oms:credentials.mapname", contentType is "XPATH", default value is "myMap" (this example uses a custom map)
- Name of parameter "oms:credentials.keyname", contentType is "XPATH", default value is "myUser"

If you use the ObjectHTTPAdapter, add the input parameters:

- Name of parameter "obj:mapname", contentType is "XPATH", default value is "osm" (this example uses the default map)
- Name of parameter "obj:keyname", contentType is "XPATH", default value is "osmUser_osm" (this example uses user "osm" stored in default map)

Upgrading Existing Cartridge Code to Use the Credential Store

Upgrade your existing OSM cartridge code to use the credential store by using the OSM credential store APIs. Upgrade your custom data provider classes and XQuery plug-in code to use the OSM credential store APIs for retrieval of credentials. See
"Developing Cartridges to Use the Credential Store" for information on developing cartridges to use the credential store.

In addition to upgrading your cartridge code, provision the credential store for the WebLogic domain for OSM applications with required credentials (see "Managing Credentials in the Credential Store") and configure the JPS policy for the WebLogic domain to allow OSM access to the credential store (see "Configuring the Java Security Policy for the OSM Credential Store").

Using Built-in SOAP Adapter as a Data Provider Class

Credential information is required to send a SOAP request. If your existing automation plug-in code that is used to test the SOAP adapter has hard coded passwords, you can use built-in SOAPAdapter as a data provider class in your cartridges to remove the need for the hard coded passwords.

When you use the default map for OSM applications, automation plug-in users pass in the user name only in the parameter. When you use your custom credential store map, automation plug-in users pass in the credential map name and key name for credentials in the parameter.

To update existing automation plug-in code that tests SOAPAdapter to remove hard coded passwords:

1. Remove hard coded passwords from the existing "oms:credentials:password" input parameter.
2. Ensure that credentials exist in the credential store under map="osm" key="osmUser_"+<SoapRequest_username>.
3. Test that the SOAP adapter works correctly after the update.

OSM Credential Store Commands

OSM offers two credential store commands available in the XML Import/Export application which can be used for OSM credential store setup or management:

- **credStoreAdmin command**
  
  You can use this command to configure the JPS policy for the credential store and manage credentials in the credential store. See "credStoreAdmin Command" for a description of this command.

- **userAdmin command**
  
  You can use this command for credential store management. This command is also used to administer OSM users and workgroups. See "userAdmin Command" for a detailed description of this command.

See "Using the XML Import/Export Application" for information on using the XML Import/Export application.

Setting Build Properties for Credential Store Commands

To use the credential store commands credStoreAdmin and userAdmin, you must set the following properties in OSM_home/SDK/XMLImportExport/build.properties:

- **xmlie.root.dir**: The XML Import/Export application directory (/XMLImportExport) in the OSM SDK installation.
- **middlewareHome**: The Fusion Middleware installation directory.
- `xmlie.root.modelDocument`: The XML data file that contains the credential information (`credential.xml`).
- `xmlie.root.configDocument`: The configuration file for the `credStoreAdmin` command (`config/config.xml`).

Note that the `credStoreAdmin` command does not use the other properties in the `build.properties` file.
To develop OSM cartridges to use the credential store offered through CSF (see “About the Credential Store”), use the OSM credential store APIs. OSM credential store APIs are wrapper APIs to the CSF APIs. Use the OSM credential store APIs in your OSM-related code that requires credential retrieval, such as in data providers and automation plug-ins.

The OSM credential store APIs and credential store-related classes are:

- userAdmin Command
- credStoreAdmin Command
- CredStore
- PasswordCredStore
- CredStoreException
- SoapAdapter
- ObjectelHPPTAdapter
- ViewRuleContext
- AutomationContext
userAdmin Command

The userAdmin command is part of the XML Import/Export application and is used to administer OSM users and workgroups. The userAdmin command also supports credential store management.

Use the userAdmin command to create an OSM user and also add the user in the credential store.

For credential-store related interface and object details, see "J2ee Manager/WLUserManager" and "UserAdminOperation".

Use the userAdmin command to add OSM users to the default OSM credential store (to the default map with default key values).

Syntax

Batch script:

userAdmin user.xml config/config.xml

Ant script:

ant userAdmin

Creating the XML Data File for the userAdmin Command

To create the XML data file for the userAdmin command (user.xml):

---

**Important:** Remove the file when the operation is complete because it contains unencrypted passwords.

---

Example input data file (user.xml):

```xml
<userConfig xmlns="http://www.metasolv.com/Provisioning/UserConfig"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <user name="testOsmUser1">
        <description>OSM test user 1</description>
        <password>mypassword1</password>
    </user>
    <user name="testOsmUser2">
        <description>OSM test user 2</description>
        <password>mypassword2</password>
    </user>
    <clientGroup>
        <user>testOsmUser1</user>
        <user>testOsmUser2</user>
    </clientGroup>
    <automationGroup>
        <user>testOsmUser1</user>
        <user>testOsmUser2</user>
    </automationGroup>
    <wsAPIGroup>
        <user>testOsmUser1</user>
        <user>testOsmUser2</user>
    </wsAPIGroup>
    <xmlAPIGroup>
        <user>testOsmUser1</user>
    </xmlAPIGroup>
</userConfig>
```
Creating the Configuration File for the userAdmin Command

To create the configuration file for the userAdmin command (config.xml):

1. Copy the sample XML Import/Export application configuration file config/config_sample.xml and rename it to config/config.xml.

2. Edit the "j2eeAdminConnection" and "log" sections of the file with your installation information.

The following is an example "j2eeAdminConnection" section which contains the data for WebLogic Server:

```xml
<j2eeAdminConnection>
  <user>weblogic</user>
  <password/>
  <hostname>localhost</hostname>
  <port>7001</port>
</j2eeAdminConnection>
```

When the password value is empty in the configuration file, which is the recommended approach for security purposes, you must:

- Input the password at run time when prompted if running the batch script or Ant task.

3. Edit the "credentialStore" section to define the credentialStore element as true:

```xml
<credentialStore addUser="true"/>
```

This enables the userAdmin command to perform credential store updates.

4. (Optional) Edit the "databaseConnection" section.

If you configure workgroups using the userAdmin command and the XML data file contains "workgroup" sections, you are required to edit this section. However, it is better to avoid configuring workgroups using the userAdmin command because it requires setting up database connection parameters in the configuration file which is not a secure approach. Instead, it is recommended that you configure workgroups after OSM user is created using OSM Administrator or during cartridge deployment.

Usage Notes

The userAdmin command can create a new WebLogic user and add the user to the OSM default credential store map at the same time.

Calling UserAdmin Target in Another Ant Script

The following is an example on how to invoke the userAdmin Ant script in your own Ant script:
<target description="Configure OSM user" name="setupUsers" depends="wls_password">
    <echo message="Create users in WebLogic and Credential Store"/>
    <ant inheritRefs="true" antfile="${xmlieRoot}/build.xml" dir="${xmlieRoot}" target="userAdmin">
        <property name="wls_admin_user" value="weblogic"/>
        <property name="wls_admin_password" value="${wls.password}"/>
        <property name="wls_host" value="localhost"/>
        <property name="wls_port" value="7001"/>
        <property name="middlewareHome" value="${middleware.home}"/>
        <property name="xmlie.root.modelDocument" value="user.xml"/>
        <property name="xmlie.root.configDocument" value="config.xml"/>
    </ant>
</target>

<target name="wls_password">
    <input message=" Enter WebLogic Admin User Password:  " addproperty="wls.password">
        <handler classname="org.apache.tools.ant.input.SecureInputHandler"/>
    </input>
</target>
</target>
</target>
</input>
</target>
</input>
</target>

**J2ee Manager/WLUserManager**

Business Object Name: J2eeManager/WLUserManager  
Business Object Component Name: Package name: com.mslv.oms.j2ee.useradmin  
Description: This class is used to create J2EE user in WebLogic and add the user to appropriate J2EE groups. It can also add the user in the WebLogic CSF credential store.

**Attributes**

- **credStoreName**  
  Type: ObjectName  
  Description: MBean object for credential store:  
  JpsJmxConstants.MBEAN_JPS_CREDENTIAL_STORE

**Business Object Operations**

Operation Name: createUserInCredentialStore  
Description: Method which adds the user in credential store.  
If the map/key pair exists in the credential store already, it will be overwritten with new values.

**UserAdminOperation**

Business Object Name: UserAdminOperation  
Business Object Component Name: Package name  
com.mslv.oms.metadatahandler.operation  
Description: This class is used to create J2EE user in WebLogic, and add the user to appropriate J2EE groups. It also can add the user in the credential store.

**Attributes**

- **OSM_CREDENTIAL_MAPNAME**
**Type:** String (static final)
**Sensitive:** Value is "osm"
**Description:** Pre-defined map name for OSM application in credential store.

- **OSM_CREDENTIAL_KEYNAME_PREFIX**
  
  **Type:** String (static final)
  **Sensitive:** Value is "osmUser_"
  **Description:** Prefix of key names used for OSM users in credential store.

---

**Business Object Operations**

**Operation Name:** configureJ2eeUsers

**Description:** This method can add users to the credential store.

After a user is created in the J2EE server, a check is made if configuration is defined to add the user in the credential store. The following line is the example configuration (the default value of this configuration is set to "false"):

```
<credentialStore addUser="true"/>
```

The user is added to the credential store using the default map name `OSM_CREDENTIAL_MAPNAME` and default key name `OSM_CREDENTIAL_KEYNAME_PREFIX` + OSM_username. For example, if OSM user name is "osmlf", then the map and key values used for it will be map="osm" and key="osmUser_osmlf".
credStoreAdmin Command

Use the credStoreAdmin command to configure the JPS policy for the credential store and to manage credentials in the credential store.

Cartridges can use the credStoreAdmin command to create and configure credential stores during setup.

The credStoreAdmin command is available as an Ant script and as a batch script in the XML Import/Export application (which is included in the OSM SDK package). The batch script supports interactive mode which allows users to input passwords at runtime; this is the recommended method of using the credStoreAdmin command because entering the password at runtime is a more secure approach.

See "Configuring the Java Security Policy for the OSM Credential Store" for instructions on configuring the JPS policy for the OSM credential store map using the credStoreAdmin command.

See "Managing Credentials in the Credential Store" for information on using the credStoreAdmin command to manage credentials in the credential store.

ANT Task Name

credStoreAdmin

Batch Script Name

credStoreAdmin.bat (Windows)

Schema File

The schema file for the credStoreAdmin command is OSM_home/SDK/XMLImportExport/models/CredStoreAdmin.xsd

Task Arguments

XML data file that contains credential information: credentials.xml (see "Creating the XML Data File for the credStoreAdmin Command" for information on creating this file).

XMILE configuration file: config.xml

WLS administrator password (if not provided in config.xml)

If the WLS administrator password is provided in the command line, the following values can be passed in also:

---

**Note:** This mode is used when a cartridge uses this command to create and configure credential stores during setup.

---

- WebLogic administrator user name
- WebLogic server host
- WebLogic server port
Schema File Input Data Format

The following is the schema for the XML Import/Export application configuration file (config.xsd):

```xml
<xs:element name="configuration">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="credentialStore" type="oms:credentialStoreType" minOccurs="0">
                <xs:annotation>
                    <xs:documentation>It determines if user should be added in OSM credential store for new OSM user. The default would be no if node not exist.</xs:documentation>
                </xs:annotation>
            </xs:element>
        </xs:sequence>
    </xs:complexType>
</xs:element>
```

Creating the XML Data File for the credStoreAdmin Command

The following is an example XML data file for the credStoreAdmin command (credential.xml). This example uses the map name osm, the default map for OSM applications. If you do not use the default map, replace osm with your map name.

```xml
<?xml version="1.0" encoding="UTF-8"?>
    <ns2:jpsPolicy operation="add">
        <ns2:mapname>osm</ns2:mapname>
    </ns2:jpsPolicy>
</ns2:CredentialConfig>
```

Creating the Configuration File for the credStoreAdmin Command

To create the configuration file (config/config.xml):

1. Copy the sample configuration file config/config_sample.xml and rename it to config/config.xml.
2. Edit the "j2eeAdminConnection" and "log" section of the file with your installation information.

Note that other sections of the file are not used in the credStoreAdmin command, but they must exist and can use dummy values.

The following is an example "j2eeAdminConnection" section which contains the data for WebLogic Server; for example.
When the password value is empty in the configuration file (which is the recommended approach for better security):

- You must input the password at run time when prompted if running the batch script or Ant task.

### Calling credStoreAdmin Target in Another Ant Script

The credStoreAdmin target can be called directly in Ant scripts or batch scripts; this capability can be used during an OSM installation with OSM cartridges and custom cartridges.

The following is an example of how to call the credStoreAdmin target in another Ant script:

```xml
<target description="Configure JPS Policy" name="setupJPSPolicy" depends="wls_password">
  <echo message="Configure JPS Policy for default credential store in WebLogic"/>
  <ant inheritRefs="true" antfile="${xmlieRoot}/build.xml" dir="${xmlieRoot}" target="credStoreAdmin">
    <property name="wls_admin_user" value="weblogic"/>
    <property name="wls_admin_password" value="${wls.password}"/>
    <property name="wls_host" value="localhost"/>
    <property name="wls_port" value="7001"/>
    <property name="middlewareHome" value="${middleware.home}"/>
    <property name="xmlie.root.modelDocument" value="credential.xml"/>
    <property name="xmlie.root.configDocument" value="config.xml"/>
  </ant>
</target>

<target name="wls_password">
  <input message=" Enter WebLogic Admin User Password: ">
    <handler classname="org.apache.tools.ant.input.SecureInputHandler"/>
  </input>
</target>
```

If the Ant target "credStoreAdmin" is called by another Ant script, which is running in your Design Studio workspace, interactive mode is not supported. In this case, passwords must be provided in the XML data file.

**Note:** It is recommended that you delete this data file immediately after use because it contains unencrypted passwords.

The following is an example of an XML data file that contains the passwords for user osm and osmlf:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ns2:CredentialConfig
  xmlns:ns2="http://www.metasolv.com/Provisioning/CredentialConfig"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="../XMLImportExport/models/CredStoreAdmin.xsd">
  <user>weblogic</user>
  <password/>
  <hostname>localhost</hostname>
  <port>7001</port>
</ns2:CredentialConfig>
```
Business Object Operations for credStoreAdminOperation

For business operations for credStoreAdminOperation, see "CredStoreAdminOperation".

CredStoreAdminOperation

Business Object Name: CredStoreAdminOperation
Business Object Component Name: Package name: com.mslv.oms.metadatahandler.operation
Description: This new class is used to configure the JPS policy for your custom credential store map and to manage credentials in the credential store.

Attributes

- **credStoreName**
  
  Type: ObjectName
  
  Description: Mbean object for credential store: JpsJmxConstants.MBEAN_JPS_CREDENTIAL_STORE

- **globalPolicyName**
  
  Type: ObjectName
  
  Description: Mbean object for global policy: JpsJmxConstants.MBEAN_JPS_ADMIN_POLICY_STORE

Business Object Operations

**configJPSPolicy**

This method is used to:

- Update the JPS policy to use the default credential store map (the default map is not configured during OSM installation).
- Configure the JPS policy with an entry for your custom credential store map. The supported operations are **add** and **remove**.

Example credential data in XML file with JPS policy information:

```xml
......
<jpspolicy operation="add">
  <mapname>osm_systemAmap</mapname>
</jpspolicy>
```
configCredentialStore
This method is used to manage credentials in the WebLogic Server credential store. Use this command to manage credentials of external systems. Use the "userAdmin Command" to configure the OSM user in the credential store.

Example credential data in the XML file with credential information:

```xml
<credential operation="create">
  <mapname>osm_systemAmap</mapname>
  <keyname>user1</keyname>
  <user>mobileUser1</user>
  <password>user1pwd</password>
</credential>
```

Supported operations are create, update, and delete.

**Note:** If create fails when specified map/key values already exist in the credential store, set attribute "overwrite" to "false".

**Note:** Password value can be provided through console input.
CredStore

Credential store object.

The credential store object is the domain credential store class which contains a single instance of the CredentialStore object. The JpsServiceLocator APIs in the Credential Store Framework look up the single instance of the CredentialStore object.

Package name: com.mslv.oms.security.credstore

Attributes

Name: store
Type: Oracle.security.jps.service.credstore.CredentialStore
Description: A reference object to the JPS credential store object.

Business Object Operations

getInstance
Description: Return an instance of the object. Only a single instance of the class is ever created. If "store" is not initiated, look up the credential store from class "oracle.security.jps.service.credstore.CredentialStore".
Operation Outputs: Output Name: store; Type: CredStore; Description: An instance of the CredentialStore object.

def getJPSCredentialStore
Description: Retrieving attribute "store".
Operation Outputs: Output Name: store; Type: oracle.security.jps.service.CredentialStore.

Output of new methods
An instance of the object is returned by getInstance(). At the first time invocation, object will be initiated, and a credential store of class Oracle.security.jps.service.credstore.CredentialStore is resolved through the CSF lookup API.

Error Conditions

Improper JPS configuration can cause credential store lookup to fail.

Usage Notes

This API can be used directly if you have your own implementation JAVA class of "ViewRuleContext" and "AutomationContext".
PasswordCredStore

Password credential store object.

Use com.mslv.oms.security.credstore.PasswordCredStore APIs in your JAVA classes to retrieve user name and password from the credential store.

Package Name

com.mslv.oms.security.credstore

Attributes

- **credstore**
  
  Type: CredStore
  
  Description: A reference object to OSM credential store object.

- **OSM_CREDENTIAL_MAPNAME**
  
  Type: String (static final)
  
  Sensitive: Value is "osm"
  
  Description: Pre-defined map name for OSM application in credential store.

- **OSM_CREDENTIAL_KEYNAME_PREFIX**
  
  Type: String (static final)
  
  Sensitive: Value is "osmUser_"
  
  Description: Prefix of key names used for OSM users in credential store.

Business Object Operations

**Operation Name: getPasswordCredential**

**Description**

Return a PasswordCredential object stored with specified map and key names.

**Input Parameters**

- **mapName**
  
  Type: String
  
  Description: Map name of the stored password credential object

- **keyName**
  
  Type: String
  
  Description: Key name of the stored password credential object

**Operation Outputs**

- **passwordCredential**
  
  Type: PasswordCredential
  
  Description: An object of Oracle.security.jps.service.credstore.PasswordCredential, which contains credential information stored under map and key name pair.

**Operation Name: getCredential**

**Description**

Return a string of user name and password for specified map and key names.
Input Parameters

mapName
Type: String
Description: Map name of the stored password credential object

keyName
Type: String
Description: Key name of the stored password credential object

Operation Outputs

Type: String
Description: A string contains user name and password information stored under map and key name pair. Format is "username/password".

Operation Name: getOsmCredentialPassword

Description
Return password value for specified OSM user. This API is used to access credentials stored in the credential store using the default map and key names that follow OSM naming convention:

- Map name is osm
- Key name is osmUser_+username

Input Parameters

username
Type: String
Description: OSM user name.

Operation Outputs

Type: String
Description: A string contains password value for specified OSM user. OSM user name and password values are stored under credential store with map value OSM_CREDENTIAL_MAPNAME, and key value starts with OSM_CREDENTIAL_KEYNAME_PREFIX, following with user name.

Operation Name: getCredentialAsXML

Description
Return user name and password in XML format for specified map and key names.

Input Parameters

mapName
Type: String
Description: Map name of the stored password credential object

keyName
Type: String
Description: Key name of the stored password credential object

Operation Outputs

Type: org.w3c.dom.Element
Description: An element that contains user name and password information stored under map and key name pair.
Output of Methods

These methods will return a PasswordCredential/String/Element object if the credential store contains a credential with specified map name and key name. If a match is not found, null value will be returned.

Error Conditions

Improper JPS configuration can cause “read” operation on the credential store to fail due to “no permission” error. Incorrect map and key names can cause “no credential found” problem.

Usage Notes

This API can be used directly if you have your own implementation JAVA class of "ViewRuleContext" and "AutomationContext".

Example: Retrieve Password from OSM Default Map Given User Name

```java
PasswordCredStore pwdCredStore;
try {
    pwdCredStore = new PasswordCredStore();
    return pwdCredStore.getOsmCredentialPassword(username);
} catch (final Exception e) {
    throw new AutomationException("Fail to find password credential with specified map and key name.", e);
}
```

Example: Retrieve Password from Custom Map Given Map and Key Names Used to Store the Credentials

```java
PasswordCredStore pwdCredStore;
try {
    pwdCredStore = new PasswordCredStore();
    return pwdCredStore.getCredentialAsXML(map, key);
} catch (final Exception e) {
    throw new AutomationException("Fail to find password credential with specified map and key name.", e);
}
```
CredStoreException

Credential store exception object.

Package Name

com.mslv.oms.security.credstore

Attributes

Name: target
Type: Exception
Description: Target exception is the original exception caught in the three OSM credential store classes: CredStore, PasswordCredStore, JPSPasswordCredential.

Business Object Operations

Operation Name: getTargetException
Description
Get attribute "target".

Operation Outputs
exception
Type: Exception

Usage Notes

This API can be used directly if you have your own implementation JAVA class of "ViewRuleContext" and "AutomationContext".
SoapAdapter

Use the attributes for the credential store when you define data provider instances in your cartridges.

For detailed information on data provider adapters, see the discussion on behaviors in OSM Developer’s Guide.

Description

Built-in adapter.

Attributes

- CREDENTIAL_MAPNAME_PARAM
  Type: String
  Description: Defines the parameter name to be specified in data provider for SOAP. A constant with value "oms:credentials.mapname".

- CREDENTIAL_KEYNAME_PARAM
  Type: String
  Description: Defines the parameter name to be specified in data provider for SOAP. A constant with value "oms:credentials.keyname".

Business Object Operations

Operation Name: retrieveInstance

Description

This method includes support to retrieve credential information from the credential store, from map and key name parameters if provided.

Business Logic

The business logic for retrieveInstance is as follows:

- If "oms:credentials.username" is provided in parameters:
  - If "oms:credentials.password" is also provided in parameter, then input values are used directly.
  - If "oms:credentials.password" is not provided in the parameter, call context API "getOsmCredentialPassword(username)" to retrieve the password value from the credential store and use it in the SOAP request.

- Otherwise, if "oms:credentials.mapname" and "oms:credentials.keyname" are provided in the parameters, call context API "getCredential(mapname, keyname)" to retrieve user name and password, and use them in the SOAP request.

Error Conditions

Invalid map and key names can cause credential lookup to return a "null" object.

Message text is "Password credential with map name %s and key name %s does not exist in the credential store."

Usage Notes

Do not use operation APIs directly in this object.
Objectel HPPTAdapter

Use the attributes for the credential store when you define data provider instances in your cartridges.

For detailed information on data provider adapters, see the discussion on behaviors in OSM Developer’s Guide.

Description

Built-in adapter. Objectel HTTP adapter.

Attributes

- **CREDSITUAL_MAPNAME_PARAM**
  - Type: String
  - Description: Defines the parameter name to be specified in data provider for Objectel HTTP type. A constant with value "obj:mapname".

- **CREDSITUAL_KEYNAME_PARAM**
  - Type: String
  - Description: Defines the parameter name to be specified in data provider for Objectel HTTP type. A constant with value "obj:keyname".

- **mapname**
  - Type: String
  - Description: Value specified for map name parameter.

- **keyname**
  - Type: String
  - Description: Value specified for key name parameter.

Business Object Operations

**Operation Name: parseParameters**

**Description**

This method includes support to parse parameters for credential store map and key names. Add context to input parameter. Same method in the super class will be changed as well.

**Input Parameters**

**Context**

Type: ViewRuleContext

**Operation Name: sendCommand**

**Description**

This method includes support to retrieve credential information from the credential store, from map and key name parameters if provided.

**Business Logic**

The business logic for sendCommand is as follows:

- If "obj.user_name" is provided in the parameters:
If "obj:password" is also provided in the parameter, then input values are used directly.

If "obj:password" is not provided in the parameter, call context API "getOsmCredentialPassword(username)" to retrieve password value from the credential store and use it in the SOAP request.

- Otherwise, if "obj:mapname" and "obj:keyname:" are provided in parameters, call context API "getCredential(mapname, keyname)" to retrieve user name and password and use them in the SOAP request (after the command, the code will send a SOAP message via HTTP to the specified URL).

**Usage Notes**

Do not use operation APIs directly in this object.

**Error Conditions**

Invalid map and key names can cause credential lookup to return a "null" object.

Message name: ViewRuleFailedException

Message text: "Password credential with map name %s and key name %s does not exist in the credential store."
**ViewRuleContext**

Use operation APIs defined in this interface object for the credential store.

**Description**

Interface object.

**Business Object Operations**

**Operation Name: getCredential**

**Description**

Return a string of user name and password for specified map and key names.

**Input Parameters**

- **map**
  
  Type: String
  
  Description: Map name

- **key**
  
  Type: String
  
  Description: Key name

**Operation Outputs**

Type: String

Description: A string contains user name and password information stored under map and key name pair. Format is "username/password".

Details on operation getCredential():

```java
/**
 * Get user name and password values in string format from credential store, given map and key values.
 */

String getCredential(final String map, final String key) throws TransformerException;
```

**Operation Name: getOsmCredentialPassword**

**Description**

Return password value for specified OSM user. This API is used to access credentials stored in the credential store using the default map and key names that follow OSM naming convention:

- Map name is **osm**
Key name is osmUser_+username

Input Parameters
username
Type: String
Description: OSM user name.

Operation Outputs
Type: String
Description: Return password value for specified OSM user. OSM user name and password values are stored under credential store with map value OSM_CREDENTIAL_MAPNAME, and key value starts with OSM_CREDENTIAL_KEYNAME_PREFIX, following with user name.

Error Conditions
Improper JPS configuration can cause creation of PasswordCredStore to fail.
Message Name: ViewRuleFailedException
Message Text: "Fail to create PasswordCredStore."

Usage Notes
This API is often used in XQuery scripts.
Use operation APIs from AutomationContext interface to retrieve credentials in XQuery code for automation tasks.

See "Example: Retrieve Password from OSM Default Map Given User Name".
See "Example: Retrieve Password from Custom Map Given Map and Key Names Used to Store the Credential".

Description

Interface object.

Business Object Operations

Operation Name: getCredentialAsXML
Description
Get user name and password values in XML format given map and key values of the credential.

Input Parameters

map
Type: String
Description: Map name

key
Type: String
Description: Key name

Operation Outputs

Type: org.w3c.dom.Element
Description: An element that contains user name and password information stored under map and key name pair.

Details on operation getCredentialAsXML():

/**
 * Get user name and password values in XML format given map and key values of the credential.
 * @param map
 * @param key
 * @return User name and password for the user in this XML format:
 *     <Credential xmlns="urn:com:metasolv:oms:xmlapi:1">
 *         <Username>NAME</Username>
 *         <Password>PASSWORD</Password>
 *     </Credential>
 * @throws CredStoreException If the application cannot access credential store, or if there is no permission to read the credential store with given map and key values, or if the credential is expired.
 */

Document getCredentialAsXML(final String map, final String key) throws
Operation Name: getOsmCredentialPassword

Description
Return password value for specified OSM user. This API is used to access credentials stored in the credential store using the default map and key names that follow OSM naming convention:

- Map name is osm
- Key name is osmUser_+username

Input Parameters
username
Type: String
Description: OSM user name.

Operation Outputs
Type: String
Description: Password value for specified OSM user. OSM user name and password values are stored under credential store with map value OSM_CREDENTIAL_MAPNAME, and key value starts with OSM_CREDENTIAL_KEYNAME_PREFIX, following with user name.

Error Conditions
Fail to read credential store due to improper JPS configuration or invalid map and key names.

Message Name: AutomationException
Message Text: "Fail to create PasswordCredStore. Password credential with map name %s and key name %s does not exist in the credential store."

Example: Retrieve Password from OSM Default Map Given User Name

```
declare variable $context external;
let $osmPwd := context:getOsmCredentialPassword($context, $username)
```

Example: Retrieve Password from Custom Map Given Map and Key Names Used to Store the Credential

```
declare namespace oms="urn:com:metasolv:oms:xmlapi:1";
declare variable $context external;
let $customCred := context:getCredentialAsXML($context, "osmTest", $username)/oms:Credential
let $customerName := $customCred/oms:Username/text()
let $customPwd := $customCred/oms:Password/text()
```
This chapter describes how to change the appearance and functionality of Oracle Communications Order and Service Management (OSM) GUI applications.

About Configuring the User Experience

OSM supports the customization of the information presented to and collected from Web Client users to match a wide variety of user roles and tasks. This makes it easy to customize the user experience for edits and data validation without having to code Java Server Pages (JSPs).

The Web Client user experience can be customized in various ways:

- Adding default values and calculating values from different fields on the order
- Viewing and selecting data dynamically from a source outside of OSM
- Adding fonts, colors, and other formatting; adding conditional formatting
- Organizing order data in tables and tabs
- Adding your own custom HTML-based online help (with hyperlinks) and tool tips for your users
- Presenting data in one or more presentation languages
- Enforcing formats of input data, required fields and so on
- Hiding and showing fields relative to other fields
- Adding check boxes and radio buttons
- Conditionally making fields read-only or read-write
- Provide custom information, warning, and error messages to the user to help them

About Behaviors

Behaviors provide the mechanism to exercise greater control over validation and presentation of order data to Task, and Order Management Web Client users. (In earlier versions of OSM, this capability was called the View Framework.) Behaviors are used mainly in the context of manual tasks that you manage with the Task Web Client.

There are nine behavior types that enable you to dynamically control a specific aspect of the order data model. (You can also add new behavior types using Oracle Communications Design Studio). The included behavior types are:
- **Calculation**: Computes the value of a field value based on a formula that references order data.
- **Constraint**: Specifies a condition that must be met for the data to be considered valid.
- **Data Instance**: Declares an instance that can be used by other behaviors.
- **Event**: Specifies an action that is performed when data is modified.
- **Information**: Specifies the label, hint, and help information for the element instance.
- **Lookup**: Specifies a set of dynamically generated choices from which you can select.
- **Read Only**: Dynamically determines whether a value can be modified or not.
- **Relevant**: Dynamically determines whether data is visible or hidden.
- **Style**: Specifies the visual appearance of fields.

Behaviors can be created only for manual tasks. They can be created at the data element level (most general), order level (more specific), or task level (most specific). After the behavior is created in Design Studio, you can model the actions you want it to perform through its properties settings.

See *OSM Developer’s Guide* and Design Studio Modeling OSM Processes Help for more information about behaviors.

You can use Design Studio to define additional behaviors.
Configuring OSM with oms-config.xml

This chapter explains how to configure Oracle Communications Order and Service Management (OSM) using oms-config.xml and provides a detailed reference of available parameters.

Working with the oms.ear File

The oms.ear file is an OSM WebLogic application component deployed to the WebLogic server. It contains OSM configuration directories and files, including oms-config.xml.

Unpacking the oms.ear file lets you access and change the OSM configuration files, and packing and redeploying the oms.ear file implements your changes. The scripts for unpacking and packing the oms.ear file are located in the OSM_home/SDK/Customization directory. This directory exists only if you installed the SDK during a Custom installation.

Note: You can install only the SDK on a Windows, UNIX, or Linux machine by running the OSM installer again, and selecting Custom installation.

Undeploying and redeploying the oms.ear allows the WebLogic server to pick up any changes you have made to oms.ear.

Unpacking the oms.ear File

Before you can work with the oms-config.xml file or any other configuration file packed in the oms.ear file, you must unpack the oms.ear file.

To unpack the oms.ear file:

1. Copy the oms.ear file from the following directory to your OSM_home/SDK/Customization directory:

   domain_home/servers/admin_server_name/upload/oms/app/

   where admin_server_name is the name of the WebLogic administration server.

Note: Oracle recommends that you make a backup copy of your original oms.ear file, in case you need to restore the default settings.
2. From the `OSM_home/SDK/Customization` directory, use a text editor to do one of the following:
   - In UNIX or Linux, edit `unpackOMS.sh` and set the JAVA_HOME variable to the path to the JDK on your system.
   - In Windows, edit `unpackOMS.bat` and set the JAVA_HOME variable to the path to the JDK on your system.

   **Note:** If the path in JAVA_HOME contains a space, enclose the path in quotation marks. For example:

   ```
   set JAVA_HOME="C:\oracle\middleware test\jdk170_51"
   ```

3. Save and close the file.

4. Do one of the following:
   - In UNIX or Linux, run `unpackOMS.sh`.
   - In Windows, run `unpackOMS.bat`.

   The script unpacks the `oms.ear` file and creates the following new sub-directories in the `Customization` directory:
   - osm-ear
   - osm-ejb
   - osm-war
   - osmwebui
   - resources

### Packing the `oms.ear` File

When you finish editing a configuration file, you must pack the `oms.ear` file and redeploy it to the OSM WebLogic server so that your edits take effect.

To pack the `oms.ear` file:

1. From the `OSM_home/SDK/Customization` directory, use a text editor to do one of the following:
   - In UNIX or Linux, edit `packOMS.sh` and set the JAVA_HOME variable to the path to the JDK on your system.
   - In Windows, edit `packOMS.bat` and set the JAVA_HOME variable to the path to the JDK on your system.

   **Note:** If the path in JAVA_HOME contains a space, enclose the path in quotation marks. For example:

   ```
   set JAVA_HOME="C:\oracle\middleware test\jdk170_51"
   ```

2. Save and close the file.

3. Do one of the following:
   - In UNIX or Linux, run `packOMS.sh`.
   - In Windows, run `packOMS.bat`.
The script creates a new oms.ear file.

4. Copy the new oms.ear file to following directory:
   
   domain_home/servers/admin_server_name/upload/oms/app/

   where admin_server_name is the name of the WebLogic administration server.

5. Do one of the following:
   - Stop and restart the WebLogic server. See "Starting and Stopping OSM".
   - Redeploy the oms.ear file:
     a. Log in to the WebLogic Administration Console.
     b. In the Domain Structure pane, click Deployments.
     c. In the Deployments window, select the box next to oms, click Stop, and select the option appropriate for your situation.
     d. With the box next to oms selected, click Delete.
     e. Click Activate Changes.
     f. Click Install.
     g. In the Install Application Assistant window, browse to the location of the updated oms.ear file and click Finish.
     h. Select the box next to oms and click Start and Servicing all Requests.

Undeploying and Redeploying the oms.ear File

Undeploying and redeploying the oms.ear file makes any changes you have made active on the OSM server.

To undeploy and redeploy the oms.ear file:

1. Log in to the WebLogic Server Administration console as a user with administrative privileges.

2. In the Domain Structure pane, click Deployments.

3. Select the oms enterprise application deployment in the Deployments table by clicking the check box for that row.

4. From the Stop menu for the Deployments table, select either When work completes or Force stop now.

5. When the server has stopped, select the oms deployment again and click Delete.

6. Click Install.
   The Install Application Assistant window is displayed.

7. Browse to the location of the new version of the oms.ear file and select oms.ear.

8. Click Next.

9. Select Install this deployment as an application and click Next.

10. Click Finish.
   You are returned to the Deployments window with the oms deployment added and active.
Working with oms-config.xml

You can configure various OSM runtime parameters in the oms-config.xml file. The file can be stored in oms.ear, or it can be located externally.

By default, OSM uses oms-config.xml packed in oms.ear for configuration parameters.

Oracle recommends the using an external oms-config.xml file for the following reasons:

- It is quicker to make changes to an external oms-config.xml file because you do not have to unpack and undeploy the old oms.ear file and repack and redeploy the updated oms.ear file.
- OSM monitors the external oms-config.xml file. If any changes are made to it when OSM is running, it automatically reloads oms-config.xml and refreshes metadata.

Using an External oms-config.xml File

To use an external oms-config.xml file:

1. If you have not already done so, unpack the oms.ear file as described in "Unpacking the oms.ear File".
2. Make a copy of the oms-config.xml file and store it in a location of your choosing.
3. Open domain_home/bin/startWebLogic.sh (startWebLogic.bat on Windows) in a text editor and locate the line that starts WebLogic. It will appear similar to the following:

   ```bash
   ${JAVA_HOME}/bin/java ${JAVA_VM} ${MEM_ARGS} -Dweblogic.Name=${SERVER_NAME} -Djava.security.policy=${WL_HOME}/server/lib/weblogic.policy ${JAVA_OPTIONS} ${PROXY_SETTINGS} ${SERVER_CLASS}
   ```

4. Add the following parameter:

   ```bash
   -Dmslv.oms.config=omsconfigxml_path/oms-config.xml
   ```

   where omsconfigxml_path is the path to your customized oms-config.xml file.

   The updated line will appear similar to the following:

   ```bash
   ${JAVA_HOME}/bin/java ${JAVA_VM} ${MEM_ARGS} -Dweblogic.Name=${SERVER_NAME} -Djava.security.policy=${WL_HOME}/server/lib/weblogic.policy -Dmslv.oms.config=/Users/wls/oms-config.xml ${JAVA_OPTIONS} ${PROXY_SETTINGS} ${SERVER_CLASS}
   ```

5. Save and close the file.

Using an oms-config.xml File Stored in the oms.ear File

Every time you edit an oms-config.xml file stored in the oms.ear file, you must unpack the oms.ear file, edit the oms-config.xml file, and pack and redeploy the oms.ear file.

To configure OSM by editing the oms-config.xml file:

1. Unpack the oms.ear file as described in "Unpacking the oms.ear File".
2. Edit osm-ejb\security\META-INF\oms-config.xml to customize the parameters you want to modify. See Table 6–1 for a list of parameters in the oms-config.xml file.

**Important:** Oracle recommends that you make a backup copy of oms-config.xml before making any changes to it.

3. Save and close the oms-config.xml file.
4. Repack and redeploy oms.ear as described in "Packing the oms.ear File".

### oms-config.xml Parameters

Table 6–1 describes the parameters that can be configured in the oms-config.xml file.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Type</th>
<th>Min</th>
<th>Max</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>max_worklist_rows</td>
<td>Maximum number of rows shown in the worklist</td>
<td>integer</td>
<td>1</td>
<td>64000</td>
<td>200</td>
</tr>
<tr>
<td></td>
<td>Changing this parameter can affect system performance.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Oracle recommends that you not change this parameter to a value</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>significantly larger than the default.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>max_notification_rows</td>
<td>Maximum number notifications returned in one request in the Task Web client</td>
<td>integer</td>
<td>1</td>
<td>64000</td>
<td>200</td>
</tr>
<tr>
<td></td>
<td>Changing this parameter can affect system performance.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Oracle recommends that you not change this parameter to a value</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>significantly larger than the default.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>max_query_rows</td>
<td>Maximum number of rows returned in the Query Results page in the Task Web client</td>
<td>integer</td>
<td>1</td>
<td>64000</td>
<td>200</td>
</tr>
<tr>
<td></td>
<td>Changing this parameter can affect system performance.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Oracle recommends that you not change this parameter to a value</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>significantly larger than the default.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>worklist_page_size</td>
<td>Maximum number of rows shown on each Worklist page in the Task Web client</td>
<td>integer</td>
<td>1</td>
<td>64000</td>
<td>15</td>
</tr>
<tr>
<td>userlist_page_size</td>
<td>Maximum number of rows shown on each User List page in the Task Web client</td>
<td>integer</td>
<td>1</td>
<td>64000</td>
<td>15</td>
</tr>
<tr>
<td>notifications_page_size</td>
<td>Maximum number of lines shown on the Notifications page in the Task Web client</td>
<td>integer</td>
<td>1</td>
<td>64000</td>
<td>15</td>
</tr>
<tr>
<td>query_results_page_size</td>
<td>Maximum number of lines shown on one page in the Query Results page in the Task Web client</td>
<td>integer</td>
<td>1</td>
<td>64000</td>
<td>15</td>
</tr>
</tbody>
</table>
Table 6–1 (Cont.) Description of oms-config.xml Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Type</th>
<th>Min</th>
<th>Max</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>jdbc_fetch_size</td>
<td>JDBC fetch size</td>
<td>integer</td>
<td>1</td>
<td>64000</td>
<td>200</td>
</tr>
<tr>
<td>url_root</td>
<td>The base URL for the OSM Web applications</td>
<td>string</td>
<td>N/A</td>
<td>N/A</td>
<td>oms</td>
</tr>
<tr>
<td>email_server</td>
<td>IP address or server name of the email server used to send OSM notifications</td>
<td>string</td>
<td>N/A</td>
<td>N/A</td>
<td>127.0.0.1</td>
</tr>
<tr>
<td>email_server_port</td>
<td>Port number of the email server used to send OSM notifications</td>
<td>string</td>
<td>N/A</td>
<td>N/A</td>
<td>25</td>
</tr>
<tr>
<td>admin_email_address</td>
<td>Default email address to which to send notifications. The default value set by oms-parameter-default is ignored. Oracle recommends that you not change this parameter from the installation setting.</td>
<td>string</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>email_server_ssl</td>
<td>Enables SSL connection to the email server when set to True</td>
<td>boolean</td>
<td>N/A</td>
<td>N/A</td>
<td>True</td>
</tr>
<tr>
<td>email_server_authentication</td>
<td>Enables use of SSL authentication when connecting to the email server when set to True</td>
<td>boolean</td>
<td>N/A</td>
<td>N/A</td>
<td>True</td>
</tr>
<tr>
<td>remark_change_timeout_hours</td>
<td>Time in hours after the date and time a remark is added to an order that changes are no longer allowed to the remark.</td>
<td>integer</td>
<td>1</td>
<td>1000</td>
<td>120</td>
</tr>
<tr>
<td>attachment_file_system_name</td>
<td>Name of the T3 file service configured in WebLogic that manages order attachments for OSM</td>
<td>string</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>max_attachment_size</td>
<td>Maximum size in MB of documents attached to orders in the Task Web client Changing this parameter impacts OSM WebLogic host machine file system usage.</td>
<td>integer</td>
<td>1</td>
<td>256</td>
<td>10</td>
</tr>
<tr>
<td>database_timezone_offset</td>
<td>Offset in seconds of the database server’s time zone. The database time zone offset is used to calculate the time zone of the database, which may be different from the time zone of the application. The maximum offset is 14 hours. The default value is ignored: this parameter is defaulted to the database server’s time zone offset. It is not recommended to change this parameter from the installation setting.</td>
<td>integer</td>
<td>-50400</td>
<td>50400</td>
<td>N/A</td>
</tr>
<tr>
<td>max_read_only_field_length</td>
<td>Maximum length of a read-only field in the Order Editor</td>
<td>integer</td>
<td>1</td>
<td>512</td>
<td>30</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
<td>Type</td>
<td>Min</td>
<td>Max</td>
<td>Default</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>--------</td>
<td>------</td>
<td>------</td>
<td>---------</td>
</tr>
<tr>
<td>create_order_show_namespace</td>
<td>If set to False, no list is displayed in the Task Web client to select a namespace when creating an order.</td>
<td>boolean</td>
<td>N/A</td>
<td>N/A</td>
<td>True</td>
</tr>
<tr>
<td>load_users_from_database</td>
<td>In the XML API, you can call ListUsers. This tells the handler to either load the users from the database or from the J2EE security server (WebLogic).</td>
<td>boolean</td>
<td>N/A</td>
<td>N/A</td>
<td>False</td>
</tr>
<tr>
<td>workstream_refresh_interval</td>
<td>In the Task Web client, the number of milliseconds to wait before attempting to retrieve the next available task in a workstream.</td>
<td>integer</td>
<td>500</td>
<td>60000</td>
<td>1000</td>
</tr>
<tr>
<td></td>
<td>It is not recommended to change this parameter from the installation setting. Changing this parameter can affect system performance.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>max_workstream_retry</td>
<td>In the Task Web client, the number of retries to attempt to retrieve the next available task in a workstream before redirecting the user back to the Worklist screen.</td>
<td>integer</td>
<td>1</td>
<td>1000</td>
<td>30</td>
</tr>
<tr>
<td></td>
<td>It is not recommended to change this parameter from the installation setting. Changing this parameter can affect system performance.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>workstream_predefined_status_display_fixed</td>
<td>Controls how task completion status buttons are displayed in the workstream order editor. If set to True, predefined task statuses are displayed on a separate line in their predefined order. User-defined statuses are displayed on the next line. The display order of user-defined statuses is controlled by the model designer. If set to False, all task completion statuses are displayed in a single line. The display order of all task completion statuses is controlled by the model designer.</td>
<td>boolean</td>
<td>N/A</td>
<td>N/A</td>
<td>True</td>
</tr>
<tr>
<td>hide_dirty_order</td>
<td>Normally, after a task is completed and you return to the worklist, the task is displayed in bold italics to indicate it has been completed. If set to True, the completed task will not be displayed in the worklist.</td>
<td>boolean</td>
<td>N/A</td>
<td>N/A</td>
<td>False</td>
</tr>
<tr>
<td>disable_edit_on_server_refresh</td>
<td>If set to True, if a server refresh occurs while an order is being edited, the edit function becomes disabled.</td>
<td>boolean</td>
<td>N/A</td>
<td>N/A</td>
<td>True</td>
</tr>
</tbody>
</table>
### Table 6–1 (Cont.) Description of oms-config.xml Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Type</th>
<th>Min</th>
<th>Max</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>login_screen</td>
<td>Initial screen displayed when the Task Web client is started. Valid values include: about default home query worklist</td>
<td>string</td>
<td>N/A</td>
<td>N/A</td>
<td>worklist</td>
</tr>
<tr>
<td>auto_logout_warning_offset_minutes</td>
<td>Number of minutes prior to session timeout (auto-logout) to display a warning message to the user. A value of -1 indicates no warning is issued. The maximum value is 1440 minutes (24 hours). It is not recommended to change this parameter from the installation setting.</td>
<td>integer</td>
<td>-1</td>
<td>1440</td>
<td>5</td>
</tr>
<tr>
<td>com.mslv.oms.handler.order.cache.OrderCacheManager</td>
<td>Cache manager</td>
<td>string</td>
<td>N/A</td>
<td>N/A</td>
<td>com.mslv.oms.handler.order.cache.jboss.JBossOrderCacheManager</td>
</tr>
<tr>
<td>com.mslv.oms.util.xml.XMLHelper.DocumentBuilderFactory</td>
<td></td>
<td>string</td>
<td>N/A</td>
<td>N/A</td>
<td>oracle.xml.jaxp.JDOMDocumentBuilderFactory</td>
</tr>
<tr>
<td>custommenuaction_model_location</td>
<td>Name of the configuration file containing metadata definitions for a custom menu action</td>
<td>string</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>event_poller_interval</td>
<td>Time in seconds OSM waits before polling for new events</td>
<td>integer</td>
<td>100</td>
<td>60000</td>
<td>1000</td>
</tr>
<tr>
<td>event_poller_mutex_timeout</td>
<td></td>
<td>integer</td>
<td>0</td>
<td>60000</td>
<td>10000</td>
</tr>
<tr>
<td>com.mslv.oms.security.HandlerFuncRegistry.HandlerFactory</td>
<td>Name of the process that manages the handling of high activity orders in clustered systems</td>
<td>string</td>
<td>N/A</td>
<td>N/A</td>
<td>com.mslv.oms.security.HandlerFuncFactory</td>
</tr>
<tr>
<td>com.mslv.oms.handler.cluster.ClusteredHandlerFactory.HighActivityOrder.CollectionCycle.Enabled</td>
<td>Enables high-activity order collection cycles</td>
<td>boolean</td>
<td>N/A</td>
<td>N/A</td>
<td>True</td>
</tr>
<tr>
<td>com.mslv.oms.handler.cluster.ClusteredHandlerFactory.HighActivityOrder.CollectionCycle.InitialDelay</td>
<td>The amount of time in milliseconds to wait before the first collection cycle. This wait period allows servers to start and the system to stabilize before statistics used to determine high activity orders are collected.</td>
<td>integer</td>
<td>1000</td>
<td>3600000</td>
<td>10000</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
<td>Type</td>
<td>Min</td>
<td>Max</td>
<td>Default</td>
</tr>
<tr>
<td>---------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-------</td>
<td>-------</td>
<td>-------</td>
<td>----------</td>
</tr>
<tr>
<td>com.mslv.oms.handler.cluster.ClusteredHandlerFactory.HighActivityOrder.RequestPerSecondThreshold</td>
<td>The number of requests per second per order that must be processed for an order to be considered a high-activity order and eligible for special load balancing</td>
<td>integer</td>
<td>1</td>
<td>1000</td>
<td>50</td>
</tr>
<tr>
<td>com.mslv.oms.security.HandlerCallbackFactoryRegistry.HandlerCallbackFactory</td>
<td></td>
<td>string</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A com.mslv.oms.security.HandlerCallbackFactory</td>
</tr>
<tr>
<td>com.mslv.oms.security.OrderViewAccessProvider</td>
<td>Registers a security callback for the OrderViewAccessException exception</td>
<td>string</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>com.mslv.oms.cartridgemgmt.DeployCartridgeMDB.CartridgeDeploymentTransactionTimeout</td>
<td>Default transaction timeout interval in seconds for Oracle Communications Design Studio cartridge deployment. Design Studio can override the default value through the environment property.</td>
<td>integer</td>
<td>100</td>
<td>3600</td>
<td>600</td>
</tr>
<tr>
<td>com.mslv.oms.cartridgemgmt.cache.DeployCartridgeCache.DeployCartridgeRequestTimeToLiveMinutes</td>
<td>Default eviction timeout interval in minutes for Design Studio cartridge deployment requests to be cleaned up from the cache. Design Studio can override the default value through the environment property.</td>
<td>integer</td>
<td>5</td>
<td>360</td>
<td>30</td>
</tr>
<tr>
<td>com.mslv.oms.model.transform.OrderTransformer.ModelURL</td>
<td></td>
<td>string</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>oracle.communications.ordermanagement.cluster.BusinessRequestBalancer.OrderRelease.Timeout</td>
<td>The time in seconds to wait for a non-exclusive lock to be placed on an order A non-exclusive lock is required prior to OSM attempting to process an order. Exclusive locks are acquired by OSM when an order is about to be ejected from the order cache or when an order is being transferred from one node in an OSM cluster to a different node. Exclusive locks prevent non-exclusive locks from being acquired.</td>
<td>integer</td>
<td>200</td>
<td>500</td>
<td>200</td>
</tr>
<tr>
<td>oracle.communications.ordermanagement.cluster.BusinessRequestBalancer.ServerState.Scanning.Interval</td>
<td>Time in milliseconds to wait between scanning server states. This value is used to determine how frequently the WebLogic server is checked to see if it is in a RUNNING state prior to enabling application services such as intracluster communication or the JMS server. These services cannot be safely enabled until after the WebLogic server is in a RUNNING state.</td>
<td>integer</td>
<td>1000</td>
<td>10000</td>
<td>5000</td>
</tr>
</tbody>
</table>
Table 6–1 (Cont.) Description of oms-config.xml Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Type</th>
<th>Min</th>
<th>Max</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>oracle.communications.ordermanagement.orchestration.generation.ControlDataLocation</td>
<td>Specifies the node in an orchestration order's order template that contains control data for order items. The data at this location is automatically populated by OSM when the orchestration plan is generated.</td>
<td>string</td>
<td>N/A</td>
<td>N/A</td>
<td>ControlData/OrderItem</td>
</tr>
<tr>
<td>oracle.communications.ordermanagement.orchestration.generation.TransformedOrderItemLocation</td>
<td>Specifies the node in an orchestration order's order template that contains control data for transformed order items. The data at this location is automatically populated by OSM when the orchestration plan is generated.</td>
<td>string</td>
<td>N/A</td>
<td>N/A</td>
<td>ControlData/TransformedOrderItem</td>
</tr>
<tr>
<td>oracle.communications.ordermanagement.util.net.CatalogUriResolver.DefaultXmlCatalogsUris</td>
<td>List of URIs specifying the XML Catalogs that are used system-wide. On all OS platforms, entries are separated by a semicolon (;).</td>
<td>string</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>oracle.communications.ordermanagement.config.ModelResourceClasspath</td>
<td>List of URIs specifying either JAR files or directories containing class files that will be made available to the OSM class loader. On all OS platforms, entries are separated by a semicolon (;).</td>
<td>string</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>com.mslv.oms.automation.AutomationDispatcher.DefaultAutomationPluginDispatchMode</td>
<td>The dispatch mode to use by default for all automation plug-ins LEGACY: run in external EAR file INTERNAL: run in oms.ear</td>
<td>string</td>
<td>N/A</td>
<td>N/A</td>
<td>INTERNAL</td>
</tr>
<tr>
<td>order_editor_submit_mode_threshold</td>
<td>The response time of the Order Editor page increases with the number of nodes in an order. To avoid slow response times, the order_editor_submit_mode_threshold parameter is configured to a threshold value for node instances saved in the system. If the number of saved instances increases this threshold value, the system automatically switches from AJAX to form-submit mode when edited orders are saved or submitted for processing.</td>
<td>integer</td>
<td>1</td>
<td>999999999</td>
<td>15</td>
</tr>
<tr>
<td>bindParameter_optimization_threshold</td>
<td>If you assign a large number of tasks (or all tasks) to a workgroup, and find that the users assigned to this workgroup experience a delayed Task Client login time, set this parameter to a small value, for example, 2000</td>
<td>integer</td>
<td>N/A</td>
<td>N/A</td>
<td>99999999</td>
</tr>
<tr>
<td>is_tablelayout_height_fixed</td>
<td>If set to True, the table height equals the value of height_of_tablelayout. If set to False, the table height adjusts according to the number of rows in the table.</td>
<td>boolean</td>
<td>N/A</td>
<td>N/A</td>
<td>True</td>
</tr>
<tr>
<td>height_of_tablelayout</td>
<td>Height in pixels of the table layout</td>
<td>string</td>
<td>N/A</td>
<td>N/A</td>
<td>400px</td>
</tr>
<tr>
<td>no_of_rows_in_textarea_without_scroll</td>
<td></td>
<td>integer</td>
<td>3</td>
<td>1000</td>
<td>3</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
<td>Type</td>
<td>Min</td>
<td>Max</td>
<td>Default</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
<td>------</td>
<td>-----</td>
<td>-----</td>
<td>---------</td>
</tr>
<tr>
<td>max_no_of_rows_in_textarea_with_scroll</td>
<td>Time in seconds that OSM will wait while trying to acquire an exclusive lock on an orchestration plan. This lock is required only when OSM detects that all order components within the orchestration plan have completed and the order can complete.</td>
<td>integer</td>
<td>3</td>
<td>1000</td>
<td>3</td>
</tr>
<tr>
<td>com.mslv.oms.handler.completeorder.CompleteOrderHandlerEJB.OrchPlanLock.Timeout</td>
<td>If set to True, OSM saves a copy of every generated orchestration plan in XML format to the file orderId_orderType_orchestrationSequence_orchestrationPlanOutput.xml. <strong>Note:</strong> This parameter should be used only at the request of Oracle Support for diagnostic purposes.</td>
<td>boolean</td>
<td>N/A</td>
<td>N/A</td>
<td>False</td>
</tr>
<tr>
<td>oracle.communications.ordermanagement.orchestration_generation.DumpOrchestrationPlan</td>
<td>Determines whether spaces are normalized within XML values that are used in the results of a Lookup behavior. If set to True, the results are normalized by trimming leading and trailing spaces and replacing repeating spaces with a single space, in accordance with the <code>normalize-space</code> XPath function as defined on the World Wide Web Consortium (W3C) web site here: <a href="http://www.w3.org/TR/xpath/#function-normalize-space">http://www.w3.org/TR/xpath/#function-normalize-space</a></td>
<td>boolean</td>
<td>N/A</td>
<td>N/A</td>
<td>True</td>
</tr>
<tr>
<td>webui_order_info_pane_order_item_sort_ascending</td>
<td>The amount of time to wait before throwing the JMS response timeout.</td>
<td>integer</td>
<td>15000</td>
<td>300000</td>
<td>15000</td>
</tr>
<tr>
<td>oracle.communications.ordermanagement.cache.UserPreferenceCache</td>
<td>Specifies the name of the Coherence cache configuration to use for user preference information. By default, a local cache is used for non-clustered environments. For clustered environments a “near cache” is used to ensure changes to user preference information is automatically synchronized between cluster nodes.</td>
<td>string</td>
<td>N/A</td>
<td>N/A</td>
<td>near</td>
</tr>
<tr>
<td>oracle.communications.ordermanagement.RuleDelayTaskPoller.Interval</td>
<td>Specifies the pause time in milliseconds between consecutive executions of the rule and delay task processors.</td>
<td>integer</td>
<td>1000</td>
<td>60000</td>
<td>5000</td>
</tr>
<tr>
<td>oracle.communications.ordermanagement.RuleDelayTaskPoller.MaxRuleTaskProcessors</td>
<td>Specifies the maximum number of rule task processors</td>
<td>integer</td>
<td>1</td>
<td>50</td>
<td>1</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
<td>Type</td>
<td>Min</td>
<td>Max</td>
<td>Default</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
<td>------</td>
<td>-----</td>
<td>-----</td>
<td>---------</td>
</tr>
<tr>
<td>oracle.communications.ordermanagement.RuleDelayTaskPoller.MaxDelayTaskProcessors</td>
<td>Specifies the maximum number of delay task processors</td>
<td>integer</td>
<td>0</td>
<td>50</td>
<td>1</td>
</tr>
</tbody>
</table>
| oracle.communications.ordermanagement.security.GlobalQueryRoles | An order type/source can be queried if the user has one of the following permissions on the order type/source:  
  ■ Permission to create the order (creation task)  
  ■ Permission on at least one task  
  ■ Permission on at least one flex header where the flex header node is in that order’s order template  
  ■ At least one query view assigned.  
The workgroups where one of the previous permissions is set must all have the Search View permission assigned. Users can always see default and defaultOrchestration cartridges. For backwards compatibility, this configuration option can be used to force this function to return all order type/source in the system. The workgroup names should be separated by a semicolon, a comma, and a colon (;,:).  
This parameter makes the OSM consistent with pre 7.0.3.1 behavior. | string | N/A | N/A | N/A |
| oracle.communications.ordermanagement.order.UseUnionOfFiltersAcrossRoles | Filters are OR’ed across roles and the results are a union of data across all the results so that if a user is a member of a workgroup with privileges on an order within a cartridge and that workgroup has no filters, the user can see all orders in that cartridge. To see all the results AND’ed so that an intersection of the results is obtained, the names of the specific workgroup should be given in the configuration file separated by a semicolon, a comma, and a colon (;,:).  
This parameter has been added for backward compatibility with older versions of OSM. | string | N/A | N/A | N/A |
### Table 6–1  (Cont.) Description of oms-config.xml Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Type</th>
<th>Min</th>
<th>Max</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>oracle.communications.ordermanagement.amendment.DataEnrichmentAware</td>
<td>Determines whether OSM is aware of changes to order data from Task Web client users assigned to manual tasks or from automation plug-ins. If set to True, OSM compares revision order data to the current order including any changes from manual tasks or automation-plug-ins. If set to false, OSM compares revision orders to the last submitted order data excluding changes from manual tasks or automation plug-ins.</td>
<td>boolean</td>
<td>N/A</td>
<td>N/A</td>
<td>True</td>
</tr>
<tr>
<td>oracle.communications.ordermanagement.security.access.summary</td>
<td>Grants access to the Summary tab in the Order Management Web client, by workgroup names. Only users of specific workgroups can view the Summary tab. Separate values with commas, semicolons, or colons. Values can contain the asterisk wildcard character, where the asterisk can match a string of characters (for example, a value of user* matches workgroup names user1, user2, user3, and so on).</td>
<td>string</td>
<td>N/A</td>
<td>N/A</td>
<td>* (meaning all workgroups)</td>
</tr>
<tr>
<td>oracle.communications.ordermanagement.security.access.data</td>
<td>Grants access to the Data tab in the Order Management Web client, by workgroup names. Only users of specific workgroups can view the Data tab. Separate values with commas, semicolons, or colons. Values can contain the asterisk wildcard character, where the asterisk can match a string of characters (for example, a value of user* matches workgroup names user1, user2, user3, and so on).</td>
<td>string</td>
<td>N/A</td>
<td>N/A</td>
<td>* (meaning all workgroups)</td>
</tr>
<tr>
<td>oracle.communications.ordermanagement.security.access.orchestration-plan</td>
<td>Grants access to the Orchestration Plan tab in the Order Management Web client, by workgroup names. Only users of specific workgroups can view the Orchestration Plan tab. Separate values with commas, semicolons, or colons. Values can contain the asterisk wildcard character, where the asterisk can match a string of characters (for example, a value of user* matches workgroup names user1, user2, user3, and so on).</td>
<td>string</td>
<td>N/A</td>
<td>N/A</td>
<td>* (meaning all workgroups)</td>
</tr>
</tbody>
</table>
Table 6–1  Description of oms-config.xml Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Type</th>
<th>Min</th>
<th>Max</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>oracle.communications.order management.security.access.dependencies</td>
<td>Grants access to the Dependencies tab in the Order Management Web client, by workgroup names. Only users of specific workgroups can view the Dependencies tab. Separate values with commas, semicolons, or colons. Values can contain the asterisk wildcard character, where the asterisk can match a string of characters (for example, a value of user* matches workgroup names user1, user2, user3, and so on).</td>
<td>string</td>
<td>N/A</td>
<td>N/A</td>
<td>* (meaning all workgroups)</td>
</tr>
<tr>
<td>oracle.communications.order management.security.access.amendments</td>
<td>Grants access to the Amendments tab in the Order Management Web client, by workgroup names. Only users of specific workgroups can view the Amendments tab. Separate values with commas, semicolons, or colons. Values can contain the asterisk wildcard character, where the asterisk can match a string of characters (for example, a value of user* matches workgroup names user1, user2, user3, and so on).</td>
<td>string</td>
<td>N/A</td>
<td>N/A</td>
<td>* (meaning all workgroups)</td>
</tr>
<tr>
<td>oracle.communications.order management.security.access.activity</td>
<td>Grants access to the Activity tab in the Order Management Web client, by workgroup names. Only users of specific workgroups can view the Activity tab. Separate values with commas, semicolons, or colons. Values can contain the asterisk wildcard character, where the asterisk can match a string of characters (for example, a value of user* matches workgroup names user1, user2, user3, and so on).</td>
<td>string</td>
<td>N/A</td>
<td>N/A</td>
<td>* (meaning all workgroups)</td>
</tr>
<tr>
<td>oracle.communications.order management.security.access.data-tree</td>
<td>Controls how users see the Order Info region in the Order Management Web client, by workgroup names. The Order Info region can be seen by all users, but only users of specific workgroups can expand the order item child items. Separate values with commas, semicolons, or colons. Values can contain the asterisk wildcard character, where the asterisk can match a string of characters (for example, a value of user* matches workgroup names user1, user2, user3, and so on).</td>
<td>string</td>
<td>N/A</td>
<td>N/A</td>
<td>* (meaning all workgroups)</td>
</tr>
</tbody>
</table>
**Table 6–1 (Cont.) Description of oms-config.xml Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Type</th>
<th>Min</th>
<th>Max</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>fast_cartridge_undeploy</td>
<td>If set to <strong>True</strong>, cartridges are undeployed from OSM but cartridge metadata and order data are not purged from the database. After undeploying a cartridge, its status in the OSM database is set to <strong>UNDEPLOYED</strong>. If set to <strong>False</strong>, when cartridges are undeployed, cartridge order data and metadata are removed from the database, which can take time if there are a large number of orders.</td>
<td>boolean</td>
<td>N/A</td>
<td>N/A</td>
<td>True</td>
</tr>
<tr>
<td>oracle.communications.order_management.table-layout.threshold</td>
<td>This feature introduced a threshold system parameter above which multi-instance group nodes will automatically be displayed as a table. The system parameter will be read from the existing oms-config.xml configuration file at application startup and metadata refresh, as with current system parameters. A non-integer value or a negative value disables this feature.</td>
<td>integer</td>
<td>N/A</td>
<td>N/A</td>
<td>50</td>
</tr>
<tr>
<td>oracle.communications.order_management.table-layout.size</td>
<td>The number of rows displayed in the table in a single view (i.e. without scrolling), can be configured.</td>
<td>integer</td>
<td>0</td>
<td>50</td>
<td>10</td>
</tr>
<tr>
<td>oracle.communications.order_management.table-layout.fetch-size</td>
<td>The number of rows fetched at a time from the server is configurable; which means that not all the rows that are available for the component on the server are fetched and displayed on the client. The number of rows that are displayed on the client are just enough to fill the viewport. More rows are fetched as the user scrolls the component vertically.</td>
<td>integer</td>
<td>5</td>
<td>100</td>
<td>25</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
<td>Type</td>
<td>Min</td>
<td>Max</td>
<td>Default</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
<td>------</td>
<td>-----</td>
<td>-----</td>
<td>---------</td>
</tr>
</tbody>
</table>
| oracle.communications.ordermanagement.resource.FullScopeAccess | The level of access to system resources. This parameter has one of the following values:  
  - _strict_access_  
    This value indicates that no cartridges will have full scope access to system resources.  
  - *  
    (asterisk) This value indicates that all cartridges will have full scope access to system resources.  
  - cartridgeMnemonic,cartridgeMnemonic, etc.  
    This type of value indicates that only the listed cartridges will have full scope access. For example, if the value is:  
    OSM_Cart1:1.0.0.0.0,OSM_Cart1:2.0.0.0.0  
    it means that versions 1.0.0.0.0 and 2.0.0.0.0 of the OSM_Cart1 cartridge have full scope access to system resources, and no other cartridges have that access. | string | N/A | N/A | _strict_access_ |
| show_all_data_history_logs_for_orderdetails | Controls which events the Order Management Web Client displays on the Activity tab of the Order Details page.  
  If set to False, the Activity tab displays only events that occur after the order is created. If set to True, the Activity tab displays all events, including those that occur before the order is created.  
  If this property does not appear in the oms-config.xml file, the Activity tab behaves as though the property is set to False. | boolean | N/A | N/A | False |
| oracle.communications.ordermanagement.compliance.snapshot.output.directory | Specifies the directory that contains the system configuration snapshot files that are captured while you are running the compliance tool. | string | N/A | N/A | oms_compliance/snapshots |
| oracle.communications.ordermanagement.compliance.evaluator.output.directory | Specifies the directory that contains the evaluation results files after you run the compliance tool. | string | N/A | N/A | oms_compliance/results |
Table 6–1  (Cont.) Description of oms-config.xml Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Type</th>
<th>Min</th>
<th>Max</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>oracle.communications.ordermanagement.compliance.evaluator.snapshot.directory</td>
<td>Specifies the directory that contains the generated snapshot files that need to be evaluated. If this directory contains multiple sub-directories that contain snapshot files, the sub-directories are sorted alphabetically and the last one is used in the evaluation process. Each directory name is a timestamp of when the snapshot was captured, in the format: <strong>yyyymmdd-hhmmss</strong>.</td>
<td>string</td>
<td>N/A</td>
<td>N/A</td>
<td>osm_compliance/snapshots</td>
</tr>
<tr>
<td>oracle.communications.ordermanagement.compliance.evaluator.rule.directory</td>
<td>Specifies the directory that contains the rules that the snapshot is to be evaluated against. Directory rules run in addition to internal rules.</td>
<td>string</td>
<td>N/A</td>
<td>N/A</td>
<td>osm_compliance/rules</td>
</tr>
<tr>
<td>OrderCacheMaxEntries</td>
<td>Maximum number of orders that can be in the cache in a managed server at one time. When there is an attempt to exceed the max cache size, the order with the longest period of inactivity is forcefully ejected from the cache to make room for orders being loaded into the cache. A value of zero means the cache size is unlimited. The only way that orders can be removed from the cache when the value of this parameter is zero is when the value of the OrderCacheInactivityTimeout is reached.</td>
<td>integer</td>
<td>0</td>
<td>86400000</td>
<td>1000</td>
</tr>
<tr>
<td>OrderCacheInactivityTimeout</td>
<td>Maximum number of seconds that an order can be inactive (no attempt to access or update it) before being ejected from cache. A value of zero means there is no inactivity timeout.</td>
<td>integer</td>
<td>0</td>
<td>86400000</td>
<td>3600</td>
</tr>
</tbody>
</table>
### Table 6–1  (Cont.) Description of oms-config.xml Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Type</th>
<th>Min</th>
<th>Max</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>ClosedOrderCacheMaxEntries</td>
<td>Maximum number of orders that recently changed from an open state to a closed state, which can exist in a managed server at any time. When there is an attempt to exceed the closed order size, the recently closed order with the longest period of inactivity is forcefully ejected from cache. A typical strategy for this value is to set it to a fraction of the OrderCacheMaxEntries value. For example, the default is 50, which is 5% of the default of OrderCacheMaxEntries. This value along with the ClosedOrderCacheTimeout value is to set a number of seconds that represents the grace period that recently closed orders remain in cache, after which they are ejected.</td>
<td>integer</td>
<td>0</td>
<td>86400000</td>
<td>50</td>
</tr>
<tr>
<td>ClosedOrderCacheTimeout</td>
<td>Maximum number of seconds that recently closed orders remain in the cache after they have changed from an open state to a closed state. This allows sufficient time for post processing that might happen after an order is completed or aborted, but encourages closed orders to otherwise be ejected from cache more quickly than if they were ejected based on the cache.InactivityTimeout parameter. Typically, this value is set to a relatively low value, such as 60 seconds. To be effective, this value should be much smaller than the OrderCacheInactivityTimeout value.</td>
<td>integer</td>
<td>0</td>
<td>86400000</td>
<td>60</td>
</tr>
</tbody>
</table>
You can specify a delimited list of OACC policy xml files that the system can use to limit the concurrency of automation plug-ins. Each policy specifies which automation plug-ins are limited, the scope of the policy, and the maximum allowed concurrency level. All automation plug-ins not targeted by a policy have unlimited concurrency.

For example:

```xml
<oms-parameter>
  <oms-parameter-name>AutomationConcurrencyModels</oms-parameter-name>
  <oms-parameter-value>file:///Users/sample/automationConcurrencyModel.xml</oms-parameter-value>
</oms-parameter>
```

You can put the policy file in any reachable location and the policy file can have any name when being referenced from the oms-config.xml. You can also include OACC policy files directly in the resource folder of OSM cartridges instead of referencing them from oms-config.xml. If both the referenced oms-config.xml OACC policy file and the local cartridge OACC policy file have the same name, then the oms-config.xml policy file overrides the local version.

For more information about OACC policy files, see OSM Developer’s Guide.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Type</th>
<th>Min</th>
<th>Max</th>
<th>Default</th>
</tr>
</thead>
</table>
| AutomationConcurrencyModels   | You can specify a delimited list of OACC policy xml files that the system can use to limit the concurrency of automation plug-ins. Each policy specifies which automation plug-ins are limited, the scope of the policy, and the maximum allowed concurrency level. All automation plug-ins not targeted by a policy have unlimited concurrency. For example: <oms-parameter>
  <oms-parameter-name>AutomationConcurrencyModels</oms-parameter-name>
  <oms-parameter-value>file:///Users/sample/automationConcurrencyModel.xml</oms-parameter-value>
</oms-parameter> You can put the policy file in any reachable location and the policy file can have any name when being referenced from the oms-config.xml. You can also include OACC policy files directly in the resource folder of OSM cartridges instead of referencing them from oms-config.xml. If both the referenced oms-config.xml OACC policy file and the local cartridge OACC policy file have the same name, then the oms-config.xml policy file overrides the local version. For more information about OACC policy files, see OSM Developer’s Guide. | List of files | N/A  | N/A | N/A     |
This chapter describes how to configure the rule engine to improve performance and handle rule-processing errors for Oracle Communications Order and Service Management (OSM). In most cases, you can use the default configuration for the rule engine.

This chapter also describes how to evaluate your system’s compliance against a set of established rules to ensure your system is optimally configured for your environment.

About Configuring the Rule Engine

The rule engine evaluates rules, event delays, and timer delays to determine when to transition to the next task in a process. The engine is implemented as one or more Task Processor threads running on WebLogic servers. There are two types of task processors: the rule task processor evaluates orders at rules and the delay task processor evaluates orders at delays. You can configure additional task processors to improve performance.

Configuring Rule-Engine Error Handling

You can configure how the OSM rule engine handles errors during processing.

To configure error handling, you use SQL to set a value in the OSM database. Table 7–1 shows the configuration parameters.

The first two parameters enable you to change the behavior of the rule engine when an Oracle internal error (ORA-00600) occurs. When that error occurs during the rule execution, the rule engine immediately tries to re-execute the current rule. If the rule executes, the rule engine continues processing. If the rule does not execute, the rule engine tries to re-execute the current rule (the number of retries is dictated by the value you specify in the rule_retries parameter). If it fails, the rule engine generates an error message. You can clear the error message using the System Events tab in the OSM Administrator application.

When an error occurs during the rule or delay processing, a message is recorded as a system event and the rule or delay is marked as invalid by default. The client has an option not to invalidate the task immediately when an error occurs during processing, but after a specific number of retries. Number of retries is defined by the order_rule_threshold parameter.
### Table 7–1  Rule Engine Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>rule_delay</td>
<td>Specifies the delay between retries when an Oracle internal error (ORA-00600) occurs.</td>
<td>N/A</td>
</tr>
<tr>
<td>rule_retries</td>
<td>Determines how many times you want the rule engine to retry executing a rule before generating an event.</td>
<td>N/A</td>
</tr>
<tr>
<td>order_rule_threshold</td>
<td>Specifies how the rule engine handles processing errors. Values are:</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>■ 0 - Unlimited retries without invalidation of rule/delay (this option can waste CPU if errors constantly occur).</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ 1 - Immediately invalidates the rule after an error occurs.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ 2 or higher - Specifies how many uncleared error occurrences should disable the rule/delay. Error occurrences are counted in Total column on the System Events tab in the OSM Administrator application.</td>
<td></td>
</tr>
</tbody>
</table>

### To change value of a parameter

You change these parameters by using SQL to edit the OM_PARAMETER table in the OSM schema:

1. Connect to OSM schema using SQL*PLUS.

2. Execute one of the following:

   ■ To specify the delay between retries when an Oracle internal error occurs:

     ```
     update om_parameter
     set value = 'value'
     where mnemonic = 'rule_delay';
     commit;
     ```

   ■ To specify the number of retries when an Oracle internal error occurs:

     ```
     update om_parameter
     set value = 'value'
     where mnemonic = 'rule_retries';
     commit;
     ```

   ■ To specify how to handle processing errors:

     ```
     update om_parameter
     set value = 'value'
     where mnemonic = 'order_rule_threshold';
     commit;
     ```

### Configuring the Rule Engine for Performance

The number and type of task processors used in a given environment is determined at installation. You can also reconfigure them at a later time using the `oms-config.xml` file. After reconfiguration, restart the WebLogic server instance to apply the changes. The configuration parameters apply only to the WebLogic instance that uses the modified `oms-config.xml` file (your environment may have multiple WebLogic instances with multiple `oms-config.xml` files).
There are two types of task processors:
- Rule Task Processor, which evaluates only rules.
- Delay Task Processor, which evaluates only delays.

You may want to reconfigure the number or type of task processors as the amount of data you handle grows.

For rule task processors, the following parameters apply:
- `oracle.communications.ordermanagement.RuleDelayTaskPoller.MaxRuleTaskProcessors`. Specifies the maximum number of rule task processors; minimum value 1; maximum 50; default value is 1.
- `oracle.communications.ordermanagement.RuleDelayTaskPoller.Interval`. Specifies the pause time in milliseconds between two invocations of the rule and delay task processors; minimum value 1000; maximum value 60000; default value is 5000.

For delay task processors, the following parameters apply:
- `oracle.communications.ordermanagement.RuleDelayTaskPoller.MaxDelayTaskProcessors`. Specifies the maximum number of delay task processors; minimum value 0; maximum 50; default value is 1.
- `oracle.communications.ordermanagement.RuleDelayTaskPoller.Interval`. Specifies the pause time in milliseconds between two invocations of the rule and delay task processors; minimum value 1000; maximum value 60000; default value is 5000.

There must be at least one rule task processor running. The number of delay task processors can be 0 or above.

If there is a backlog of rule or delay tasks, you can increase the number of rule or delay task processors.

OSM will adjust the number of rule and delay task processors to use no more than 10% of the connection pool size that is configured for the WebLogic instance. The adjusted numbers are written in the managed server’s log file. If the adjusted number of rule and delay task processors does not meet your performance requirement, increase the connection pool size or decrease the parameter `oracle.communications.ordermanagement.RuleDelayTaskPoller.Interval`. 
Managing the OSM Database Schema

This chapter describes how to manage an Oracle Communications Order and Service Management (OSM) database schema.

Implementing a Strategy for OSM Information Lifecycle Management

Information Lifecycle Management (ILM) is a process for managing information through its lifecycle in a manner that optimizes storage and access. This section discusses how to implement an OSM-specific strategy for ILM.

An OSM deployment includes these database schemas:

- The core schema, which contains order cartridge metadata, order data, configuration, and other data.
- The rule engine schema, which contains logic for rule processing.
- The reporting schema, which is used for reporting.

All schemas are installed and upgraded together (although you might have to install the reporting schema manually depending on your release and patch level).

OSM provides tools to help you manage data classes rather than individual tables, so you do not need a detailed understanding of the OSM schemas. The core schema is the only schema that contains data that accumulates, ages, becomes obsolete, and eventually must be purged:

- **Cartridge metadata**: Static metadata that is populated in the OSM database when the cartridge is deployed or redeployed. This data does not grow or change when orders are created or processed. For each cartridge there is a Java EE application deployed to the OSM WebLogic domain. Cartridge metadata and the associated Java EE applications consume resources and take time to initialize on server startup.

- **Order data**: The bulk of OSM data. Storage sizing depends on order volume, order size, retention policies, and your order purge strategy. OSM supports partitioning, which helps you manage order data efficiently.

The remainder of the data includes relatively small volumes of configuration and other data that is static, is updated infrequently, or it is managed by OSM.

The following sections discuss important aspects of an ILM strategy:

- **Creating Tablespaces**: Presents options and provides recommendations for creating tablespaces for OSM schemas.

- **Using Partitioning**: Provides an overview of partitioning in OSM and discusses the benefits and pitfalls of partitioning. Oracle strongly recommends partitioning in
Creating Tablespaces

all production deployments or production test environments, particularly those with high order volumes or any volume of large or complex orders. Moreover, partitioning is required if you plan to use active-active Oracle RAC.

- **Order Purge Strategies**: Helps you decide on an order purge strategy. This is one of the most important decisions that you must make, not only before going into production but also before you start performance testing.

- **Sizing Partitions**: Discusses how to size partitions for order data. Partition sizing depends on your order purge strategy.

- **Cartridge Management Strategy**: Recommends a strategy for managing cartridges.

- **Online vs. Offline Maintenance**: Gives a brief overview of which maintenance operations can be performed online.

Creating Tablespaces

The OSM installer prompts you to select the following permanent database tablespaces:

- The default tablespace for all OSM schemas.

- Model Data and Model Indexes tablespaces: Used mainly for cartridge metadata and configuration data.

- Order Data and Order Indexes: Used for order data and auxiliary order-related tables.

You can choose different tablespaces or a single tables pace. Typically model data and indexes are separate from order data and indexes.

If your schema is partitioned and you choose the same tablespace for order data and order indexes, the OSM installer creates local index partitions with tablespace DEFAULT, which means that local index partitions are stored in the same tablespace as table partitions.

If your schema is partitioned, you can also create new table partitions in different tablespaces for increased administration and availability, for example on a rotation basis. If a tablespace is damaged, the impact and restoration effort could be limited to one or just a few partitions. See "Adding Partitions (Online or Offline)" for more information.

Oracle recommends the following:

- Create tablespaces dedicated to OSM, so that OSM performance and availability are not affected by other applications, for example due to I/O contention or if a tablespace must be taken offline. Store the datafiles of these tablespaces on different disk drives to reduce I/O contention with other applications.

- Create locally managed tablespaces with automatic segment space management by specifying EXTENT MANAGEMENT LOCAL and SEGMENT SPACE MANAGEMENT AUTO in the CREATE TABLESPACE statement. Both options are the default for permanent tablespaces because they enhance performance and manageability.

- Configure automatic database extent management by using the AUTOALLOCATE clause of the CREATE TABLESPACE statement. This is the default. For production deployments, avoid UNIFORM extent management for OSM order data and indexes because the volume of data varies widely from table to table.
If you use smallfile tablespaces, do not create hundreds of small datafiles. These files need to be checkpointed, resulting in unnecessary processing. Note that Oracle Database places a limit on the number of blocks per datafile depending on the platform. The typical limit is 222-1, which limits the datafile size to 32GB for 8k blocks.

Additional considerations if you use bigfile tablespaces:

- If data is stored in bigfile tablespaces instead of traditional tablespaces, the performance of database opens, checkpoints, and DBWR processes should improve. However, increasing the datafile size might increase time to restore a corrupted file or create a new datafile. You can mitigate the risk of corruption by using multiple tablespaces for partitions, for example on a rotating basis.

- Bigfile tablespaces are intended to be used with Automatic Storage Management (Oracle ASM) or other logical volume managers that supports striping or RAID, and dynamically extensible logical volumes.

- Avoid creating bigfile tablespaces on a system that does not support striping because of negative implications for parallel query execution and RMAN backup parallelization.

- Using bigfile tablespaces on platforms that do not support large file sizes is not recommended and can limit tablespace capacity.

For more information about managing tablespaces, see Oracle Database Administration Guide.

Using Partitioning

OSM database partitioning enhances the performance, manageability, and availability of data in an OSM deployment.

The OSM installer allows you to choose whether or not to enable partitioning.

**Important:** Oracle strongly recommends partitioning in all production deployments or production test environments, particularly those with high order volumes or any volume of large or complex orders. If you choose not to partition your OSM schema, it could be expensive to later reverse your decision. Changing a non-partitioned schema that has accumulated a large volume of data to a partitioned schema involves time-consuming and resource intensive export and import.

When you select the partitioning option, OSM automatically partitions tables that accumulate order-related information using range partitioning based on OSM order ID ranges. Figure 8–1 shows this aspect of OSM partitioning.
The `OM_ORDER_HEADER` table stores a synopsis for each order, such as the order Id, priority, state, milestone timestamps, and so on. This table is range-hash partitioned by order Id. More precisely:

- The non-inclusive upper bound of each range partition is an order Id. For example, if the upper bound of the first partition is 1,000,001 and partitions are sized to contain 1,000,000 order Ids each, the first partition contains orders with an order Id between 1 and 1,000,000, the next partition contains orders with an order Id between 1,000,001 and 2,000,000, and so on.
- Hash sub-partitioning reduces I/O contention. In production deployments, range partitions typically have 16, 32, or 64 sub-partitions.
- Range partition names are generated by OSM and they include the partition upper bound. For example, the upper bound of partition `P_00000000001000001` is 1000001. Sub-partition names are generated by Oracle Database (for example, `SYS_P2211001`).

The rest of the tables that accumulate order data are equipartitioned with `OM_ORDER_HEADER`. They are either range-hash partitioned or reference partitioned.

For more information about the different types of partitioning, see Oracle Database VLDB and Partitioning Guide.

**Benefits of Partitioning**

Partitioning your OSM schema allows you to subdivide tables and indexes into smaller segments. This provides the following benefits:

- **Improved Manageability**
- **Increased Availability**
- **Increased Concurrency**
- **Support for Active-Active Oracle RAC**
- **Increased Query Performance** (for certain queries)

**Improved Manageability**

Improved manageability is the result of partition independence, plus managing smaller segments is easier, faster and less resource intensive. The benefits increase with
the schema size, because partitioning allows you to divide large tables and indexes into smaller more manageable segments.

You can purge several weeks’ worth of order data in minutes by dropping or purging a partition, without affecting other partitions. You can set up routine purges of older partitions containing obsolete and complete orders, while creating and bringing on-line new partitions as existing partitions fill up.

Data Definition Language (DDL) operations on tables and indexes are less resource intensive and less likely to fail if they are performed one partition at a time. For example, consider a 128 GB non-partitioned index and a partitioned index of the same size with 32 partitions. The partitioned index can be rebuilt one partition at a time (32 transactions), whereas rebuilding the non-partitioned index is a single transaction that requires 32 times more free space. If the rebuild fails, the entire transaction is rolled back.

**Increased Availability**
Partitioning increases availability mainly by reducing downtime in the event of error. For example, the time to recover from a corruption in a large table could be reduced significantly if that table was partitioned and the corruption was isolated to a single partition.

**Increased Concurrency**
Hash sub-partitioning increases concurrency by reducing I/O contention, specifically by spreading DML statements (data modifications) over several physical sub-partitions. Contention is usually manifested as “buffer busy” waits in Automatic Workload Repository (AWR) reports.

Note that range partitioning does not help with contention because new orders are created in the same range partition. The only exception is if you use active-active Oracle RAC, in which case order creation is spread over two or more range partitions.

**Support for Active-Active Oracle RAC**
If you use active-active Oracle RAC, range partitioning is critical for good performance. Therefore, active-active Oracle RAC is supported only if your OSM schema is partitioned.

If you use a single instance database, new orders are assigned order Ids and are all stored in the same partition. When that partition is exhausted, new orders are created in the next partition, and so on. To avoid conflicts in order Id generation, each OSM server running in a WebLogic cluster is assigned one or more “slots” and generates only order Ids mapped to its slots based on a proprietary algorithm. For example, in a cluster with two equally weighted managed servers, each server might generate every other order Id. (The order Id generation algorithm is internal and may be different from release to release.)

If you configure OSM to use Oracle RAC in active-passive mode or a single node of Oracle RAC, order Ids are generated as described above. However, if you configure OSM to use Oracle RAC in active-active mode with N nodes, new orders are created concurrently in N partitions. The goals are load balancing and increased scalability and performance. These are achieved by dividing managed servers into N groups, so that each group interacts with a single Oracle RAC instance.

**Figure 8–2** shows an example with four managed servers divided into two groups. Each group is configured with a different multi data source in failover mode. However, the primary data source in each multi data source points to a different Oracle RAC instance in an alternating fashion.
Figure 8–3 shows that each group generates order IDs from a different range, so that the orders created by each group are stored in different partitions, for example, P_0000000000010001 and P_0000000000020001. In this way:

- All database transactions on an order that take place on a single managed server also take place on a single database instance. This minimizes cluster waits by avoiding buffer cache transfers among database instances.
- The workload is balanced across Oracle RAC nodes and WebLogic Managed Servers.

Figure 8–2 OSM Data Source Configuration for Active-Active Oracle RAC

Figure 8–3 OSM Order ID Generation for Active-Active Oracle RAC

The downside of creating orders concurrently in N partitions is that some order IDs are skipped, which causes partitions to be only partially filled when they are considered exhausted. For example, Figure 3 shows that MS01 and MS03 skip order IDs 2, 6, 10, 14, and so on. This is because these are mapped to slots owned by MS02 and MS04. However, MS02 and MS04 do not generate those order IDs because they generate order IDs from a different range. As a result, each partition is only half-full when it is exhausted.
The overall size of order Id gaps depends on the number of Oracle RAC nodes, regardless of how the load is balanced across those nodes. For example, when you remove managed server MS4 from the cluster of the previous example, so that each managed server processes 1/3 of the load, the managed servers are still divided into two groups. This means that partition P_00000000000100001 contains 2/3 of the order Ids and P_00000000000200001 contains the remaining 1/3. Thus, when P_00000000000100001 is exhausted, it will be 1/3 empty. Because MS2 skips slots assigned to MS1 and MS3, its partition will be exhausted at about the same time and it will be 2/3 empty. Although the two Oracle RAC nodes are not balanced (they process 2/3 and 1/3 of the load each), on average both partitions are half empty.

In summary, if you switch from a single database to N-node active-active Oracle RAC, the number of partitions increases N-fold, whereas the actual number of order Ids stored in a partition decreases N-fold. Storage consumption is about the same.

For more information, refer to the OSM High-Availability Guidelines and Best Practices in the OSM Installation Guide.

Increased Query Performance

OSM query performance benefits the least from partitioning because queries are designed to use indexes and return quickly. In fact, a very large number of physical partitions could negatively effect query performance, as discussed in "Pitfalls of Partitioning." However, there are the following performance benefits:

- Many indexes are based on order Id and are therefore right-handed. Partitioning alleviates imbalances by dividing indexes into smaller segments.
- Queries that perform index range scans can be parallelized if the index is partitioned. Parallel query execution could dramatically improve the performance of maintenance operations, reporting, and ad-hoc user searches.

Pitfalls of Partitioning

Tables that store order data are equipartitioned with OM_ORDER_HEADER. The rest of the tables are not partitioned. Therefore, the number of physical partitions in an OSM schema is at least T x R x H, where T is the number of partitioned tables, R is the number of OM_ORDER_HEADER range partitions, and H is the number of hash sub-partitions per range partition (excluding LOB index partitions). For example, an OSM schema with 48 order tables, 10 OM_ORDER_HEADER range partitions, and 32 hash sub-partitions has at least 15360 physical partitions. If you let the number of physical partitions increase unchecked, you are likely to run into the performance problems discussed below, even if the space used by most partitions is very small. It is recommended that you review the "Sizing Partitions" section for sizing guidelines.

Order Search Performance

The majority of OSM queries return quickly because they perform index unique scans or range scans on a single partition. Most query access is based on order Id, which is the partitioning key, and a database technique called partition pruning that narrows access to a single or a small subset of partitions. Partitioning does not benefit such queries because the performance increase achieved by scanning smaller index partitions is negligible. In contrast, a large number of partitions could have a negative impact on queries that iterate over all or a large subset of local index partitions. This happens when order Id is not part of the query criteria and therefore partition pruning cannot be used. In order search, probing a large number of small index segments is slower than probing a small number of large index segments.
Purge Performance
A very large number of partitions could significantly slow down partition purge. Experience shows that the tipping point is around 300,000 physical partitions, although this varies depending on the specific OSM installation environment.

The time to purge a partition using EXCHANGE PARTITION operations depends on the number of hash sub-partitions. For example, if you decrease the number of sub-partitions from 64 to 32, the duration of the EXCHANGE PARTITION stage of the purge decreases to nearly half.

A partitioned table is considered in the library cache as one object, regardless of the number of partitions. Partition purge operations use DDL statements, which invalidate the cursors associated with the underlying partitioned tables. When a cursor is re-parsed, all the partition metadata for the underlying tables must be reloaded and the amount of time increases with the number of partitions. This is less of an issue when you drop a partition, because the DROP PARTITION statement is cascaded. However, partition purge also uses EXCHANGE PARTITION, which is not cascaded in 11g. A partition purge executes several exchange operations per reference-partitioned table, causing repeated metadata reloads that could significantly slow down the purge (for example, from minutes to hours).

Shared Pool
Oracle Database stores partitioning metadata in the data dictionary, which is loaded in the row cache of the shared pool. A very large number of partitions could stress the shared pool. If the shared pool is undersized, you could run into ORA-4031 errors (unable to allocate shared memory), especially while purging partitions.

Development and Testing Environments
Starting with Oracle Database 11.2.0.2, the initial extent size for partitioned tables is 8 MB. If you have many hash sub-partitions, partitioned tables can consume a lot of space even if they contain very little data. For example, even with deferred segment allocation, injecting a few orders could quickly expand a table with 64 sub-partitions to 512 MB. Although this is not typically an issue for production environments, which should already have sufficient storage available, it could be an issue in development or testing environments with limited storage capacity. In such environments, you can use a smaller number of sub-partitions (for example 4), or use a tablespace with uniform extent allocation and a small extent size (for example, 64 KB).

Using the OM_ORDER_NODE_ANCESTRY Table
OM_ORDER_NODE_ANCESTRY is a table that stores the hierarchy of order group nodes. The table improves the efficiency and response time of worklist and order search queries, mainly for cartridges that have multi-instance subprocesses and a large number of flexible headers. For more information about parallel processes and multi-instance subprocesses, see the topic about understanding parallel process flows in OSM Concepts.

The downside of enabling the OM_ORDER_NODE_ANCESTRY table is increased CPU usage for order creation and updates, increased order creation response time, and most importantly increased disk usage. Specifically, OM_ORDER_NODE_ANCESTRY is one of the largest tables in OSM. It is often responsible for more than 20% of the space, depending on the depth of order templates, especially for large orders, such as O2A. Therefore, this table is disabled by default. For more information, see "Performance Recommendations for the OM_ORDER_NODE_ANCESTRY Table".
An Oracle database package called OM_ORDER_NODE_ANCESTRY_PKG contains the stored procedures that allow you enable and disable the OM_ORDER_NODE_ANCESTRY table.

**Note:** If you deploy cartridges with multi-instance subprocesses and are considering running OSM with the OM_ORDER_NODE_ANCESTRY table disabled, you must evaluate factors such as the ancestry depth in the master order template and the number of flexible headers, which could impact performance in the UI worklist and search results.

---

**Performance Recommendations for the OM_ORDER_NODE_ANCESTRY Table**

The configuration of your OSM cartridges determines whether you run OSM with the OM_ORDER_NODE_ANCESTRY table enabled or disabled.

Table 8–1 shows the performance implications of running different cartridges in OSM with the OM_ORDER_NODE_ANCESTRY table enabled or disabled.

**Table 8–1 Performance Implications of the OM_ORDER_NODE_ANCESTRY Table**

<table>
<thead>
<tr>
<th>OSM Solution</th>
<th>OM_ORDER_NODE_ANCESTRY Table Status</th>
<th>Performance Implications</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cartridges that do not require multi-instance subprocesses</td>
<td>Disabled</td>
<td><strong>Positive impact:</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ Saves CPU time</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ Reduces order creation time</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ Improves throughput</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ Reduces OSM schema disk storage</td>
</tr>
<tr>
<td>Cartridges that require multi-instance subprocesses</td>
<td>Enabled</td>
<td><strong>Positive impact:</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ Improves response time when users retrieve worklist tasks and search orders</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Negative impact:</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ Increases CPU time</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ Increases order creation time</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ Degrades throughput</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ Increases OSM schema disk storage</td>
</tr>
<tr>
<td></td>
<td></td>
<td>In this case, consider compressing the ancestry table. For more information about Oracle advanced compression, see Oracle Technology Network. Note that compression has the following negative impact:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ Further increases order creation time</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ Increases SQL database CPU per execution (INSERTs)</td>
</tr>
</tbody>
</table>
Using the OM_ORDER_NODE_ANCESTRY Table

Table 8–1 (Cont.) Performance Implications of the OM_ORDER_NODE_ANCESTRY Table

<table>
<thead>
<tr>
<th>OSM Solution</th>
<th>OM_ORDER_NODE_ANCESTRY Table Status</th>
<th>Performance Implications</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cartridges that require multi-instance subprocesses</td>
<td>Disabled</td>
<td>Positive impact:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>▪ Saves CPU time</td>
</tr>
<tr>
<td></td>
<td></td>
<td>▪ Reduces order creation time</td>
</tr>
<tr>
<td></td>
<td></td>
<td>▪ Improves throughput</td>
</tr>
<tr>
<td></td>
<td></td>
<td>▪ Reduces OSM schema disk storage</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Negative impact:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>▪ Increases the response time when users retrieve worklist tasks and search orders</td>
</tr>
<tr>
<td></td>
<td></td>
<td>In this case, Oracle recommends:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>▪ Avoiding deep order template node hierarchies</td>
</tr>
<tr>
<td></td>
<td></td>
<td>▪ Eliminating unnecessary flexible headers</td>
</tr>
</tbody>
</table>

Enabling the OM_ORDER_NODE_ANCESTRY Table

When the OM_ORDER_NODE_ANCESTRY table is enabled, OSM populates the OM_ORDER_NODE_ANCESTRY table with data and uses queries on this table to support UI worklist and order searches. Running OSM in this mode is effective for new order id blocks. A new block is allocated when the current partition where new orders are created (known as the active partition) is exhausted.

Order ids are stored in the OM_ORDER_ID_BLOCK table. In this table, a column called ANCESTRY_POPULATED_UP_TO indicates the last order id in the block of order ids that has data in the OM_ORDER_NODE_ANCESTRY table.

An active order id block can be split logically, as in the following example:

- An order id block contains order ids from 0 to 100000. The order id block is NOT split yet and all order ids in this block contain ancestry data.
- An order id block is split. Orders ids between order id 0 and 2000 have ancestry data. Orders between order id 2001 and 100000 do not have ancestry data.

A block of order ids is active if it is the latest block for the current database instance (DBINSTANCE). The previous blocks for the database instance are inactive blocks.

When users retrieve worklist tasks or search for orders, OSM uses the data in order id blocks to determine if queries are run against the OM_ORDER_NODE_ANCESTRY table (old queries) or the OM_ORDER_INSTANCE table (new queries).

Note: Ancestry data is used only if the cartridge includes multi-instance tasks (pivot nodes).

You might need to switch several times between running OSM with the OM_ORDER_NODE_ANCESTRY table enabled and disabled. The following example scenarios illustrate circumstances that might necessitate switching between the two modes.

Scenario 1: Introducing multi-instance subprocess entities (enable, disable, enable)

1. You have upgraded OSM to a later version that includes this functionality. OSM continues to run with the OM_ORDER_NODE_ANCESTRY table enabled.
2. Because of large volumes of orders, you determine that OSM cartridges do not use multi-instance subprocesses and decide to disable the OM_ORDER_NODE_ANCESTRY table.

3. Some time later, you introduce multi-instance sub-process entities (for example, OSM needs to run a sub-process for each of the multiple addresses a customer has) by redeploying existing, or deploying new, cartridges.

4. You then determine that the worklist demonstrates performance degradation and decide to re-enable the OM_ORDER_NODE_ANCESTRY table.

**Scenario 2: Eliminating multi-instance subprocess entities (disable, enable, disable)**

1. You install the latest release of OSM, which includes this functionality. The OM_ORDER_NODE_ANCESTRY table is disabled.

2. You deploy a cartridge that uses multi-instance subprocesses, and leave the table disabled because performance test results are satisfactory.

3. You then determine that the worklist demonstrates performance degradation and decide to enable the OM_ORDER_NODE_ANCESTRY table.

4. Some time later, you redeploy updated cartridges so that all multi-instance subprocesses are eliminated. You then disable the OM_ORDER_NODE_ANCESTRY table.

You can run this procedure when OSM is online or offline.

To enable the OM_ORDER_NODE_ANCESTRY table:

1. Log in to SQL*Plus as the OSM core schema user.

2. Run the following command:

   ```
   begin
   om_order_node_ancestry_pkg.enable_ancestry_table;
   end;
   ```

---

**Disabling the OM_ORDER_NODE_ANCESTRY Table**

Running OSM with the OM_ORDER_NODE_ANCESTRY table disabled is suitable if you are deploying cartridges that do not include multi-instance subprocesses. When you run OSM with the table disabled, the OM_ORDER_NODE_ANCESTRY table is not populated and hierarchical queries (for cartridges with multi-instance subprocesses) that are run using UI worklist or search functionality return ancestry data from the OM_ORDER_INSTANCE table.

You must disable the OM_ORDER_NODE_ANCESTRY table when OSM is offline because the procedure uses the last order id to split the block of order ids into two parts: populated and non-populated. For example, if the current order id is 100 and the last order id in the active block is 10000:

- [1...100] is logically split into:
  - [1...100]: order ids with populated ancestry
  - [101...10000]: order ids with non-populated ancestry

To disable the OM_ORDER_NODE_ANCESTRY table:

1. Log in to SQL*Plus as the OSM core schema user.

2. Take the OSM server offline. For more information about stopping OSM, see "Starting and Stopping OSM".
3. Run the following command:

```sql
begin
    om_order_node_ancestry_pkg.disable_ancestor_table;
end;
```

**Note:** You can run the disable OM_ORDER_NODE_ANCESTRY table procedure only once on a single block of ids because the current block of ids can be split only once.

---

## Order Purge Strategies

The database size increases as orders enter the system. If left unchecked, the size becomes difficult for database administrators to manage and you might eventually run out of space. A purge strategy is an important decision to make before going into production. If you choose a continuous order purge strategy, you should incorporate it in your order load tests. Partition sizing depends on the purge strategy.

OSM supports these purge strategies:

- Partition-Based Order Purge Strategy
- Row-Based Order Purge Strategy
- Hybrid Purge Strategy

### Partition-Based Order Purge Strategy

The `om_part_maintain` PL/SQL package allows you to drop and "purge" partitions. These operations are based on efficient DDL statements that modify the table structure, such as DROP PARTITION and EXCHANGE PARTITION. They are designed to reclaim large amounts of space as fast and efficiently as possible. You can purge several weeks' worth of order data in minutes. However, they usually require downtime and a regular maintenance schedule.

When a partition is dropped, all its data is deleted and space is immediately reclaimed. Populated partitions can be dropped offline only. Empty partitions can be dropped either offline or online.

Often a partition cannot be dropped because it contains orders that do not satisfy the purge criteria (for example, closed orders that are still in retention and open orders). In this case, you can use "partition purge". This operation is more versatile than dropping because it allows you to retain some orders and consolidate multiple partitions. In addition, if all orders in a partition are closed and satisfy the purge criteria, you might be able to purge it online.

The following sections provide additional information about this strategy:

- **Partition Purge Example**: Provides an example of the strategy.
- **Advantages and Disadvantages of Partition-Based Purge**: Discusses the pros and cons of partition-based purge.
- **Sizing Range Partitions for Partition-Based Order Purge**: Provides partition sizing guidelines.
- **Using Partition-Based Order Purge**: Partition-based order purge is discussed in more detail.
Partition Purge Example

The following figures show a partition purge example over a period of 14 weeks with three maintenance windows.

- The purge frequency is biweekly.
- Each partition is sized to contain about 2 weeks' worth of orders.
- The order retention period is 4 weeks (after closure).
- Purge performance tests show that the period of downtime is acceptable if less than 2% of the orders in the partition are retained (including both open orders and closed orders in retention) but the system can tolerate up to 5%. The statistics (or estimates) suggest that 98% of the orders close within 3 weeks.

For this example, to purge an exhausted partition you must wait for at least 7 weeks (3 weeks for 98% of the orders to close and 4 weeks retention). Because the partition size is equal to 2 weeks of orders, you can schedule the first maintenance window after 9 weeks in production. After that, maintenance is biweekly. Before going into production, you should also add enough partitions to store orders until the first maintenance window.

Figure 8–4 shows the first maintenance window. P1 is purged and less than 2% of its orders are retained, including closed orders that are still in retention. Because a partition contains 2 weeks' worth of orders, you also add at least one partition (only P6 shown).

Partitions P2, P3, P4, and P5 are not purged:

- It is not cost-effective to purge P2: It contains a high percentage of open orders and closed orders that are still in the retention period (for example 15%).
- It is not cost-effective to purge P3: The second half of P3 is within the retention period, so it contains a very high percentage of open orders, and closed orders that are still in the retention period (for example 75%).
- P4 and P5 are ineligible for purge: All orders are either open or within the retention period. In addition, P5 is where new orders are created (shown half full).

In other words, 3.5 partitions are not purged. If you follow an uninterrupted biweekly schedule, the number of partitions that are not purged is always about 3.5.

Figure 8–4  Partition Purge Maintenance, Window 1
Figure 8–5 shows the second maintenance window (after 2 weeks). Because a partition contains 2 weeks’ worth of orders, P5 is exhausted and P6 is half full. As in the previous maintenance, it is cost-effective to purge only one partition that has not yet been purged, which is P2 in this example. You also add at least one partition (not shown). Notice that the number of partitions that are not purged is 3.5 again.

Figure 8–6 shows the third maintenance window. As in the previous maintenance, it is cost-effective to purge only one partition (P3) that has not yet been purged, and 3.5 partitions are not purged. This time, however, the previously purged partitions P1 and P2 are purged again and consolidated with P3 (3-to-1 consolidation). The number of orders retained with this consolidation is less than 3.2%, which is less than the 5% tolerance limit. Periodic consolidation is recommended to maintain a low number of partitions and maximize reclaimed space.

Advantages and Disadvantages of Partition-Based Purge
Advantages of the partition purge strategy:
Order Purge Strategies

- The ability to reclaim large amounts of space (for millions of orders) in a very short period of time (normally measured in minutes). Partition drop and purge operations are based on efficient DDL statements, such as DROP PARTITION and EXCHANGE PARTITION.

- If your database and storage are well-tuned for order processing, you do not need extra hardware for purging (CPU, RAM and I/O bandwidth). Because the purge happens when OSM is offline and not processing orders, all the system resources are available for purging.

- You can place partitions on different tablespaces for increased administration and availability. For example, if a tablespace is damaged, the impact and restoration effort could be limited to one or just a few partitions, possibly an old partition with few open orders.

- You can choose the correct partition size to facilitate administration and maintenance according to your needs.

Disadvantages of the partition purge strategy:

- This strategy requires downtime. The amount of downtime depends on the purge frequency, purge performance, and several additional factors discussed in "Sizing Range Partitions for Partition-Based Order Purge." In general, with relatively little extra storage you can reduce the frequency of purge cycles considerably. For example, you might be able to switch from biweekly to monthly maintenance with only a 20% increase in storage. This is discussed in detail in the "Purge Frequency" section. If you require 24x7 operations, consider row-based order purge.

- If you have very high volumes of orders and you cannot afford frequent downtime, the large storage size could become hard to manage.

- This strategy does not work well if you have a mix of long-lived orders and a relatively high-volume of short-lived orders. Because the high-volume orders reside together with long-lived orders, the latter dictate the purge strategy. Unless long-lived orders are a fraction of the total volume, it might not be cost-effective to purge a partition soon after all short-lived orders are closed. (This is because retaining a large number of long-lived orders would increase considerably the purge time and therefore the downtime.) Also, as explained in "Pitfalls of Partitioning," if you let the number of partitions increase significantly, performance of partition purge operations suffers. In this case, consider a hybrid purge strategy or row-based order purge.

To mitigate the disadvantages of this strategy, choose the partition size carefully and adjust it as conditions change. As a rule of thumb, size your partitions to contain as many orders as will be purged in one purge maintenance window. For sizing guidelines, refer to the "Sizing Range Partitions for Partition-Based Order Purge" section.

Row-Based Order Purge Strategy

If you cannot take periodic downtime to purge orders, consider row-based order purge, which is implemented by the om_new_purge_pkg package. Because you can run order purge online, it allows for continuous business operations. You can also use it as an auxiliary purge strategy, as discussed in "Hybrid Purge Strategy," and for ad-hoc deletes (for example, if you need to purge and resubmit an order).

Because order purge uses deletes instead of DDL statements, you can run order purge online with minimal contention with normal order processing:
Order processing can take place concurrently because all indexes remain usable, foreign keys remain enabled and validated, and only the orders that satisfy the purge criteria are affected (as opposed to partition-based purge, which moves around orders that do not satisfy the purge criteria).

Contention is minimal because deletes acquire row level locks (as opposed to partition-based purge, which uses DDL operations that acquire object level locks).

As row-based order purge allows OSM to stay online and perform normal order processing while purge operations are taking place concurrently, this increases the total workload of the system. The database and storage I/O must be sized to handle the additional workload of deleting orders concurrently with order processing. For sizing guidelines refer to “Sizing Range Partitions for Row-Based Order Purge.”

To use order purge as your main purge strategy (or in a hybrid strategy), schedule it to run as a background activity that continuously frees space to be reused by new orders. Ideally you should free space at the same rate it is consumed. The space freed by deletes in a table or index segment does not lower the high water mark (the boundary between used and unused space) of that segment. It can be reused by inserts into the same segment but it is not released to the tablespace. Therefore the primary purge target should be the partition(s) where new orders are created. (In the case of active-active Oracle RAC with N nodes, orders are created on N partitions concurrently.) For example, you could run it once daily during the lowest volume period with high parallelism, several times a day for shorter periods or continuously throughout the day with low parallelism.

Partitions should be sized with a wide range of order Ids to store several months' worth of orders. Partition sizing depends on the retention policy, the order life time, and how you reclaim the space occupied by old partitions. This leads to the following two variations of this strategy:

- **Zero downtime**: If you have a strict 24x7 requirement, you could delete orders from old partitions until they are empty, which enables you to drop them online. The downside is that it might be a long wait before you can fully reclaim that space by dropping the empty partitions, especially if you have orders that remain open for several months or even years (as mentioned earlier, deletes do not lower the segment high water mark and the space freed by deletes cannot be used for new orders because new orders are created on the latest partition).

- **Infrequent maintenance** (for example, once a year): If you have a near 24x7 requirement with occasional maintenance windows, you could use those windows to drop or purge old partitions offline.

The following sections provide additional information about this strategy:

- **Row-Based Order Purge Example**
- **Advantages and Disadvantages of Row-Based Order Purge**
- **Using Row-Based Order Purge**

**Row-Based Order Purge Example**

Figure 8–7 show a row-based order purge example over a period of several months. The retention period is 2 weeks, and the maximum lifetime of an order is 5 weeks. Closed orders are deleted daily, as they come out of retention. Some of the orders close within a day, and the order aging curve allows order purge to target an additional 25% of the orders each week for any week prior to the past two weeks.

This means that by the end of week 6, all orders created in the first week have been purged. By the end of week 7, all orders created in the first two weeks have been
purged. This pattern continues until the partition is exhausted on the N-th week. Then it repeats for the next partition.

Figure 8–7  Partition Lifetime Using Row-Based Order Purge

Advantages and Disadvantages of Row-Based Order Purge

The key advantage of this strategy is that you can run it online and therefore it can support 24x7 operations.

The disadvantages of row-based order purge:

- Row-based order purge is the least efficient way to purge because delete is an expensive DML statement. Deleting row by row is CPU intensive and consumes I/O bandwidth, so you must plan for spare capacity. Normally this is not an issue because you size your hardware for peak volume. However, if you do not have enough idle or low-volume periods for purge to catch up with new order creation, you will have to add hardware (for example, CPUs and storage links).

- The space freed by deletes in a table or index segment does not lower the high water mark of that segment. It can be reused by inserts into the same segment but it is not released to the tablespace. Therefore you must run order purge as frequently as possible (at least once a day) to free space at the same rate it is consumed by new orders, which has an objective of restraining the growth of the high water mark.

- If order purge runs continuously, it might be harder to manage. You must be prepared to deal with unexpected load spikes and adjust the order purge schedule if load patterns change. And you might have to disable order purge for occasional maintenance tasks or batch jobs.

- Row-based order purge makes it difficult troubleshoot performance issues. If you run into performance issues with normal order processing, you might have to stop order purge to reproduce the problem in isolation. This is not an issue if you configure a purge job class that runs on a dedicated database service (see "purge_job_class").

Hybrid Purge Strategy

You can use both partition purge and order purge, and realize the benefits of both strategies. For example, consider a hybrid strategy if you have a mixed load of
long-lived orders and short-lived orders, and you can afford periodic downtime. In this example:

- Use order purge to delete short-lived orders aggressively from the partition(s) where new orders are created. The space freed by deletes will be reused by new orders. The goal is to restrain the growth of the high water mark (the boundary between used and unused space) in table and index segments. This will allow you to have larger partitions and less frequent purge maintenance.

- Drop/purge partitions periodically (for example, once a month) to purge long-lived orders and the remaining short-lived orders. Follow the partition sizing guidelines for partition-based order purge, which are discussed in "Sizing Range Partitions for Partition-Based Order Purge."

**Cartridge Management Strategy**

There are three main components to a deployed cartridge:

- The Java EE application that is automatically generated by Design Studio and deployed to the OSM WebLogic domain.
- The cartridge metadata that is populated in the OSM database when the cartridge is deployed or redeployed.
- The order data populated in the OSM database whenever an order is created and as it is being processed.

Cartridge Java EE applications consume memory resources and take time to initialize on startup. Cartridge metadata consumes little space in the database. However, it consumes memory resources, takes time to load into the OSM server on startup, and is re-loaded when cartridges are deployed.

Follow these guidelines to reduce the memory footprint and startup time of OSM, and to deploy and undeploy cartridges quickly online:

- Undeploy obsolete cartridges from time to time to reduce the memory footprint and startup time of OSM.

- Use Fast Undeploy to undeploy cartridges instead of conventional undeploy. Fast Undeploy allows you to undeploy cartridges quickly online by undeploying the Java EE application only. Instead of purging cartridge metadata, it sets the cartridge status in the database to UNDEPLOYED. Fast Undeploy also offloads purging of order data to your order purge strategy, which does not need to distinguish between deployed and undeployed cartridges.

- Consider purging metadata of an undeployed cartridge only if it has no associated orders. In general, defer purging metadata of undeployed cartridges as frequently as possible. Although cartridge metadata consumes little space in the database, it is expensive to purge it and it must be purged offline.

For more information see "Managing Cartridges."

**Sizing Partitions**

The following values, which you enter on the Database Schema Partition Information installer screen, specify the size and number of partitions created during and after an OSM installation.

- **Orders per Partition**: Specifies the number of orders that the Oracle Database allows in a range partition. This is also referred to as the range partition size.
Sizing Partitions

Managing the OSM Database Schema

8-19

- **Number of Sub-partitions**: Specifies the number of hash sub-partitions in a range partition.

You can change the values that you selected during the installation process by updating the `range_partition_size` and `subpartitions_number` OSM database parameters. Updates to these parameters do not affect existing partitions. For more information about these parameters, see "Configuration Parameters."

Sizing of partitions depends on the purge strategy and several other factors, as discussed in the following sections:

- Sizing Hash Sub-Partitions
- Sizing Range Partitions for Partition-Based Order Purge
- Sizing Range Partitions for Row-Based Order Purge

**Sizing Hash Sub-Partitions**

Hash sub-partitioning increases concurrency by reducing I/O contention, specifically by spreading DML statements (data modifications) over several physical sub-partitions. Contention is usually manifested as "buffer busy" waits in AWR reports.

Oracle strongly recommends that you set the number of sub-partitions to the smallest power of 2 that either eliminates "buffer busy" waits or reduces them to low single digits (for example, less than 2%). Typical values in production environments are 16, 32, and 64.

- Using values other than powers of 2 results in data skew, that is, data unevenly distributed across partitions.
- Increasing unnecessarily the number of sub-partitions could have a negative impact on purge performance. Test with 16 and 32 before trying 64. For more information refer to "Pitfalls of Partitioning."

Oracle recommends that you conduct performance tests at the expected peak workload to find the right number of sub-partitions for your environment. Periodically, also review the "buffer busy" waits in AWR reports from production. Consider increasing the `subpartitions_number` parameter if there are signs of increased I/O contention. Similarly, if there are no signs of contention and you experience performance issues due to a very large number of partitions, consider decreasing it if the total number of physical partitions is very large. The new value will be used when you add partitions (existing partitions are not affected).

In development and testing environments with limited storage capacity you should use a small number of hash sub-partitions, such as 2 or 4. For more information see "Development and Testing Environments."

**Sizing Range Partitions for Partition-Based Order Purge**

If your purge strategy is partition-based, typical range partition sizes vary between 100 thousand and 1 million. Creating partitions that are overly large makes it more difficult to free space by purging partitions that contain orders that no longer need to be retained. Creating partitions that are too small increases the frequency that new partitions must be created and the duration of maintenance windows, and could cause issues with performance and resource consumption.

As a rule of thumb, each range partition should be sized to contain as many orders as will be purged in one purge maintenance window. For example, if you target to purge partitions every month, you could size your partitions to contain the average expected
monthly volume. The feasibility of purge frequency will need to be validated based mainly on the amount of storage available and the duration of the outage that may be required to run the purge maintenance window.

Several factors influence sizing. Because these factors are correlated, it is recommended that you create a few scenarios by adjusting those parameters that are under your control. This document uses the example in the “Partition Purge Example” section to discuss sizing criteria in detail.

- Purge Performance
- Estimating Storage
- "All-In" Order Volume
- Partition Size Restrictions
- Retention Policy
- Time-to-Close Wait
- Oracle RAC
- Purge Frequency

**Purge Performance**
The main factors that affect partition-based purge downtime are purge frequency and purge performance.

There are a number of ways to improve purge performance:

- If range partitions are undersized, according to the guideline that each range partition should ideally contain as many orders as will be purged in one purge maintenance window, consider increasing the partition size. For example, the time to purge a 200 GB partition is only slightly more than the time to purge a 100 GB partition. This guideline also helps minimize partition consolidations.

- Decrease the number of hash sub-partitions. The time to purge a 200 GB partition with 64 hash sub-partitions is nearly double the time to purge a 200 GB partition with 32 sub-partitions. For more information refer to “Pitfalls of Partitioning” and “Sizing Hash Sub-Partitions.”

- Decrease the overall number of physical partitions. For more information refer to “Pitfalls of Partitioning.”

- Increase the time-to-close wait to reduce the number of retained orders.

- Tune purge operations, for example increase the degree of parallelism.

- Tune the database and operating system.

- Tune storage. For example, consider enabling more storage ports or converting disks from RAID-5 to RAID-10, which has better write performance.

- If, after exhausting all of the above options, performance is still inadequate, consider hardware upgrades depending on the nature of the bottleneck (for example, CPU, RAM, I/O bandwidth).

**Estimating Storage**
To determine the size of partitions, you need to also consider the amount of storage that is allocated to OSM. This is necessary to provision sufficient storage capacity, ensure that you are comfortable managing it, and validate the trade-off between storage and the frequency and duration of maintenance windows (outages).
It is recommended that you conservatively estimate the amount of required storage. Consider possible changes in sizing criteria and purge frequency, such as a volume or order size increase due to a rollout of new services, orders requiring more space due to additional functional requirements introduced during a solution or product upgrade or a purge embargo during holidays. Add contingency for unforeseen events that might delay purging or lead to increased space consumption.

For the purpose of estimating minimum storage requirements, consider the following partition breakdown:

- The oldest partitions that have been purged at least once.
- Partitions that have never been purged, including exhausted partitions and the latest partition(s) where new orders are created. (If you use Oracle RAC with N nodes in active-active mode, orders are created concurrently on N partitions as explained in the Oracle RAC section.)

The oldest partitions that have been purged at least once normally contain a small number of orders. It is recommended that you consolidate these partitions regularly (every few purges). If you do, the total space consumed by those partitions should be a fraction of a single partition.

Partitions that have never been purged consume the bulk of your storage. The number of these partitions depends on the partition size, the order retention period, the time-to-close wait, the purge frequency and whether you use active-active Oracle RAC. At the time of purge, these partitions can be further distinguished as eligible and ineligible for purge. If you follow a regular schedule, you can estimate the space consumed by these partitions as follows:

- Where P is the partition size (for example, 4 week's worth of data), R the retention period, T the time-to-close wait, and F the purge frequency (all using the same units, such as days or weeks).
- Where N is the number of active-active Oracle RAC nodes. If you use a single instance database, N=1.
- Where S is the space consumed by a single partition. Refer to the "All-In Order Volume" section for estimating S.

To estimate the number of partitions that are eligible for purge: $F / P \times N$

To estimate the number of partitions that are ineligible for purge: $(T + R) / P \times N$

To estimate the total number of partitions that have never been purged: $(F + T + R) / P \times N$

To estimate the total space consumed by these partitions: $(F + T + R) / P \times N \times S$

If you use a single instance database and the partition size is the same as the purge frequency, the above formula can be simplified: $(P + T + R) / P \times S$

Oracle strongly recommends that you increase your estimate by some contingency based on your ability to quickly add storage if necessary, reschedule a failed purge, and other risks (for example, by 25% or the amount of space reclaimed by each purge).

**Example:**

*Figure 8–8 is an example of estimating minimum storage requirements.*

- Partition size (P): 2 weeks’ worth of orders
- Purge frequency (F): Biweekly
- Time-to-close (T): 3 weeks
Retention period (R): 4 weeks

The number of partitions that have never been purged before is:

\[(F + T + R) / P \times N = (2 + 3 + 4) / 2 \times 1 = 4.5\]

Assuming that the space consumed by a single partition is about 200 GB, the total space consumed by those partitions is about 900 GB. Specifically, the four exhausted partitions P3-P6 consume about 800 GB, while the half-full partition P7 consumes about 100 GB. Partitions P1 and P2 have already been purged at least once. Assuming that you do not purge a partition unless at least 98% of its orders are closed, P1 and P2 consume less than 4 GB each (2% of 200 GB). In total, the used space is about 908 GB.

The used space should fluctuate between roughly 712 GB after a purge and 908 GB, as long as there are no unexpected events or delays. In addition, you must add some contingency in case you miss a maintenance window, for example, 200 GB.

**Figure 8-8 Estimating Minimum Space Consumption**

"All-In" Order Volume

To estimate the space consumed by a single partition, you must first estimate the "all-in" order volume. "All-in" means a representative order mix, including SOM orders created by COM orders (if they are stored in the same schema), revision orders, fallouts, technical orders, and so on. Some cartridge implementations might generate a lot of small orders that consume a disproportionate share of order Ids compared to their size (for example, for trouble tickets).

This is how you could estimate the space consumed by a single partition:

- Estimate the average all-in order volume over a period that accounts for volume fluctuations. One week is a good starting point, because it includes weekends.
- Populate a partition with a representative mix of orders and states for the same period. If that period is too long due to storage or time constraints, you may use a shorter period. However, it is important that you use a substantial data set to improve the accuracy of estimates - typically at least one day’s worth of orders.
- Use the `om_part_maintain.estimate_ptn_purged_space` procedure to estimate the space that would be reclaimed if you purged the entire partition, and extrapolate for various partition sizes. For more information, see "Estimating Partition Disk Space (Online or Offline)."
Partition Size Restrictions

If the estimated space consumed by a range partition is too big to manage? For example, suppose you want to purge monthly and your estimates show that a month's worth of orders will consume about 400 GB. If you do not want to manage partitions as big as 400 GB but you want to stick to a monthly purge frequency, decrease the partition size (for example, to two weeks' worth of orders). The downside is an increase in purge time, normally measured in minutes. Refer to the “Purge Frequency” section for an example.

Retention Policy

The retention policy is one of the most important sizing factors, yet you have the least control over it because normally it is determined by the business. The retention period starts counting after an order is closed. Therefore, in order to determine when an exhausted partition will be both eligible and cost-effective to purge, add the retention period to the “time-to-close” wait period.

Example:

Figure 8–9 shows the impact of the retention period. Decreasing the retention period by 2 weeks requires less storage, equal to the space consumed by a single partition. This is because each partition is sized to contain 2 weeks' worth of orders. Similarly, if you increased the retention period to 6 weeks, you would consume additional space for 2 weeks' worth of orders and you would have to maintain an extra partition.

Figure 8–9 Impact of Retention Policy

![Retention Policy Diagram]

Time-to-Close Wait

Time-to-close wait is the period until "most" orders in the partition are closed. The objective is to wait until a partition purge is cost-effective. As a starting point, you should wait until at least 98% of the orders are closed. Your final decision is a trade-off between storage and purge performance (duration of outage), as discussed below.

The first concern is the impact to purge performance of the time-to-close wait. When you purge a partition, retained orders are temporarily copied into so-called backup tables, and they are later restored (copied back) into the partition. These copy operations could add significant downtime to the maintenance window depending on the volume of retained data, your hardware, and the degree of parallelism. You can
decrease execution time by increasing parallelism. In general, you should aim to maximize resource utilization in order to improve purge performance. However, increased parallelism comes with more overhead. For example, you might find out that if you double the parallelism, the execution time is reduced by only one third. And there is a tipping point where parallelism overhead outweighs gains. Therefore it is recommended that you tune the degree of parallelism and evaluate the performance of purge operations on production quality hardware - ideally of the same caliber as your production hardware. For additional information about tuning, see the "Performance Tuning" section.

It is easier to use percentages in your initial time-to-close calculations (for example, the time to close 98% of orders). Performance tests help to nail it down to absolute numbers. For example, suppose your acceptable range for copying retained orders (backup and restore) is 15-30 minutes, and that according to performance tests this is enough to retain 10000-20000 orders. In order to allow for partition consolidations, you could use 10000 in your calculations, which also provides a safety buffer. For example, if the partition size in one of your scenarios is one million orders, 10000 orders is 1%. In this case, time-to-close is the time it takes to close 99% of the orders.

With regard to storage, a shorter time-to-close wait is better. Decreasing the time-to-close wait alone by X days is the same as decreasing the retention period alone by X days or decreasing both by X days in total.

**Example:**
Figure 8–10 shows the impact of the time-to-close wait period. Each partition is sized to contain 2 weeks’ worth of orders. All things being equal, decreasing this wait by 2 weeks requires less storage, equal to the space consumed by a single partition. Alternatively, the number of retained orders increased five times to about 10%, which might add several minutes to the duration of a maintenance window. You must decide whether these storage savings justify a longer outage (perpetually).

**Figure 8–10 Impact of Time-to-Close Wait Period**

![Figure 8–10 Impact of Time-to-Close Wait Period](image)

**Oracle RAC**

As explained in the section “Support for Active-Active Oracle RAC,” if you switch from a single database to Oracle RAC with N active-active nodes, the number of partitions increases N-fold whereas the actual number of order Ids stored in a partition decreases N-fold. This means that:
The space consumed by N partitions is about the same as that consumed previously by a single partition.

You do not necessarily need to change the partition size, storage capacity, the purge frequency, or any other purge-related policies.

During a purge window, you must purge N partitions instead of one and consolidate them N-to-1.

Consolidating partitions might sound contrary to the way OSM is designed to use partitions on active-active Oracle RAC. However, it is unlikely that order processing on a consolidated partition will experience cluster waits. The number of retained orders is normally small, the consolidated order IDs are far apart, and there is typically little activity on those orders. If a significant increase in cluster waits proves to be the result of partition consolidation, consider avoiding consolidation when a partition is purged for the first time.

Another concern is that a large number of physical partitions could potentially cause performance issues, as discussed in the "Pitfalls of Partitioning" section.

Using Oracle RAC in active-passive mode is similar to using a single instance database. The only difference is that order creation might be switched to another partition and then back to the original in the events of failover and failback, although a switch might not occur right away or even not at all. This means that you may end up with a sparsely populated partition, which at some point could be consolidated with another partition.

**Example:**

Figure 8–11 compares a single instance database to active-active Oracle RAC. Specifically, OSM is configured to use two nodes in active-active mode. The Oracle RAC database may have additional nodes that are either not used by OSM or they are used in passive mode. The partition size, time-to-close wait, retention period and purge frequency are the same. However, OSM uses twice as many partitions on Oracle RAC, which are half-full when they are exhausted (half of the order IDs are skipped). This means that you must purge and consolidate two partitions instead of one to reclaim the same amount of space.
Purge Frequency

As explained in "Estimating Storage," if you follow a regular purge schedule, the number of partitions purged during each maintenance window is \( F/P \) for a single instance database, where \( F \) is the purge frequency (for example, 30 days) and \( P \) is the partition size (for example, 30 days' worth of orders). As a starting point, it is recommended that you size each range partition to contain as many orders as will be purged in one purge maintenance window, that is, \( F=P \). As you evaluate scenarios for different purge frequencies, adjust the partition size accordingly so that \( F=P \). If the partition size is less than the purge frequency, you will have to consolidate partitions \( N \)-to-1, where \( N=F/P \). This will add some extra time to purge maintenance (normally measured in minutes). You might do this if you are uncomfortable using large partitions. In this case, if you like a constant (predictable) consolidation ratio, choose the partition size so that \( N=F/P \) is an integral number.

A desire to purge as infrequently as possible is likely limited by the storage capacity and/or the administrative burden of managing a very large schema (whatever your criteria may be for "large"). Fortunately, you can often decrease the purge frequency \( N \)-fold with a relatively small increase in storage capacity. For simplicity, consider a single instance database and assume that the purge frequency is the same as the partition size. As explained in Estimating Storage, in this case you can use the following formula to estimate the storage consumed by partitions that have never been purged, where \( P \) is the partition size (for example, in days), \( T \) is the time-to-close wait, \( R \) is the retention period, and \( S \) is the space consumed by a single partition:

\[
\frac{P + T + R}{P} \times S = \frac{1 + (T + R)}{P} \times S
\]

Based on this formula, if the period \( T + R \) is large compared to \( P \), you could double or triple the partition size and the purge frequency with a relatively small increase in storage. This is demonstrated with the following example.

Example (from biweekly to monthly maintenance):

Suppose that you have a 4-week retention period, a 3-week time-to-close wait and a biweekly purge frequency. The partition size is also 2 weeks. The following formula is used to calculate the storage consumed by never-purged-before partitions, where \( S \) is the space consumed by a single partition:

\[
\frac{P + T + R}{P} \times S = \frac{P + 3/2P + 2P}{P} \times S = 4.5 \times S
\]

Now suppose that you want to reduce the purge frequency to every 4 weeks. You can double the partition size and estimate the storage capacity:

\[
\frac{P + T + R}{P} \times S = \frac{P + 3/4P + P}{P} \times 2 \times S = 5.5 \times S
\]

This means that the extra storage is \( S \). Thus, you can achieve a 100% reduction in downtime for 22% increase in storage capacity or 2 weeks' worth of orders. This is demonstrated in the figure below.

Note that because there is no change in the time-to-close wait, the larger partition will have more retained orders depending on the order aging curve. If the difference is large enough to have a material impact in the purge time, you may want to consider increasing the time-to-close wait slightly at the cost of a bit more extra storage.
Sizing Range Partitions for Row-Based Order Purge

Sizing range partitions for row-based order purge is different from sizing for partition-based purge. Partitions sized for row-based purge must have a wide range of order IDs to store several months' worth of orders.

The main characteristic of this strategy is that closed orders, when they come out of their retention period, must be deleted from the partition(s) where new orders are created. The objective is to maximize reuse of the free space, to restrain the growth of the segment high water mark. In addition, the space freed by deletes in an exhausted partition cannot be reused because new orders are created on a different partition. The space consumed by an exhausted partition can be released to the tablespace by using other operations, for example by dropping or shrinking the partition.

This means that your goal should be to minimize the number of exhausted partitions. The main criteria are the maximum order lifetime (OLT), the retention period, and your availability requirements. Here are some guidelines for sizing partitions for row-based order purge:

- If you have little or no operational experience using this purge strategy, be conservative by starting with a partition size you are comfortable with, and increase it as you gain experience (if necessary).
- If you cannot afford outages, the partition size should be large enough to contain the order volume over a period that is greater than or equal to the sum of the maximum order lifetime and the retention period. This approach requires the least amount of storage, as discussed in "Sizing Range Partitions for Zero Downtime."
- If the maximum order lifetime is too long (years), the above recommendation would result in oversized partitions that could take years to exhaust. If you do not have operational experience using row-based order purge and/or you feel uncomfortable with oversized partitions, and you can afford some occasional downtime (for example, every 6 months or once a year), you can size partitions for infrequent maintenance as discussed in "Sizing Range Partitions for Infrequent Maintenance."
- The partition sizing for a single instance database and active-active Oracle RAC is the same. As explained in "Support for Active-Active Oracle RAC," if you switch from a single database to Oracle RAC with N active-active nodes, the number of
partitions will increase N-fold, whereas the actual number of order Ids stored in a partition will decrease N-fold.

**Sizing Range Partitions for Zero Downtime**

If you cannot afford outages, you can size partitions to avoid outages as follows:

- Size partitions to contain the order volume over a period that is greater than or equal to the sum of the maximum order lifetime and the retention period.
- Keep purging orders from exhausted partitions until they are empty (using row-based order purge).
- Drop empty exhausted partitions online.

This approach requires the least amount of storage because it restricts the number of partitions to two at any point in time (2xN if you use Oracle RAC with N nodes in active-active mode). Specifically, the previous partition is purged empty and dropped online before the current partition is exhausted. Figure 8–13 illustrates this by example.

**Figure 8–13  Sizing Partitions to Avoid Outages**

If both the retention period and the maximum order lifetime are relatively short, there is more flexibility in sizing partitions. You do not necessarily need oversized partitions because you can drop them online in a relatively short period after they are exhausted.

If the current partition is exhausted and the previous partition has still long-lived open orders that are expected to remain open for much longer, you might have to schedule a maintenance window to purge that partition using partition-based purge.

**Sizing Range Partitions for Infrequent Maintenance**

If you have a near 24x7 availability requirement with occasional maintenance windows, you could use those windows to drop or purge old partitions offline (for example, every 6 months or once a year).

If the retention period or the maximum time-to-close wait are long, you should plan the partition size with the next maintenance window in mind, so that eventually you are able to either drop it or purge it cost-effectively (for example less than 2% of the orders will be retained).

As a rule, the partition size (P) should be less than the period until the next maintenance window (M), minus the time-to-close wait (T), minus the retention period.
Sizing Partitions

Managing the OSM Database Schema

(R), as shown below. In this case, time-to-close is the wait period until "most" orders in the partition are closed, as discussed in "Time-to-Close Wait."

\[ P < M - (T + R) \]

If there is uncertainty about the date of the next maintenance window and/or that date is based on external factors and it will be immovable, it is a good idea to make the partition a bit smaller as contingency.

The partition size must also be (a lot) greater than the sum of the retention period and the time-to-close wait, otherwise it would be exhausted before there is an opportunity to delete a substantial amount of orders:

\[ P > T + R \]

Note that the partition size may vary, at least initially, because it depends on a maintenance schedule. Instead of pre-allocating two or more partitions, you may want to wait until the current partition is closer to being exhausted and there is less uncertainty about the next maintenance window.

**Example:**

Suppose 98% of the orders close within 1 week, the retention period is 4 weeks, and you have a maintenance window every 24 weeks. You want the first partition to be exhausted after 19 weeks or less \((24 - 1 - 4)\). Using 1 week as contingency, 18 weeks is a good size. After that, the partition size is 24 weeks (the same as the purge frequency), everything else being the same.

**Online vs. Offline Maintenance**

All database maintenance operations that can be performed online, which means OSM is running, can also be performed offline. However, some operations can be performed offline only.

In order to execute a procedure offline you must either stop all WebLogic servers where OSM is deployed (for example, all managed servers) or stop the OSM application and cartridges. Beginning with 7.2.0.9, database management procedures stop and restart OSM database jobs automatically.

The table below summarizes which operations can be performed online. Offline execution is always faster than online execution. If a procedure supports online execution, it is recommended only under low volume. In particular, online execution of partition management operations causes lock contention in the database and waits, such as `cursor: pin S wait on X`. Under high volume, such contention could result in severe performance degradation, transaction timeouts, and even order failures.

<table>
<thead>
<tr>
<th>Operation</th>
<th>Online</th>
<th>Offline</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add a partition</td>
<td>Avoid if possible</td>
<td>Recommended</td>
</tr>
<tr>
<td>Row-based order purge</td>
<td>OK</td>
<td>OK</td>
</tr>
<tr>
<td>Drop a populated partition</td>
<td>Not supported</td>
<td>OK</td>
</tr>
<tr>
<td>Drop an empty partition</td>
<td>OK</td>
<td>Recommended</td>
</tr>
</tbody>
</table>

Table 8–2  Summary of Online Versus Offline Maintenance Operations
Managing Order Data

The most common maintenance operations are adding partitions and purging orders to reclaim storage.

See "Adding Partitions (Online or Offline)" for information about how to add partitions.

As discussed in "Order Purge Strategies", there are two main purge strategies, namely row-based order purge and partition-based order purge. These are discussed in more detail in these sections:

- **Using Row-Based Order Purge**: Explains how row-based order purge works and how to use it.
- **Using Partition-Based Order Purge**: Explains how partition-based order purge works and how to purge and drop partitions.
- **Order Purge Policies**: Discusses purge policies.
- **Managing Exchange Tables for Partition-Based Order Purge**: Describes exchange tables and explains how to create and manage them. Exchange tables are required for purging partitions. They are not used when dropping partitions or using row-based order purge.
- **Estimating Partition Disk Space (Online or Offline)**: Explains how to estimate the amount of space that could be reclaimed during a maintenance window, the amount of space consumed by a partition, or the average order size in a partition.

### Adding Partitions (Online or Offline)

You can add partitions manually using the following procedures:

- `om_part_maintain.add_partition`
- `om_part_maintain.add_partitions`

For production and performance environments, Oracle strongly recommends that you add partitions manually either when OSM is offline or during periods of low system activity. This is particularly important if you use Oracle RAC.

<table>
<thead>
<tr>
<th>Operation</th>
<th>Online</th>
<th>Offline</th>
</tr>
</thead>
</table>
| Partition purge                                | • Purging a partition partially is not supported online.  
• Purging an entire partition is supported conditionally. For a list of these conditions, see the `om_part_maintain.purge_partitions` procedure. | OK      |
| Exchange table maintenance (exchange tables are used by partition purge) | OK                                  | OK      |
| Cartridge Fast Undeploy                        | OK                                  | OK      |
| Cartridge conventional undeploy                | Not supported                       | OK      |

*Table 8–2  (Cont.) Summary of Online Versus Offline Maintenance Operations*
 ■ Adding partitions online causes lock contention in the database and waits, such as cursor: pin S wait on X. Under high volume, such contention could result in severe performance degradation, transaction timeouts, and even order failures.

 ■ OSM can also add a partition automatically when a new order ID does not map to any partition. This is to prevent order creation failures if all partitions are exhausted. However, it is strongly recommended that you create partitions manually and disable automatic creation for all production and performance environments, especially if you use Oracle RAC, in order to avoid automatic creation under high volume. Automatic creation of partitions can be disabled with the partition_auto_creation parameter.

 ■ The size of new partitions is specified by the range_partition_size parameter. Specifically, the upper bound of a new partition is the greatest partition upper bound plus the value of range_partition_size.

 ■ You can optionally specify the tablespace of each new partition as the input argument to om_part_maintain.add_partition. Using a different tablespace for each partition (usually a circular list) facilitates administrative tasks, such as backups.

 The installer creates the first partition. Always create a sufficient number of new partitions to handle the orders for a given period until the next maintenance window that includes a safety buffer in case an unexpected increase in order volume occurs or you skip a maintenance window.

 If you have configured OSM to use Oracle RAC with N database nodes in active-active mode, you must add partitions in multiples of N. This is because OSM creates orders on the last N partitions concurrently. For example, if you use a 2-node Oracle RAC database in active-active mode, new orders are created on the last two partitions. If OSM is configured with a single instance database, an Oracle RAC database in active-passive mode or Oracle RAC One Node, new orders are created on the last partition only.

 For more information, see "range_partition_size", "om_part_maintain.add_partition (Offline Only)", and "om_part_maintain.add_partitions (Offline Only)".

 Example (Add 2nd partition): Consider a new installation with range_partition_size equal to 100,000. The upper bound of the partition created by the installer is 100001 (the upper bound of a partition is non-inclusive). The following statement adds a second partition with upper bound 200001 on tablespace OSMTS.

 execute om_part_maintain.add_partition('OSMTS');

 Example (Add Nth partition, N > 2): The following statement adds three more partitions on the same tablespace as the most recently added partition with upper bounds 300001, 400001 and 500001.

 execute om_part_maintain.add_partitions(3);

Using Row-Based Order Purge

 As discussed in "Row-Based Order Purge Strategy," you can use row-based order purge as an online purge strategy. Row-based order purge uses deletes to purge individual orders, whereas partition-based purge uses ALTER TABLE ... DROP PARTITION and ALTER TABLE ... EXCHANGE PARTITION to change the table structure.

 The space freed by deletes in a table or index segment does not lower the high water mark of that segment. That space can be reused by inserts into the same segment but...
the space is not released to the tablespace. Therefore, you must run order purge as frequently as possible (at least once a day) to free space at the same rate it is consumed by new orders. The objective is to restrain the growth of the high water mark.

Figure 8–14 shows how row-based order purge deletes orders in two stages. The API also provides procedures that allow you to execute each stage separately, as follows:

1. Selects the order Ids that satisfy given purge criteria into the OM_ORDER_ID_FOR_PURGE staging table.
2. Deletes the selected orders.

![Figure 8–14 How Row-Based Order Purge Works](image)

In the first stage, order purge scans OM_ORDER_HEADER and inserts the order Ids of all orders that satisfy the given purge criteria into the OM_ORDER_ID_FOR_PURGE staging table. You can restrict the scope of the search to an order Id range, for example to the latest partition(s) where new orders are created.

In the second stage, the selected orders are purged in parallel using the dbms_parallel_execute package. More precisely:

- Order purge splits the work into smaller pieces by dividing the data blocks of OM_ORDER_ID_FOR_PURGE into chunks. Then it spawns N database jobs to purge the chunks in parallel, where N is the degree of parallelism (possibly 1). Each job processes one chunk at a time by deleting one order at a time, automatically committing every few deletes. In the event of error (for example, a deadlock), the job continues with the next chunk.

- After finishing the processing of all chunks, order purge retries processing of any failed chunks until either all chunks are processed successfully (all orders in the chunk are purged) or a pre-defined retry threshold is reached.

- At the end of a successful purge, order purge clears OM_ORDER_ID_FOR_PURGE.

This approach ensures that a) an order is either purged entirely or not at all b) a purge may succeed partially even in the event of errors and c) the purge handles recoverable errors, such as deadlocks.

For performance reasons, the staging table is hash partitioned. To minimize contention among purge jobs, OM_ORDER_ID_FOR_PURGE must have the same number of partitions as the number of hash sub-partitions in an OM_ORDER_HEADER range.
Managing Order Data

Managing the OSM Database Schema

8-33

partition. Otherwise, order IDs in different OM_ORDER_ID_FOR_PURGE blocks that are processed by different purge jobs might be stored in the same OM_ORDER_ HEADER block.

The default degree of parallelism is specified by the degree_of_parallelism configuration parameter. The number of chunks generated depends on a number of factors, including the number of OM_ORDER_ID_FOR_PURGE hash partitions, the volume and distribution of data in the staging table, and the desired chunk size. The latter is specified by parallel_execute_chunk_size, which is an advanced configuration parameter (you rarely need to change the default).

Row-based order purge is implemented by the om_new_purge_pkg package. This package allows you to:

■ Purge a single order by order ID.
■ Purge orders that satisfy given purge criteria.
■ Schedule an order purge.
■ Stop and resume an order purge.
■ Audit row-based purge operations. For more information see "Auditing and Monitoring Order Purges."

Purging a Single Order by Order ID

The om_new_purge_pkg.delete_order procedure allows you to delete an order by its order ID. This is convenient when you want to delete only one order or a small number of orders (for example, to resubmit).

Note that this procedure does not issue commit, in contrast to most purge procedures. You must manually issue commit or rollback.

Purging Orders that Satisfy Given Criteria

If you need to purge large quantities of orders (for example, if row-based order purge is your purge strategy), purging orders one by one serially using delete_order would be very slow. Use one of the following procedures instead, which allow you to purge orders in parallel:

■ The om_new_purge_pkg.purge_orders procedure purges all orders that satisfy given purge criteria, including your retention policy (for example, you can purge all orders that were closed 30 days ago or more). First, it finds the order IDs that satisfy the purge criteria. Then it spawns database jobs to purge the selected orders in parallel, as discussed in Understanding Row-Based Order Purge.

■ The om_new_purge_pkg.select_orders and om_new_purge_pkg_purge_selected_orders procedures are equivalent to purge_orders but offer more flexibility because you can execute the order selection and order purge steps separately. For example, you can select the orders to purge piecemeal by executing select_orders several times. You can also update the OM_ORDER_ID_FOR_PURGE staging table manually, especially if you have complex purge criteria that are not supported by purge_orders and select_orders.

If the purge is performed while OSM is online, adjust the degree of parallelism to ensure that the database can handle the additional workload of deleting orders concurrently with order processing.

Both purge_orders and purge_selected_orders provide the a_stop_date parameter, which allows you to specify an end date and time for the purge. This is useful if you
want to run order purge during specific periods (for example, during a low-volume period at night).

**Scheduling Order Purge**
The `om_new_purge_pkg.schedule_order_purge_job` procedure allows you to schedule a one-time execution of order purge. The purge is scheduled using the `dbms_job` package. If row-based order purge is your main strategy, this procedure is inadequate. In this case it is recommended that you use the Oracle Database scheduler to schedule periodic execution of order purge.

**Stopping and Resuming an Order Purge**
The `om_new_purge_pkg.stop_purge` procedure allows you to stop an order purge. This might be necessary, for example, if the host is too busy or you want to perform other maintenance. This procedure returns immediately. However, the purge will stop after all currently assigned chunks are processed, possibly after several minutes.

Later you can resume the same purge by executing `om_new_purge_pkg.resume_purge`. You can also restart the purge with different parameters by executing `om_new_purge_pkg.purge_selected_orders` (for example, if you want to change the time when the purge window ends or the degree of parallelism), or start a new purge.

**Using Partition-Based Order Purge**
As discussed in the “Partition-Based Order Purge Strategy” section, partition-based order purge allows you to purge several weeks’ worth of order data in minutes by dropping or “purging” partitions. These operations are based on efficient DDL statements that modify the table structure, such as `ALTER TABLE ... DROP PARTITION` and `ALTER TABLE ... EXCHANGE PARTITION`.

The following sections discuss partition-based purge in detail:

- **Differences Between Purging and Dropping Partitions**: Summarizes the differences between purging and dropping partitions.

- **Purging Partitions (Online or Offline)**: Describes how to purge partitions. When you purge a partition, all the data that satisfies given purge criteria is deleted and storage is immediately reclaimed. This operation allows you purge a partition partially, that is, to retain orders that do not satisfy given purge criteria. It also allows you to consolidate multiple partitions into one. The limitation is that the partition must be purged offline. However, if all orders in a partition are closed and satisfy the purge criteria, you might be able to purge the entire partition online.

- **Dropping Partitions (Offline Only)**: Describes how to drop partitions. When you drop a partition, all its data is deleted and storage is immediately reclaimed. The limitations are that the partition must be dropped offline and it must have no open orders.

- **Dropping Empty Partitions (Online or Offline)**: Describes how to drop empty partitions online.

**Differences Between Purging and Dropping Partitions**
These are the main differences between purging and dropping partitions:

- You cannot drop a partition if it has open orders. However, you can purge a partition if the number of retained orders does not exceed a configurable
Managing Order Data

Managing the OSM Database Schema

Threshold. In this case, OSM copies retained orders to the so-called backup tables prior to the partition exchange, and restores them afterwards.

- To purge partitions you must create exchange tables first. This must be done once after a new installation and subsequently after schema changes. Exchange tables are not used when dropping partitions or when using row-based order purge.
- If you have only one partition, you cannot drop it unless you add another partition first. This restriction does not apply to purging.
- You can only drop partitions when the OSM is offline, so you must plan for downtime. If a partition does not have any open orders and all orders satisfy the purge criteria, you might be able to purge the partition online. However, note that purging online is slower than offline, and results in increased contention and lower throughput because DDL operations lock the entire table in exclusive mode. Therefore, you should only purge online during the lowest volume hours and only after you have tested it successfully in your environment.

Purging Partitions (Online or Offline)

When you purge a partition, all the data that satisfies given purge criteria is deleted and storage is immediately reclaimed. This operation allows you purge a partition partially, that is, to retain orders that do not satisfy given purge criteria. This is done using ALTER TABLE ... EXCHANGE PARTITION statements. The EXCHANGE PARTITION statement exchanges a partition with a table (known as the exchange table). The exchange operation swaps the segments of the partition with the segments of the exchange table but does not delete or move data between segments. When the exchange table is empty, the partition is purged, but the partition remains associated with the partitioned table, for example, it still appears in user_tab_partitions.

In general, the EXCHANGE PARTITION statement is a fast operation because it updates metadata only in the data dictionary (unless the table has global indexes and the exchange operation is performed with the UPDATE GLOBAL INDEXES option). However, to reclaim the storage previously used by the partition you must also purge the exchange tables.

To purge partitions and reclaim storage space, do the following using PL/SQL stored procedures:

1. Create exchange tables (once after a new installation and subsequently after schema changes). For more information see "Managing Exchange Tables for Partition-Based Order Purge."

2. During periodic maintenance windows, do the following:
   - Purge partition and possibly consolidate partitions. See "Partition-Based Order Purge Strategy" for examples. Typically this is an offline operation because a small percentage of orders must be retained (for example, open orders and closed orders that are still in the retention period). See "Purging Partitions Partially (Offline Only)" for more information. If all orders in a partition are closed and satisfy the purge criteria, you might be able to purge the entire partition online, see "Purging Entire Partitions That Do Not Contain Open Orders (Online or Offline)" for more information.
   - Add partitions. You can add partitions online under low volume, however adding partitions offline is less likely to cause problems. See "Adding Partitions (Online or Offline)" for more information.
   - Purge exchange tables to reclaim storage space. You can purge the exchange tables at any time to reclaim storage (preferably during off-peak hours). If the database has enough spare CPU and I/O capacity, this operation does not
affect performance, even when it is performed online, because it does not cause contention (the exclusive locks acquired on the exchange tables do not block other processing). For more information see "Managing Exchange Tables for Partition-Based Order Purge."

**Figure 8–15  Typical Maintenance Using Partition-Based Order Purge**

---

**Purging Entire Partitions That Do Not Contain Open Orders (Online or Offline)**

You can entirely purge partitions that do not contain any open orders with the `om_part_maintain.purge_partitions` procedure. If purging online, this procedure exchanges each partition you want to purge with empty purge table(s), effectively swapping the data of that partition out of the table. If purging offline, partitions that can be entirely purged are dropped, unless you disallow it. However, the partition where retained orders are consolidated is always exchanged, even if that partition has no retained orders itself.

- Purging partitions online causes lock contention in the database and waits, such as cursor: pin S wait on X. Under high volume, such contention could result in severe performance degradation, transaction timeouts, and even order failures.
- You can disallow dropping partitions by passing `a_drop_empty_ptns=false`. However, this prevents partitions from being consolidated and affects purge performance.
- The name and upper bound of an exchanged partition do not change.
- If the exchanged partition and the purge table are on different tablespaces then after the exchange the two tablespaces are swapped (there is no movement of data).
- If the parameter `purge_policy_purge_related_orders_independently` is set to 'N' and the partition contains orders that are associated directly or indirectly with orders that do not satisfy the purge criteria (for example, open follow-on orders in a different partition), the partition cannot be purged entirely. For more
information, see the purge policy section in "Purging Related Orders Independently."

For more information, see om_part_maintain.purge_partitions.

Purging Partitions Partially (Offline Only)
The om_part_maintain.purge_partitions procedure can purge partitions that contain orders that must be excluded from purging (retained), such as open orders and orders that do not satisfy the purge criteria. This is permitted if the OSM application is offline and the number of retained orders in a partition is "small", that is, it does not exceed a pre-defined threshold.

In addition, purge_partitions can move retained orders from multiple partitions into a single partition in order to maximize reclaimed space, reduce the number of partitions and minimize downtime. This is done by purging successive partitions in iterations. The maximum number of partitions purged in each iteration is limited by the purge capacity. More precisely, purge_partitions purges successive partitions that qualify for purging as follows:

- Copies the orders that do not satisfy the purge criteria from those partitions into the backup tables.
- Purges each partition entirely by exchanging it with purge tables.
- Drops N-1 of those partitions.
- Restores the retained orders from the backup tables into the Nth partition with their order IDs unchanged.

For more information, see "om_part_maintain.purge_partitions (Online or Offline)".

Example: Assume that the purge capacity is 3, and consider these partitions, as shown in Figure 8–16:

- P_000000000000600001: All orders satisfy the purge criteria. This partition can be purged entirely.
- P_000000000000700001: This partition can be purged but some orders do not satisfy the purge criteria.
- P_000000000000800001: This partition can be purged but some orders do not satisfy the purge criteria.
- P_000000000000900001: This partition cannot be purged because the number of orders that do not satisfy the purge criteria exceeds a configured threshold.
**Figure 8–16  Purging Partitions That Contain Orders That Must be Excluded from Purging**

These orders must be retained:

<table>
<thead>
<tr>
<th>OSM Purge Tables</th>
<th>OSM Order Tables</th>
<th>OSM Backup Tables</th>
</tr>
</thead>
<tbody>
<tr>
<td>XCHG_OM_PRG_001$*</td>
<td>P_000000000000600001</td>
<td></td>
</tr>
<tr>
<td>Empty</td>
<td>Purge entirely</td>
<td></td>
</tr>
<tr>
<td>XCHG_OM_PRG_002$*</td>
<td>P_000000000007000001</td>
<td></td>
</tr>
<tr>
<td>Empty</td>
<td>Purge</td>
<td></td>
</tr>
<tr>
<td>XCHG_OM_PRG_003$*</td>
<td>P_000000000008000001</td>
<td>XCHG_OM_BCK_$*</td>
</tr>
<tr>
<td>Empty</td>
<td>Purge</td>
<td>Empty</td>
</tr>
<tr>
<td></td>
<td></td>
<td>P_000000000009000001</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Do not purge</td>
</tr>
</tbody>
</table>

**Figure 8–17 to Figure 8–21** show how `purge_partitions` purges these partitions step by step:

1. It copies the orders that do not satisfy the purge criteria from P_000000000000700001 and P_000000000000800001 into the backup tables.

2. It purges partitions P_000000000006000001, P_000000000007000001 and P_000000000008000001 by exchanging them with the purge tables.

3. It drops partitions P_000000000006000001 and P_000000000007000001, which are now empty.

4. It restores the retained orders from the backup tables into partition P_000000000000800001.

5. (Optional) It purges the purge tables and continues the same process for any remaining partitions. This is possible only if you allowed it to purge the purge tables. Otherwise, it cannot proceed because the purge capacity is exhausted.
Figure 8–17  Step 1: Back Up Retained Orders

OSM Purge Tables

<table>
<thead>
<tr>
<th>XCHG_OM_PRG_001$*</th>
<th>Empty</th>
</tr>
</thead>
<tbody>
<tr>
<td>XCHG_OM_PRG_002$*</td>
<td>Empty</td>
</tr>
<tr>
<td>XCHG_OM_PRG_003$*</td>
<td>Empty</td>
</tr>
</tbody>
</table>

OSM Order Tables

<table>
<thead>
<tr>
<th>P_000000000000060001</th>
</tr>
</thead>
<tbody>
<tr>
<td>P_000000000000070001</td>
</tr>
<tr>
<td>P_000000000000080001</td>
</tr>
</tbody>
</table>

OSM Backup Tables

<table>
<thead>
<tr>
<th>Purge entirely</th>
</tr>
</thead>
<tbody>
<tr>
<td>Purge</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>P_000000000000090001</th>
</tr>
</thead>
</table>

Figure 8–18  Step 2: Purge the Partitions

OSM Purge Tables

<table>
<thead>
<tr>
<th>XCHG_OM_PRG_001$*</th>
<th>Empty</th>
</tr>
</thead>
<tbody>
<tr>
<td>XCHG_OM_PRG_002$*</td>
<td>Empty</td>
</tr>
<tr>
<td>XCHG_OM_PRG_003$*</td>
<td>Empty</td>
</tr>
</tbody>
</table>

OSM Order Tables

<table>
<thead>
<tr>
<th>P_000000000000060001</th>
</tr>
</thead>
<tbody>
<tr>
<td>P_000000000000070001</td>
</tr>
<tr>
<td>P_000000000000080001</td>
</tr>
</tbody>
</table>

OSM Backup Tables

<table>
<thead>
<tr>
<th>Purge entirely</th>
</tr>
</thead>
<tbody>
<tr>
<td>Purge</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>P_000000000000090001</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Do not purge</th>
</tr>
</thead>
</table>
Figure 8–19  Step 3: Drop the Partitions

Figure 8–20  Step 4: Restore the Retained Orders from the Backup Tables
Dropping Partitions (Offline Only)

If most of your orders are short-lived, you may be able to purge orders by dropping old partitions. This is the most efficient way to reclaim space but it involves downtime. During a maintenance window, stop OSM, drop old partitions, add enough partitions to store new orders until your next maintenance window, and restart OSM, as shown in Figure 8–22. Exchange tables are not used when you drop partitions.

When you drop a partition, all its data is deleted and storage is immediately reclaimed. You can use the stored procedure om_part_maintain.drop_partitions to drop partitions that contain orders with order IDs within a specified range if you no longer require the order data they contain and they have no open orders. If the schema contains only a single partition then Oracle Database does not allow you to drop it.

If the parameter `purge_policy_purge_related_orders_independently` is set to 'N' and the partition contains orders that are associated directly or indirectly with orders that
do not satisfy the purge criteria (for example, open follow-on orders in a different partition), the partition cannot be dropped. For more information, see the purge policy section in "Purging Related Orders Independently."

Because global indexes become unusable when partitions are dropped, this procedure also rebuilds unusable indexes and index partitions. This can be done in parallel.

For more information, see `om_part_maintain.drop_partitions`.

**Example:**
Consider an OSM schema with partitions P_000000000001000001, P_000000000002000001, P_000000000003000001 and so on. If P_000000000003000001 has open orders then the following statement will drop only P_000000000001000001 and P_000000000002000001.

execute om_part_maintain.drop_partitions(4000000);

**Dropping Empty Partitions (Online or Offline)**
Eventually, you must drop empty purged partitions for performance and operational reasons. Dropping empty partitions is relatively fast because they have few segments. You can use one of the following procedures:

- `om_part_maintain.purge_partitions` drops empty partitions by default, unless specified otherwise.
- `om_part_maintain.drop_empty_partitions` (online and offline): Oracle recommends this procedure because it does not disable foreign keys and therefore can be executed online. For more information, see "om_part_maintain.drop_empty_partitions (Online or Offline)".
- `om_part_maintain.drop_partitions` (offline only): For more information, see "om_part_maintain.drop_partitions (Offline Only)".

**Order Purge Policies**
Purge policies are configured in the `om_parameter` table.

**Purging Related Orders Independently**
By default, related orders are purged independently. This means that the decision whether an order can be purged is based solely on whether that order satisfies the purge criteria or not, regardless of any dependencies. For example, if the predecessor of an open follow-on order satisfies the purge criteria, the predecessor order can be purged even though the follow-on must be retained.

Beginning with OSM 7.2.0.10 and 7.2.2.3.5, you can disable this policy by setting the `purge_policy_purge_related_orders_independently` parameter to the value N in the `om_parameter` table. When this policy is disabled, an order with related orders can be purged only if all directly and indirectly related orders are ready to purge, that is they satisfy the purge criteria (for example, `a_delete_before` and `a_order_states`). However, the order IDs of the related orders may be within a different partition, or even outside the given purge range.

---

**Note:** Currently this policy is supported by partition-based purge only.
Important: Setting `purge_policy_purge_related_orders_independently` to N may add several minutes to the time it takes to purge or drop a partition.

Example (Purging related orders independently):
Assume that `purge_policy_purge_related_orders_independently` is set to N and that you want to purge all orders in partition P_000000000000300001 that were closed before midnight on the specified date of February 13, 2014:

```sql
execute om_part_maintain.purge_partitions(
    a_online => false,
    a_delete_before => trunc(to_date('02/13/2014', 'MM/DD/YY')),
    a_order_states => om_new_purge_pkg.v_closed_orders,
    a_order_id_lt => 300001);
```

Figure 8–23 shows some of the orders in partition P_000000000000300001 and the related orders:
- Order 291001 is purged because it satisfies the purge criteria and has no related orders.
- Order 291002 satisfies the purge criteria and is related to 391002, which is amended by 491002. Both 391002 and 491002 satisfy the purge criteria and are therefore ready to purge although they are outside the given purge range. Therefore order 291002 is purged even though 391002 and 491002 are not.
- Order 291003 is retained because it is indirectly related to 491003, which has a completion date that is after the specified date.
- Order 291004 is retained because it is indirectly related to 491004, which is open.

**Figure 8–23  Related Orders that Satisfy the Purge Criteria**

The following types of relationships are considered when looking for related orders:
- Successor Orders
- Predecessor Orders
- Amendment Orders
- Base Orders (for amendments)

Keep the following in mind when `purge_policy_purge_related_orders_independently` is set to N:
Both direct and indirect relationships are considered.

- It does not matter if the related orders are within the range of order ids being purged; it matters only whether they match the purge criteria (`a_delete_before` and `a_order_states` parameters).

- When purging partitions online, if any order in a partition has a related order that does not match the purge criteria, the partition cannot be purged. Purging online requires that there are no orders retained in the partition.

- When dropping partitions, if any order in a partition has a related order that does not match purge criteria, the partition cannot be dropped. Dropping partitions requires that there are no orders retained in the partition.

- If the total number of orders to be retained exceeds the threshold defined by the parameter `xchg_retained_orders_thres` (default 10000), the partition is not purged. This includes orders that satisfy the purge criteria but they must be retained because they are related to orders that do not satisfy that criteria.

**Auditing and Monitoring Order Purges**

OSM monitors all row-based order purge operations in the database but not partition-based purge operations. OSM audits the following operations:

- `om_new_purge_pkg.purge_orders`
- `om_new_purge_pkg.purge_selected_orders`
- `om_new_purge_pkg.delete_order`
- `om_new_purge_pkg.purge_cartridge_orders`

---

**Note:** You do not manually run the `om_new_purge_pkg.purge_cartridge_orders` package. Design Studio runs this package when the `PURGE_ORDER_ON_UNDEPLOY` cartridge management variable is set to `true` and you undeploy a cartridge. Design Studio does not run this package when the `FAST_CARTRIDGE_UNDEPLOY` cartridge management variable is set to `true` when you undeploy a cartridge.

OSM assigns each purge operation a unique purge Id that OSM associates with all audit records. You can monitor in-progress purges, review past purges, and analyze purge performance using the following views:

- **OM_AUDIT_PURGE_LATEST**: This view returns information about the latest order purge.
- **OM_AUDIT_PURGE_ALL**: This view returns information about all order purges.

You must set the `nls_date_format` database initialization parameter for queries to return the time portion in audit views that have `DATE` datatype columns. For example:

```
alter session set nls_date_format = 'DD-MM-YYYY HH24:MI:SS';
```

**Example (Monitoring an order purge):** You can use the `OM_AUDIT_PURGE_LATEST` view to monitor the latest order purge. If a purge is running, you can use a query like this one to find out the purge rate and estimated completion time.

```
select status,
       est_or_actual_end_date,
```
percent_complete,
orders_purged_per_minute as purge_rate,
parallelism
from om_audit_purge_latest;

<table>
<thead>
<tr>
<th>STATUS</th>
<th>EST_OR_ACTUAL_END_DATE</th>
<th>PERCENT_COMPLETE</th>
<th>PURGE_RATE</th>
<th>PARALLELISM</th>
</tr>
</thead>
<tbody>
<tr>
<td>RUNNING</td>
<td>15-09-2015 08:23:39</td>
<td>95.45</td>
<td>1911</td>
<td>16</td>
</tr>
</tbody>
</table>

Audit Tables
In most cases, the purge audit views provide sufficient data; however, you may need to query the underlying tables. For example you may need to review the orders that were purged, or the purge criteria, or both for troubleshooting purposes. OSM stores audit records in these tables:

- **OM_AUDIT_PURGE**: This is the main audit table. It stores the operation name, status, critical dates and other data.
- **OM_AUDIT_PURGE_ORDER**: Stores a synopsis of each purged order with a timestamp.
- **OM_AUDIT_PURGE_PARAM**: Stores the purge criteria, the parameters supplied to the purge procedure, and a snapshot of relevant session and configuration parameters when the purge was started.

By default, OSM retains the audit data for at least 90 days. The minimum retention period is specified by the `purge_audit_retention_days` configuration parameter in the `om_parameter` table. Order purge procedures purge obsolete audit records automatically before adding new audit records.

If you have partitioned the OSM schema, then OSM partitions the purge audit tables on a monthly basis. OSM manages the audit partitions automatically. The first order purge operation in a month automatically adds a new partition for the month. Order purge procedures drop partitions with obsolete audit records automatically before adding new audit records.

Managing Exchange Tables for Partition-Based Order Purge
Partition purge can purge a partition by exchanging it with tables that are known as exchange table. The exchange operation swaps the segments of the partition with the segments of the exchange table but does not delete or move data between segments. In general, the EXCHANGE PARTITION statement is a fast operation because it updates metadata only in the data dictionary (unless the table has global indexes and the exchange operation is performed with the UPDATE GLOBAL INDEXES option). However, to reclaim the storage previously used by the partition you must also purge the exchange tables.

There are two types of exchange tables used for purging:

- Purge tables
- Backup tables

About OSM Purge Tables
Purge tables are used for purging partitions using exchange. For each partition to be purged, `purge_partitions` decides whether to drop it or exchange it. For example, when purging online, it is possible to drop only empty partitions. When purging
offline, the partition where retained orders are consolidated must be exchanged, whereas the remaining partitions can be dropped.

The structure of each purge table is similar to the structure of the corresponding partitioned table. The number of purge tables that OSM creates depends on how many partitions you want to be able to exchange without having to purge the purge tables. This is called the exchange purge capacity.

For example, if the purge capacity is 3 then OSM creates 3 sets of purge tables as follows:

- If an OSM table is range-hash partitioned, OSM creates 3 hash-partitioned purge tables.
- If an OSM table is range partitioned, OSM creates 3 non-partitioned purge tables.
- If an OSM table is reference-partitioned, OSM creates 3\times N non-partitioned purge tables, where by default N is the number of hash sub-partitions of the oldest OM_ORDER_HEADER partition. You can override the default N when you setup the exchange tables.

The format of a purge table name is XCHG_OM_PRG_p$xchg_table_id$r, where:

- The XCHG_OM_PRG_ allows quick identification of purge tables.
- p is sequence number between 1 and the purge capacity, referred to as the logical exchange partition (formatted to 3-digits 001 to 999). Each partition to be exchanged is mapped to a logical exchange partition. This means that the maximum supported purge capacity is 999. This is the first generated component so that purge table names are grouped by partition when sorted.
- xchg_table_id is an OSM-generated sequence ID for each partitioned table, called the exchange table ID (formatted to 3 digits). OSM stores the exchange table IDs in the om_xchg_table table when OSM creates exchange tables. OSM purges this table when it drops exchange tables. You do not need to know the exchange table IDs.
- r is a 3-digit suffix that identifies which reference partition is exchanged when the table is reference partitioned; otherwise this value is omitted. r is referred to as the reference partition position because reference partitions are exchanged in order based on their position.

**Example:**

Figure 1 shows how OSM maps OM_ORDER_HEADER partitions to exchange tables when purging partitions. OSM maps OM_ORDER_HEADER to exchange table ID 001. All range-hash partitioned tables and the corresponding exchange tables have 64 hash partitions. The purge table capacity is 2, for example, there are two purge tables for OM_ORDER_HEADER. Assuming that the purge tables are empty, OSM can purge partitions P_000000000008000000 and P_000000000009000000 by exchanging them with exchange tables having logical exchange partitions 001 and 002, respectively.
**About OSM Backup Tables**

Backup tables are used for making a backup of orders to be retained before purging a partition. Orders are restored from the backup tables after the partition is purged. The structure of each backup table is similar to the structure of the corresponding partitioned table. Backup tables are always hash-partitioned in order to enable parallel inserts.

OSM creates a single set of backup tables, one for each partitioned table (whereas the number of purge tables depends on the purge capacity). The format of a backup table name is `XCHG_OM_BCK_$xchg_table_id$`, where:

- `$xchg_table_id` is the OSM-generated exchange table ID (formatted to 3 digits). This is identical to the exchange table ID used for purge tables.
- `XCHG_OM_BCK_` allows quick identification of backup tables.

**Creating Exchange Tables (Online or Offline)**

You must manually create exchange tables using the `om_part_maintain.setup_xchg_tables` procedure after you install the OSM schema. You must re-create them after upgrades, schema import, and after any ad hoc schema changes.

For more information see "`om_part_maintain.setup_xchg_tables (Online or Offline)`".

**Example**

If you want to exchange 3 partitions without having to purge exchange tables, run the `om_part_maintain.setup_xchg_tables` procedure as shown below. The procedure creates one set of backup tables and 3 sets of purge tables.

```
execute om_part_maintain.setup_xchg_tables(3);
```
Purging Exchange Tables (Online or Offline)
Eventually, you must reclaim the storage for exchanged partitions. You can do this with the `om_part_maintain.purge_xchg_prg_tables` procedure, which executes a TRUNCATE TABLE statement for each purge table with the DROP STORAGE option to reclaim the space. For more information see "om_part_maintain.purge_xchg_prg_tables (Online or Offline)."

It is not necessary to purge backup tables. These are purged automatically immediately after all order data is restored with the REUSE STORAGE option to retain the space of deleted rows. If you want to purge the backup tables without restoring orders, or if you want to reclaim their space, you can do this with the "om_part_maintain.purge_xchg_bck_tables (Online or Offline)" procedure.

Dropping Exchange Tables (Online or Offline)
You can drop exchange tables with the `om_part_maintain.drop_xchg_tables` procedure. By default, OSM automatically drops exchange tables before re-creating them.

For more information, see "om_part_maintain.drop_xchg_tables (Online or Offline)."

Estimating Partition Disk Space (Online or Offline)
Beginning with 7.2.2.4, the function `estimate_ptn_purged_space` returns the estimated amount of disk space (in bytes) that could be reclaimed by purging or dropping partitions. You can use this function to estimate:

- The amount of space that could be reclaimed during a maintenance window.
- The amount of space consumed by a partition, including global indexes.
- The average order size in a partition.

The `estimate_ptn_purged_space` function simulates a purge execution and determines the total numbers of bytes used by successive partitions that qualify for purging. The following valuable information about the purge simulation is available in the DBMS output:

- The partitions that qualify for purging and whether they are purged entirely or partially.
- Number of orders purged.
- Number of orders retained.
- Average size of an order (bytes).
- Estimated amount of space used by retained orders (bytes).
- Estimated amount of space reclaimed (bytes).

Consider the following when executing the `estimate_ptn_purged_space` function:

- Exchange tables must be created before calling this function.
- This function can be executed offline or online.
- This function assumes that you will execute `purge_partitions` with arguments `a_drop_empty_ptns` and `a_purge_xchg_prg_tables` set to true.
- The average order size is used to calculate the space used by retained orders. If retained orders are not typical sized orders the estimate returned from this function may not closely match the actual space reclaimed.
Example (Estimate the space that could be reclaimed in a maintenance window):
Consider an OSM schema with partitions P_000000000001000001, P_000000000002000001, and P_000000000003000001. The following statement estimates the space that could be reclaimed if partitions P_000000000001000001 and P_000000000002000001 were purged of all orders that were closed more than 30 days ago.

```sql
declare
begin
  dbms_output.put_line('Space Reclaimed(bytes):' ||
    om_part_maintain.estimate_ptn_purged_space(
      a_delete_before=> trunc(sysdate)-30,
      a_order_states=>om_new_purge_pkg.v_closed_orders,
      a_order_id_lt=>2000001,
      a_order_id_ge=>1));
end;
```

Example (Estimate the space consumed by a partition):
Consider an OSM schema with partitions P_000000000001000001, P_000000000002000001, and P_000000000003000001. The following statement estimates the space consumed by the first two partitions (including their share of global indexes).

```sql
declare
begin
  dbms_output.put_line('Partition size:' ||
    om_part_maintain.estimate_ptn_purged_space(
      a_delete_before=> om_const_pkg.v_no_date,
      a_order_states=>om_new_purge_pkg.v_all_orders,
      a_order_id_lt=>2000001,
      a_order_id_ge=>1));
end;
```

Managing Cartridges

There are three main components to a deployed cartridge:

- The Automation plugin EAR file, which is a Java EE application that is automatically generated by Design Studio and deployed to the OSM WebLogic domain. The Automation plugin EAR file contains the necessary logic to listen to external queues and invoke automation plugins when messages arrive on those queues. The automation plugin EAR file is initialized at startup.

- The static cartridge metadata that is populated in the OSM database when the cartridge is deployed or redeployed. This data does not grow or change when orders are created or processed. Cartridge metadata is loaded into the OSM server at startup and re-loaded when cartridges are deployed.

- The dynamic order data that is populated in the OSM database whenever an order is created and as it is being processed.

Your primary goals should be a) to minimize the memory needs and startup time of OSM and b) to deploy, redeploy, and undeploy cartridges quickly online. Because cartridge metadata consumes relatively little space in the database, purging cartridge metadata is not a major concern.

Cartridge metadata and the Automation plugin EAR file consume memory resources and take time to initialize on startup. You can minimize the memory needs and startup time of OSM by undeploying from the run-time production environment old cartridges that are no longer required.
To undeploy and redeploy cartridges quickly online, use Fast Undeploy instead of conventional undeploy. Fast Undeploy allows you to decouple deploy of the Automation plugin EAR file, purging the order data associated with a cartridge and purging the cartridge metadata.

Using Fast Undeploy

A cartridge can be undeployed when all associated orders are closed. There are two ways to undeploy a cartridge:

- Using conventional undeploy, which removes from the database both the cartridge metadata and all associated orders. This operation can be extremely expensive if you have a large volume of order data.
- Beginning with OSM 7.2.0.9, a new Fast Undeploy option is provided to rapidly undeploy a cartridge without removing the cartridge metadata and the associated order data from the database. When an OSM cartridge is undeployed using Fast Undeploy, OSM behaves the same as if the cartridge was undeployed using a conventional undeploy, that is, as if the cartridge and associated orders do not exist. The benefit of Fast Undeploy is that it allows the undeploy operation to complete quickly regardless of the number of orders that may have been created against that cartridge. Fast Undeploy is the default undeploy mode.

Oracle strongly recommends that you use Fast Undeploy instead of conventional undeploy. This enables you to undeploy unwanted cartridges quickly while offloading data purge to regular partitioned-based or row-based order purge, based on your data retention policies and maintenance schedule. This is useful both in development and production environments.

When you redeploy a cartridge, you have the option to undeploy the cartridge first. If you deployed the cartridge using fast undeploy, this operation is called a fast redeploy because the cartridge is fast undeployed before it is redeployed.

Fast Undeploy removes cartridges from the OSM WebLogic domain only. You must later remove undeployed cartridges from the database. For performance reasons, it is recommended that you remove undeployed cartridges only after all associated orders have been purged. Because cartridge metadata consumes relatively little space in the database, this can be an infrequent operation.

Purging Metadata of Undeployed Cartridges

A cartridge that was undeployed using Fast Undeploy has UNDEPLOYED status in the database. You can use the following statement to query the database for undeployed cartridges:

```sql
select * from om_cartridge where status = 'UNDEPLOYED';
```

You can purge an undeployed cartridge using the following statement:

```sql
execute om_cartridge_pkg.drop_cartridge(cartridge_id);
```

Configuration Parameters

The following configuration parameters affect partition maintenance operations:

- Parameters in the `om_parameter` table:
  - `range_partition_size`
  - `subpartitions_number`
- default_xchg_capacity
- xchg_retained_orders_thres
- degree_of_parallelism
- degree_of_parallelism_rebuild_indexes
- degree_of_parallelism_rebuild_xchg_indexes
- purge_job_class
- parallel_execute_chunk_size
- partition_auto_creation
- purge_policy_rebuild_unusable_indexes
- purge_policy_purge_related_orders_independently
- purge_policy_consolidate_partitions
- purge_policy_time_to_close_wait
- purge_audit_retention_days
- deferred_segment_creations (Oracle Database initialization parameter)

**range_partition_size**

This parameter in the om_parameter table specifies the size of new partitions. You choose the initial value of this parameter during installation. You can change it with the following SQL statement, where N is the new value (for example 100000):

```sql
update om_parameter
set value = N
where mnemonic = 'range_partition_size';
commit;
```

Updates to this parameter do not affect existing partitions. The upper bound of a new partition is the greatest partition upper bound plus the value of this parameter.

**subpartitions_number**

Specifies the number of hash sub-partitions. You choose the initial value of this parameter during installation. You can change it with the following SQL statement, where N is the new value (for example, 32).

```sql
update om_parameter
set value = N
where mnemonic = 'subpartitions_number';
commit;
```

Updates to this parameter do not affect existing partitions. If you change this parameter and you use om_part_maintain.purge_partitions for purging, you must re-execute om_part_maintain.setup_xchg_tables when it is time to purge partitions that were added after the change. This is because the number of hash partitions of the purge tables must match the number of hash sub-partitions of the range partitions to be purged.

**default_xchg_capacity**

Specifies the default purge capacity if om_part_maintain.setup_xchg_tables is called with an unspecified capacity. If it is not configured, the default is 3.
xchg_retained_orders_thres
If the number of orders to be excluded from purging in a partition exceeds this
threshold, the partition cannot be purged for performance reasons. The default is
10000. You can override the default in the om_parameter table.

degree_of_parallelism
Specifies the default degree of parallelism for statements that are executed in parallel.
It applies to queries, DML, and DDL statements. However, the degree of parallelism
for rebuilding indexes is configured by the degree_of_parallelism_rebuild_indexes
and degree_of_parallelism_rebuild_xchg_indexes parameters. If this parameter is not
specified, the default degree of parallelism is 4.

This parameter is also used for recreating global partitioned indexes when the
RECREATE GLOBAL policy is used. However, the degree of parallelism for rebuilding
index partitions is configured by the degree_of_parallelism_rebuild_indexes
and degree_of_parallelism_rebuild_xchg_indexes parameters. For more information, see
"purge_policy_rebuild_unusable_indexes." You can use the "om_part_maintain.set_
dop (Online or Offline)" procedure to set this parameter.

For more information, see "Parallel Execution".

degree_of_parallelism_rebuild_indexes
Specifies the default degree of parallelism for rebuilding index partitions of OSM
tables except exchange tables. If this parameter is not specified, the default degree of
parallelism is 2. This is less than the default value for degree_of_parallelism because
you cannot rebuild an entire partitioned index with a single statement. You must
rebuild each partition or sub-partition, which contains only a fraction of the data.
Therefore the overhead of increased parallelism may have negative impact on rebuild
performance. For example, performance tests might show that an optimal value for
degree_of_parallelism is 32 whereas the optimal value for degree_of_parallelism_
rebuild_indexes is only 4.

You can use the "om_part_maintain.set_dop_rebuild_indexes (Online or Offline)"
procedure to set this parameter.

The degree of parallelism for rebuilding indexes of exchange tables is configured with
the degree_of_parallelism_rebuild_xchg_indexes parameter.

degree_of_parallelism_rebuild_xchg_indexes
Specifies the default degree of parallelism for rebuilding index partitions of exchange
tables. If this parameter is not specified, the default degree of parallelism is 1. This is
because you cannot rebuild an entire partitioned index with a single statement. You
must rebuild each partition or sub-partition, which contains only a fraction of the data.
Because the size of exchange indexes is usually small rebuilding them serially is
usually faster.

You can use the "om_part_maintain.set_dop_rebuild_xchg_indexes (Online or
Offline)" procedure to set this parameter.

purge_job_class
This parameter in the om_parameter table specifies the class for purge jobs. A
database job must be part of exactly one class. The default value is DEFAULT_JOB_
CLASS, which is also the default database job class. If your database is Oracle RAC, jobs in the DEFAULT_JOB_CLASS class can run on any node.

If you use a partition purge strategy, restricting purge jobs to a single node significantly improves performance. Specifically, if the jobs that restore retained orders run on all nodes, cluster waits could account for 40% or more of the database time. Cluster waits increase with the degree of parallelism and the number of nodes. You can eliminate cluster waits by restricting job execution on a single node as follows:

1. Create a database service, for example, OSM_MAINTAIN, with a single preferred node and any number of available nodes. Refer to Oracle Database documentation for instructions about how to create a service using Oracle Enterprise Manager or srvctl.

2. Create a job class, for example, OSM_MAINTAIN, and associate it with the new service:

   ```
   exec dbms_scheduler.create_job_class(
       'OSM_MAINTAIN', service => 'OSM_MAINTAIN');
   ```

3. Grant EXECUTE permission on the job class to the OSM user:

4. Grant execute on sys. OSM_MAINTAIN to <user>;

5. Set the `purge_job_class` to the job class.

   Purge jobs will be spawned on the preferred node for this database service, if it is running; otherwise on an available node.

   If you use a row-based order purge strategy, running purge jobs on all nodes does not negatively affect performance. In fact, you may want to distribute the purge load on all nodes. However, if you do not want order purge to compete for resources with order processing, this parameter allows you to run order purge on a different node. For example, if you have an Oracle RAC database with 3 nodes, you could use two nodes for order processing and the third node for continuous order purge.

**parallel_execute_chunk_size**

This is an advanced parameter that specifies the chunk size for parallel execution using jobs. For more information, see "Tuning parallel_execute_chunk_size."

**partition_auto_creation**

This parameter in the om_parameter table specifies whether OSM is enabled to add a partition automatically when a new order ID does not map to any partition. Valid values are Y (enabled) and N. Oracle strongly recommends that you plan to add partitions manually and disable automatic creation for all production and performance environments, especially if you use Oracle RAC. Adding partitions online causes high contention in the database, resource busy exceptions and transaction timeouts that could result to failed orders and instability of OSM (especially during a busy period).

**purge_policy_rebuild_unusable_indexes**

This parameter in the om_parameter table specifies the default policy for rebuilding unusable indexes. Possible values are:

- `om_part_maintain.c_rebuild_idx_recreate_global` (RECREATE GLOBAL): This means that the preferred method to rebuild a global partitioned index that became unusable after a partition maintenance operation is to drop and recreate the entire index. This is the default, unless the global index is not partitioned, it supports a
unique constraint, or OSM is offline. Recreating a global partitioned index scans the table only once and it can be done efficiently with a high degree of parallelism, so it is more efficient and much faster than rebuilding each index partition separately. The default degree of parallelism for recreating global indexes is specified by the degree_of_parallelism parameter.

- **om_part_maintain.c_rebuild_idx_rebuild (REBUILD):** This means that the preferred method to rebuild global partitioned indexes is one partition at a time using ALTER INDEX REBUILD PARTITION. The default degree of parallelism for rebuilding index partitions is specified by the degree_of_parallelism_rebuild_indexes parameter.

**purge_policy_purge_related_orders_independently**

This parameter in the om_parameter table specifies whether orders should be purged independently of any related orders they may have. Valid values are Y (purge independently is enabled) and N (purge independently is disabled). By default, orders are purged independently. For more information, see the purge policy section in "Purging Related Orders Independently."

**Important:** Setting purge_policy_purge_related_orders_independently to N may add several minutes to the time it takes to purge or drop a partition.

**purge_policy_consolidate_partitions**

This parameter in the om_parameter table specifies the number of partitions to consolidate into a single partition when purging. Valid values are between 1 and 10 and the default value is 3. For example, a value of 5 means the purge procedure can combine the retained orders of up to 5 successive partitions into a single partition and drop the other 4 partitions.

In order for partitions to be consolidated, the following conditions must be satisfied:

- Partitions can be dropped (argument a_drop_empty_ptns is true)
- Purging is done offline (argument a_online is false)
- Purge capacity is not exhausted

**purge_policy_time_to_close_wait**

This purge policy can improve the performance of row-based purges and decrease purge rate fluctuations. The policy specifies a delays time before beginning to purge eligible orders so that the majority of the orders that were created on the same day are closed. The goal is to decrease I/O. For example, if 80% of orders complete in 4 days and the remaining 20% complete slowly over a much longer period, you could set purge_policy_time_to_close_wait to 4.

**purge_audit_retention_days**

This parameter in the om_parameter table specifies the minimum number of days to retain purge audit data. The default is 90 days. OSM automatically purges the audit data after the data exceeds this time limit. For more information see "Auditing and Monitoring Order Purges."
deferred_segment_creation

Oracle Database introduced deferred segment creation in 11gR2. If the `deferred_segment_creation` initialization parameter is set to true (the default), it forces the database to wait until the first row is inserted into a table/partition before creating segments for that table/partition and its dependent objects. In general, deferred segment creation saves disk space for unused tables/partitions. The main benefit to OSM is that it minimizes the time it takes to create a partition. However, in high volume deployments, especially on Oracle RAC, deferred segment creation can lead to serious performance issues when the database is forced to create the deferred segments of a partition in order to store new orders. This occurs when the previous partition is exhausted. The result is high "library cache lock" waits that could last for an extended period of time (frequently more than 30 minutes). In high volume deployments, it is strongly recommended that you disable deferred segment creation.

To disable deferred segment creation, log in to the database as the SYS user and execute the following statements:

```sql
alter system set deferred_segment_creation=false scope=both sid='*';
exeexecute dbms_space_admin.materialize_deferred_segments('<schema_name>');
```

About PL/SQL API

This section provides an overview of PL/SQL API.

DBMS Output

It is strongly recommended that you spool DBMS output to a file, especially for partition maintenance operations. The DBMS output includes valuable information for troubleshooting and performance tuning, such as elapsed execution times and error traces.

**Note:** The DBMS output is sent to the client at the end of execution. Oracle Database does not provide any mechanism to flush output during the execution of a procedure.

For example, if you use SQL*Plus:

```sql
SQL> set serveroutput on
SQL> spool part_maintain.out
SQL> execute om_part_maintain.drop_partitions(4000000);
SQL> execute om_part_maintain.add_partitions(2);
SQL> spool off;
```

Specifying Purge Criteria

Table 8–3 lists and describes the Purge criteria:

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Parameters</th>
<th>Partition-Based Purge</th>
<th>Row-Based Order Purge</th>
</tr>
</thead>
<tbody>
<tr>
<td>Order state</td>
<td><code>a_order_states</code></td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Order timestamp</td>
<td><code>a_delete_before</code></td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Order id range</td>
<td><code>a_order_id_lt</code></td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Each bit in the `a_order_states` parameter represents an order state. A number of constants are defined in the `om_new_purge_pkg` package. For example, if you want to purge only closed, cancelled, or failed orders, use `om_new_purge_pkg.v_closed_or_cancelled_orders + om_new_purge_pkg.v_failed_orders`.

Table 8–4 shows the order state constants and their corresponding values.

### Table 8–4  Order State Constants and Values

<table>
<thead>
<tr>
<th>Constant</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>v_completed_orders</code></td>
<td>1</td>
</tr>
<tr>
<td><code>v_aborted_orders</code></td>
<td>2</td>
</tr>
<tr>
<td><code>v_closed_orders</code></td>
<td><code>v_completed_orders + v_aborted_orders</code></td>
</tr>
<tr>
<td><code>v_not_started_orders</code></td>
<td>4</td>
</tr>
<tr>
<td><code>v_suspended_orders</code></td>
<td>8</td>
</tr>
<tr>
<td><code>v_cancelled_orders</code></td>
<td>16</td>
</tr>
<tr>
<td><code>v_closed_or_cancelled_orders</code></td>
<td><code>v_completed_orders + v_aborted_orders + v_cancelled_orders</code></td>
</tr>
<tr>
<td><code>v_wait_for_revision_orders</code></td>
<td>32</td>
</tr>
<tr>
<td><code>v_failed_orders</code></td>
<td>64</td>
</tr>
<tr>
<td><code>v_not_running_orders</code></td>
<td><code>v_failed_orders + v_wait_for_revision_orders + v_cancelled_orders + v_suspended_orders + v_not_started_orders</code></td>
</tr>
<tr>
<td><code>v_in_progress_orders</code></td>
<td>128</td>
</tr>
<tr>
<td><code>v_amending_orders</code></td>
<td>256</td>
</tr>
<tr>
<td><code>v_cancelling_orders</code></td>
<td>512</td>
</tr>
<tr>
<td><code>v_compensating_orders</code></td>
<td><code>v_cancelling_orders + v_amending_orders</code></td>
</tr>
<tr>
<td><code>v_running_orders</code></td>
<td><code>v_compensating_orders + v_in_progress_orders</code></td>
</tr>
<tr>
<td><code>v_open_orders</code></td>
<td><code>v_running_orders + v_not_running_orders</code></td>
</tr>
</tbody>
</table>
The `a_delete_before` parameter allows you to further narrow the purge criteria based on the order timestamp (for example, you might want to retain closed orders for at least 30 days). Table 8–5 shows which timestamp in the `om_order_header` table is compared to `a_delete_before` depending on `a_order_states` and the order status.

### Table 8–5  Order Purge Based on Timestamp, Order State, and Order Status

<table>
<thead>
<tr>
<th><code>a_order_states</code></th>
<th>Order Status</th>
<th>Timestamp</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>v_all_orders</code></td>
<td>N/A</td>
<td><code>a_ord_creation_date</code></td>
</tr>
<tr>
<td><code>v_closed_orders</code></td>
<td>7 (complete) or 9 (aborted)</td>
<td><code>a_ord_completion_date</code></td>
</tr>
<tr>
<td><code>v_completed_orders</code></td>
<td>7</td>
<td><code>a_ord_completion_date</code></td>
</tr>
<tr>
<td><code>v_aborted_orders</code></td>
<td>9</td>
<td><code>a_ord_completion_date</code></td>
</tr>
<tr>
<td><code>v_suspended_orders</code></td>
<td>2</td>
<td><code>a_ord_txn_completion_date</code></td>
</tr>
<tr>
<td><code>v_cancelled_orders</code></td>
<td>3</td>
<td><code>a_ord_txn_completion_date</code></td>
</tr>
<tr>
<td><code>v_in_progress_orders</code></td>
<td>4</td>
<td><code>a_ord_txn_completion_date</code></td>
</tr>
<tr>
<td><code>v_amending_orders</code></td>
<td>5</td>
<td><code>a_ord_txn_completion_date</code></td>
</tr>
<tr>
<td><code>v_cancelling_orders</code></td>
<td>6</td>
<td><code>a_ord_txn_completion_date</code></td>
</tr>
<tr>
<td><code>v_wait_for_revision_orders</code></td>
<td>8</td>
<td><code>a_ord_txn_completion_date</code></td>
</tr>
<tr>
<td><code>v_failed_orders</code></td>
<td>10</td>
<td><code>a_ord_txn_completion_date</code></td>
</tr>
<tr>
<td><code>v_not_started_orders</code></td>
<td>1 (not started)</td>
<td><code>a_ord_creation_date</code></td>
</tr>
</tbody>
</table>

**Parallel Execution**

The `om_part_maintain` API performs many operations in parallel:

- Parallel queries and most DML and DDL statements are executed in parallel using parallel servers, which apply multiple CPU and I/O resources to the execution of a single database operation. Examples include copying orders into the backup tables and rebuilding unusable indexes.

- Some operations are executed in parallel using the `dbms_parallel_execute` package, which divides work into chunks processed in parallel by database jobs. Row-based order purge and the restore stage of `purge_partitions` are performed this way. If your database is Oracle RAC, it is recommended that you create a database job class to restrict job execution on a single node to eliminate cluster waits. For more information, see "purge_job_class."

Procedures that support parallelism use the `a_parallelism` parameter, which allows you to specify the desired degree of parallelism for those statements that can be executed in parallel.

The degree of parallelism can be:

- Greater than 1: Statements that can be executed in parallel are executed with the specified degree of parallelism.
1: All statements are executed serially.

0: The degree of parallelism is computed by the database and it can be 2 or greater. Statements that can be executed in parallel always run in parallel.

Less than 0: The degree of parallelism is computed by the database and it can be 1 or greater. If the computed degree of parallelism is 1, the statement runs serially. Indexes are always rebuilt in parallel.

If you leave a \texttt{parallelism} unspecified, OSM uses the default parallelism configured by these parameters:

- \texttt{degree\_of\_parallelism}
- \texttt{degree\_of\_parallelism\_rebuild\_indexes}
- \texttt{degree\_of\_parallelism\_rebuild\_xchg\_indexes}

### Concurrency Restrictions

Exchange table, partition management, and purge procedures acquire an exclusive user lock to prevent concurrent execution of other management procedures, which could result in unrecoverable errors. Each OSM schema uses a different lock specifically for this package. The lock is released automatically at the end of execution. The database also releases user locks automatically when a session terminates.

Specifically, the following procedures acquire an exclusive lock to prevent concurrent execution:

- \texttt{setup\_xchg\_tables}
- \texttt{drop\_xchg\_tables}
- \texttt{purge\_xchg\_prg\_tables}
- \texttt{purge\_partitions}
- \texttt{drop\_empty\_partitions}
- \texttt{drop\_partitions}
- \texttt{add\_partition} and \texttt{add\_partitions}
- \texttt{equipartition}
- \texttt{purge\_orders}
- \texttt{select\_orders}
- \texttt{purge\_selected\_orders}
- \texttt{resume\_purge}

### PL/SQL API Reference

The PL/SQL API provides procedures and functions for:

- Setup and tuning
- Maintenance
- Troubleshooting and recovery

### Setup and Tuning Procedures

This section provides information about setup and tuning PL/SQL API procedures.
om_part_maintain.setup_xchg_tables (Online or Offline)
This procedure creates exchange tables for purging partitions with om_part_maintain.purge_partitions.

procedure setup_xchg_tables(
    a_xchg_purge_capacity natural default null,
    a_tablespace varchar2 default null,
    a_force boolean default false,
    a_subpartition_count_override positive default null
) ;

If you purge partitions, you must create exchange tables after a new installation and each time you upgrade the schema. If the exchange tables are not up to date, om_part_maintain.purge_partitions reports an error. If you only drop partitions, exchange tables are not required.

This procedure first calls drop_xchg_tables to drop all existing exchange tables and reclaim space. If a_force is false and an exchange table is not empty, it throws an exception. Upon successful completion, it sets the sys$xchg_purge_capacity and sys$xchg_purge_seq system parameters to the purge capacity and 1, respectively (in the om_parameter table).

The parameters are:

- **a_xchg_purge_capacity**: Specifies the exchange capacity in the range 0-999. If it is not specified, it uses the value of the default_xchg_capacity parameter configured in the om_parameter table. If default_xchg_capacity is not set, the default capacity is 3. If the specified capacity is 0 then it creates backup tables but not purge tables. If the specified or configured capacity is illegal, it throws an exception.
- **a_tablespace**: Specifies the tablespace where you want the exchange tables to be created. If you do not specify it, the database default tablespace is used.
- **a_force**: Specifies whether existing exchange tables should be dropped even if they are non-empty. If this is false and an exchange table is not empty, it throws an exception. In this case, exchange tables are left in an inconsistent state (new exchange tables are not created but existing exchange tables might be partially dropped).
- **a_subpartition_count_override**: Specifies the number of hash partitions for exchange tables. Oracle Database does not allow a range-hash partition to be exchanged with the hash-partitioned table if the number of hash partitions of the range partition and the table do not match. By default, the number of hash partitions of the exchange tables for om_order_header is the same as the number of hash sub-partitions of the oldest om_order_header partition. If you need to purge partitions with a different number of hash sub-partitions (because you changed the subpartitions_number parameter), re-execute setup_xchg_tables and supply the right value for this parameter.

om_part_maintain.drop_xchg_tables (Online or Offline)
This procedure drops all exchange tables. It is executed automatically when you execute setup_xchg_tables.

procedure drop_xchg_tables(a_force boolean default false) ;

The implementation first purges exchange metadata. Specifically, it purges the om_xchg_table table, and sets sys$xchg_purge_capacity and sys$xchg_purge_seq in the om_parameter table to 0.

The DROP TABLE statements are executed with the PURGE option, so the space is released immediately (you cannot recover the exchange tables from the recycle bin).
If a_force is false and an exchange table is not empty, it throws an exception. In this case exchange tables are left in an inconsistent state.

**om_part_maintain.set_dop (Online or Offline)**
This procedure sets the degree_of_parallelism parameter in the om_parameter table to the specified degree of parallelism.

```plsql
procedure set_dop(a_parallelism binary_integer);
```

**om_part_maintain.set_dop_rebuild_indexes (Online or Offline)**
This procedure sets the degree_of_parallelism_rebuild_indexes parameter in the om_parameter table to the specified degree of parallelism.

```plsql
procedure set_dop_rebuild_indexes(a_parallelism binary_integer);
```

**om_part_maintain.set_dop_rebuild_xchg_indexes (Online or Offline)**
This procedure sets the degree_of_parallelism_rebuild_xchg_indexes parameter in the om_parameter table to the specified degree of parallelism.

```plsql
procedure set_dop_rebuild_xchg_indexes(a_parallelism binary_integer);
```

### Maintenance Procedures and Functions
This section provides information about maintenance procedures and functions.

**om_part_maintain.add_partition (Offline Only)**
This procedure adds a single partition.

```plsql
procedure add_partition(a_tablespace varchar2 default null);
```

The implementation is equivalent to this call:

```plsql
add_partitions(1, a_tablespace);
```

**om_part_maintain.add_partitions (Offline Only)**
This procedure adds one or more partitions. At the end, it also rebuilds any unusable indexes as a precaution (normally indexes should remain usable).

```plsql
procedure add_partitions(
    a_count positive,
    a_tablespace varchar2 default null);
```

The upper bound of each new partition is the greatest partition upper bound plus the value of the range_partition_size parameter. The upper bound is used in the partition name. For example, if the upper bound is 100,000 then the partition name is P_00000000000100000 (always formatted to 18 characters).

This procedure must be executed offline. Online execution causes high contention in the database and transaction timeouts that could result in failed orders and instability of OSM.

The parameters are:

- **a_count**: The number of partitions to add.
- **a_tablespace**: The tablespace for the new partitions. This procedure modifies the default tablespace attribute of partitioned tables with the specified tablespace before adding partitions. If you do not specify the tablespace or the input
argument is null, each partition is created on the default tablespace of the partitioned table (for example, on the same tablespace as the most recently added partition).

**Dropping newly added partitions**: If you want to drop several new partitions, perhaps because you want to re-create them (for example, with a different number of hash sub-partitions and/or on a different tablespace) or because you inadvertently added a large number of partitions, you can drop those partitions that are still empty using *drop_empty_partitions*.

**om_part_maintain.drop_partitions (Offline Only)**

```plsql
procedure drop_partitions(
    a_order_id_lt number,
    a_order_id_ge integer default null,
    a_parallelism_rebuild_indexes binary_integer default null);
```

This procedure drops partitions that satisfy the following conditions:

- The order IDs mapped to this partition are within the specified range.
- All orders are either closed (complete or aborted) or canceled.
- If the *om_parameter purge_policy_purge_related_orders_independently* is set to 'N' and the partition contains orders with related open orders, the partition cannot be dropped. For more information, see the purge policy section in "Purging Related Orders Independently."

More precisely:

- It disables all foreign keys that reference the partitioned tables.
- It drops partitions that satisfy the aforementioned conditions. However, if all partitions satisfy those conditions, the partition with the greatest upper bound is not dropped. Oracle Database requires that a partitioned table have at least one partition. For example, if you have only one partition, you cannot use *drop_partitions* to reclaim space. In this case, use *om_part_maintain.purge_partitions*.
- It re-enables the disabled foreign keys (with NOVALIDATE, for performance reasons).
- It rebuilds unusable indexes and index partitions in parallel.
- It deletes any remaining order data that references orders in the partitions dropped.

Oracle recommends that you back up the OSM schema prior to executing this procedure.

This procedure must be executed offline.

The parameters are:

- **a_order_id_lt**: Specifies a non-inclusive upper bound for the range of order IDs mapped to the partitions to be dropped. If it is not null then only partitions with an upper bound less than or equal to this value are considered. (The upper bound of a partition is non-inclusive, that is, the order IDs mapped to that partition are strictly less than its upper bound.)

- **a_order_id_ge**: Specifies an inclusive lower bound for the range of order IDs mapped to the partitions to be dropped. If it is not null then only partitions with a lower bound greater than or equal to this value are considered. (The lower bound of a partition is the upper bound of the previous partition, if any; otherwise 1.)
- **a_parallelism_rebuild_indexes**: Specifies the degree of parallelism for rebuilding unusable indexes. It is recommended that you leave it null. The implementation will choose the optimal method for each unusable index depending on the index type and configuration parameters. For more information, see "purge_policy_rebuild_unusable_indexes."

**om_part_maintain.drop_empty_partitions (Online or Offline)**

This procedure drops empty partitions with mapped order IDs within the specified range.

```plsql
procedure drop_empty_partitions(
    a_order_id_lt integer default null,
    a_order_id_ge integer default null);
```

This procedure is similar to `drop_partitions` except that:

- It ignores non-empty partitions. It exits when it encounters a partition with an upper bound greater than `a_order_id_lt`.
- It does not delete data from non-partitioned tables. It assumes it is already deleted.
- It does not disable foreign keys.
- It can be executed online. However, in this case you might experience high contention due to exclusive locks acquired by Oracle Database. Oracle recommends that you execute this procedure either offline or off-peak.

If all partitions are empty and within the specified range, the partition with the greatest upper bound are not dropped. This is because Oracle Database requires that each partitioned table have at least one partition.

The parameters are:

- **a_order_id_lt**: Specifies a non-inclusive upper bound for the range of order IDs mapped to the partitions to be dropped. If it is not null then only partitions with an upper bound less than or equal to this value can be dropped. (The upper bound of a partition is non-inclusive, that is, the order IDs mapped to that partition are strictly less than its upper bound.)

- **a_order_id_ge**: Specifies an inclusive lower bound for the range of order IDs mapped to the partitions to be dropped. If it is not null then only partitions with a lower bound greater than or equal to this value can be dropped. (The lower bound of a partition is the upper bound of the previous partition, if any; otherwise 1.)

Exceptions:

- **ORA-20166**: There is another in-progress maintenance operation.
- **ORA-20170**: Failed to suspend database jobs.
- **ORA-20171**: OSM is running.

**Example (dropping empty partitions after a purge)**: Assume that `purge_partitions` left some partitions empty as shown in Table 8–6.

```sql
select partition_name, high_value
from user_tab_partitions
where table_name = 'OM_ORDER_HEADER'
order by partition_name;
```
### Example: Dropping Empty Partitions

<table>
<thead>
<tr>
<th>PARTITION_NAME</th>
<th>HIGH_VALUE</th>
<th>Empty?</th>
</tr>
</thead>
<tbody>
<tr>
<td>P_000000000000100001</td>
<td>100001</td>
<td>Yes</td>
</tr>
<tr>
<td>P_000000000000200001</td>
<td>200001</td>
<td>--</td>
</tr>
<tr>
<td>P_000000000000300001</td>
<td>300001</td>
<td>Yes</td>
</tr>
<tr>
<td>P_000000000000400001</td>
<td>400001</td>
<td>Yes</td>
</tr>
<tr>
<td>P_000000000000500001</td>
<td>500001</td>
<td>--</td>
</tr>
</tbody>
</table>

The following statement drops P_000000000000100001, ignores P_000000000000200001, drops P_000000000000300001, and stops at P_000000000000400001 because the upper bound of this partition is greater than 300001:

```sql
execute om_part_maintain.drop_empty_partitions(300001);
```

### Example (dropping newly added partitions)

Suppose you want to drop several new partitions, perhaps because you want to re-create them (for example, with a different number of hash sub-partitions and/or on a different tablespace) or because you inadvertently added a large number of partitions. Specifically, assume that you want to drop partitions P_000000000000600001, P_000000000000700001 and P_000000000000800001 shown in Table 8–7.

### Example: Dropping Newly Added Partitions

<table>
<thead>
<tr>
<th>PARTITION_NAME</th>
<th>HIGH_VALUE</th>
<th>Mapped Order IDs</th>
<th>Empty?</th>
</tr>
</thead>
<tbody>
<tr>
<td>P_000000000000200001</td>
<td>200001</td>
<td>1-200000</td>
<td>--</td>
</tr>
<tr>
<td>P_000000000000400001</td>
<td>400001</td>
<td>200001-400000</td>
<td>Yes</td>
</tr>
<tr>
<td>P_000000000000500001</td>
<td>500001</td>
<td>400001-500000</td>
<td>--</td>
</tr>
<tr>
<td>P_000000000000600001</td>
<td>600001</td>
<td>500001-600000</td>
<td>Yes</td>
</tr>
<tr>
<td>P_000000000000700001</td>
<td>700001</td>
<td>600001-700000</td>
<td>Yes</td>
</tr>
<tr>
<td>P_000000000000800001</td>
<td>800001</td>
<td>700001-800000</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Execute the following statement:

```sql
execute om_part_maintain.drop_empty_partitions(
    a_order_id_ge => 500001);
```

### om_part_maintain.purge_partitions (Online or Offline)

```sql
procedure purge_partitions(
    a_online boolean,
    a_delete_before date,
    a_order_states integer default om_new_purge_pkg.v_closed_orders,
    a_order_id_lt integer default null,
    a_order_id_ge integer default null,
    a_stop_date date default om_const_pkg.v_no_date,
    a_drop_empty_ptns boolean default true,
    a_purge_xchg_prg_tables boolean default false,
    a_parallelism binary_integer default null);
```

This procedure purges the partitions that satisfy these conditions:

- If executed online:
- All of the orders are either closed (complete or aborted) or canceled.
- All of the contained orders satisfy the purge criteria specified by the `a_delete_before` and `a_order_states` arguments.
- All of the contained order IDs are within the purge range specified by `a_order_id_lt` and `a_order_id_ge`. The range of mapped order IDs does not need to be a subset of the specified range. What matters is the range of actual order IDs.

- If executed offline:
  - Some or all of the contained orders satisfy the purge criteria specified by the `a_delete_before` and `a_order_states` arguments.
  - Some or all of the contained order IDs are within the purge range specified by `a_order_id_lt` and `a_order_id_ge`. The range of mapped order IDs does not need to be a subset of the specified range. What matters is the range of actual order IDs.
  - The number of orders to be excluded from purging (for example, those orders that do not satisfy the previous two conditions) does not exceed the threshold specified by the `xchg_retained_orders_thres` parameter.

Oracle recommends that you back up the OSM schema prior to executing this procedure and that you gather statistics after you finished purging.

If you execute this procedure online, you might experience high contention due to exclusive locks acquired by Oracle Database. Oracle recommends that you execute this procedure either offline or off-peak.

If you execute this procedure offline, you can purge a partition that contains orders that do not satisfy the purge criteria as long as the number of retained orders in that partition does not exceed the threshold specified by the `xchg_retained_orders_thres` parameter. In this case, the retained orders are copied to the backup tables prior to the exchange operation and they are restored (copied again) into the partitioned tables after the exchange operation. Because these are relatively expensive operations, the threshold ensures that they will complete in a timely fashion. Both backup and restore are executed in parallel as specified by the `a_parallelism` argument.

If this procedure is executed offline, it disables foreign keys. This is necessary when purging partitions with retained orders. Disabling foreign keys is unsafe to do when the OSM application is online as it can result in data integrity violations. Therefore disabling foreign keys requires OSM be offline until they are re-enabled.

Partitions are purged one by one end-to-end, that is, from all partitioned tables. For example, if you want to purge partitions `P_000000000001000001`, `P_000000000002000001`, and `P_000000000003000001` then `P_000000000001000001` will be purged first from all partitioned tables, then `P_000000000002000001` and so on.

This procedure can consolidate retained orders from multiple partitions into a single partition, to maximize reclaimed space, reduce the number of partitions, and minimize downtime. This is done by purging successive partitions in iterations. The maximum number of partitions consolidated in each iteration is limited by the parameter `purge_policy Consolidate Partitions`. More precisely, this procedure purges successive partitions that qualify for purging as follows:

1. Copies the orders that do not satisfy the purge criteria from those partitions into the backup tables. This is a relative fast operation because it is performed in parallel and the backup tables have few indexes and constraints.
2. Purges each partition entirely by exchanging it with purge tables. This is a fast operation because EXCHANGE PARTITION only updates metadata in the data dictionary.

3. Drops N-1 of those partitions. This is a fast operation because the partitions are now empty.

4. Restores the retained orders from the backup tables into the Nth partition with their order IDs unchanged. This is also performed in parallel using the `dbms_parallel_execute` package. However, this step is slower than backup because the partitioned tables have more indexes and constraints.

The EXCHANGE PARTITION operation is performed with the following options:

- INCLUDING INDEXES: This means that local index partitions or subpartitions are also exchanged. This ensures that local indexes remain usable during the exchange, for example, they do not have to be rebuilt.

- WITHOUT VALIDATION: By default, the exchange operation is performed WITH VALIDATION, which means that Oracle Database returns an error if any rows in the exchange table do not map into partitions or subpartitions being exchanged. This check is unnecessary when the exchange table is empty.

- If this procedure is executed online and the table has global indexes that enforce unique constraints then the exchange is performed with the following options:
  - UPDATE GLOBAL INDEXES: This means that global indexes are updated during the exchange and therefore remain usable. Otherwise, unusable global indexes that enforce unique constraints would result in ORA-01502 exceptions. (By default, unusable global indexes that do not enforce unique constraints are ignored and therefore are not an issue – this is controlled by the `SKIP_UNUSABLE_INDEXES` initialization parameter. Therefore, if a table has no such global indexes or if this procedure is executed offline, rebuilding unusable global indexes is deferred for performance reasons.)
  - PARALLEL: This means that global indexes are updated in parallel for performance reasons. It does not alter the global indexes to parallel.

After each partition is purged end-to-end, the `sys$xcg_purge_seq` counter in the `om_parameter` table increments to the next logical exchange partition. When the logical exchange partition exceeds the purge capacity, this counter cycles to 1.

The procedure exits when:

- Time expires.
- It encounters a partition with a lower bound greater than or equal to the upper bound of the specified range.
- The number of hash sub-partitions of the next `om_order_header` partition is different than the number of partitions of the corresponding exchange table. The number of hash partitions of each exchange table is the same as the same number of hash sub-partitions of the oldest partition of the corresponding range-hash partitioned table. If newer partitions have a different number of hash sub-partitions (because you changed the `subpartitions_number` parameter) then you will not be able to purge the newer partitions until you drop the older partitions and re-execute `setup_xchg_tables`.

All disabled constraints are re-enabled at the end (with `NOVALIDATE` for performance reasons).

The parameters are:
- **a_online**: Specifies whether this procedure is being executed online. If it is true, it ignores partitions with open orders and partitions with orders that do not satisfy the purge criteria (only entire partitions can be purged online).

- **a_delete_before**: Only orders with a timestamp older than this date and time are eligible for purging. For more information, see "Specifying Purge Criteria".

- **a_order_states**: Only orders with one of these states are eligible for purging. By default, only closed orders are eligible for purging. For more information, see "Specifying Purge Criteria".

- **a_order_id_lt** and **a_order_id_ge**: If **a_order_id_ge** is not null then only orders with order ID greater than or equal to this value are eligible for purging. If **a_order_id_lt** is not null then only orders with order ID less than to this value are eligible for purging. If **a_order_id_lt** is null, it will be defaulted to the non-inclusive upper bound of the latest used partition. (This ensures that new empty partitions beyond the currently active partition are not dropped accidentally.) If a partition contains both order IDs in this range and outside this range then the partition cannot be purged unless the out-of-range orders can be retained (for example, the purge is done offline and the total number of retained orders in that partition does not exceed the threshold specified by the **xchg_retained_orders_thres** parameter).

- **a_stop_date**: If it is not null then the procedure exits when the date and time are reached. This is done on a best-effort basis, since a premature exit could leave data in inconsistent state. The time is checked periodically. The elapsed time between checks could be as high as the time it takes to purge as many partitions as the spare purge capacity. Only non-critical deferrable operations are skipped when the time expires, such purging exchange tables.

- **a_drop_empty_ptns**: Specifies whether empty partitions should be dropped. The default is true, since dropping empty partitions is a fast operation. In this case, this procedure can purge as many successive partitions at a time as the spare capacity, which reduces the time it takes to restore orders and therefore downtime. If this is argument is false, each partition to be purged must go through the backup-purge-restore process separately.

- **a_purge_xchg_prg_tables**: Specifies whether exchange tables should be purged as well. If it is true then it purges exchange tables, as long as time has not expired and at least one partition was purged. This is relatively slow operation, so the default is false. In this case, the number of partitions that can be purged by a single execution of this procedure is limited by the space purge capacity.

- **a_parallelism**: Specifies the degree of parallelism for backup and restore operations. If it is null, it uses the parallelism configured by the **degree_of_parallelism** parameter. For more information, see "Parallel Execution".

**Exceptions**: This procedure performs a number of checks to ensure it can proceed with purge. If a check fails, it throws one of the following exceptions:

- **ORA-20142**: The schema is not equi-partitioned. Execute the equi-partition procedure.

- **ORA-20160**: The schema is not partitioned. You can only use this procedure if your schema is partitioned.

- **ORA-20162**: There are no exchange tables. Execute the **setup_xchg_tables** procedure.
Managing the OSM Database Schema

ORA-20163: The exchange tables are not up-to-date. This means that the schema has been upgraded after the exchange tables were created. Re-execute the `setup_xchg_tables` procedure.

ORA-20166: There is another in-progress maintenance operation.

ORA-20170: Failed to suspend database jobs.

ORA-20171: The procedure was executed with `a_online`=false and it detected that OSM is running.

Example (purge all orders that were closed at least 180 days ago): Suppose you want to purge all complete or aborted orders that were closed at least 180 days ago. Assuming that most partitions contain some orders that do not satisfy these criteria, you decided to execute `purge_partitions` offline. You also want to defer dropping empty partitions and purging the exchange tables until the system is restarted. This is how you can do it:

```sql
begin
    om_part_maintain.purge_partitions(
        a_online => false,
        a_delete_before => trunc(sysdate) - 180,
        a_order_states => om_new_purge_pkg.v_closed_orders,
        a_drop_empty_ptns => false,
        a_purge_xchg_prg_tables => false,
        a_parallelism => 4)
end;
```

Example (ignore old partitions that contain only a few orders): This example adds to the scenario of the previous example. Assume that old partitions with non-inclusive upper bound up to 5600000 contain a small number of orders that can be purged but cannot be purged entirely (for example, because they still contain open orders). Purging those partitions would be unproductive, since it could exhaust the exchange capacity. Therefore you decided to use the `a_order_id_ge` parameter to ignore them for now:

```sql
begin
    om_part_maintain.purge_partitions(
        a_online => false,
        a_delete_before => trunc(sysdate) - 180,
        a_order_states => om_new_purge_pkg.v_closed_orders,
        a_order_id_ge => 5600000,
        a_drop_empty_ptns => false,
        a_purge_xchg_prg_tables => false,
        a_parallelism => 4)
end;
```

om_part_maintain.purge_entire_partition (Online or Offline)

```sql
procedure purge_entire_partition(
    a_online            boolean,
    a_partition_name    varchar2,
    a_purge_xchg_prg_tables boolean default false,
    a_purge_orphan_data     boolean default true) ;
```

This procedure purges the given partition entirely (all orders). The partition is not dropped. The following two calls are equivalent, assuming that the partition size is 100000:

```sql
execute om_part_maintain.purge_entire_partition(
    a_online => true,
    a_partition_name => 'P_000000000000400001');
```
execute om_part_maintain.purge_partitions(
    a_online => true,
    a_delete_before => om_const_pkg.v_no_date,
    a_order_states => om_new_purge_pkg.v_all_orders,
    a_order_id_lt => 400001,
    a_order_id_le => 300001,
    a_stop_date => null,
    a_drop_empty_ptns => false);

Parameters:

- **a_online**: Specifies whether this procedure is being executed online. If this parameter is true, it ignores partitions with open orders and partitions with orders that do not satisfy the purge criteria (only entire partitions can be purged online).

- **a_partition_name**: The name of the partition to purge.

- **a_purge_xchg_prg_tables**: Specifies whether exchange tables should be purged as well. If this parameter is true, it purges exchange tables, as long as time has not expired and at least one partition was purged. This is a relatively slow operation, so the default is false. In this case, the number of partitions that can be purged by a single execution of this procedure is limited by the space purge capacity.

- **a_purge_orphan_data**: Specifies whether you want orphan data to be purged after the partition is purged. The default is true. You may want to defer purging of orphan data if you used om_part_maintain.backup_selected_ords to manually backup selected orders, which you plan to restore with om_part_maintain.restore_orders.

**om_part_maintain.estimate_ptn_purged_space (Online or Offline)**

```plsql
function estimate_ptn_purged_space(
    a_delete_before date,
    a_order_states integer default om_new_purge_pkg.v_closed_orders,
    a_order_id_lt integer default null,
    a_order_id_ge integer default null)
return number;
```

This function estimates amount of disk space (in bytes) that is reclaimed by purging or dropping partitions.

This function simulates the execution of `om_part_maintain.purge_partitions`, therefore refer to the purge partitions API reference for a description of the parameters, exit conditions, and possible exceptions.

**Example (estimate the space reclaimed by purging all orders that were closed at least 180 days ago):**

```sql
declare
begin
    dbms_output.put_line('Space Reclaimed (bytes): '||om_part_maintain.estimate_ptn_purged_space(  
        a_delete_before => trunc(sysdate) - 180,  
        a_order_states => om_new_purge_pkg.v_closed_orders))  
end;
```

**Example (estimate the space reclaimed by dropping partitions):** The following example shows how to estimate the space reclaimed by dropping all partitions with an upper bound less than or equal to 300001. Note that the **a_delete_before** and **a_order_states** parameters have been set to values that include all orders in the partition.
declare
begin
dbms_output.put_line('Space Reclaimed (bytes): '||om_part_maintain. estimate_ptn_purged_space(
    a_delete_before => om_const_pkg.v_no_date,
    a_order_states => om_new_purge_pkg.v_all_orders,
    a_order_id_lt => 300001));
end;

om_part_maintain.purge_xchg_bck_tables (Online or Offline)
procedure purge_xchg_bck_tables(a_drop_storage boolean default false);

This procedure purges all exchange backup tables. Normally you do not need to execute this procedure because backup tables are purged automatically when all order data is restored. The implementation executes TRUNCATE TABLE, so purged data cannot be restored.

If a_drop_storage is true, backup tables are truncated with the DROP STORAGE option to reclaim space. Otherwise, they are truncated with the REUSE STORAGE option to retain the space from the deleted rows. If you never reclaim the space, its size is limited by the largest volume of order data copied into the backup tables. By default, space is reused for performance reasons and in order to minimize downtime: First, inserts are more efficient if space is already allocated. Second, purging the backup tables is faster if space is reused.

om_part_maintain.purge_xchg_prg_tables (Online or Offline)
procedure purge_xchg_prg_tables;

This procedure purges all exchange purge tables to reclaim space. It does not purge backup tables. The implementation executes TRUNCATE TABLE … DROP STORAGE, so purged data cannot be restored.

om_new_purge_pkg.delete_order (Online or Offline)
procedure delete_order(a_order_seq_id integer);

This procedure unconditionally deletes the given order from the database. Note that this procedure does not issue commit, in contrast to most purge procedures. It is the responsibility of the user to issue commit or rollback.

This operation is audited.

om_new_purge_pkg.purge_orders (Online or Offline)
procedure purge_orders(
    a_status                     out integer,
    a_delete_before              date,
    a_order_states               integer,
    a_stop_date                  date     default om_const_pkg.v_no_date,
    a_order_id_lt                integer  default null,
    a_order_id_ge                integer  default null,
    a_order_source_mnemonic      varchar2 default null,
    a_order_type_mnemonic        varchar2 default null,
    a_namespace_mnemonic         varchar2 default null,
    a_version_mnemonic           varchar2 default null,
    a_cartridge_id               integer  default null,
    a_commit_count               integer default v_default_count,
    a_parallelism binary_integer default null);
This procedure purges orders that satisfy the given criteria. It is the main implementation of row-based order purge. Orders are purged by database job. The procedure finds the order Ids that satisfy the purge criteria, inserts them into the OM_ORDER_ID_FOR_PURGE staging table, splits them into chunks, and distributes the chunks to database jobs for parallel purge. Each chunk is processed by deleting one order at a time with periodic commits. This approach ensures that a) an order is either purged entirely or not at all and b) a purge may succeed partially even in the event of errors.

This operation is audited.

Executing this procedure is equivalent to executing select_orders and purge_selected_orders. However, purge_orders always starts a new purge by clearing the OM_ORDER_ID_FOR_PURGE staging table, whereas select_orders only adds orders Ids to the staging table.

Table 8–8 describes the possible outcomes. The a_status output parameter is set accordingly.

**Table 8–8  Possible Outcomes**

<table>
<thead>
<tr>
<th>Outcome</th>
<th>a_status</th>
<th>OM_ORDER_ID_FOR_PURGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>No orders satisfy the purge criteria.</td>
<td>om_new_purge_pkg.v_status_nopurge</td>
<td>Empty</td>
</tr>
<tr>
<td>The purge finished successfully and all orders that satisfied the purge criteria were purged.</td>
<td>om_new_purge_pkg.v_status_finished</td>
<td>Cleared</td>
</tr>
<tr>
<td>The purge finished with errors (some orders were not purged). This procedure throws an exception. Note that this procedure retries purging of failed chunks a few times. An exception means that the retries also failed (or only partially succeeded). You can run resume_purge to retry.</td>
<td>N/A</td>
<td>Contains all order Ids that satisfy the purge criteria.</td>
</tr>
<tr>
<td>The purge finished prematurely because the expiration date specified by a_stop_date was reached.</td>
<td>om_new_purge_pkg.v_status_expired</td>
<td>Cleared</td>
</tr>
<tr>
<td>The purge was stopped by the user (using stop_purge), and it can be resumed using resume_purge.</td>
<td>om_new_purge_pkg.v_status_stopped</td>
<td>Contains all order Ids that satisfy the purge criteria.</td>
</tr>
</tbody>
</table>

Parameters:

- **a_status**: Returns the purge status.

- **a_delete_before**: Only orders with a timestamp older than this date and time are eligible for purging. See the Specifying Purge Criteria section for more information.

- **a_order_states**: Only orders with one of these states are eligible for purging. See the Specifying Purge Criteria section for more information.

- **a_stop_date**: The end of the purge window. If it is not null then the procedure exits when this date and time are reached. The time is checked after each order delete.

- **a_order_id_lt** and **a_order_id_ge**: If **a_order_id_ge** is not null, only orders with order ID greater than or equal to this value are eligible for purging. If **a_order_id_
It is not null then only orders with order ID less than this value are eligible for purging. If either of these is set to `om_new_purge_pkg.v_ptn_scope_latest`, the purge scope is restricted to the latest partition(s) where new orders are created.

- **a_order_source_mnemonic**: If it is not null, only orders with this order source are eligible for purging. Wildcards are not supported.
- **a_order_type_mnemonic**: If it is not null, only orders with this order type are eligible for purging. Wildcards are not supported.
- **a_namespace_mnemonic**: If it is not null, only orders in this cartridge namespace are eligible for purging. Wildcards are not supported.
- **a_version_mnemonic**: This is used in combination with **a_namespace_mnemonic**. If it is not null, only orders in the specified cartridge namespace and version are eligible for purging. Wildcards are not supported.
- **a_cartridge_id**: If it is not null, only orders in the specified cartridge are eligible for purging.
- **a_parallelism**: Specifies the degree of parallelism (the number of database jobs performing the purge). If it is null, it uses the parallelism configured by the `degree_of_parallelism` parameter. If it is 1, the purge is executed serially (with a single database job). See "Parallel Execution" for more information.
- **a_commit_count**: Specifies how often each job should issue commit. Unless you performed extensive performance purge tests to determine the optimal value for this parameter, it is recommended that you leave it null.

**Example**: The following execution purges orders with a time limit of 15 minutes and a parallelism of 8. The purge criteria specify all orders that were closed 30 days ago or more.

```sql
declare
    v_status integer;
begin
    om_new_purge_pkg.purge_orders(
        a_status=>v_status,
        a_stop_date => sysdate + 15/24/60, -- 15m
        a_delete_before=>trunc(sysdate) - 30,
        a_order_states=> om_new_purge_pkg.v_closed_orders,
        a_parallelism => 8);
end;
```

**om_new_purge_pkg.schedule_order_purge_job (Online or Offline)**

```sql
procedure schedule_order_purge_job(
    a_start_date                 date,
    a_delete_before              date,
    a_order_states               integer,
    a_stop_date                  date     default om_const_pkg.v_no_date,
    a_order_id_lt                integer  default null,
    a_order_id_ge                integer  default null,
    a_order_source_mnemonic      varchar2 default null,
    a_order_type_mnemonic        varchar2 default null,
    a_namespace_mnemonic         varchar2 default null,
    a_version_mnemonic           varchar2 default null,
    a_cartridge_id               integer  default null,
    a_commit_count               integer  default v_default_count,
    a_parallelism binary_integer default null);
```
This procedure schedules an execution of `purge_orders` using the `dbms_job` package. The `a_start_date` parameter specifies the start date and time.

**om_new_purge_pkg.select_orders (Online or Offline)**

```plsql
procedure select_orders(
  a_selected_count out integer,
  a_delete_before date,
  a_order_states integer,
  a_order_id_lt integer default null,
  a_order_id_ge integer default null,
  a_order_source_mnemonic varchar2 default null,
  a_order_type_mnemonic varchar2 default null,
  a_namespace_mnemonic varchar2 default null,
  a_version_mnemonic varchar2 default null,
  a_cartridge_id integer default null);
```

This procedure inserts into the staging table `OM_ORDER_ID_FOR_PURGE` the order IDs that satisfy the given purge criteria. This is useful when you cannot identify all orders to be purged in a single execution of `purge_orders`, and you do not want to execute multiple purges. In this case:

- You can populate `OM_ORDER_ID_FOR_PURGE` piecemeal by executing `select_orders` several times with different purge criteria. You can also insert or delete order IDs from this table manually.
- After you finish populating this table, execute `purge_selected_orders`.

**Parameters:**

- **a_selected_count**: Returns the number of order IDs inserted into `OM_ORDER_ID_FOR_PURGE` by this call. This count ignores order IDs that were already inserted into this table, even if they match the given purge criteria.
- The rest of the parameters specify the purge criteria and they are the same as in `purge_orders`.

**Example**: The following selects for purge all orders in cartridge namespace X that were closed 7 days ago and reside on the latest partition(s) (where new orders are created).

```plsql
declare
  v_status integer;
  v_selected_count  integer;
begin
  om_new_purge_pkg.select_orders(
    a_selected_count=>v_selected_count,
    a_delete_before=>trunc(sysdate) - 7,
    a_order_states=>om_new_purge_pkg.v_closed_orders,
    a_order_id_ge=>om_new_purge_pkg.v_ptn_scope_latest,
    a_namespace_mnemonic => 'X');
end;
```

**om_new_purge_pkg.purge_selected_orders (Online or Offline)**

```plsql
procedure purge_selected_orders(
  a_status out integer,
  a_purged_count out integer,
  a_stop_date date default om_const_pkg.v_no_date,
  a_commit_count number default v_default_count,
  a_parallelism binary_integer default null);
```

Example: The following selects for purge all orders in cartridge namespace X that were closed 7 days ago and reside on the latest partition(s) (where new orders are created).
This procedure purges the orders specified in the staging table `OM_ORDER_ID_FOR_PURGE`. It works like `purge_orders` except that the purge criteria are not supplied (the orders are already selected).

You can also use this procedure to restart a stopped purge with different parameters (for example, if you want to change the time when the purge window ends or the degree of parallelism).

See "om_new_purge_pkg.purge_orders (Online or Offline)" for possible outcomes.

This operation is audited.

Parameters:
- `a_status`: Returns the purge status.
- `a_purged_count`: The number of orders purged.
- `a_stop_date`: The end of the purge window. If it is not null then the procedure exits when this date and time are reached. The time is checked after each order delete.
- `a_commit_count`: Specifies how often each job should issue commit. Unless you performed extensive performance purge tests to determine the optimal value, it is recommended that you leave it null.
- `a_parallelism`: Specifies the degree of parallelism (the number of database jobs performing the purge). If it is null, it uses the parallelism configured by the `degree_of_parallelism` parameter. If it is 1, the purge is executed serially (with a single database job). See "Parallel Execution" for more information.

```
pl/SQL API Reference

om_new_purge_pkg.stop_purge (Online or Offline)
procedure stop_purge;

This procedure stops the current order purge if one is running. This procedure call returns when the purge stops, which normally takes a few seconds.

Later you can resume the same purge by running `resume_purge`, restart the purge with different parameters by running `purge_selected_orders` (for example, if you want to change the time when the purge window ends or the degree of parallelism), or start a new purge.

```

om_new_purge_pkg.resume_purge (Online or Offline)
procedure resume_purge(
    a_stop_date date default null,
    a_commit_count number default null,
    a_parallelism binary_integer default null);

This procedure resumes a stopped order purge or an order purge that finished with errors.

If you do not supply any arguments or if the given arguments are the same as those of the initial purge operation, this procedure resumes processing of existing chunks that are either unassigned or finished processing with errors, using the same degree of parallelism. If you do supply new or changed arguments, this procedure regenerates chunks allows you to change certain parameters of the purge operation. For example:

- You can resume a stopped the purge operation with the `a_stop_date` parameter if you want to change the end of the purge window.
You can resume a stopped purge operation with the `a_parallelism` parameter if you want to lower the degree of parallelism of an online purge operation (for example, due to an unexpected increase in order volume).

Parameters:

- **a_stop_date**: This parameter specifies the end of the purge window. If it is null, the initial value supplied to the purge operation remains in effect.
- **a_commit_count**: This parameter specifies how often each job should commit. If it is null, the initial value supplied to the purge operation remains in effect.
- **a_parallelism**: This parameter specifies the degree of parallelism (the number of database jobs performing the purge). If it is null, the initial value supplied to the purge operation remains in effect.

---

**Note**: Do not use `resume_purge` to expand the scope of a purge. `resume_purge` does not regenerate order Id chunks and any order Ids that fall outside the range of existing unassigned chunks are not be purged. If you want to expand the scope of a purge, add order Ids to `OM_ORDER_ID_FOR_PURGE` and run `purge_selected_orders` instead.

---

Advanced Procedures

This section provides information about advanced procedures.

### om_part_maintain.backup_selected_ords (Offline)

```sql
procedure backup_selected_ords(
   a_parallelism binary_integer default null);
```

The `purge_partitions` procedure inspects each partition in the given range and inserts into the `OM_ORDER_ID_FOR_BACKUP` table the order IDs of the orders that do not satisfy the purge criteria. The specified orders are copied into the backup tables and they are restored after the partitions are purged. The `backup_selected_ords` and `restore_orders` procedures allow you to do the same for arbitrary order IDs, for example, if you want to retain orders for a particular cartridge.

---

**Note**: This procedure does not modify data in partitioned tables.

---

**Example**: The following example shows how to purge partition `P_000000000000400001` but retain all orders in the HSI cartridge. Error handling is omitted for simplicity. Note that orphan data is purged after orders are restored, which is the reason why the `a_purge_orphan_data` argument of `purge_entire_partition` is false.

```sql
declare
   v_jobs_suspended dbms_sql.number_table;
begin
   om_job_pkg.disable_suspend_jobs(60, v_jobs_suspended);
   insert into om_order_id_for_backup
   (select h.order_seq_id
    from om_cartridge c,
    om_order_header partition (P_000000000000400001) h
    where c.namespace_mnemonic = 'HSI'
    and h.cartridge_id = c.cartridge_id
   );
```

om_part_maintain.backup_selected_ords(a_parallelism=>4);
om_part_maintain.purge_entire_partition{
    a_online => false,
    a_partition_name => 'P_000000000000400001',
    a_purge_orphan_data => false);
om_part_maintain.purge_entire_partition(a_online => false,
a_partition_name => 'P_000000000000400001',
a_purge_orphan_data => false);
om_part_maintain.restore_orders(a_parallelism=>4);
om_part_maintain.purge_orphan_order_data();
om_job_pkg.enable_resume_jobs(v_jobs_suspended);
end;

om_part_maintain.restore_orders (Offline)
This procedure restores orders from the backup tables into the partitioned tables, and
purges the backup tables.

procedure restore_orders{
    a_parallelism binary_integer default null);

Normally you do not have to use this procedure because purge_partitions restores
orders automatically. However, it might be needed for recovery purposes, as discussed
in the "Troubleshooting and Error Handling" section. It can also be used in conjunction
with backup_selected_ords to exclude arbitrary order IDs from a purge.

Troubleshooting Functions
This sections provides information about troubleshooting functions.

om_part_maintain.get_partitions (Online or Offline)
This function returns all om_order_header range partitions as well as any partitions
missing from om_order_header (if the schema is not equi-partitioned).

function get_partitions return om_t_order_partitions;

The returned information includes the table name, partition name, number of
subpartitions, tablespace name, and partition upper bound. If the table name is not
OM_ORDER_HEADER, the specified partition is missing. This function is useful for
troubleshooting.

om_part_maintain.is_equipartitioned (Online or Offline)
This function tells whether the schema is equi-partitioned and returns missing
partitions.

function is_equipartitioned{
    a_missing_ptns out om_t_order_partitions)
return boolean;

It returns false if the number of range partitions differs from table to table or the
schema is not partitioned. The implementation does not compare the number of
subpartitions.

If number of range partitions differs from table to table, this could be the result of
interrupted or failed attempts to add or drop partitions. If the schema is not
equi-partitioned, EXCHANGE PARTITION cannot be used for purging partitions;
therefore om_part_maintain.purge_partitions returns right away. In this case, use
om_part_maintain.equipartition to partition your schema.
Recovery Procedures

This section provides information about recovery procedures.

**om_part_maintain.equipartition (Offline only)**

This procedure equi-partitions the schema by adding the partitions in the specified collection that are missing from the schema.

```plsql
procedure equipartition(
    a_missing_ptns in om_t_order_partitions default null);
```

Partitions are added through ALTER TABLE ADD PARTITION and ALTER TABLE SPLIT PARTITION operations. It throws an exception if the schema is not partitioned.

Parameters:
- **a_missing_ptns**: The missing partitions to be added. If it is null, the procedure calls `is_equipartitioned` to find all missing partitions.

Exceptions:
- **ORA-20166**: There is another in-progress maintenance operation.
- **ORA-20170**: Failed to suspend database jobs.
- **ORA-20171**: OSM is running.

Error handling: After you resolve the issue, re-execute this procedure.

**om_part_maintain.purge_orphan_order_data (Online or Offline)**

```plsql
procedure purge_orphan_order_data;
```

This procedure is not part of regular maintenance operations. It purges orphan order data from tables that are not range-partitioned (specifically, order data with an order ID that is less than the minimum order ID in `om_order_header`). Orphan data could be the result of a failed execution of `purge_partitions` or `drop_partitions`.

**om_part_maintain.rebuild_unusable_indexes (Online or Offline)**

```plsql
procedure rebuild_unusable_indexes(
    a_table_name_like varchar2 default 'OM_%',
    a_parallelism binary_integer default null,
    a_online boolean default true,
    a_preferred_method varchar2 default null);
```

These procedures rebuild unusable indexes, and unusable index partitions and sub-partitions. They are called automatically by other procedures that may leave indexes in an unusable state, especially global indexes, such as `drop_partitions`, `purge_partitions`, and `equipartition`.

Parameters:
- **a_table_name_like**: Restricts the scope of the operation to indexes of the specified table name(s). You may use wildcards. The default is OM_% (for example, exchange tables are ignored).
- **a_indexes**: The names of the indexes to be rebuilt.

- **a_parallelism**: Specifies the degree of parallelism. Indexes are altered back to NOPARALLEL afterward they are rebuilt. It is recommended that you leave it null. The implementation will choose the optimal method for each unusable index depending on the index type and configuration parameters. For more information see `purge_policy_rebuild_unusable_indexes`.

- **a_online**: Tells whether indexes should be rebuilt online in order to avoid failure from contention.

- **a_preferred_method**: The preferred rebuild method. Valid values are:
  - `om_part_maintain.c_rebuild_idx_rebuild` (REBUILD)
  - `om_part_maintain.c_rebuild_idx_recreate_global` (RECREATE GLOBAL).

For more information see `purge_policy_rebuild_unusable_indexes`.

```plsql
om_part_maintain.rebuild_index (Online or Offline)
procedure rebuild_index(
    a_index_name varchar2,
    a_parallelism binary_integer default null,
    a_online boolean default true,
    a_preferred_method varchar2 default null,
    a_only_if_unusable boolean default true);
```

This procedure rebuilds the specified index.

Parameters:

- **a_index_name**: The index name.

- **a_parallelism**: Specifies the degree of parallelism. The index is altered back to NOPARALLEL afterward it is rebuilt. It is recommended that you leave it null. The implementation will choose the optimal method depending on the index type and configuration parameters. For more information see `purge_policy_rebuild_unusable_indexes`.

- **a_online**: Tells whether the index should be rebuilt online in order to avoid failure from contention.

- **a_preferred_method**: The preferred rebuild method. Valid values are:
  - `om_part_maintain.c_rebuild_idx_rebuild` (REBUILD)
  - `om_part_maintain.c_rebuild_idx_recreate_global` (RECREATE GLOBAL).

For more information see `purge_policy_rebuild_unusable_indexes`.

```plsql
om_part_maintain.sys$undo_restore_table (Offline)
procedure sys$undo_restore_table(
    a_table_name          varchar2,
    a_parallelism binary_integer default null);
```

Important: This is an internal procedure that should be used strictly for recovery purposes.
If `om_part_maintain.purge_partitions` fails while restoring retained orders into a partitioned table, the procedure automatically invokes `om_part_maintain.sys$undo_restore_table` to delete the partially restored rows from that table. This leaves the table in a clean state and prevents unique key violations when another restore attempt is made. The OM_SQL_POINTER table shows the error and points to the line in OM_SQL_LOG where execution failed. That line includes a call to `om_part_maintain.sys$restore_table` with the name of a partitioned table as input argument. If you have any doubts whether all the partially restored data was deleted from that table successfully, execute `om_part_maintain.sys$undo_restore_table` manually.

```sql
om_part_maintain.sys$undo_restore_orders(Offline)
procedure sys$undo_restore_orders(
    a_parallelism binary_integer default null);
```

This procedure invokes `om_part_maintain.sys$undo_restore_table` for each partitioned table to delete all the order data that was restored from the backup tables. This is a slow operation because it uses DELETE statements. But it might be necessary if `purge_partitions` fails because there is an issue with the data to be restored from the backup tables. In this case you must call `om_part_maintain.sys$undo_restore_orders`, fix the data in the backup tables, and finally restore the orders. This procedure has no effect if the backup tables were purged. For more information, see “Troubleshooting and Error Handling”.

**Database Reference**

The following sections provide information about database views and database tables.

**Database Views**

The following sections provide information about database audit views.

**OM_AUDIT_PURGE_ALL**

The `OM_AUDIT_PURGE_ALL` view returns information about all order purges in descending order (the latest purge is returned first).

*Table 8–9* lists and describes all the columns in the `OM_AUDIT_PURGE_ALL` table.
### Table 8–9  OM_AUDIT_PURGE_ALL Table

<table>
<thead>
<tr>
<th>Column</th>
<th>Datatype</th>
<th>NULL</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PURGE_SEQ_ID</td>
<td>NUMBER(18)</td>
<td>NOT NULL</td>
<td>The system-generated unique Id assigned to the purge operation.</td>
</tr>
<tr>
<td>OPERATION_NAME</td>
<td>VARCHAR(64)</td>
<td>NOT NULL</td>
<td>The name of the operation (e.g. om_new_purge_pkg.purge_orders)</td>
</tr>
<tr>
<td>STATUS</td>
<td>VARCHAR2(20)</td>
<td>NOT NULL</td>
<td>The purge status.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>■ STARTED: This short-lived initial status occurs when the purge operation is selected.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>■ RUNNING: This status occurs when the purge operation is purging the orders.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>■ STOPPED: This status occurs when the om_new_purge_pkg.stop_purge stops a purge operation.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>■ EXPIRED: This status occurs when the purge exceeds the end of the purge window specified by the a_stop_date argument.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>■ FINISHED: This status occurs when the purge finishes successfully. All orders that satisfy the purge criteria are purged.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>■ FINISHED_WITH_ERROR: This status occurs when the purge finishes with some errors. Some orders that satisfy the purge criteria are not purged.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>■ FAILED: This status occurs when the purge failed. No orders are purged.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>■ NO_PURGE: This status occurs when no order satisfies the purge criteria.</td>
</tr>
<tr>
<td>START_DATE</td>
<td>DATE</td>
<td>NOT NULL</td>
<td>The start date and time of the purge.</td>
</tr>
<tr>
<td>PURGE_DURATION_MINUTES</td>
<td>NUMBER(9)</td>
<td>NOT NULL</td>
<td>The purge run time in minutes. If the purge status is STARTED or RUNNING, this duration includes the current elapsed time, for example, since the purge was started or last resumed. This behavior is unlike the PURGE_DURATION_SECS column of the underlying OM_AUDIT_PURGE table, where the current elapsed time is not included.</td>
</tr>
<tr>
<td>EST_OR_ACTUAL_END_DATE</td>
<td>DATE</td>
<td>N/A</td>
<td>The actual or estimated end date and time. If the purge is completed (for example, with status FINISHED, FINISHED_WITH_ERROR or FAILED), the actual end date and time appears. Otherwise the estimated date and time appears based on the purge rate and the purge status:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>■ STARTED: The estimated date and time is NULL because the purge rate is unknown.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>■ RUNNING: The end date and time is estimated based on the number of orders to be purged and the current purge rate. The estimated date and time may exceed the end of the purge window, if specified.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>■ STOPPED: The estimated date and time is calculated as if the status is RUNNING. In other words, it tells you when the purge would finish if you resumed it immediately. (This is useful if you need to find out whether it would finish before a certain time.)</td>
</tr>
<tr>
<td>PURGE_WINDOW_END_DATE</td>
<td>DATE</td>
<td>N/A</td>
<td>The end of the purge window as specified by the a_stop_date argument of the purge procedure.</td>
</tr>
</tbody>
</table>
Database Reference

**OM_AUDIT_PURGE_LATEST**

The **OM_AUDIT_PURGE_LATEST** view is identical to the **OM_AUDIT_PURGE_ALL** view except that it returns information only about the latest purge (see "OM_AUDIT_PURGE_ALL"). This view is useful for monitoring.

**Database Tables**

The following sections provide information about audit related database tables.

**OM_AUDIT_PURGE**

The **OM_AUDIT_PURGE** table describes each order purge. Each audited purge operation adds a record to this table to monitor the purge operation as soon as the purge starts.

Table 8–10 describes the **OM_AUDIT_PURGE** table. This table is partitioned by START_DATE. Each partition corresponds to a different month.

<table>
<thead>
<tr>
<th>Column</th>
<th>Datatype</th>
<th>NULL</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>STOPPED_DATE</td>
<td>DATE</td>
<td>N/A</td>
<td>The last date and time when the purge was stopped.</td>
</tr>
<tr>
<td>RESUMED_DATE</td>
<td>DATE</td>
<td>N/A</td>
<td>The last date and time when the purge was resumed.</td>
</tr>
<tr>
<td>SELECTED_ORDER_COUNT</td>
<td>NUMBER(9)</td>
<td>NOT NULL</td>
<td>The number of orders selected for purge.</td>
</tr>
<tr>
<td>SELECTED_ORDER_PURGED_COUNT</td>
<td>NUMBER</td>
<td>NOT NULL</td>
<td>The number of orders selected for purge that have been purged so far (cascaded deletes are excluded).</td>
</tr>
<tr>
<td>PERCENT_COMPLETE</td>
<td>NUMBER</td>
<td>NOT NULL</td>
<td>The completion percent based on SELECTED_ORDER_COUNT and SELECTED_ORDER_PURGED_COUNT.</td>
</tr>
<tr>
<td>ORDERS_PURGED_PER_MINUTE</td>
<td>NUMBER</td>
<td>NOT NULL</td>
<td>The purge rate per minute.</td>
</tr>
<tr>
<td>ORDERS_INJECTED_PER_MINUTE</td>
<td>NUMBER</td>
<td>NOT NULL</td>
<td>The rate per minute that orders are created in OSM while the order purge executes. This value allows you to identify any purge overlaps with high order volume periods.</td>
</tr>
<tr>
<td>PARALLELISM</td>
<td>NUMBER</td>
<td>NOT NULL</td>
<td>The effective purge degree of parallelism. You can specify this value on the operation using the optional a_parallelism argument. If the argument is null, the purge procedures uses the parallelism configured by the degree_of_parallelism parameter. If degree_of_parallelism is not set, the default parallelism is 4.</td>
</tr>
<tr>
<td>ERROR_MESSAGE</td>
<td>VARCHAR2(4000)</td>
<td>N/A</td>
<td>The reason of failure if the purge failed or finished with errors.</td>
</tr>
</tbody>
</table>
Table 8–10  OM_AUDIT_PURGE Table

<table>
<thead>
<tr>
<th>Column</th>
<th>Datatype</th>
<th>NULL</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PURGE_SEQ_ID</td>
<td>The name of the operation (for example, om_new_purge_pkg.purge_orders)</td>
<td>NOT NULL</td>
<td>A system-generated unique Id assigned to the purge operation.</td>
</tr>
<tr>
<td>OPERATION_NAME</td>
<td>VARCHAR2(64)</td>
<td>NOT NULL</td>
<td>The name of the purge operation (for example, om_new_purge_pkg.purge_orders).</td>
</tr>
<tr>
<td>STATUS</td>
<td>VARCHAR2(20)</td>
<td>NOT NULL</td>
<td>The purge status.</td>
</tr>
<tr>
<td>START_DATE</td>
<td>DATE</td>
<td>NOT NULL</td>
<td>The start date and time of the purge.</td>
</tr>
<tr>
<td>PURGE_DURATION_SEC</td>
<td>NUMBER(9)</td>
<td>NOT NULL</td>
<td>The purge execution time in seconds. This value is updated when the purge completes. If the status is RUNNING, it does not include the elapsed time because the purge is started or resumed. If the purge was stopped and resumed, it includes the total execution time excluding idle periods.</td>
</tr>
<tr>
<td>INJECTED_ORDER_COUNT</td>
<td>NUMBER(9)</td>
<td>NOT NULL</td>
<td>The number of orders injected while the purge was running online. This value helps identify any purge overlaps with high order volume periods.</td>
</tr>
<tr>
<td>SELECTED_ORDER_COUNT</td>
<td>NUMBER(9)</td>
<td>NOT NULL</td>
<td>The number of orders selected for purge.</td>
</tr>
<tr>
<td>END_DATE</td>
<td>DATE</td>
<td>N/A</td>
<td>The date when the purge ended. This is set when the status is FINISHED, FINISHED_WITH_ERROR, EXPIRED or FAILED.</td>
</tr>
<tr>
<td>STOP_REQUESTED_DATE</td>
<td>DATE</td>
<td>N/A</td>
<td>The last date and time when the user submitted a purge stop request.</td>
</tr>
</tbody>
</table>
The **OM_AUDIT_PURGE_ORDER** table stores a synopsis for each purged order including the order Id and all attributes that are used to determine whether an order satisfies the purge criteria. Orders are added to this table as they are purged and become visible as transactions commit. This ability allows you to monitor the purge rate.

Table 8–11 describes the **OM_AUDIT_PURGE_ORDER** table. This table is reference-partitioned with **OM_AUDIT_PURGE** as the parent table.

### Table 8–11  **OM_AUDIT_PURGE_ORDER** Table

<table>
<thead>
<tr>
<th>Column</th>
<th>Datatype</th>
<th>NULL</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PURGE_SEQ_ID</td>
<td>NUMBER(18)</td>
<td>N/A</td>
<td>The system-generated unique Id assigned to the purge operation.</td>
</tr>
<tr>
<td>ORDER_SEQ_ID</td>
<td>NUMBER(18)</td>
<td>NOT NULL</td>
<td>The order Id.</td>
</tr>
<tr>
<td>CASCADED_DELETE</td>
<td>VARCHAR2(1)</td>
<td>N/A</td>
<td>Indicates whether the order was deleted because of a cascaded delete. For example, whether this is an amendment order of a deleted base order.</td>
</tr>
<tr>
<td>DELETED_DATE</td>
<td>DATE</td>
<td>NOT NULL</td>
<td>The date/time when the order was deleted.</td>
</tr>
<tr>
<td>ORDER_TYPE_ID</td>
<td>NUMBER(9)</td>
<td>NOT NULL</td>
<td>Copied from <strong>OM_ORDER_HEADER</strong>.</td>
</tr>
<tr>
<td>ORDER_SOURCE_ID</td>
<td>NUMBER(9)</td>
<td>NOT NULL</td>
<td>Copied from <strong>OM_ORDER_HEADER</strong>.</td>
</tr>
<tr>
<td>REFERENCE_NUMBER</td>
<td>VARCHAR2(255)</td>
<td>NOT NULL</td>
<td>Copied from <strong>OM_ORDER_HEADER</strong>.</td>
</tr>
<tr>
<td>ORD_STATE_ID</td>
<td>NUMBER(9)</td>
<td>NOT NULL</td>
<td>Copied from <strong>OM_ORDER_HEADER</strong>.</td>
</tr>
<tr>
<td>CARTRIDGE_ID</td>
<td>NUMBER(6)</td>
<td>NOT NULL</td>
<td>Copied from <strong>OM_ORDER_HEADER</strong>.</td>
</tr>
<tr>
<td>ORD_CREATION_DATE</td>
<td>DATE</td>
<td>NOT NULL</td>
<td>Copied from <strong>OM_ORDER_HEADER</strong>.</td>
</tr>
<tr>
<td>ORD_START_DATE</td>
<td>DATE</td>
<td>NOT NULL</td>
<td>Copied from <strong>OM_ORDER_HEADER</strong>.</td>
</tr>
<tr>
<td>ORD_COMPLETION_DATE</td>
<td>DATE</td>
<td>NOT NULL</td>
<td>Copied from <strong>OM_ORDER_HEADER</strong>.</td>
</tr>
<tr>
<td>ORD_TXN_COMPLETION_DATE</td>
<td>TIMESTAMP</td>
<td>N/A</td>
<td>Copied from <strong>OM_ORDER_HEADER</strong>.</td>
</tr>
<tr>
<td>VERSION</td>
<td>NUMBER(9)</td>
<td>N/A</td>
<td>Copied from <strong>OM_ORDER_HEADER</strong>.</td>
</tr>
<tr>
<td>STOPPED_DATE</td>
<td>DATE</td>
<td>N/A</td>
<td>The last date and time when the purge was stopped.</td>
</tr>
<tr>
<td>RESUMED_DATE</td>
<td>DATE</td>
<td>N/A</td>
<td>The last date and time when the purge was resumed.</td>
</tr>
<tr>
<td>ERROR_MESSAGE</td>
<td>VARCHAR2(4000)</td>
<td>N/A</td>
<td>The reason of failure if the purge failed or finished with errors.</td>
</tr>
</tbody>
</table>
OM_AUDIT_PURGE_PARAM
The OM_AUDIT_PURGE_PARAM table stores the purge arguments and criteria supplied to the purge procedure and a snapshot of relevant session and configuration parameters at the time the purge was started. The following parameters are included:

- Arguments of the purge procedure that specify purge criteria, such as `a_delete_before`, `a_order_states`, `a_order_id_lt`, `a_order_id_ge`, `a_order_source_mnemonic`, `a_order_type_mnemonic`, `a_namespace_mnemonic`, `a_version_mnemonic`, and `a_cartridge_id`.
- Arguments of the purge procedure other than purge criteria, such as `a_stop_date`, `a_parallelism` and `a_commit_count`.
- Database session parameters that identify who executed the purge and where, such as `BG_JOB_ID`, `FG_JOB_ID`, `HOST`, `INSTANCE_NAME`, `OS_USER`, `SERVICE_NAME`, `SESSION_USER`, and `SID`.
- Purge-related configuration parameters in the `om_parameter` table, such as `degree_of_parallelism`, `parallel_execute_chunk_size`, `oms_timezone`, and `purge_job_class`.

Table 8–12 describes the OM_AUDIT_PURGE_PARAM table. This table is reference-partitioned with OM_AUDIT_PURGE as the parent table.

<table>
<thead>
<tr>
<th>Column</th>
<th>Datatype</th>
<th>NULL</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PURGE_SEQ_ID</td>
<td>NUMBER(18)</td>
<td>NOT NULL</td>
<td>The system-generated unique Id assigned to the purge operation.</td>
</tr>
<tr>
<td>PARAMETER_NAME</td>
<td>VARCHAR2(254)</td>
<td>NOT NULL</td>
<td>The parameter name.</td>
</tr>
<tr>
<td>PARAMETER_TYPE</td>
<td>VARCHAR2(1)</td>
<td>NOT NULL</td>
<td>The parameter type:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- P: This is a parameter of the purge procedure that specifies a purge criterion.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- U: This is a parameter of the purge procedure other than a purge criterion.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- S: This is a database session parameter.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- C: This is a purge-related configuration parameter.</td>
</tr>
<tr>
<td>PARAMETER_VALUE</td>
<td>VARCHAR2(255)</td>
<td></td>
<td>The parameter value.</td>
</tr>
</tbody>
</table>

Troubleshooting and Error Handling

**Important:** In the event of failure during a purge operation, Oracle strongly recommends that you stop OSM and perform all troubleshooting and recovery offline.

The PL/SQL API provides functions and procedures to troubleshoot and recover from errors. Most procedures for managing partitions use `om_sql_log_pkg`, which is an internal package that enables procedures to persist and execute SQL statements so that execution can be resumed in the event of an error. This is particularly useful for DDL statements.

The `om_sql_log_pkg` package persists SQL statements in the `om_sql_log` table, which includes the following columns:
Troubleshooting and Error Handling

- **sid**: The session ID. The default value is the current session ID, for example, `sys_context('USERENV', 'SID')`. This allows concurrent execution.
- **name**: This is usually the name of the procedure that generated the SQL statement. It is useful to Oracle Support.
- **line**: This is a line number used for ordering the SQL statements to be executed.
- **sql_text**: The SQL statement.

SQL statements persisted in `om_sql_log` are executed by `om_sql_log_pkg.exec`. This procedure executes all SQL statements with the specified session ID, ordered by line number. If you do not specify the session ID, it uses the current one. During execution, the line number of the current statement is updated in the `om_sql_pointer` table. This allows you to monitor execution. Upon successful execution, it deletes all statements with that session ID. In the event of failure, however, it inserts in the `om_sql_pointer` table the error message with the session ID and line number of the failed statement. In this case, when `om_sql_log_pkg.exec` is re-executed, it resumes execution with the failed statement.

Therefore you can troubleshoot and recover from a failed partition maintenance operation even if it was executed by a scheduled job. The contents of `om_sql_log` and `om_sql_pointer` allow for faster assistance from Oracle Support. After you fix the root cause of a failure, in some cases you can resume the operation from the point of failure. This ensures that your data is not left in an inconsistent state (although in some cases you might have to take additional actions if you want to complete that operation).

If you resume a failed operation from a different database session, or you abandon that operation, you must manually delete the rows for the failed session by executing the following statement:

```sql
execute om_sql_log_pkg.remove(sid);
```

where **sid** is the session ID specified by the error in `om_sql_pointer`.

**Example (Monitoring execution)**: You can monitor execution by retrieving from `om_sql_log` the current statement of each session. If the **error_code** section is populated, the execution failed.

```sql
select l.*, p.error_code, p.error_message
from om_sql_log l, om_sql_pointer p
where l.sid = p.sid and l.line = p.line;
```

**Example**: You can review the set of SQL statements of a partition maintenance operation that failed in the current session as follows:

```sql
select * from om_sql_log
where sid = sys_context('USERENV', 'SID')
order by line;
```

### Error Handling for `add_partitions`

The `add_partitions` procedure logs the DDL statements to be executed in the `om_sql_log` table. If the reason for the failure is unknown, you can check `om_sql_log` and `om_sql_pointer` for clues. If `om_sql_pointer` contains an error, it is likely because partition creation succeeded only partially and the schema is therefore no longer equi-partitioned. After you resolve the issue, you can finish the operation as follows:
If the failure occurred while rebuilding unusable indexes, the new partitions have been created. Execute `om_sql_log_pkg.exec(sid)` to finish rebuilding, where `sid` is the session ID specified by the error in `om_sql_pointer`.

- If the failure occurred during the partition creation phase, execute `om_sql_log_pkg.exec(sid)` to finish partition creation. Then execute `om_part_maintain.rebuild_unusable_indexes` with `a_online` set to true or false depending on whether OSM is running or not.

- When in doubt, execute `om_part_maintain.is_equi_partitioned` to check whether the schema is equi-partitioned. If it is not, you can execute `om_part_maintain.equipartition` to fix it.

### Error Handling for `drop_partitions`

In the event of error, `drop_partitions` re-enables disabled foreign keys and throws the exception again. However, execution failures could result in partitioning inconsistencies, orphan order data and unusable indexes or index partitions. After you resolve the issue, you can take the following actions:

- The recommended action is to re-execute the procedure with the same argument. If this is not possible (for example, because you cannot afford further downtime), do the following:
  - You must at least execute `rebuild_unusable_indexes` to ensure indexes are usable.
  - (Optional) You can execute `purge_orphan_order_data` to delete orphan order data. Otherwise orphan data is deleted by the next execution of `purge_partitions` or `drop_partitions`.
  - Execute `om_job_pkg.resume_jobs` to resume database jobs.

- When in doubt, execute `is_equi_partitioned` to check whether the schema is equi-partitioned. If it is not, you can execute `equipartition` to fix it.

### Error Handling for `purge_partitions`

In the event of unexpected error, `purge_partitions` re-enables any disabled constraints and throws the exception again. However, execution failures could result in partitioning inconsistencies, orphan order data and even data loss if you do not follow the error handling procedure to recover.

### Troubleshooting

In the event of an unexpected error, `purge_partitions` re-enables any disabled constraints and throws the exception again. However, execution failures could result in partitioning inconsistencies, orphan order data, and even data loss if you do not follow recovery procedures.

If you spooled the output of the stored procedure to a file (recommended), review the file to determine the reason and point of failure. If the purge capacity is greater than 1, the file also indicates which purge tables were involved. You can identify the point of failure by reviewing the started and finished messages that mark the beginning and end of procedure execution.

A failure may occur in these procedures:

- `sys$bpdr_backup`: Copies the orders that do not satisfy the purge criteria into the backup tables.
Troubleshooting and Error Handling

- **sys$bpdr_purge**: Purges one or more partitions entirely by exchanging them with purge tables.
- **sys$bpdr_drop**: Drops N-1 empty partitions, where N is the number of purged partitions.
- **sys$bpdr_restore**: Restores the retained orders from the backup tables into the Nth partition.
- **rebuild_unusable_indexes**: Rebuilds all or specific unusable indexes as required. It is executed:
  - By **sys$bpdr_backup** prior to copying orders into the backup tables.
  - By **sys$bpdr_restore** prior to restoring retained orders.
  - At the end, prior to **sys$purge_orphan_order_data**.
- **sys$purge_orphan_order_data**: Purges orphan order data (executed at the end).
- **sys$purge_xchg_prg_tables**: If the **a_purge_xchg_prg_tables** argument is true, it is executed at the end to purge the purge tables. It may also be executed prior to purging a group of successive partitions, if the purge capacity is exhausted.

If the procedure output is not available, inspect the following for clues.

- Most of the time you can determine the error and the point of failure by reviewing the **om_sql_log** and **om_sql_pointer** tables.
  - **om_sql_pointer** points to the SID (session ID) and line of the failed statement in **om_sql_log**. If there are several errors in **om_sql_pointer**, check the **error_date** column to find the SID of the most recent error.
  - If **om_sql_log** includes EXCHANGE PARTITION statements, execution failed in **sys$bpdr_purge**. Partitions are in an inconsistent state.
  - If **om_sql_log** includes INSERT statements into the backup tables, execution failed in **sys$bpdr_backup**. Partitions are in a consistent state.
  - If **om_sql_log** includes calls to **sys$restore_table**, execution failed in **sys$bpdr_restore**. Partitions contain partially restored orders.
  - If **om_sql_log** includes statements to rebuild indexes, execution failed in **rebuild_unusable_indexes**.

- Review the backup tables:
  - If the backup tables are empty then there are a number of possibilities, such as:
    a) no orders were retained, b) the failure occurred prior to **sys$bpdr_backup**, or c) **sys$bpdr_restore** purged the backup tables after a successful restore.
  - If the backup tables are not empty then the failure occurred either during or after **sys$bpdr_backup** and possibly during **sys$bpdr_restore**.
  - If none of the order IDs in XCHG_OM_BCK_$001$ exist in OM_ORDER_HEADER then most likely the failure occurred during or after **sys$bpdr_purge**. Check the remaining partitioned tables listed in the OM_XCHG_TABLE table. If you cannot find those order IDs in any of those tables then **sys$bpdr_purge** completed successfully (the data was exchanged into the purge tables). There is also a remote possibility that the failure occurred in **sys$bpdr_restore** while restoring retained orders into OM_ORDER_HEADER (the first table to be restored). In this case, **user_parallel_execute_tasks** should include a task with **task_name** equal to **restore:om_order_header**.
– If some but not all of the order IDs in XCHG_OM_BCK_$001$ exist in OM.ORDER_HEADER then the failure occurred in sys$bpdr_restore while restoring retained orders into OM.ORDER_HEADER (the first table to be restored). In this case, user_parallel_execute_tasks should include a task with task_name equal to restore:om_order_header.

– If all of the order IDs in XCHG_OM_BCK_$001$ exist in OM.ORDER_HEADER, check whether all the data in the remaining backup tables exist in the corresponding partitioned tables (the OM_XCHG_TABLE table specifies the exchange table ID for each partitioned table). If this is not the case then the failure occurred during sys$bpdr_purge or sys$bpdr_restore.

■ Review the purge tables, especially those that correspond to OM_ORDER_HEADER (for example, XCHG_OM_PRG_001$001$). If they do not contain any data, most likely the purge failed before the sys$bpdr_purge procedure. If the purge capacity is greater than 1, check the sys$xchg_purge_seq parameter in the om_parameter table to find out which set of purge tables was used for the latest purge.

■ Review the affected partitions. If a partition in the purge range is empty, most likely it was exchanged with the purge tables (it is also possible that it was previously empty). Check the purge tables to confirm.

■ Review the user_parallel_execute_tasks view in the OSM core schema. If the view contains any tasks with task_name equal to restore:tableName, execution failed in sys$bpdr_restore while restoring data into the tableName table (assuming the previous execution of purge_partitions was successful).

Error Handling

When you determine the point of failure, as discussed in “Troubleshooting”, and you resolve the issue, you can recover and finish the purge operation as follows:

■ If the failure occurred during sys$bpdr_backup, the partitions are in a consistent state. Execute om_part_maintain.purge_xchg_bck_tables and om_sql_log_pkg.remove(SID) to purge the backup tables, om_sql_log and om_sql_pointer.

■ If the failure occurred during sys$bpdr_purge, the partitions are in an inconsistent state (partially purged):

1. Execute om_sql_log_pkg.exec(SID) to finish the purge (exchange), where SID is the session ID of the failed execution (the SID is recorded in om_sql_pointer together with the error message).

2. If you were consolidating partitions N-to-1, drop the N-1 partitions before the N-th partition, which was exchanged. To drop those partition, use the following statements instead of drop_partitions:

   ALTER TABLE OM_ORDER_HEADER DROP PARTITION partition_name;
   ALTER TABLE OM_SEQUENCE DROP PARTITION partition_name;

3. If the backup tables are not empty, execute om_part_maintain.restore_orders with the desired degree of parallelism to rebuild unusable indexes and restore the retained orders.

■ If the failure occurred during sys$bpdr_drop while consolidating partitions: When you consolidate partitions N-to-1, purge_partitions copies retained orders into the backup tables, purges (exchanges) the N-th partition, drops N-1 partitions, and restores the retained orders into the N-th partition.
- If the `om_sql_log` table contains the DROP PARTITION statements, execute `om_sql_log_pkg.exec(SID)` where `SID` is the session ID of the failed execution (the SID is recorded in `om_sql_pointer` together with the error message). In some releases, the DROP PARTITION statements are not logged in the `om_sql_log` table. In this case, you can find them in the DBMS output. If you do not have the DBMS output, execute these statements:

  ```sql
  ALTER TABLE OM_ORDER_HEADER DROP PARTITION partition_name;
  ALTER TABLE OM_SEQUENCE DROP PARTITION partition_name;
  ```

- If the backup tables are not empty, execute `om_part_maintain.restore_orders` with the desired degree of parallelism to rebuild unusable indexes and restore the retained orders.

  If the failure occurred during `sys$bpdr_restore` and you fixed the root cause, it is recommended that you finish the restore operation. The partitions are in an inconsistent state (retained orders are not fully restored). The backup tables are not affected and they contain all retained orders. This is how you can resume the restore operation from the point of failure:

  1. Execute `om_part_maintain.disable_ptned_fks` to disable foreign keys of partitions tables.
  2. In the event of failure, `sys$bpdr_restore` automatically deletes the partially restored data from the last partitioned table in order to ensure that a second attempt will not fail due to unique key violations. However, it is recommended that you execute `om_part_maintain.sys$undo_restore_table(t)` anyway, where `t` is the name of the partitioned table that caused the failure. This procedure deletes the data restored into the given table.
  3. Execute `om_sql_log_pkg.exec(SID)` to finish the restore, where `SID` is the session ID of the failed execution (the SID is recorded in `om_sql_pointer` together with the error message).
  4. Execute `om_part_maintain.reenable_ptned_fks` to re-enable foreign keys.

  If the root cause lies with the data to be restored, you must execute `om_part_maintain.sys$undo_restore_orders` to delete all restored data from the partitioned tables. This is a slow operation because it uses DELETE statements. Then fix the orders retained in the backup tables, and execute `om_part_maintain.restore_orders` with the desired degree of parallelism to rebuild unusable indexes and restore the retained orders.

  Oracle recommends that you always execute `om_part_maintain.rebuild_unusable_indexes`.

  (Optional) In any case, you can execute `om_part_maintain.purge_orphan_order_data` to delete orphan order data. Otherwise, orphan data will be deleted by the next execution of `purge_partitions` or `drop_partitions`.

  Execute `om_job_pkg.resume_jobs` to resume database jobs.

**Error Handling for `rebuild_unusable_indexes`**

This procedure logs the DDL statements to be executed in the `om_sql_log` table. In the event of an error, it is important that you re-execute `om_sql_log_pkg.exec` to finish the rebuild operation. Otherwise:

- Unusable indexes are likely to impact performance. Moreover, unusable indexes that enforce unique constraints report `ORA-01502` errors. Error reporting is
disabled for other unusable indexes, unless SKIP_UNUSABLE_INDEXES is set to false (the default is true).

- An index may be left in PARALLEL state, which could result in undesirable behavior.

**Error Handing for setup_xchg_tables**

This procedure logs the DDL statements to be executed in the om_sql_log table. If the reason of failure is unknown, check om_sql_pointer and om_sql_log for clues. When you resolve the issue, you can take one of the following actions:

- Re-execute setup_xchg_tables with a force=true.
- Execute drop_xchg_tables to drop any partially created exchange tables.

**Performance Tuning**

This section explains how to tune the following:

- **degree_of_parallelism**
- **degree_of_parallelism_rebuild_indexes**
- **degree_of_parallelism_rebuild_xchg_indexes**
- Parallel job execution
- Row-based purge

**Tuning degree_of_parallelism**

This parameter specifies the default DOP for queries, DML, and most DDL operations. In particular, it affects the performance of order backup and restore statements performed by purge_partitions. You can use the following procedure to evaluate the optimal degree_of_parallelism without performing a purge.

To evaluate the optimal degree_of_parallelism:

1. Clear the om_order_id_for_backup table.
   
   ```sql
   delete from om_order_id_for_backup;
   ```

2. Select a representative number of order IDs that does not exceed the value of xchg_retained_orders_thres from a single partition. For example, if you frequently retain 10,000 orders when you purge partitions:

   ```sql
   insert into om_order_id_for_backup (select order_seq_id from om_order_header partition (P_000000000003000001) where rownum <= 10000);
   commit;
   ```

3. Back up the selected order IDs with the desired parallelism (for example, 16) and record the elapsed time:

   ```sql
   exec om_part_maintain.backup_selected_ords(16);
   ```

4. Purge the backup tables:

   ```sql
   exec om_part_maintain.purge_xchg_bck_tables;
   ```
5. Repeat with a different degree of parallelism and compare the elapsed times until you find the optimal DOP.

**Tuning degree_of_parallelism_rebuild_indexes**

The best way to determine the optimal DOP for `degree_of_parallelism_rebuild_indexes` is through trials. Try purging or dropping partitions with different settings for this parameter, and review the DBMS output to compare the elapsed times for rebuilding indexes.

**Tuning degree_of_parallelism_rebuild_xchg_indexes**

The optimal DOP for `degree_of_parallelism_rebuild_xchg_indexes` is normally 1 (the default) because these indexes are very small and they are rebuilt one partition at a time. There is rarely a reason to increase this value.

**Tuning Parallel Job Execution**

You can use these parameters to tune parallel job execution:

- **degree_of_parallelism**: Specifies the degree of parallelism for the majority of database operations performed by parallel jobs and parallel servers. For more information, see "Tuning degree_of_parallelism."

- **purge_job_class**: Specifies the job class for purge operations. If your database is Oracle RAC, it is important that you configure this as described in the purge_job_class section.

- **parallel_execute_chunk_size**: This is an advanced parameter that you rarely need to tune, especially beginning with 7.2.0.10.2, 7.2.2.5, and 7.2.4.2. For more information see the following section.

**Tuning parallel_execute_chunk_size**

The implementation of `purge_partitions` uses the `dbms_parallel_execute` package to restore retained orders, which uses database jobs to parallelize execution. Order data is restored one table at a time, and each table is divided into chunks. Each job is assigned a chunk, commits the transaction, gets the next chunk, and so on. The process repeats for the next table. For example, if the degree of parallelism is 32 and 64 chunks are created, 32 chunks will be processed concurrently by jobs and they will be committed at about the same time before the remaining 32 chunks are processed.

The number of chunks created depends primarily on the volume of data, the number of sub-partitions and the specified chunk size. The default value of `parallel_execute_chunk_size` is 2000 blocks. If the size of the retained order data is small to moderate, this chunk size normally results in as many chunks as sub-partitions (for example, 32 or 64), which is found to work well.

Beginning with 7.2.0.10.2, 7.2.2.5, and 7.2.4.2, each table to be restored is divided separately into chunks. This means that the number of chunks is different for each table. However, the volume of data for each chunk is about the same, regardless of the table. This results in shorter transactions (more frequent commits) that require less UNDO. Therefore, the default `parallel_execute_chunk_size` (2000 blocks) results in good performance, regardless of the volume of data retained, and there is rarely a need to change it.

Prior to 7.2.0.10.2, 7.2.2.5, and 7.2.4.2, the number of restore chunks is the same for all tables because chunks are generated from the `XCHG_OM_BCK_S001$` table. However, the volume of data for each chunk varies from table to table. If the volume
of retained order data is very large (for example, tens of thousands of orders), the
chunks for large tables such as OM_ORDER_INSTANCE are large transactions that
generate a lot of UNDO and therefore require a large UNDO tablespace.

In this case, it might be better to increase the number of chunks in order to increase the
frequency of commits and reduce the UNDO size. For example, if your performance
tests show that committing every 500 orders is more efficient in terms of elapsed time
and/or UNDO size, and you normally retain about 100000 orders, the optimal number
of chunks would be 200. To increase the number of chunks you must decrease the
parallel_execute_chunk_size.

If you are not sure how chunks are generated at your patch level, review the restore
statements. If they are joins, chunks are generated as in 7.2.0.10 or earlier.

Prior to 7.2.0.10.2, 7.2.2.5 and 7.2.4.2, use the following procedure to find out how
different parallel_execute_chunk_size settings affect the number of chunks created. If
you are using 7.2.0.10.2, 7.2.2.5, 7.2.4.2 or later, there is rarely a need to tune parallel_
execute_chunk_size. However, if you want to do so, you can substitute om_order_
header and xchg_om_bck_$001$ in the following procedure with any other
partitioned table and the corresponding xchg_om_bck_table to find out the number of
chunks that will be created for that partitioned table.

To find out how different parallel_execute_chunk_size settings affect the number of
chunks created (7.2.0.10.2, 7.2.2.5, 7.2.4.2, or earlier):

1. Ensure the exchange tables are created.
2. Populate the backup tables with a large number of orders to retain, preferably all
in the same partition (substitute x and y, so that the range (x, y) contains the
desired number of orders):
   
   ```
   insert into xchg_om_bck_$001$
   (select * from om_order_header where order_seq_id between x and y) ;
   ```

3. Repeat the following executions and with different values for chunk_size (20, 100,
200, and so on), until the query returns a count close to the desired number of
chunks:
   
   ```
   exec dbms_parallel_execute.create_task('CHECK_CHUNK_TASK') ;
   exec dbms_parallel_execute.drop_chunks('CHECK_CHUNK_TASK') ;
   exec dbms_parallel_execute.create_chunks_by_rowid('CHECK_CHUNK_TASK', user, 'XCHG_OM_BCK_$001$', by_row => false, chunk_size =>20) ;
   select count(*) from user_parallel_execute_chunks
   where task_name = 'CHECK_CHUNK_TASK'
   order by chunk_id;
   exec dbms_parallel_execute.drop_task('CHECK_CHUNK_TASK') ;
   ```

4. When you are done testing, purge the backup tables:
   
   ```
   exec om_part_maintain.purge_xchg_bck_tables;
   ```

**Tuning Row-Based Purge**

Row-based purges are I/O intensive. The purge_policy_time_to_close_wait policy
can reduce I/O, improve performance, and decrease purge rate fluctuations. For more
information see "purge_policy_time_to_close_wait."
Database Terms

This chapter uses the following database terms:

- **Automatic Workload Repository (AWR)**: A built-in repository in every Oracle database. Oracle Database periodically makes a snapshot of its vital statistics and workload information and stores them in AWR.

- **closed and open orders**: An order is closed if it is complete or aborted. Otherwise it is open. Open orders restrict certain operations. For example, if a partition has open orders, you cannot purge it online. Those restrictions are relaxed for cancelled orders: cancelled orders are treated as closed.

- **DDL (Data Definition Language)**: Includes database statements that define or change a data structure, such as CREATE TABLE or ALTER INDEX.

- **DML (Data Manipulation Language)**: Includes database statements that manipulate data, such as SELECT, INSERT, UPDATE, and DELETE.

- **exhausted partition**: Each OSM partition stores a range of order Ids. The upper bound of the range is specified by the (non-inclusive) partition upper bound. Order Ids increase monotonically. When the next order Id reaches or exceeds the upper bound of the current partition, the partition is said to be exhausted.

- **high water mark**: The boundary between used and unused space in a database segment.

- **OLTP (Online Transaction Processing)**: OLTP systems are optimized for fast and reliable transaction handling. Compared to data warehouse systems, most OLTP interactions involve a relatively small number of rows, but a larger group of tables.

- **Oracle RAC (Real Application Clusters)**: A database option that allows multiple concurrent database instances to share a single physical database.

- **partitioning**: The ability to decompose very large tables and indexes into smaller and more manageable pieces called partitions.

- **retained orders**: Retained orders excluded from purging, for example, because they are still in the retention period or they do not satisfy other purge criteria.

- **retention period**: The period of time an order should be kept after it is closed. This varies depending on policies, laws, and regulations prescribed by the business or governments. After its retention period, an order is eligible for purge.

- **tablespaces, segments, and extents**: A database is divided into one or more logical storage units called *tablespaces*. Tablespaces are divided into logical units of storage called *segments*. Each segment consists of a set of *extents* allocated for a specific database object, such as a table, index, or partition. An extent is a logically contiguous allocation of space. A partitioned table has a segment for each partition. For example, a range-hash partitioned table with two range partitions and 32 hash sub-partitions per range partition has 64 segments.

Managing Optimizer Statistics

Oracle Database internally prioritizes the database objects that require statistics, so that those objects that most need updated statistics are processed first. For more information about optimizer statistics, see the documentation for best practices for gathering optimizer statistics:

Automated Optimizer Statistics Collection

The automatic statistics collection job is enabled by default and collects statistics during a predefined maintenance window for all database objects that are missing statistics or have stale statistics.

Oracle recommends that you:

- Leave the automatic statistics collection enabled. If you have a valid reason to disable it, it should be enabled for at least the dictionary tables.
- Schedule maintenance operations, which would render statistics stale, before the automatic statistics collection, such as purging partitions, and deploying, undeploying and purging cartridges. In this way, statistics will be updated by the automatic collection.

For more information about automated optimizer statistics collection, see the following knowledge articles, available from the Oracle Support Web site:

- FAQ: Automatic Statistics Collection Job (10g and 11g) [Doc ID 1233203.1]
- Best Practices for Automatic Statistics Collection [Doc ID 377152.1]
- How to Manually Execute the Optimizer Statistics Auto Task in Oracle 11g [Doc ID 731935.1]
- How to Extend Maintenance Windows For GATHER_STATS_JOB for More Than 8 Hours [Doc ID 368475.1]

Online vs. Offline

The procedures provided by the `om_db_stats_pkg` package, as well as native Oracle database statistics procedures, can all be executed online. However, when gathering statistics during peak hours you should temporarily lower the value of the DEGREE preference as described in Parallel Collection.

Gathering Cartridge Metamodel Statistics

If the automatic statistics gathering job is enabled, statistics of cartridge metadata tables are kept up-to-date and should not cause performance issues. However, if you perform extensive cartridge management operations, such as deploy, undeploy, purge, and so on, after the automatic statistics collection and before normal business hours, it is recommended that you gather cartridge metadata statistics manually before you start processing orders.

```sql
execute om_db_stats_pkg.gather_cartridge_stats;
```

Gathering Order Statistics

A volatile order table is one where the volume of data changes over time because the data is deleted automatically after it is processed.

High Volatility Order Tables

The following order tables are always highly volatile because they are polled periodically and rows are very short-lived (seconds). These tables may have thousands of rows at peak workload but have little data or no data overnight when the automatic statistics collection job runs.

- OM_JMS_EVENT
- OM_JMS_THREAD
You should not enable incremental statistics on highly volatile tables. However, Oracle recommends that you lock statistics for these tables.

```
execute om_db_stats_pkg.lock_volatile_order_stats;
```

**Medium Volatility Order Tables**

OM_ORDER_FLOW_COORDINATOR and OM_COORD_NODE_INSTANCE have both partitions that retain order data and volatile partitions from which data is deleted after it is processed. These tables are configured with a medium level of volatility.

Oracle recommends that you enable incremental statistics for medium volatility tables.

```
execute om_db_stats_pkg.set_table_prefs_incremental(a_incremental => true,
     a_volatility => om_const_pkg.v_volatility_medium);
```

**Low Volatility Order Tables**

The majority of order tables retain data until that data is purged. These tables are configured with a low level of volatility. Oracle recommends that you enable incremental statistics for low volatility tables.

```
execute om_db_stats_pkg.set_table_prefs_incremental(a_incremental => true,
     a_volatility => om_const_pkg.v_volatility_low);
```

**High, Low, or Medium Volatility Order Tables**

Data from the following tables is deleted automatically after it is processed.

- OM_ORDER_FLOW
- OM_AUTOMATION_CTX
- OM_AUTOMATION_CORRELATION
- OM_ORDER_POS_INPUT
- OM_UNDO_BRANCH_ROOT
- OM_ORCH_DEPENDENCY_PENDING

The volatility of these tables varies depending on the solution, as discussed in the scenarios in this section.

**Scenario 1**

If most of your orders complete in seconds, it is recommended that you manage these tables in the same manner as the highly volatile tables. By default, OSM is configured so that the tables in this section are set to a medium level of volatility. You can modify the configuration of these tables to a high level of volatility using the `om_db_stats_pkg.set_table_volatility` procedure.

```
execute om_db_stats_pkg.set_table_volatility('OM_ORDER_FLOW',
     om_const_pkg.v_volatility_high);
execute om_db_stats_pkg.set_table_volatility('OM_AUTOMATION_CTX',
     om_const_pkg.v_volatility_high);
execute om_db_stats_pkg.set_table_volatility('OM_AUTOMATION_CORRELATION',
     om_const_pkg.v_volatility_high);
execute om_db_stats_pkg.set_table_volatility('OM_ORDER_POS_INPUT',
     om_const_pkg.v_volatility_high);
execute om_db_stats_pkg.set_table_volatility('OM_UNDO_BRANCH_ROOT',
     om_const_pkg.v_volatility_high);
execute om_db_stats_pkg.set_table_volatility('OM_ORCH_DEPENDENCY_PENDING',
     om_const_pkg.v_volatility_high);
```
Managing Optimizer Statistics

Scenario 2
If the life cycle of your orders is mostly days or weeks, these tables contain data at the end of the day, although likely not as much data as during the day. In this case, you can manage these tables like low volatility order tables. By default, OSM is configured so that these tables are set to a medium level of volatility. You can modify the configuration of the tables above to a low level of volatility using the `om_db_stats_pkg.set_table_volatility` procedure.

execute om_db_stats_pkg.set_table_volatility('OM_ORDER_FLOW', om_const_pkg.v_volatility_low);
execute om_db_stats_pkg.set_table_volatility('OM_AUTOMATION_CTX', om_const_pkg.v_volatility_low);
execute om_db_stats_pkg.set_table_volatility('OM_AUTOMATION_CORRELATION', om_const_pkg.v_volatility_low);
execute om_db_stats_pkg.set_table_volatility('OM_ORDER_POS_INPUT', om_const_pkg.v_volatility_low);
execute om_db_stats_pkg.set_table_volatility('OM_UNDO_BRANCH_ROOT', om_const_pkg.v_volatility_low);
execute om_db_stats_pkg.set_table_volatility('OM_ORCH_DEPENDENCY_PENDING', om_const_pkg.v_volatility_low);

Scenario 3
If the life cycle of your orders is mixed, with both short-lived and long-lived orders, the recent partitions where new orders are created have a mixed load of orders and experience both inserts and deletes. Whereas older partitions are less volatile because they have long-lived orders (or no orders at all) and few inserts. In this case, you can manage these tables like medium volatility order tables. By default, OSM is already configured so that these tables are set to a medium level of volatility.

Manual or Scheduled Maintenance of Volatile Order Tables
It is recommended that you gather statistics for high volatility tables when the workload is representative, preferably at the peak, as long as the database has spare resources. You could schedule a job to re-gather statistics periodically on these tables, for example, once a day:

execute DBMS_STATS.SET_SCHEMA_PREFS(user, 'DEGREE', 2);
execute om_db_stats_pkg.gather_order_stats(a_force => true, a_volatility => om_const_pkg.v_volatility_high);
execute DBMS_STATS.SET_SCHEMA_PREFS(user, 'DEGREE', 'DBMS_STATS.AUTO_DEGREE');

It is recommended that you gather statistics for medium volatility tables when the workload is representative, preferably near a peak as long as the database has spare resources. You could schedule a job to re-gather statistics periodically on these tables, for example, once every few days:

execute DBMS_STATS.SET_SCHEMA_PREFS(user, 'DEGREE', 2);
execute om_db_stats_pkg.gather_order_stats(a_force => true, a_volatility => om_const_pkg.v_volatility_medium);
execute DBMS_STATS.SET_SCHEMA_PREFS(user, 'DEGREE', 'DBMS_STATS.AUTO_DEGREE');

Impact of Statistics on Purging
A large volume of statistics, histograms, and incremental statistics synopses have a significant performance impact on partition purge and drop operations. When you
drop partitions, statistics are deleted. When you purge partitions, the order data of the partitioned tables are moved into empty exchange tables using `ALTER TABLE ... EXCHANGE PARTITION` operations. The exchange operation also exchanges statistics, including histograms and incremental statistics synopses, if possible. This is expensive because it involves a lot of inserts, deletes, and updates. Therefore, beginning with 7.2.0.10, the `om_part_maintain.purge_xchg_prg_tables` procedure deletes exchange table statistics automatically.

If a partition is exchanged with a non-partitioned table, synopses cannot be exchanged. In Oracle Database 12c, they are invalidated. However, in Oracle Database 11g the synopses are deleted immediately, unless you install patch 18238463: `DML WAITS ON EXCHANGE PARTITION (LIB CACHE LOCK - WRIS_OPTSTAT_SYNOPSIS$)`. Install this patch on Oracle Database 11.2.0.3 or 11.2.0.4 to ensure the synopses are marked STALE (or invalidated) instead of being deleted. Oracle strongly recommends applying this patch. The patch is disabled by default; enable the patch by entering the following:

```
alter system set "_fix_control"='18238463:1' scope=both sid='*';
```

Oracle also recommends that you disable dynamic sampling before you execute `OM_PART_MANTAIN.SETUP_XCHG_TABLES` to create exchange tables for purging partitions. This should be done for the session only:

```
alter session set OPTIMIZER_DYNAMIC_SAMPLING=0;
```

---

**Gathering Statistics on Other OSM Tables**

If the automatic statistics collection is enabled, you do not have to do anything additional to gather statistics on the other OSM tables.

---

**Gathering Fixed Object Statistics**

The automatic statistics gathering job does not gather fixed object statistics, and dynamic sampling is not automatically used for SQL statements involving `XS` tables when optimizer statistics are missing. If fixed object statistics are missing, the optimizer uses predefined default values that may not be representative and could lead to suboptimal execution plans. RMAN, Data Guard, Streams, and Grid Control make heavy use of the fixed tables through the `DBA/V$` views and so often experience the worst performance issues. Another common symptom is extreme TEMP space usage, driven by poor plans against the fixed tables.

Oracle recommends that:

- You gather statistics on fixed objects using the `DBMS_STATS.GATHER_FIXED_OBJECTS_STATS` procedure when there is a representative workload on the system.
- After you install OSM, or perform a major OSM upgrade, you re-gather fixed object statistics when there is a representative workload.
- In general, you re-gather fixed object statistics if you do a major database or application upgrade, implement a new module, or make changes to the database configuration.

For more information, see the following knowledge articles, available from the Oracle Support Web site:

- How to gather statistics on SYS objects and 'Fixed' Objects [Doc ID 457926.1]
- `GATHER_FIXED_OBJECTS_STATS` Considerations [Doc ID 798257.1]
Gathering System Statistics

System statistics enable the optimizer to more accurately cost each operation in an execution plan by using information about the actual system hardware that is executing the statement, such as CPU speed and I/O performance. System statistics are enabled by default, and are automatically initialized with default values that are representative for most systems. Therefore system statistics are not automatically collected as part of the automatic statistics gathering job.

If you choose to gather system statistics, use the `DBMS_STATS.GATHER_SYSTEM_STATS` during a representative workload time window, ideally at peak times. You may want to consider gathering system statistics under the following circumstances:

- If you changed the CPU speed or I/O subsystem.
- If you want to record the actual system performance for a representative OSM workload, for example, when you go live or after a significant increase in order volume.

If you gather system statistics, review the new statistics for correctness and compare with previous values. For more information, see the following knowledge articles, available from the Oracle Support Web site:

- How to Collect and Display System Statistics (CPU and I/O) for CBO use [Doc ID 149560.1]
- Using Actual System Statistics (Collected CPU and I/O information) [Doc ID 470316.1]
- System Statistics: Scaling the System to Improve CBO optimizer [Doc ID 153761.1]
- System Statistics: How to gather system stats for a particular batch of work [Doc ID 427939.1]

Populating New Partition Statistics

Populating statistics on new partitions improves performance by avoiding the issues that can arise when queries are made on newly active partitions for which statistics have yet to be collected.

Using Statistics from Another Partition

You can manually copy statistics to a new partition using the `om_db_stats_pkg.copy_order_ptn_stats` procedure. This procedure allows you to specify the partition from which to obtain statistics, as well as the partition to which to copy statistics. For example:

```sql
declare
  v_copied boolean;
begin
  om_db_stats_pkg.copy_order_ptn_stats(v_copied,
    a_src_partition_name => 'P_000000000000400001',
    a_dst_partition_name => 'P_000000000000700001');
end;
```

To copy recent and valid statistics to the most recently created order partition, as well as to the corresponding partitions of all reference-partitioned tables, you can use:

```sql
declare
  v_copied boolean;
begin
  om_db_stats_pkg.copy_order_ptn_stats(v_copied);
```
Managing Optimizer Statistics

You can avoid manually copying recent and valid statistics to the most recently created partition by setting the `db_stats_auto_copy_to_new_partition` om_parameter to `ON_ADD`. By default, this parameter is set to `OFF`. For more information, see "Configuration Parameter for Managing Optimizer Statistics".

Using Statistics from a Statistics Table

You can also use the `om_db_stats_pkg.export_order_ptn_stats` to export partition statistics to a statistics table and then use `om_db_stats_pkg.import_order_ptn_stats` to import them to a partition. You could use this to save one or more snapshots of representative partition statistics and then use the underlying statistic tables as templates.

Using Statistics from Another System

Partition statistics exported to a statistics table using `om_db_stats_pkg.export_order_ptn_stats` can also be saved to the file system using `om_db_stats_pkg.expdp_order_ptn_stats`. Files are saved to the `DATA_PUMP_DIR` directory. As sysdba, you can find out the location of that directory using:

```sql
select directory_name, directory_path
from dba_directories
where directory_name='DATA_PUMP_DIR';
```

After you save partition statistics to a file, that file can be transferred to the `DATA_PUMP_DIR` directory on another system (for example, using FTP) and you can load that file to a statistics table on that system using `om_db_stats_pkg.impdp_order_ptn_stats`. As long as user names are the same on both systems, you can then import these partition statistics using `om_db_stats_pkg.import_order_ptn_stats`.

This could, for example, be used to enhance the performance of a newly installed production system by importing partition statistics from a lab system.

---

**Note:** If you are using a pluggable database (PDB) you cannot use the default `DATA_PUMP_DIR` directory. You must specify a different variable and directory for the PDB instance. Each PDB instance that you want to use expdp and impdp with must have its own data pump directory.

---

**Configuration Parameter for Managing Optimizer Statistics**

The following configuration parameter in the `om_parameter` table affects optimizer statistics maintenance operations:

- `odb_stats_auto_copy_to_new_partition`

This parameter specifies the policy for automatically copying statistics to new order partitions. Possible values are:

- **OFF** (default): Statistics are not automatically copied to new order partitions.
- **ON_ADD**: Valid and recent statistics are automatically copied to new order partitions when they are added. This happens when a new partition is created manually or automatically.

You can change the default `OFF` setting to `ON_ADD` with the following SQL statement:
update om_parameter
set value = 'ON_ADD'
where mnemonic = 'db_stats_auto_copy_to_new_partition';
commit;

Optimizer Statistics Management Error Handling

Optimizer statistics management error handling is available for automated copy partition statistics jobs and datapump jobs.

Automated Copy Partition Statistics Jobs

OSM submits asynchronous database jobs to automatically copy partition statistics. While it is unlikely that these jobs will fail and become stuck, you can use the `om_db_stats_pkg.lstj_copy_order_ptn_stats` to determine if that is the case. You can then use `om_db_stats_pkg.remj_copy_order_ptn_stats` to remove individual jobs.

Datapump Jobs

The `om_db_stats_pkg.expdp_order_ptn_stats` and `om_db_stats_pkg.impdp_order_ptn_stats` procedures submit datapump jobs to save or load partition statistics to or from the file system.

While it is unlikely that these jobs will fail and become stuck, as sysdba, you can determine if datapump jobs are stuck using the following:

```
select owner_name, job_name, operation, job_mode, state, attached_sessions
from dba_datapump_jobs
where job_name not like 'BIN$%'
order by owner_name, job_name;
```

As a regular user, you can remove stuck export jobs using the following:

```
declare
dpj number;
begin
    dpj := dbms_datapump.attach('EXPORT_ORDER_PTN_STATS', user);
    dbms_datapump.stop_job(dpj, 1, 0);
end;
```

You can remove stuck import jobs using the following:

```
declare
dpj number;
begin
    dpj := dbms_datapump.attach('IMPORT_ORDER_PTN_STATS', user);
    dbms_datapump.stop_job(dpj, 1, 0);
end;
```

Optimizer Statistics Management Performance Tuning

This section describes various ways to tune optimizer statistics management performance.

Parallel Collection

As your OSM database grows, it is important that you gather statistics in parallel. Otherwise the automatic statistics collection might not be able to process all tables and
parts. You can configure the **DEGREE** as a global or schema preference using one of the **DBMS_STATS.SET_*_PREFS** procedures. **DEGREE** controls the number of parallel server processes that are used to gather statistics on each table partition and each non-partitioned table.

By default, Oracle Database uses the same number of parallel server processes specified as the Degree of Parallelism attribute of the table in the data dictionary. Because the degree of parallelism is 1 for all OSM tables and indexes, Oracle recommends that you set the **DEGREE** as a schema preference:

```sql
execute DBMS_STATS.SET_SCHEMA_PREFS(user, 'DEGREE', 'DBMS_STATS.AUTO_DEGREE');
```

However, if you gather statistics manually while the database is processing a workload, you should temporarily set a low value for **DEGREE**.

Note that the actual degree of parallelism can be between 1 (serial execution) for small objects to **DEFAULT_DEGREE** (PARALLEL_THREADS_PER_CPU X CPU_COUNT) for large objects.

---

**WARNING:** Do not change the degree of parallelism attribute of any OSM table or index. This is not supported.

---

**Concurrent Collection**

Oracle Database 11.2.0.2 introduced inter-object parallelism when gathering statistics. This is controlled by the global statistics gathering preference **CONCURRENT**. Gathering statistics on multiple tables, partitions, and subpartitions concurrently allows Oracle Database to fully utilize a multi-processor environment.

Because this is a new feature, Oracle recommends that you first test it in a non-production environment. If you have any issues with the concurrent collection, contact Oracle Database support.

---

**Incremental Statistics**

Gathering statistics on partitioned tables consists of gathering statistics at both the table level (global statistics) and (sub)partition level. If the **INCREMENTAL** preference for a partitioned table is set to **TRUE**, Oracle Database will accurately derive all global level statistics by scanning only those partitions that have been added or modified, and not the entire table. Incremental global statistics works by storing a synopsis for each partition in the table. Aggregating the partition level statistics and the synopses from each partition accurately generates global level statistics, thus eliminating the need to scan the entire table.

If you have a high volume of order data, the automatic statistics collection will likely be unable to gather all statistics even if it is done in parallel. In this case, Oracle recommends that you enable incremental statistics for order tables that retain order data, as discussed in the sections about "Low Volatility Order Tables" and "Medium Volatility Order Tables".

For more information, see the knowledge article Collect Incremental Statistics For a Large Partitioned Table in 10g and in 11g [Doc ID 1319225.1], available from the Oracle Support Web site.

---

**Cursor Invalidations**

When statistics are modified by **DBMS_STATS**, new cursors that are not yet cached in the shared pool use them to get execution plans. However, existing cursors that are
cached in the shared pool cannot update their execution plans. Instead, such cursors are invalidated and new versions (children cursors) are created, which get execution plans based on the updated statistics. This involves a hard-parse operation that is more expensive than a soft-parse, which simply reuses a cached cursor. Therefore, Oracle Database spreads cursor invalidations over a time period long enough for hard-parses not to cause noticeable spikes in resource usage. This time period is 5 hours by default and it is controlled by the \_optimizer\_invalidation\_period initialization parameter (in seconds).

If your database has performance issues that are caused by bad execution plans because of stale or missing statistics, 5 hours is a long time to wait. Oracle therefore recommends that you decrease the value of \_optimizer\_invalidation\_period. For example, the following command sets \_optimizer\_invalidation\_period to 600 seconds.

```sql
alter system set \"_optimizer\_invalidation\_period\"=600;
```

If 10 minutes turns out to be too short to avoid significant spikes caused by parsing in your environment, increase the value accordingly.

For more information, see the knowledge article Rolling Cursor Invalidations with DBMS_STATS.AUTO_INVALIDATE [Doc ID 557661.1], available from the Oracle Support Web site.

**Dynamic Statistics (or Dynamic Sampling)**

If statistics are missing or stale, the optimizer uses dynamic sampling to improve execution plans. This is controlled by the OPTIMIZER_DYNAMIC_SAMPLING initialization parameter. The default setting is 2, which means that the optimizer uses dynamic statistics if at least one table in the statement has no statistics.

Setting OPTIMIZER_DYNAMIC_SAMPLING to a value greater than 2 is not recommended for OSM and OLTP systems in general because it could increase parse times.

**METHOD_OPT**

The METHOD_OPT parameter controls column statistics collection and histogram creation. While you can set it as a preference, Oracle strongly recommends that you leave METHOD_OPT at its default setting: FOR ALL COLUMNS SIZE AUTO. Oracle Database monitors the use of columns in queries and creates histograms for the columns that need it:

Setting METHOD_OPT to FOR ALL COLUMNS SIZE 254 causes Oracle Database to gather a histogram on every column. This unnecessarily extends the elapsed time and the system resources needed for statistics gathering, increases the amount of space required to store the statistics, and significantly increases the time it takes to purge partitions.

You should also refrain from setting METHOD_OPT to FOR ALL INDEX COLUMNS SIZE 254 because this setting causes Oracle Database to gather histograms on every column used in an index, which might waste system resources. This setting also has an undesirable side-effect of preventing Oracle Database from collecting basic column statistics for non-index columns.

Note that histograms with more than 255 distinct values are more likely to be inaccurate as the NDV increases because cardinality estimations become more inaccurate as more are added.
Optimizer Statistics Management PL/SQL API Reference

This section describes: setup and tuning, maintenance, advanced, troubleshooting, and recovery procedures.

Setup and Tuning Procedures

This section describes setup and tuning procedures.

**om_db_stats_pkg.lock_volatile_order_stats**

This procedure locks statistics on volatile order tables.

```plsql
procedure lock_volatile_order_stats;
```

An order table is considered volatile if its volatility level is set to `om_const_pkg.v_volatility_high`.

**om_db_stats_pkg.unlock_volatile_order_stats**

This procedure unlocks statistics on volatile order tables.

```plsql
procedure unlock_volatile_order_stats;
```

An order table is considered volatile if its volatility level is set to `om_const_pkg.v_volatility_high`.

**om_db_stats_pkg.set_table_prefs_incremental**

This procedure sets the `INCREMENTAL` statistics preference for partitioned OSM tables that have the specified volatility level.

```plsql
procedure set_table_prefs_incremental(
    a_incremental boolean,
    a_volatility number);
```

Parameters:

- `a_incremental`: Specifies whether you want statistics to be gathered incrementally on partitioned OSM tables that have the specified volatility level. When set to true, the PUBLISH preference is also set to true because this is required for incremental statistics collection.

- `a_volatility`: Specifies the volatility level of partitioned OSM tables for which the `INCREMENTAL` statistics preference should be set.

Exception:

- **ORA-20165**: Illegal argument: Invalid volatility level.

**om_db_stats_pkg.set_table_volatility**

This procedure sets the volatility level for an OSM table. The volatility level for OSM tables is configured in `OM_\$INSTALL\$TABLE`.

```plsql
procedure set_table_volatility(
    a_table_name varchar2,
    a_volatility number);
```

Parameters:

- `a_table_name`: Specifies the name of the table on which to set the volatility level.

- `a_volatility`: Specifies the volatility level to set. Valid values are `om_const_pkg.v_volatility_none`, `om_const_pkg.v_volatility_low`, `om_const_pkg.v_volatility_medium`, and `om_const_pkg.v_volatility_high`.
Maintenance Procedures

This section describes maintenance procedures.

**om_db_stats_pkg.gather_cartridge_stats**

This procedure gathers statistics for cartridge metadata tables.

```plsql
procedure gather_cartridge_stats;
```

**om_db_stats_pkg.gather_order_stats**

This procedure gathers statistics for order tables.

```plsql
procedure gather_order_stats(
    a_volatility number default null,
    a_force boolean default false);
```

Parameters:

- **a_volatility**: The level of volatility of order tables for which statistics should be gathered. Null by default, which means all volatility levels.
- **a_force**: Specifies whether you want statistics to be gathered on order tables even if their statistics are locked. The default is false.

Exception:

- **ORA-20165**: Illegal argument: Invalid volatility level.

**om_db_stats_pkg.gather_volatile_order_stats**

This procedure copies order partition statistics from the specified source order partition to the specified destination order partition.

```plsql
procedure copy_order_ptn_stats(
    a_copied out boolean,
    a_dst_partition_name varchar2 default null,
    a_src_partition_name varchar2 default null);
```

Parameters:

- **a_copied**: Output parameter indicating whether statistics were successfully copied.
- **a_dst_partition_name**: Specifies the name of the order partition to which you want to copy statistics. If you do not specify this parameter, the most recently added partition is used.
- **a_src_partition_name**: Specifies the name of the order partition from which you want to copy statistics. If not specified, a partition with the most recent valid statistics is used, if available. If no valid partition statistics are available, a_copied is set to false.

Exceptions:

- **ORA-20142**: Operation is not allowed: OSM schema is not partitioned.
- **ORA-20165**: Illegal argument: Partition does not exist.
- **ORA-20165**: Illegal argument: The source partition cannot be the same as the destination partition.
- **ORA-20144**: Function returned unexpected value. Internal error. Contact support: Cannot find the newest partition.
om_db_stats_pkg.export_order_ptn_stats

This procedure exports order partition statistics from the specified order partition to the specified statistics table. If that table already exists, it is dropped before exporting statistics to the statistics table.

```sql
procedure export_order_ptn_stats(
    a_exported out boolean,
    a_src_partition_name varchar2 default null,
    a_stat_table_name varchar2 default c_om_order_stat_table);
```

Parameters:
- **a_exported**: Output parameter indicating whether statistics were successfully exported.
- **a_src_partition_name**: Specifies the name of the order partition from which you want to export statistics. If not specified, a partition with the most recent valid statistics is used, if available. If no valid partition statistics are available, `a_exported` is set to `false`.
- **a_stat_table_name**: Specifies the name of the statistics table to which to export statistics. Defaults to `c_om_order_stat_table` ('OM_ORDER_STAT_TABLE'). If this statistics table already exists, it is dropped and recreated before exporting statistics from the specified partition; if it is not a statistics table, the table is not dropped and an exception is raised.

Exceptions:
- **ORA-20142**: Operation is not allowed: OSM schema is not partitioned.
- **ORA-20165**: Illegal argument: Partition does not exist.
- **ORA-20165**: Illegal argument: Invalid table name.
- **ORA-20165**: Illegal argument: Table is not a statistics table.

om_db_stats_pkg.import_order_ptn_stats

This procedure imports order partition statistics from the specified statistics table to the specified destination order partition.

```sql
procedure import_order_ptn_stats(
    a_imported out boolean,
    a_dst_partition_name varchar2 default null,
    a_stat_table_name varchar2 default c_om_order_stat_table);
```

Parameters:
- **a_imported**: Output parameter indicating whether statistics were successfully imported.
- **a_dst_partition_name**: Specifies the name of the order partition to which you want to import statistics. If you do not specify this parameter, the most recently added partition is used.
- **a_stat_table_name**: Specifies the name of the statistics table from which to import statistics. The default becomes `c_om_order_stat_table` ('OM_ORDER_STAT_TABLE').

Exceptions:
- **ORA-20142**: Operation is not allowed: OSM schema is not partitioned.
- **ORA-20165**: Illegal argument: Partition does not exist.
■ ORA-20165: Illegal argument: Invalid table name.
■ ORA-20165: Illegal argument: Table is not a statistics table.
■ ORA-20144: Function returned unexpected value. Internal error. Contact support: Cannot find the newest partition.

**Advanced Procedures**

This section describes advanced procedures.

**om_db_stats_pkg.expdp_order_pntn_stats**

This procedure saves order partition statistics from the specified statistics table to the DATA_PUMP_DIR directory. A .dmp suffix is added to the table name to form the name of the file to which statistics will be saved; for example, OM_ORDER_STAT_TABLE.dmp. If that file already exists, it is deleted before saving statistics to the file system.

```sql
procedure expdp_order_pntn_stats(
    a_saved out boolean,
    a_stat_table_name varchar2 default c_om_order_stat_table,
    a_data_pump_dir in varchar2 default null);
```

**Parameters:**

- **a_saved**: Output parameter indicating whether statistics were successfully saved.
- **a_stat_table_name**: Specifies the name of the statistics table from which to obtain statistics. The default becomes **c_om_order_stat_table** (`OM_ORDER_STAT_TABLE`).

**Exceptions:**

- **ORA-20165**: Illegal argument: Table is not a statistics table.
- **ORA-20142**: Operation is not allowed: Failed to save partition statistics to file system.

**om_db_stats_pkg.impdp_order_pntn_stats**

This procedure loads order partition statistics into the specified statistics table from the DATA_PUMP_DIR directory. A .dmp suffix is added to the table name to form the name of the file from which statistics will be loaded; for example, OM_ORDER_STAT_TABLE.dmp.

**Note**: If you are using a pluggable database (PDB) you cannot use the default DATA_PUMP_DIR directory. You must specify a different variable and directory for the PDB instance. Each PDB instance that you want to use expdb and impdp with must have its own data pump directory.
procedure impdp_order_ptn_stats(
    a_loaded out boolean,
    a_stat_table_name varchar2 default c_om_order_stat_table,
    a_data_pump_dir in varchar2 default null);

Note: If partitioned statistics came from another system, they can be imported only if user names are the same in both the source and destination systems.

Parameters:
- **a_loaded**: Output parameter indicating whether statistics were successfully loaded.
- **a_stat_table_name**: Specifies the name of the statistics table into which you want to load statistics. The default becomes `c_om_order_stat_table` (`OM_ORDER_STAT_TABLE`). If this statistics table already exists, it is dropped and recreated before loading statistics from the file system; the table is not dropped and an exception is raised if it is not a statistics table.

Exceptions:
- ORA-20165: Illegal argument: Invalid table name.
- ORA-20165: Illegal argument: File not found in DATA_PUMP_DIR directory.
- ORA-20165: Illegal argument: Table is not a statistics table.
- ORA-20142: Operation is not allowed: Failed to load partition statistics from file system.

Troubleshooting Procedure
This section describes the troubleshooting procedure.

**om_db_stats_pkg.lstj_copy_order_ptn_stats**
This procedure lists active `copy_order_ptn_stats` jobs. The output includes a job id that can be used to remove the job using `remj_copy_order_ptn_stats`.

```
procedure lstj_copy_order_ptn_stats;
```

Recovery Procedure
This section describes the recovery procedure.

**om_db_stats_pkg.remj_copy_order_ptn_stats**
This procedure removes the specified `copy_order_ptn_stats` job.

```
procedure remj_copy_order_ptn_stats ( 
    a_job_id number);
```

Parameter:
- **a_job_id**: Specifies the id of the job to remove.

Exception:
- ORA-20155: Job does not exist.
This chapter helps you understand how Oracle Communications Order and Service Management (OSM) is related to WebLogic Server and the Oracle Database backup and restore procedures. For more information about these procedures, including warm or partial backup procedures, consult the Oracle Database and Oracle WebLogic Server documentation.

About Backing Up and Restoring OSM Files and Data

It is critical that you have a schedule and procedures for backing up and restoring your production-ready OSM system. This chapter includes a suggested schedule and backup and restore considerations, as well as information about the components involved in the backup. You must consider your own business needs when determining your backup and restore strategy.

Online backup of OSM data is supported if JMS JDBC store is configured.

Backup and Restore Overview

There are three main components to consider for backing up and restoring OSM. They can be performed in any order.

- Back up the OSM home directory (including the SDK directory and installation logs)
- Back up the OSM information from the Oracle Database
- Back up the files for the WebLogic domain

Backup and Restore Schedule

The three components of the OSM installation to be backed up require different schedules because they are modified at different times and for different reasons.

OSM Home Directory

The OSM program files in the OSM home directory do not change as part of the ongoing operation of OSM, so regularly scheduled backups may not be required. You should determine when it is appropriate to back up these files based on when you make changes to the files.

Oracle Database

You should perform a complete backup of the OSM database after installation. The suggested schedule for post-install backups is to take an incremental (level 0 in
Backing Up and Restoring the OSM Files

RMAN) backup of the database monthly, a cumulative (level 1 in RMAN) backup weekly and a differential (level 1 in RMAN) backup daily. It is currently not possible to take consistent backups of database and transaction logs, because the transaction logs are file-based. For highest reliability use a highly available fault-tolerant storage (for example, SAN) for database and transaction log file stores.

WebLogic Server Files

Assuming that JMS JDBC store is configured, backups of the WebLogic domain directory and any external deployments should be done primarily after making changes to the configuration. These changes include adding deployments, changing domain configuration, and other administrative tasks.

If the persistent store for JMS is on the file system, that location should be backed up on the same schedule as the database.

The OSM attachments directory (see "OSM Attachment Directory") should also be backed up on the same schedule as the database, because it contains order-related data.

Backup and Restore Considerations

Overall considerations for OSM backup and restore include:

- The backup of the WebLogic domain described in this chapter backs up not only the OSM application, but also the WebLogic configuration of any other applications that share the WebLogic domain with OSM. Restoring the domain will also return any other applications in the domain to the same restore point. Therefore, it is important (although outside the scope of this document) to coordinate the backup and restore of any applications that share the WebLogic domain.

- Test backup and restore procedures in a test or staging environment before they are used in production.

Backing Up and Restoring the OSM Files

This section describes how to backup and restore the OSM files in the \texttt{OSM\_home} directory.

Back up the OSM Files

The OSM installer creates files in a user-specified location during OSM installation. The default location in the installer is \texttt{/opt/OSM}, but usually another value is supplied during installation. This location is usually referred to as \texttt{OSM\_home}.

You can back up these files using the \texttt{tar} command to put them in a single file, and then storing the .\texttt{tar} file in a safe location.

Restoring the OSM Files

To restore the OSM files, remove the contents of the \texttt{OSM\_home} directory, and extract the contents of the backed-up tar file into the \texttt{OSM\_home} directory.

Oracle Database Backup Considerations

The Oracle Database Server provides several means of backing up information. The two recommended methods for ordinary backup and restore are provided in this
Oracle Database Backup Considerations

Backing Up and Restoring OSM Files and Data

Section. There are no special considerations for OSM in determining the actual procedure for a backup or restore. Information about how to use the backup and restore methods considered in this section can be found in the Oracle Database documentation.

Database backup and restore procedures should be performed by a qualified database administrator.

**RMAN Considerations**

Recovery Manager (RMAN) is an Oracle Database utility that backs up, restores, and recovers Oracle databases. It backs up individual datafiles, and provides complete and incremental backup options. The following are some issues you should consider for using RMAN:

- Because it backs up datafiles, this method is most appropriate for use when OSM is not sharing any tablespaces with other applications. If OSM is sharing its tablespaces with other applications, they will be backed up at the same time. This means that if the OSM data is restored, the information for any other applications will be restored as well. This may not be desired.

- You should back up all of the permanent tablespaces that you have defined for OSM. For example, if you have different tablespaces for data and indexes, you should remember to back up both of them.

- RMAN may be slower than Flashback. This might be an issue in a large production environment.

**Oracle Flashback Technology Considerations**

Oracle Flashback Technology comprises a group of features that support the viewing of past states of data without needing to restore backups. It provides the ability to restore an entire database or individual tables from a set point in time. Following are some issues you should consider if you choose to employ this backup method:

- Because it backs up the entire database, this method is most appropriate for use when OSM is not sharing the database with other applications. If OSM is sharing the database instance with other applications, this method does not allow you to restore only the OSM portion of the database. This can cause data for other applications to be overwritten with older data.

- The Flashback Database command does not restore or perform media recovery on files, so you cannot use it to correct media failures such as disk crashes.

- Some editions of the Oracle Database software may not include this feature.

**Mirroring Technology Considerations**

Split mirrored hardware and software solutions provide a higher level of performance than RMAN and Oracle Flash Technology. Mirroring technology, such as the Oracle Sun ZFS Storage Appliance, enables very fast backup and restore that you can run online without overloading the system. You can use mirror splitting to backup ASM disks, the RDBMS data files, redo logs, and control files.

Oracle recommends that you consider a mirroring technology over other technologies for larger OSM installations. Using a mirroring solution in large OSM installations provides the following benefits:

- Before you purge a partition or upgrade OSM, Oracle recommends that you backup the database in case of a failure. Mirroring technology enables fast
database backup and restore operations. This ability greatly reduces the time it takes to prepare for a purge or upgrade and reduces the time it takes to recover from purge or upgrade failures.

- Taking a database snapshot for testing an upgrade procedure, troubleshooting a problem, and so on, becomes much less time consuming.

**Backing Up and Restoring the WebLogic Server Configuration**

The procedure in this section describes how to perform a full backup and restore of the WebLogic configuration for the domain used by OSM.

You must read through this entire section before starting the procedures. There may be pieces of information that you must retrieve from the domain before you shut it down.

There are several parts of the WebLogic Server configuration that should be backed up. They are not all required to be backed up on the same schedule, so consider the following when determining your WebLogic backup schedule. See “WebLogic Server Files” for more information about the backup schedule. The parts of the WebLogic configuration that must be backed up are the following:

- WebLogic domain directory
- WebLogic persistent store
- OSM attachments directory
- External deployments
- Remote managed server directories (clustered server only)

**Backing Up the WebLogic Server Configuration**

This section relates to backing up WebLogic Server files for OSM.

**Before You Back Up the WebLogic Server**

There are some tasks that you must perform before starting the backup procedures in this chapter.

- Configure WebLogic so that when the domain is restarted, OSM will not process any JMS messages. This ensures that when you restore the domain from the backup, you can verify the configuration before OSM starts processing messages.

- Shut down all of the servers in the WebLogic domain that contains OSM. This includes any remote servers in a clustered environment.

**Setting OSM to Pause Processing of JMS Messages**

You can configure OSM to pause processing of JMS messages using the WebLogic Server Administration Console. By default, OSM has one JMS server, called oms.jms.server. If you have custom JMS servers defined, you should perform this procedure for each of them as well.

1. Access the WebLogic Server Administration console Home window. (See "Accessing the WebLogic Server Administration Console")

2. In the Messaging subsection of the Services section, click JMS Servers.

   The Summary of JMS Servers window is displayed.

3. Click on the name of the appropriate JMS server in the table.
4. The settings for your JMS server are displayed with the Configuration tab and the General sub-tab selected. At the bottom of the window, expand the Advanced heading to display more options.

5. Select the following options:
   - Insertion Paused At Startup
   - Production Paused At Startup
   - Consumption Paused At Startup

6. Click Save.

7. Exit the WebLogic console.

---

**Note:** After you have taken the backup and restarted the WebLogic Server, to return the server to its normal state by following the procedure above, but deselecting the options listed above.

---

**WebLogic Server Domain Directory**

The WebLogic Server domain directory contains many important parts of the server configuration, such as the main configuration files, security files, and LDAP files. You can back up these files using the `tar` command to put them in a single file, and then storing the `.tar` file in a safe location.

**WebLogic Persistent Store**

The default location for the persistent store for WebLogic is in a subdirectory of the domain directory. However, the persistent store may be required to be backed up separately, because it may be backed up on a separate schedule from the rest of the domain configuration. In addition, it is possible to configure this directory in a location outside the WebLogic domain directory. You can view the location of this directory in the WebLogic Server Administration Console. (For information about accessing the WebLogic Console, see "Accessing the WebLogic Server Administration Console".)

From the Home window of the WebLogic Console, Click **Persistent Store** in the right pane under Services. The Persistent Stores window is displayed containing a list of the persistent stores that have been defined and their types. Click the names of any file stores with a type of FileStore. The resulting window displays the location of the directory.

**Note:** If the Persistent Store type is listed as JDBCStore, you do not need to back it up separately, because it is backed up automatically with the database.

You can back up these files using the `tar` command to put them in a single file, and then storing the `.tar` file in a safe location. Ensure you back up the directories for all FileStore persistent stores if there are more than one.

**OSM Attachment Directory**

The default location for the attachments directory for OSM is in a subdirectory of the domain directory backed up in the previous section. However it is possible to configure this directory in another location. You can view the location of this directory in the WebLogic Server Administration Console. (For information about accessing the
WebLogic Console, see "Accessing the WebLogic Server Administration Console".

From the Home window of the WebLogic Console, Click FileT3 in the right pane under Services. The resulting window displays a list of the File (T3) Services that have been defined and their paths. If the path is not a complete path (for example, if it does not start with "/" on a UNIX system), the directory is located in the indicated location inside the domain directory.

You can back up these files using the `tar` command to put them in a single file, and then storing the `.tar` file in a safe location.

**External Deployments**

Any objects that are deployed into WebLogic using the OSM deployment tools are located in a subdirectory of the domain directory. If any objects have been deployed using other methods from a location outside the domain directory, these locations must be backed up as well.

You can use the WebLogic console to see if a particular object has been deployed from a directory outside the domain directory. (For information about accessing the WebLogic Console, see "Accessing the WebLogic Server Administration Console".)

From the Home window of the WebLogic Console, Click Deployments in the right pane under Your Deployed Resources. The resulting window displays a list of the deployments in your domain. Click on the name of any deployment to see the path to the source file.

You can back up these files by copying them to a safe location.

**Remote Managed Servers (Clustered Server Only)**

If you are working in a clustered WebLogic environment, you must also back up the directories for any remote managed servers.

You can back up these files using the `tar` command to put them in a single file, and then storing the `.tar` file in a safe location.

**Restoring the WebLogic Server Configuration**

This section relates to restoring WebLogic Server files for OSM.

**Restoring the WebLogic Server Files**

To restore the WebLogic Server files for the following components, remove the contents of the directories you backed up, and extract the contents of the backed-up tar files into those directories.

- WebLogic domain directory
- WebLogic persistent store
- OSM attachments directory
- Remote managed server directories (clustered server only)

For the external deployments which you have backed up, copy the backed up files to the directory in which they were originally located, and replacing the existing files with the same names.

**Setting OSM to Resume Processing of JMS Messages**

Once you have restarted the WebLogic servers from the restored files and determined that the restoration has been successful, configure OSM to resume processing of JMS
messages using the WebLogic Server Administration Console. By default, OSM has one JMS server, called oms_jms_server. If you have custom JMS servers defined, you should perform this procedure for each of them as well.

1. Access the WebLogic Server Administration console Home window. (See "Accessing the WebLogic Server Administration Console")

2. In the Messaging subsection of the Services section, click JMS Servers.
   The Summary of JMS Servers window is displayed.

3. Click on the name of the appropriate JMS server in the table.

4. The settings for your JMS server are displayed with the Configuration tab and the General sub-tab selected. At the bottom of the window, expand the Advanced heading to display more options.

5. Deselect the following options:
   - Insertion Paused At Startup
   - Production Paused At Startup
   - Consumption Paused At Startup

6. Click Save.

7. Exit the WebLogic console.
This chapter provides information about how to selectively export and import schema data, which include orders and model data (cartridge data), from an Oracle Communications Order and Service Management (OSM) database.

About Exporting and Importing OSM Schema Data

Exporting OSM schema data is useful for providing support personnel with the information that they require to troubleshoot a failed order, or range of orders. Exporting only the data that is causing an issue allows support to more quickly locate and resolve issues.

This chapter provides information about exporting and importing order and model data from one database to another. You can export an entire OSM schema, however it is typically very large, with hundreds of thousands of orders. Exporting a large schema is rarely practical because of its size, the resources it would consume, and the time it might take to export. It is more feasible to selectively export specific orders or a range of orders.

OSM provides an export/import utility package. Use the utilities contained in the package to allow you to dynamically generate the files that are required for exporting data, and to prepare and finalize the schema for import.

**Note:** The utilities do not provide an effective means of backing up and restoring database data. For more information about how to do this, see "Backing Up and Restoring OSM Files and Data".

You can follow several scenarios to export and import OSM data, depending on the reason and the type of data that you need.

- Exporting and Importing the OSM Model Data Only
- Exporting and Importing the OSM Model and a Single Order
- Exporting and Importing a Range of Orders and the OSM Model
- Exporting and Importing a Range of OSM Orders Only

Exporting and Importing the OSM Model Data Only

You can export and then import only the OSM model data (cartridge data). To do this, you use the export/import utilities to generate a PAR file with options for exporting the OSM model.

Exporting and Importing OSM Schema Data
**Exporting OSM Model Data**

To export model data only:

1. Open a terminal and log in to SQL*Plus as the OSM core schema user.

2. Verify that the export/import utility package (**om_exp_imp_util_pkg**) exists by running the following command:

   ```sql
   select count(*) from user_objects where object_type = 'PACKAGE' and OBJECT_NAME = 'OM_EXP_IMP_UTIL_PKG';
   
   COUNT(*)
   -------
   1
   
   If the count that is returned is 1, the package exists and you do not need to create it. If the count is zero, you must create the package.

3. Open another terminal and take the OSM server offline. For more information about stopping OSM, see "Starting and Stopping OSM".

4. Return to the original terminal and allow the output to be generated by running the following command:

   ```sql
   set serveroutput on
   
   set trimspool on
   
   Extra white space in the PAR file causes the export to fail.

6. Specify the file where you want to capture the generated output by running the following command. The database creates this file in the folder you were in when you logged into the sqlplus session:

   ```sql
   spool osm_expdp_model_parfile.txt
   
   Export the model by running the **om_exp_imp_util_pkg.generate_expdp_model_parfile** procedure:

   ```sql
   exec om_exp_imp_util_pkg.generate_expdp_model_parfile
   
   Stop capturing the generated output by running the following command:
9. Log off the SQL*Plus session. For example:
   exit

10. Using a text editor, remove the following lines from the osm_expdp_model_parfile.txt file:
   - exec om_exp_imp_util_pkg.generate_expdp_model_parfile
   - PL/SQL procedure successfully completed.
   - spool off

11. (Optional) Modify PAR file parameters as needed in osm_expdp_model_parfile.txt file. For more information, see “Changing PAR File Parameters”.

12. Export the model tables by running the following command:
    expdp SourceOSMCoreSchemaUserName PARFILE=osm_expdp_model_parfile.txt

13. (Optional) Purge existing model data by running the following command:
    exec om_exp_imp_util_pkg.purge_model

    **Note:** Purging the data before importing it to the target schema ensures that there are no constraint violations when importing duplicate data.

14. Print the schema data to a text file by doing the following:
   a. Log in to SQL*Plus as the OSM core schema user.
   b. Allow the output to be generated by running the following command:
      set serveroutput on
   c. Specify the file where you want to capture the generated output by running the following command:
      spool osm_expdp_schema_info.txt
   d. Print the schema by running the om_exp_imp_util_pkg.print_schema_info procedure:
      exec om_exp_imp_util_pkg.print_schema_info;
   e. Stop capturing the generated output by running the following command:
      spool off

    **Note:** Keep this text file with the OSM database dump (dmp) files because the file contains information that will be used when you import the dmp files.

    The Data Pump Export utility unloads data and metadata into a set of operating system files called a dump file set, which is then imported using the Data Pump Import utility. For more information about Oracle Data Pump utility and the dmp files, see Oracle Database documentation.
Preparing the Target OSM Schema Before Import

Before you import the OSM model data that you exported from a source schema, you must prepare the OSM target schema to which you want to import that data.

Creating the Target OSM Schema

If the target schema does not already exist, run the OSM installer (of the same OSM version as the source schema) to create it.

Adding Partitions

If the target schema is partitioned and you are importing order data, you must ensure that the order IDs that you want to import map to existing partitions. If any order ID is greater than or equal to the greatest partition upper bound, add one or more partitions as needed.

For more information about adding partitions, see "Managing the OSM Database Schema".

To add partitions to a schema:

1. Log in to SQL*Plus as the OSM core schema user.
2. Add partitions by running the following command:
   
   ```sql
   exec om_part_maintain.add_partitions(n)
   ```

   where \(n\) is the number of partitions that you want to add.

3. Query the `user_tab_partitions` table to view the newly added partitions.

   ```sql
   select * from user_tab_partitions where table_name = 'OM_ORDER_HEADER';
   ```

   Note: If you are not on the latest version of your OSM release, the `add_partitions` procedure might not work. In this case, see "Managing the OSM Database Schema" for information about how to add partitions.

Importing OSM Model Data

After you prepare the target schema, import the OSM model data and finalize the target schema by running the target schema setup script.

For information about Oracle Data Pump and the dmp files, see Oracle Database documentation.

To import OSM model data:

1. Log in to SQL*Plus as the OSM core schema user.
2. Disable constraints and triggers, and stop jobs by running the following command:

   ```sql
   exec om_exp_imp_util_pkg.pre_impdp_setup
   ```

   This command ensures that there are no errors when importing OSM data.

3. From another terminal, import the model tables. For example:

   ```sql
   impdp TargetOSMCoreSchemaUserName DIRECTORY=DATA_PUMP_DIR DUMPFILE=osm_expdp_model%U.dmp LOGFILE=osm_impdp_model.log REMAP_SCHEMA=SourceOSMCoreSchemaUserName:TargetOSMCoreSchemaUserName REMAP_TABLESPACE=SourceOSMTablespace:TargetOSMTablespace
   ```
4. Finalize the target OSM schema by running the following command:

```sql
exec om_exp_imp_util_pkg.post_impdp_setup
```

This enables the constraints and triggers, and resubmits the jobs that were disabled and stopped before importing the OSM data.

Exporting and Importing the OSM Model and a Single Order

In some cases, you might want to analyze a single known order from a large schema, and you can selectively export that specific order. To extract a single order for analysis in another environment, you need to export both the order and the model. First you export the order and then you export the model.

Exporting OSM Order Data

Order data is saved in order tables. To select the order data that you want to export, query the order ID. Most tables store the order ID in the `order_seq_id` column. If there are exceptions that use a different name for the order ID column, you must export and import these separately.

Preparing to Export Order Tables for a Single Order

To prepare to export order tables:

1. Log in to SQL*Plus as the OSM core schema user.
2. Verify that the export/import utility package (`om_exp_imp_util_pkg`) exists by running the following command:

```sql
select count(*) from user_objects where object_type = 'PACKAGE' and OBJECT_NAME = 'OM_EXP_IMP_UTIL_PKG';
```

COUNT(*)
----------
1

If the count that is returned is 1, the package exists and you do not need to create it.

**Note:** If the export/import utility package does not exist in the database, you can run the installer to create it. For information about running the installer, see OSM Installation Guide.

When you run the OSM installer, make sure that you select the "Database Schema" component to create the export/import utility package.

3. Verify that the order that you want to export is not open by running the following SQL commands:

```sql
set serveroutput on
exec om_exp_imp_util_pkg.print_open_order_count(a_min_order_id =>orderid_min, a_max_order_id =>orderid_max);
```

where both `orderid_min` and `orderid_max` are the ID of the order that you want to export. For example: `a_min_order_id => 123456, a_max_order_id => 123456`. For
more information, see "About Order Export Queries".

The following message appears:

There are no open orders

If the order specified is open and the server should be taken offline, the following message appears:

The open order count is: 1

Note: Oracle recommends that you always check for open orders before you export order data.

4. If the order that you want to export is open, take the OSM server offline. For more information about stopping OSM, see "Starting and Stopping OSM".

5. Allow the output to be generated by running the following command:

   set serveroutput on

6. Prevent extra white space in the generated PAR file by running the following command:

   set trimspool on

Extra white space in the PAR file causes the export to fail.

**Exporting Order Tables That Define an Order Sequence ID**

To export order tables that define an ID in the order_seq_id column:

1. Follow all the steps of the procedure in "Preparing to Export Order Tables for a Single Order".

2. Specify the file where you want to capture the generated output by running the following command:

   spool osm_expdp_orders_parfile.txt

3. Export orders by running the om_exp_imp_util_pkg.generate_expdp_order_parfile procedure.

   exec om_exp_imp_util_pkg.generate_expdp_order_parfile;

4. Stop capturing the generated output by running the following command:

   spool off

5. Modify the PAR file option query in osm_expdp_orders_parfile.txt to select the single order that you want to export.

   For example, modify where order_seq_id > 0 to where order_seq_id = 123456; where 123456 is the ID of the order that you want to export. For more information, see "About Order Export Queries".

6. (Optional) Modify PAR file parameters as needed in osm_expdp_model_parfile.txt. For more information, see "Changing PAR File Parameters".

7. Export the order tables by running the following command:

   expdp SourceOSMCoreSchemaUserName PARFILE=osm_expdp_orders_parfile.txt
Exporting the OSM Model Data

After you export the single order, export the OSM model data. For more information, follow the procedure to export the model data in the topic "Exporting and Importing the OSM Model Data Only".

---

**Note:** Model tables that are system-managed, or contain system-managed parameters, are excluded from the export because these tables are created by the database installer and already exist in the target schema.

---

Importing the OSM Model and Order Data

Before you import the OSM model data, prepare the target schema. For more information, see "Preparing the Target OSM Schema Before Import".

After you prepare the target schema, you import first the OSM model data, and then the OSM order data that you previously exported. Most tables store the order ID in the `order_seq_id` column. There are a few exceptions that you must export and import separately.

To import the OSM data:

1. Log in to SQL*Plus as the OSM core schema user.
2. Disable constraints and triggers, and stop jobs by running the following command:
   
   ```
   exec om_exp_imp_util_pkg.pre_impdp_setup
   ```

   Running this command ensures that there are no errors when importing OSM data.

3. Import the model data by running the following command:

   ```
   impdp TargetOSMCoreSchema UserName DIRECTORY=DATA_PUMP_DIR DUMPFILE=osm_expdp_model%U.dmp LOGFILE=osm_impdp_model.log REMAP_SCHEMA=SourceOSMCoreSchema UserName:TargetOSMCoreSchema UserName REMAP_TABLESPACE=SourceOSMTablespace:TargetOSMTablespace
   ```

   For more information about these parameters, see "About Import Parameters".

4. Import order tables that define an order sequence ID by running the following command:

   ```
   impdp TargetOSMCoreSchema UserName DIRECTORY=DATA_PUMP_DIR DUMPFILE=osm_expdp_orders%U.dmp LOGFILE=osm_impdp_orders.log REMAP_SCHEMA=SourceOSMCoreSchema UserName:TargetOSMCoreSchema UserName REMAP_TABLESPACE=SourceOSMTablespace:TargetOSMTablespace TRANSFORM=oid:n
   ```

   For more information about these parameters, see "About Import Parameters".

5. Finalize the OSM target schema by running the following command:

   ```
   exec om_exp_imp_util_pkg.post_impdp_setup
   ```

   This enables the constraints and triggers, and resubmits the jobs that were disabled and stopped before importing the OSM data.
Exporting and Importing a Range of Orders and the OSM Model

You can replicate an OSM schema using a subset, or range, of orders from a large schema.

Exporting the OSM Order Data Range

Order data is saved in order tables. To select the order data that you want to export, query the order ID. Most tables store the order ID in the order_seq_id column. There are a few exceptions, which must be exported and imported separately.

Print the schema data. For more information see, "About Import Parameters".

Preparing to Export Order Tables for a Range of Orders

To prepare to export order tables:

1. Log in to SQL*Plus as the OSM core schema user.

2. Verify that the export/import utility package (om_exp_imp_util_pkg) exists by running the following command:
   
   ```sql
   select count(*) from user_objects where object_type = 'PACKAGE' and OBJECT_NAME = 'OM_EXP_IMP_UTIL_PKG';
   
   COUNT(*)
   --------
   1
   ``

   If the count that is returned is 1, the package exists and you do not need to create it.

   **Note:** If the export/import utility package does not exist in the database, you can run the installer to create it. For information about running the installer, see OSM Installation Guide.

   When you run the OSM installer, make sure that you select the Database Schema component to create the export/import utility package.

3. Verify that none of the orders that you want to export are open by running the following SQL commands:

   ```sql
   set serveroutput on
   exec om_exp_imp_util_pkg.print_open_order_count(a_min_order_id => orderid_min);
   
   where orderid_min is the minimum bound value of a range of order IDs. For more information, see "About Import Parameters".

   The following message appears:

   There are no open orders

   If any of the orders within the range specified are open and the server should be taken offline, the following message appears:

   The open order count is: n

   where n is the number of open orders.
Note: Oracle recommends that you always check for open orders before you export data.

4. If any of the orders that you want to export are open, take the OSM server offline. For more information about stopping OSM, see "Starting and Stopping OSM".

5. Allow the output to be generated by running the following command:
   ```sql
   set serveroutput on
   ```

6. Prevent extra white space in the generated PAR file by running the following command:
   ```sql
   set trimspool on
   ```

Extra white space in the PAR file causes the export to fail.

Exporting Order Tables That Define an Order Sequence ID

To export order tables that define an order sequence ID:

1. Follow all the steps of the procedure in "Preparing to Export Order Tables for a Single Order".

2. Specify the file where you want to capture the generated output by running the following command:
   ```sql
   spool osm_expdp_orders_parfile.txt
   ```

3. Export the orders by running the `om_exp_imp_util_pkg.generate_expdp_order_parfile` procedure. For example:
   ```sql
   exec om_exp_imp_util_pkg.generate_expdp_order_parfile(a_min_order_id => 100000);
   ```

   ```sql
   DIRECTORY=DATA_PUMP_DIR
   DUMPFILE=osm_expdp_orders%U.dmp
   FILESIZE=1GB
   LOGFILE=osm_expdp_orders.log
   CONTENT=DATA_ONLY
   PARALLEL=1
   QUERY=" where order_seq_id >= 100000"
   TABLES=(
     OM_ATTACHMENT,
     OM_HIST$DATA_CHANGE_NOTIF,
     OM_HIST$FALLOUT,
     OM_HIST$FALLOUT_NODE_INSTANCE,
     OM_HIST$FLOW,
     OM_HIST$NOTIFICATION,
     OM_HIST$ORDER_HEADER,
     OM_HIST$ORDER_INSTANCE,
     OM_HIST$ORDER_STATE
     ...
     OM_JMS_THREAD,
     OM_SYSTEM_EVENT
   )
   ```

   PL/SQL procedure successfully completed.

4. Run the following command, which stops capturing the generated output:
5. (Optional) Modify PAR file options as needed in `osm_expdp_model_parfile.txt`. For more information, see "Changing PAR File Parameters".

6. Export the order tables by running the following command:

   `expdp SourceOSMCoreSchemaUserName PARFILE=osm_expdp_orders_parfile.txt`

7. (Optional) Purge existing OSM orders by running the following command:

   `exec om_exp_imp_util_pkg.purge_orders`

---

Note: Purging the data before importing it to the target schema ensures there are no constraint violations when importing duplicate data.

---

Exporting the OSM Model Data

After you export the orders, you can export the OSM model data. For information about exporting the model data, follow the procedure in the topic "Exporting and Importing the OSM Model Data Only".

---

Note: Model tables that are system-managed, or contain system-managed parameters, are excluded from the export because these tables are created by the database installer and already exist in the target schema.

---

Importing OSM Model and Order Data

Before you import the OSM model data, prepare the target schema. For more information, see "Preparing the Target OSM Schema Before Import".

After you prepare the target schema, you import first the OSM model data, and then the OSM order data that you previously exported. As with the export procedure, most tables store the order ID in the `order_seq_id` column. There are a few exceptions that must be exported and imported separately. Use the order table import procedure that corresponds to the table in which the order ID that you want to import is stored.

To import the OSM data:

1. Log in to SQL*Plus as the OSM core schema user.
2. Disable constraints and triggers, and stop jobs by running the following command:

   `exec om_exp_imp_util_pkg.pre_impdp_setup`

   Running this command ensures that there are no errors when importing OSM data.

3. Import order tables that use a different name for the order ID column by running the following command:

   `impdp TargetOSMCoreSchemaUserName DIRECTORY=DATA_PUMP_DIR DUMPFILE=osm_expdp_model%U.dmp LOGFILE=osm_impdp_model.log REMAP_SCHEMA=SourceOSMCoreSchemaUserName:TargetOSMCoreSchemaUserName REMAP_TABLESPACE=SourceOSMTablespace:TargetOSMTablespace`

   For more information about these parameters, see "About Import Parameters".
4. Import order tables that define an order sequence ID by running the following command:

   \texttt{impdp TargetOSMCoreSchemaUserName DIRECTORY=DATA\_PUMP\_DIR DUMPFILE=osm\_expdp\_orders\%U.dmp LOGFILE=osm\_impdp\_orders.log REMAP\_SCHEMA=SourceOSMCoreSchemaUserName:TargetOSMCoreSchemaUserName REMAP\_TABLESPACE=SourceOSMTablespace:TargetOSMTablespace TRANSFORM=oid:n}

   For more information about these parameters, see "About Import Parameters".

5. Finalize the OSM target schema by running the following command:

   \texttt{exec om\_exp\_imp\_util\_pkg.post\_impdp\_setup}

   This enables the constraints and triggers, and resubmits the jobs that were disabled and stopped before importing the OSM data.

---

Exporting and Importing a Range of OSM Orders Only

If you have a target schema that already contains a subset of orders and the OSM model data, you can use the information in this section to export and import a range of OSM orders only.

\textbf{Note:} This section does not provide information about importing OSM model data to a target schema. If you want to do that, see "Importing OSM Model Data."

---

Exporting an Additional Range of Orders

Order data is saved in order tables. To select the range of order data that you want to export, query the order IDs at each end of the range. Most tables store the order ID in the \texttt{order_seq_id} column. There are a few exceptions, which must be exported and imported separately.

Preparing to Export Order Tables for a Range of Orders

To prepare to export order tables for a range of orders:

1. Log in to SQL*Plus as the OSM core schema user.
2. Verify that the export/import utility package (\texttt{om\_exp\_imp\_util\_pkg}) exists by running the following command:

   \texttt{select count(*) from user_objects where object_type = 'PACKAGE' and OBJECT\_NAME = 'OM\_EXP\_IMP\_UTIL\_PKG';}

   \texttt{COUNT(*)}

   \texttt{----------}

   \texttt{1}

   If the count that is returned is 1, the package exists and you do not need to create it.
Note: If the export/import utility package does not exist in the database, you can run the installer to create it. For information about running the installer, see OSM Installation Guide.

When you run the OSM installer, make sure that you select the Database Schema component to create the export/import utility package.

3. Verify that the orders that you want to export are not open by running the following SQL commands:

   ```sql
   set serveroutput on
   exec om_exp_imp_util_pkg.print_open_order_count(a_min_order_id => orderid_min);
   ```

   where `orderid_min` is the minimum bound value of a range of order IDs. For more information, see "About Order Export Queries".

   If the orders within the range specified are not open and the server does not need to be taken offline, the following message appears:

   There are no open orders

   If any of the orders within the range specified are open and the server should be taken offline, the following message appears:

   The open order count is: n

   where `n` is the number of open orders.

   Note: Oracle recommends that you always check for open orders before you export data.

4. If any of the orders that you want to export are open, take the OSM server offline. For more information about stopping OSM, see "Starting and Stopping OSM".

5. Allow the output to be generated by running the following command:

   ```sql
   set serveroutput on
   ```

6. Prevents extra white space in the generated PAR file by running the following command:

   ```sql
   set trimspool on
   ```

   Extra white space in the PAR file causes the export to fail.

**Exporting a Range of Orders from Order Tables That Define an Order Sequence ID**

To export a range from order tables that define an order sequence ID:

1. Follow all the steps of the procedure in "Preparing to Export Order Tables for a Range of Orders".

2. Specify the file where you want to capture the generated output by running the following command:

   ```sql
   spool osm_exdpd_orders_parfile.txt
   ```
3. Export orders by running the `om_exp_imp_util_pkg.generate_expdp_order_parfile` procedure. For example:

   ```plsql
   exec om_exp_imp_util_pkg.generate_expdp_order_parfile(a_min_order_id => 100000,
               a_max_order_id => 200000);
   ```

   ```plaintext
   DIRECTORY=DATA_PUMP_DIR
   DUMPFILE=osm_expdp_orders%U.dmp
   FILESIZE=1GB
   LOGFILE=osm_expdp_orders.log
   CONTENT=DATA_ONLY
   PARALLEL=1
   QUERY=" where order_seq_id between 100000 and 200000"
   TABLES=(
               OM_ATTACHMENT,
               OM_HIST$DATA_CHANGE_NOTIF,
               OM_HIST$FALLOUT,
               OM_HIST$FALLOUT_NODE_INSTANCE,
               OM_HIST$FLOW,
               OM_HIST$NOTIFICATION,
               OM_HIST$ORDER_HEADER,
               OM_HIST$ORDER_INSTANCE,
               OM_HIST$ORDER_STATE
               ...
               OM_JMS_THREAD,
               OM_SYSTEM_EVENT
   )
   ```

   PL/SQL procedure successfully completed.

4. Stop capturing the generated output by running the following command:

   ```plsql
   spool off
   ```

5. (Optional) Modify PAR file parameters as needed in `osm_expdp_orders_parfile.txt`. For more information, see "Changing PAR File Parameters".

6. Export the order tables by running the following command:

   ```plsql
   expdp SourceOSMCoreSchemaUserName PARFILE=osm_expdp_orders_parfile.txt
   ```

7. (Optional) Purge existing order data by running the following command:

   ```plsql
   exec om_exp_imp_util_pkg.purge_orders
   ```

   **Note:** Purging the data before importing it to the target schema ensures there are no constraint violations when importing duplicate data.

You can use a variety of queries to selectively export the orders that you want. For more information, see "About Order Export Queries".

**Importing an Additional Range of Orders**

Before you import the OSM order data, prepare the target schema. For more information, see "Preparing the Target OSM Schema Before Import."

After you prepare the target schema, you import first the OSM order data that you previously exported. As with the export procedure, most tables store the order ID in the `order_seq_id` column. There are a few exceptions, which must be exported and
imported separately. Use the order table import procedure that corresponds to the table in which the order ID that you want to import is stored.

To import order data:

1. Log in to SQL*Plus as the OSM core schema user.
2. Disable constraints and triggers, and stops jobs by running the following command:
   
   `exec om_exp_imp_util_pkg.pre_impdp_setup`

   Running this command ensures that there are no errors when importing OSM data.

3. Import order tables that use a different name for the order ID column by running the following command:
   
   `impdp TargetOSMCoreSchemaUserName DIRECTORY=DATA_PUMP_DIR DUMPFILE=osm_expdp_orders%U.dmp LOGFILE=osm_impdp_orders.log REMAP_SCHEMA=SourceOSMCoreSchemaUserName:TargetOSMCoreSchemaUserName REMAP_TABLESPACE=SourceOSMTablespace:TargetOSMTablespace TRANSFORM=oid:n`

   For more information about these parameters, see "About Import Parameters".

4. Import order tables that define an order sequence ID by running the following command:
   
   `impdp TargetOSMCoreSchemaUserName DIRECTORY=DATA_PUMP_DIR DUMPFILE=osm_expdp_orders%U.dmp LOGFILE=osm_impdp_orders.log REMAP_SCHEMA=SourceOSMCoreSchemaUserName:TargetOSMCoreSchemaUserName REMAP_TABLESPACE=SourceOSMTablespace:TargetOSMTablespace TRANSFORM=oid:n`

   For more information about these parameters, see "About Import Parameters".

5. Finalize the target OSM schema by running the following command:
   
   `exec om_exp_imp_util_pkg.post_impdp_setup`

   This enables the constraints and triggers, and resubmits the jobs that were disabled and stopped before importing the OSM data.

### About Order Export Queries

You can use a variety of queries to select the orders that you want to export. Table 10–1 provides examples of queries that you can use to select the orders to export.

<table>
<thead>
<tr>
<th>Example Query</th>
<th>Data to Export</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>exec om_exp_imp_util_pkg.generate_expdp_order_parfile;</code></td>
<td>Exports all orders when exporting order tables that define an order sequence ID. You can also use this example and edit the PAR file to customize the QUERY option.</td>
</tr>
<tr>
<td><code>exec om_exp_imp_util_pkg.generate_expdp_order_parfile(a_max_order_id =&gt; 2000);</code></td>
<td>This example exports all orders below order ID 2000 for order tables that define an order sequence ID.</td>
</tr>
<tr>
<td><code>exec om_exp_imp_util_pkg.generate_expdp_order_parfile(a_min_order_id =&gt; 1000);</code></td>
<td>This example exports all orders above order ID 1000 for order tables that define an order sequence ID.</td>
</tr>
</tbody>
</table>
Changing PAR File Parameters

You can change the parameters in the generated export parameter file (PAR file). The PAR file is generated in the directory where you start SQL*Plus.

Table 10–2 describes the parameters in the PAR file.

Note: There are other parameters in the PAR file but if you change these, the export will not be successful.

For more information about the parameters that are available in the command line mode of the data pump export, see the Data Pump Export chapter of Oracle Database Utilities Guide.

Table 10–2  PAR File Parameters

<table>
<thead>
<tr>
<th>PAR File Parameter</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
</table>
| CONTENT            | ALL     | Enables you to filter what export unloads: data only, metadata only, or both.  
|                    |         | Note: If you change this parameter in any of the export scenarios in this chapter, the export will not be successful. |
Changing PAR File Parameters

DIRECTORY DATA_PUMP_DIR Specifies the directory to which export writes the dump file and the log file.

Note: For exports or imports performed in an Oracle RAC environment using Automatic Storage Management, change this parameter to point to the shared location. For more information, see the Data Pump Oracle RAC Considerations section of the Oracle Database documentation.

Note: If you are using a pluggable database (PDB) you cannot use the default DATA_PUMP_DIR variable and directory. You must specify a different variable and directory for the PDB instance for the DIRECTORY parameter. Each PDB instance that you want to use impdp with must have its own data pump directory.

The following sqlplus commands create and set permissions on a new variable PDB directory variable:

create directory pdb_variable_name as 'path';
grant read, write on directory pdb_variable_name to osm_core_schema;

where:
- pdb_variable_name is the name of the PDB directory variable
- path is the path to the data pump directory (for example /samplepath/pdbdatapumpdir)
- osm_core_schema is the core OSM schema (for example ordermgmt)

After creating the pdb_variable_name directory, update the data_pump_dir parameter value to the pdb_variable_name in the om_parameter table.

DUMPFILE osm_expdp_model%U.dmp Specifies the name and, optionally, the directory of objects of dump files for an export job.

FILESIZE 1 GB Specifies the maximum size of each dump file.

INCLUDE N/A Enables you to filter the metadata that is exported by specifying objects and object types for the current export mode.

Note: If you change this parameter in any of the export scenarios in this chapter, the export will not be successful.

LOGFILE osm_expdp_model.log Specifies the name and, optionally, the directory for the log file of the export job.

PARALLEL 1 Specifies the maximum number of processes of active execution operating on behalf of the export job.

Table 10–2 (Cont.) PAR File Parameters

<table>
<thead>
<tr>
<th>PAR File Parameter</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DIRECTORY DATA_PUMP_DIR</td>
<td>Specifies the directory to which export writes the dump file and the log file. Note: For exports or imports performed in an Oracle RAC environment using Automatic Storage Management, change this parameter to point to the shared location. For more information, see the Data Pump Oracle RAC Considerations section of the Oracle Database documentation. Note: If you are using a pluggable database (PDB) you cannot use the default DATA_PUMP_DIR variable and directory. You must specify a different variable and directory for the PDB instance for the DIRECTORY parameter. Each PDB instance that you want to use impdp with must have its own data pump directory. The following sqlplus commands create and set permissions on a new variable PDB directory variable: create directory pdb_variable_name as 'path'; grant read, write on directory pdb_variable_name to osm_core_schema; where: - pdb_variable_name is the name of the PDB directory variable - path is the path to the data pump directory (for example /samplepath/pdbdatapumpdir) - osm_core_schema is the core OSM schema (for example ordermgmt) After creating the pdb_variable_name directory, update the data_pump_dir parameter value to the pdb_variable_name in the om_parameter table.</td>
<td></td>
</tr>
<tr>
<td>DUMPFILE osm_expdp_model%U.dmp</td>
<td>Specifies the name and, optionally, the directory of objects of dump files for an export job.</td>
<td></td>
</tr>
<tr>
<td>FILESIZE 1 GB</td>
<td>Specifies the maximum size of each dump file.</td>
<td></td>
</tr>
<tr>
<td>INCLUDE N/A</td>
<td>Enables you to filter the metadata that is exported by specifying objects and object types for the current export mode. Note: If you change this parameter in any of the export scenarios in this chapter, the export will not be successful.</td>
<td></td>
</tr>
<tr>
<td>LOGFILE osm_expdp_model.log</td>
<td>Specifies the name and, optionally, the directory for the log file of the export job.</td>
<td></td>
</tr>
<tr>
<td>PARALLEL 1</td>
<td>Specifies the maximum number of processes of active execution operating on behalf of the export job.</td>
<td></td>
</tr>
</tbody>
</table>
About Import Parameters

You can add parameters in-line to data pump import command (impdp).

Table 10–3 describes the parameters you can use when impdp.

For more information about the parameters that are available in the command line mode of the data pump import, see the Data Pump Import chapter of Oracle Database Utilities Guide.

Table 10–3 Import In-line Parameters

<table>
<thead>
<tr>
<th>PAR File Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| DIRECTORY          | Specifies the directory to which import finds the dump file and the log file generated by the export.  
  Note: For imports performed in an Oracle RAC environment using Automatic Storage Management, change this parameter to point to the shared location. For more information, see the Data Pump Oracle RAC Considerations section of the Oracle Database documentation.  
  Note: If you are using a pluggable database (PDB) you cannot use the default DATA_PUMP_DIR variable and directory. You must specify a different variable and directory for the PDB instance for the DIRECTORY parameter. Each PDB instance that you want to use impdp with must have its own data pump directory.  
  The following sqlplus commands create and set permissions on a new variable PDB directory variable:  
  create directory pdb_variable_name as 'path';  
  grant read, write on directory pdb_variable_name to osm_core_schema;  
  where:  
  - pdb_variable_name is the name of the PDB directory variable  
  - path is the path to the data pump directory (for example /samplepath/pdbdatapumpdir)  
  - osm_core_schema is the core OSM schema (for example ordermgmt)  
  After creating the pdb_variable_name directory, update the data_pump_dir parameter value to the pdb_variable_name in the om_parameter table. |
| DUMPFILE           | Specifies the name and, optionally, the directory of objects of dump files for an import job. |
| LOGFILE            | Specifies the name and, optionally, the directory for the log file of the import job. |
| TRANSFORM          | Specifies whether the types being created should be assigned a new object identifier (OID). For example:  
  TRANSFORM=oid:n |
Troubleshooting Export/Import

Errors might occur during the process of exporting or importing data.

Table 10–4 lists some common export errors and their solutions.

<table>
<thead>
<tr>
<th>Error</th>
<th>Cause</th>
<th>Solution</th>
</tr>
</thead>
</table>
| ORA-39097: Data Pump job encountered unexpected error -12801 | There is an issue with the Oracle Data Pump export tool and Oracle RAC databases. For more information, see the knowledge article about the issue [Doc ID 13099577.8], available from the Oracle support website: https://support.oracle.com | Update the following database parameters: 
  ALTER SYSTEM set parallel_force_local=true
  ALTER SYSTEM set parallel_max_servers=0
  ALTER SYSTEM set parallel_min_servers=0 |
| ORA-39065: unexpected master process exception in MAIN | ORA-12801: error signaled in parallel query server PZ99, instance <instanceDetails> (4) ORA-01460: unimplemented or unreasonable conversion requested |                                                        |
| UDE-00014: invalid value for parameter, 'include' | The include parameter used in the export options PAR file contains more than 4,000 characters. This is normally due to extra white space at the end of each line when the file is spooled. | As a workaround, execute the following command in SQL Plus before generating the options PAR files: SQL> SET TRIMSPOOL ON |
Table 10–4 (Cont.) Common Export Errors

<table>
<thead>
<tr>
<th>Error</th>
<th>Cause</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>ORA-39001: invalid argument value</td>
<td>A previously generated version of the dmp file already exists.</td>
<td>Remove the previously generated version of the dmp file.</td>
</tr>
<tr>
<td>ORA-39000: bad dump file specification</td>
<td></td>
<td>or Specify the following option in the export PAR file to overwrite it: REUSE_DUMPFILES=Yes</td>
</tr>
<tr>
<td>ORA-31641: unable to create dump file &quot;+DATA/osm_expdp_model.dmp&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ORA-17502: ksfdcre:4 Failed to create file +DATA/osm_expdp_model.dmp</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ORA-15005: name &quot;osm_expdp_model.dmp&quot; is already used by an existing alias</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ORA-31693: Table data object &quot;]&quot;&lt;OSMCoreSchemaUserName&gt;&quot;.&quot;OSM_ORDER_HEADER&quot;:&quot;P_000000000000005001&quot;.&quot;SYS_SUBP6l2617&quot; failed to load/unload and is being skipped due to error: ORA-06502: PL/SQL: numeric or value error: character string buffer too small ORA-01403: no data found ORA-01403: no data found</td>
<td>The data pump export job name is too long. If no job name is specified when doing the export, the job name is automatically generated based on the schema name. If the schema name is long the job name can exceed the limit of 26 characters and causes this error. For more information, see the knowledge article about the issue [Doc ID 1502119.1], available from the Oracle support website: <a href="https://support.oracle.com">https://support.oracle.com</a></td>
<td>Specify the JOB_NAME option in the export PAR file. Make sure the value specified for the option is less than 26 characters long. JOB_NAME=osm_expdp_job</td>
</tr>
</tbody>
</table>

Table 10–5 lists some common import errors and their solutions.
### Table 10–5  Common Import Errors

<table>
<thead>
<tr>
<th>Error</th>
<th>Cause</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>ORA-31693: Table data object &quot;OSMCoreSchemaUserName&quot;.&quot;OM_ORCH_PLAN&quot;:&quot;SYS_SUBP44607&quot; failed to load/unload and is being skipped due to error: ORA-29913: error in executing ODCIEXTTABLEOPEN callout ORA-29400: data cartridge error ORA-39779: type &quot;&lt;OSMCoreSchemaUserName&gt;&quot;.&quot;OM_T_ORCH_PROCESS&quot; not found or conversion to latest version is not possible</td>
<td>The types being created should be assigned a new object identifier (OID). For more information, see the knowledge article about the issue [Doc ID 351519.1], available from the Oracle support website: <a href="https://support.oracle.com">https://support.oracle.com</a></td>
<td>Make sure the following option is specified when importing. This option assigns new OIDs. TRANSFORM=oid:n For more information, see the Data Pump: Import TRANSFORM section of Oracle Database documentation.</td>
</tr>
<tr>
<td>ORA-31693: Table data object &quot;&lt;OSMCoreSchemaUserName&gt;&quot;.&quot;&lt;OSMTableName&gt;&quot;.&quot;SYS_P44941&quot; failed to load/unload and is being skipped due to error: ORA-29913: error in executing ODCIEXTTABLEFETCH callout ORA-14400: inserted partition key does not map to any partition</td>
<td>The order IDs being imported are greater than the greatest partition upper bound.</td>
<td>Resolve this error by adding partitions. For more information, see “Adding Partitions.”</td>
</tr>
<tr>
<td>ORA-31693: Table data object &quot;&lt;OSMCoreSchemaUserName&gt;&quot;.&quot;&lt;OSMTableName&gt;&quot; failed to load/unload and is being skipped due to error: ORA-00001: unique constraint (&lt;OSMCoreSchemaUserName&gt;.&lt;OSMTableName&gt;) violated</td>
<td>The table already contains the data that is being imported.</td>
<td>Before you import, clean up the table using one or more of the following purge commands: SQL&gt; exec om_exp_imp_util_pkg.purge_orders SQL&gt; exec om_exp_imp_util_pkg.purge_model SQL&gt; exec om_exp_imp_util_pkg.purge_schema</td>
</tr>
<tr>
<td>ORA-39001: invalid argument value ORA-39046: Metadata remap REMAP_TABLESPACE has already been specified.</td>
<td>The same source tablespace has been specified more than once for the REMAP_TABLESPACE option.</td>
<td>This might happen when the OSM Core and Rule Engine schema use the same tablespace. In this case, you need to specify the REMAP_TABLESPACE for this tablespace only once.</td>
</tr>
<tr>
<td>ORA-00932: inconsistent datatypes: expected OM_T_ORCH_PROCESS got OM_T_ORCH_PROCESS</td>
<td>There is a known issue with data pump import that causes imports with REMAP_SCHEMA and TYPE definitions to generate this error.</td>
<td>Follow the steps outlined in the scenario “Exporting and Importing a Range of Orders and the OSM Model”. When generating the order PAR files, select the option to export all orders, that is, order_target_seq_id &gt; 0.</td>
</tr>
</tbody>
</table>
### Table 10–5 (Cont.) Common Import Errors

<table>
<thead>
<tr>
<th>Error</th>
<th>Cause</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDI-31626: operation generated ORACLE error 31626 ORA-31626: job does not exist ORA-39086: cannot retrieve job information ORA-06512: at &quot;SYS.DBMS_DATAPUMP&quot;, line 3326 ORA-06512: at &quot;SYS.DBMS_DATAPUMP&quot;, line 4551 ORA-06512: at line 1</td>
<td>This is an issue with data pump import privileges. For more information, see the knowledge article about the issue [Doc ID 1459430.1], available from the Oracle support website: <a href="https://support.oracle.com">https://support.oracle.com</a></td>
<td>Apply Patch 13715680 Or Follow the workaround in the notes of the associated bug to add the missing privileges. For more information, see bug 13715680 on the Oracle support website. <a href="https://support.oracle.com">https://support.oracle.com</a> The missing privileges are: SQL&gt; GRANT lock any table TO datapump_imp_full_database; SQL&gt; GRANT alter any index TO datapump_imp_full_database;</td>
</tr>
<tr>
<td>ORA-31693: Table data object &quot;&lt;InvalidSourceOSMCoreSchemaUserName&gt;&quot;.&quot;OM_SQL_LOG&quot; failed to load/unload and is being skipped due to error: ORA-00001: unique constraint (&lt;InvalidSourceOSMCoreSchemaUserName&gt;..XPKOM_SQL_LOG) violated</td>
<td>Import fails if the wrong source schema name is specified. If the wrong source schema user name is specified in the REMAP_SCHEMA option, data pump tries to import the data to the actual source schema. If the source schema exists in the target instance, the constraint violations are shown.</td>
<td>Verify that the correct source OSM core schema user name is specified in the import command: impdp TargetOSMCoreSchemaUserName DIRECTORY=DATA_PUMP_DIR DUMPFILE=osm_expdp_orders%U.dmp LOGFILE=osm_impdp_orders.log REMAP_SCHEMA=SourceOSMCoreSchemaUserName REMAP_TABLESPACE=SourceOSMTablespace TRANSFORM=oid:n</td>
</tr>
<tr>
<td>ORA-31693: Table data object &quot;&lt;InvalidSourceOSMCoreSchemaUserName&gt;&quot;.&quot;OM_MODEL_CLOB&quot; failed to load/unload and is being skipped due to error: ORA-29913: error in executing ODCIEXPORTTABLEFETCH callout ORA-00001: unique constraint (&lt;InvalidSourceOSMCoreSchemaUserName&gt;..XPKOM_MODEL_CLOB) violated</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table 10–5 (Cont.) Common Import Errors

<table>
<thead>
<tr>
<th>Error</th>
<th>Cause</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>ORA-31631: privileges are required</td>
<td>The schema user being used for import does not have the <code>imp_full_database</code> role. This role is required for imports only if you cannot use the system user because of the REMAP_SCHEMA bug. For more information, see the knowledge article about the issue <a href="https://support.oracle.com">Doc ID 1367290.1</a>, available from the My Oracle support website:</td>
<td>Grant <code>imp_full_database</code> role to the OSM core schema user. The following command grants this role:</td>
</tr>
<tr>
<td></td>
<td>REMAP_SCHEMA is not required if you do the import on a different database instance and the schema user is unchanged (uses the same ID and permissions as the source user).</td>
<td><code>SQL&gt; GRANT imp_full_database TO &lt;OSMCoreSchemaUserName&gt;;</code></td>
</tr>
<tr>
<td>ORA-39122: Unprivileged users may not perform REMAP_SCHEMA remappings.</td>
<td></td>
<td><strong>Caution:</strong> This role provides a user with access to any data in any schema in the database. Use caution when granting this role to users. For more information, see <code>Predefined Roles in an Oracle Database Installation</code> section of <em>Oracle Database Security Guide</em>. Oracle recommends removing this role after the import is complete. For more information, see <code>Guidelines for Securing User Accounts and Privileges</code> section in <em>Oracle Database Security Guide</em>. The following command removes this role:</td>
</tr>
</tbody>
</table>
This chapter describes how to monitor and manage the Oracle Communications Order and Service Management (OSM) system using the Oracle WebLogic Server Administration Console and Oracle Dynamic Monitoring Service (DMS) and aggregate metrics.

About Monitoring and Managing OSM

Oracle WebLogic Server requires a set of interrelated resources, such as database connections, Java Messaging Service (JMS) queues, execution threads, transactions, and system memory to work together in order to provide the functionality required of OSM.

You use the WebLogic Server Administration Console to manage these resources, including tasks such as starting and stopping servers, balancing the load on servers or connection pools, selecting and monitoring the configuration of resources, detecting and correcting problems, monitoring and evaluating system performance, and making sure that OSM is correctly deployed to the target servers.

The WebLogic Server Administration Console is a Web-based application that allows system administrators, support staff, and others to monitor and manage the OSM application remotely. See Oracle WebLogic documentation for more information.

To monitor run-time performance, OSM also provides the Oracle Dynamic Monitoring Service, which measures and reports performance metrics and traces performance over time.

Accessing the WebLogic Server Administration Console

To access the Oracle WebLogic Server Administration console do one of the following:

- If you are not connecting via Secure Socket Layer (SSL), enter the following URL into your browser:
  
  http://Hostname:Port/console
  
  where Hostname represents the DNS name or IP address of the computer on which the Administration server is installed and Port represents the address of the port on which the Administration server is listening for requests.

- If you are connecting via SSL, enter the following URL into your browser:

  https://Hostname:SSLPort/console
  
  where Hostname represents the DNS name or IP address of the computer on which the Administration server is installed and SSLPort represents the address of the
Using the WebLogic Console to Determine the Status of the OSM Application

After you have logged into the WebLogic console, you can access information about the status of the WebLogic servers and OSM deployments.

To access the status of the OSM server and deployments:

1. If you are not in the Home window of the WebLogic Console, click the Home icon in the upper left part of the window.

2. In the Environment subsection of the Domain Configurations section, click Servers.
   The Summary of Servers window is displayed. Server statuses are contained in the Health column of the table.

3. Click the Home icon in the upper left part of the window.

4. In the Your Deployed Resources subsection of the Domain Configurations section, click Deployments.
   The Summary of Deployments window is displayed. Deployment statuses are contained in the Health column of the table.

   **Note:** If any of the deployments are not in the status that you expected, you can use the buttons on this window to start and stop individual deployments if necessary.

Managing Log Files

This section details how to manage log files using the WebLogic Server Administration Console.

Viewing Log Files Using WebLogic Server Administration Console

You can view the OSM log files using WebLogic Server Administration Console.

To view the most recent messages in the log files:

1. Log in to the WebLogic Server Administration Console.

2. In the left pane of the Console, expand Diagnostics and select Log Files.

3. In the Log Files table, select the radio button next to the name of the log you want to view, and then click View.
The page displays the latest contents of the log file, up to 500 messages in reverse chronological order, meaning the messages at the top of the window are the most recent messages that the server has generated.

The log viewer does not display messages that have been rotated into archive log files.

**Configuring the Log View**

OSM logs everything that happens in the system, which can make the logs file quite large. Using the WebLogic Server Administration Console, you can configure the log views to show only the most recent messages.

To configure the view of the logs:

1. Log in to WebLogic Server Administration Console.
2. From the log files page, click **Customize this table**.
3. Choose the appropriate filter and view options, then click **Apply**. To add an option to the **Chosen** column, highlight it in the **Available** column, then click the arrow that points to the **Chosen** column. To deselect the option, reverse this action.

**Log Size and Rotation**

It is important to be able to manage your log file maximum size and rotations to prevent the logs from filling up your disk resources.

To manage your log settings:

1. Log in to WebLogic Server Administration Console.
2. Click **Environment**, then **Servers**, and then select a server from the list.

   This displays the General Configuration page for the selected server.

3. Click the **Logging** tab.

   The resulting page displays the logging settings for the selected server.

4. Modify the **Rotation file size** to 25,000 kilobytes for both tabs, General and HTTP.
5. Select the **Limit number of retained files** option and set the limit for the number of **Files to retain**. Do this for both tabs, General and HTTP.

**Managing Error Message Volume and Logging Levels**

OSM uses Log4j to generate and manage the system log messages. Through the use of Log4j, you can develop and maintain a logging strategy that minimizes the overall impact of logging operations on the application’s resources. It does this by letting you control the volume of log messages generated.

You also have the ability to change the level and detail of the logged messages temporarily and dynamically. This feature helps you to, for example, increase the level and detail of logged messages to help analyze performance problems within a production environment.

To read more about Log4j, refer to the Apache web site a:

http://logging.apache.org/log4j
Severity Levels
To control the volume of log messages generated and written to the output destinations (the console and the WebLogic log file, which are referred to as Appenders by Log4j), you assign severity levels to the various areas of the application that generate their own discrete messages (these areas are referred to as Categories by Log4). If you do not assign a severity level, the severity level is Off.

The following is an ascending list of the severity levels, starting with the least severe:

- **Debug**: (Least severe) Designates fine-grained informational events that are most useful to debug an application
- **Info**: Designates informational messages that highlight the progress of the application at coarse-grained level
- **Warning**: Designates potentially harmful situations
- **Error**: Designates error events that might still allow the application to continue running
- **Fatal**: (Most severe) Designates very severe error events that presumably leads the application to abort

If an event occurs inside a given category that triggers a message below the severity level assigned to the category (that is, less severe than the assigned level), Log4j does not generate the message.

**Example 1**
If you assigned the severity level Warning to a given category, Log4j does not generate any messages for that category that are flagged in the OSM code as Debug or Info level messages. Log4j will, however, generate all messages that are flagged as Warning, Error, and Fatal.

You can further control the number of messages written to the output destinations, or appenders, by also assigning a severity level to them. When you assign a severity level to an appender, it rejects messages below that severity level, even if Log4j passes the message to it.

**Example 2**
If you configure the console to the Warning severity level, and one of the categories is configured with the Info level, the console will not display the Info message, even though Log4j generates the message and passes it on to the console, because the message's severity level falls below the threshold for which the appender is configured. If, however, that same category later generates an Error level message, the console accepts and displays the message, because it carries a severity level equal to or higher than the console's threshold.

By default, the console and the WebLogic log file accepts all error messages, from the least severe to the most severe.

**Displaying Current Log Levels**
To display current log levels:

1. Open the log4jAdmin web page:
   - If you are not connecting via Secure Socket Layer (SSL), enter the following URL into your browser:
     http://host:portOrderManagement/admin/log4jAdmin
Managing Log Files

where *host* is the DNS name or IP address of the computer on which the Administration server is installed, and *port* is the port on which the Administration server is listening for requests for non-clustered environment, and is the proxy server port or the load-balancing server port in a clustered environment.

- If you are connecting via SSL, enter the following URL into your browser:

  https://host:SSLportOrderManagement/admin/log4jAdmin

  where *host* is the DNS name or IP address of the computer on which the Administration server is installed, and *SSLport* is the port on which the Administration server is listening for SSL requests for non-clustered environments, and is the proxy server port or the load-balancing server port in a clustered environment.

2. Log in using a user ID that is a member of the OMS_log_managers group. (You can add this group permission to a user using the WebLogic Server Administration Console.)

The complete list of available loggers is displayed.

3. You can use the Filter Loggers feature to check the logging level of specific categories or sub components. If you know the name of the category or sub-component that you want to check, you can use the filter to display only that category, or related categories.

If you want to see only a selection of the entries in the table:

   a. Enter the beginning of the name or a part of the name, in the Filter Loggers field.

   b. Click either:

      Begins With (if filtering on the beginning of the name)
      Contains (if filtering on just a part of the name)

The list displays the categories and subcategories that match the entry in the Filter Loggers field.

Configuring the Severity Levels

There are two methods by which you can configure the severity levels:

- **log4j.xml**: The log4j.xml file resides within the omslogging.jar of the OMS.ear file. Using this method you can change the default severity settings. This requires you to stop and restart the servers to take effect. See "Configuring the log4j.xml File".

- **log4jAdmin**: The log4jAdmin web page lets you change the severity levels while the servers are running. The changes take effect immediately (this method is temporary; the system reverts to the default levels you established in the log4j.xml file when you restart the server). See "Configuring Log Levels Temporarily."

---

**Note:** To use this method, your Web Client log-in ID must have OMS_log_manager permissions.

---

**Configuring the log4j.xml File**

Use this method to set up the default severity levels. To change these levels later, you must stop the server, modify the log4j.xml file, then restart the server.
If you need to modify the logging levels, you should use the `log4jAdmin` web page, as described in "Configuring Log Levels Temporarily".

You can also change settings in the `log4j.xml` file in order to minimize logging and improve performance. If there is no limit on logging, the garbage collection activity for large orders becomes high, which has an impact on performance.

There are two sections of the `log4j.xml` file that you should look at when configuring this file:

- **Appenders:** This section defines the output destination for the messages. At installation, the Appenders section contains two entries, as follows:
  - Console: From this entry you control the level of messages that the WebLogic console accepts.
  - WebLogic: From this entry you control the level of messages that the WebLogic log file accepts.

- **Categories:** This section contains references to all of the OSM categories that generate messages and gives you the ability to control the level of messages they generate.

To configure the `log4j.xml` file:

1. Unpack the `oms.ear` file. See "Unpacking the oms.ear File" for details.
2. Use a text editor to open `oms.ear\osm-ejb\logging\log4j.xml`.
3. Tune input/output by setting the following values:
   
   ```
   <param name="immediateFlush" value="false"/>
   <param name="bufferedIO" value="true"/>
   ```

4. Go to the **Console** entry in the Appenders section. To do this, search for the following string, which is at the top of the Appenders section:
   
   ```
   <!-- Append messages to the console -->
   ```

   The Console entry governs what level of messages are written to the console.

5. If necessary, change the threshold level. By default, it is configured to DEBUG, allowing the console to display all messages sent to it. If you want to restrict the number of messages displayed, change the threshold entry to the severity level appropriate for your installation (see "Severity Levels" for a description of the severity levels).

   In the example below, the level is changed from DEBUG to INFO.

   Before
   
   ```
   <param name="Threshold" value="DEBUG"/>
   ```

   After
   
   ```
   <param name="Threshold" value="INFO"/>
   ```

6. Go to the WebLogic's log file entry in the Appenders section. To do this, search for the following:
   
   ```
   <!-- Append messages to the weblogic's log file-->
   ```

   The weblogic log file entry governs what level of messages are written to the WebLogic log file.

7. If necessary, change the threshold level as described in step 5.
8. Go to the Categories section. To do this, go to the top of the file and search for the string:

```
<category name="org.jboss.system">
```

This takes you to the first category entry in the file.

9. Review each of the categories in this section, changing the severity level where necessary.

In the example below, the level is changed from INFO to WARNING.

**Before**

```
<category name="org.jboss.system">
<priority value="info"/>
</category>
```

**After**

```
<category name="org.jboss.system">
<priority value="warning"/>
</category>
```

10. Turn off logging of large messages by setting the following:

```
<category name="CloseCreationFailedTroubleTicketTask.CloseOrderCreationFailedTroubleTicketBean">
<priority value="fatal"/>
</category>
```

11. Restrict the length of a logged message. For example, restrict the length to 100,000 characters for each of the 3 layout patterns, by changing the following:

```
%m to %.100000m
```

12. When you finish updating the categories, save and close the **log4j.xml** file.

13. Pack and redeploy the **oms.ear** file to the WebLogic server. See "Packing the oms.ear File" for details.

**Configuring Log Levels Temporarily**

Use **log4jAdmin** web page to check the current logging levels or to change the logging levels dynamically.

---

**Note:** You can use this method only to change the severity levels of the Categories. To change the Appender level (the logfile output, for example, console or file), you must reconfigure the **log4j.xml** file (see "Configuring the log4j.xml File" for an explanation of the Categories, the Appenders, and how to configure the **log4j.xml** file).

---

The changes you make to the logging severity levels using this method are temporary; they are not written to the **log4j.xml** file. When you restart the server, the logging levels return to those that are configured in the **log4j.xml** file.

To change the logging levels temporarily:

1. Open the **log4jAdmin** web page:
If you are not connecting via Secure Socket Layer (SSL), enter the following URL into your browser:

http://host:portOrderManagement/admin/log4jAdmin

where host is the DNS name or IP address of the computer on which the Administration server is installed, and port is the port on which the Administration server is listening for requests for non-clustered environment, and is the proxy server port or the load-balancing server port in a clustered environment.

If you are connecting via SSL, enter the following URL into your browser:

https://host:SSLportOrderManagement/admin/log4jAdmin

where host is the DNS name or IP address of the computer on which the Administration server is installed, and SSLport is the port on which the Administration server is listening for SSL requests for non-clustered environments, and is the proxy server port or the load-balancing server port in a clustered environment.

2. Log in using a user ID that is a member of the OMS_log_managers group. (You can add this group permission to a user using the WebLogic Server Administration Console.)

The complete list of available loggers is displayed. You can use the Filter Loggers feature to filter the list of loggers. See "Displaying Current Log Levels" for more information.

3. Scan down the entries in the left-hand column of the page. This column contains a list of the categories and their related sub-components. Find the category or sub-component for which you want to change the logging level.

Example

com.mslv.oms.automation (category name)

or

com.mslv.oms.automation.AutomationDispatcher (sub-component name)

Note: To show specific category and sub-component names, at least one corresponding order should have been previously submitted to OSM. For example, to show sub-components beginning with "oracle.communications.ordermanagement.requestprocessor", at least one instance of the corresponding SalesOrder10000DeliverEBM.xml sample order should have been previously submitted.

4. Scan across the row to the severity levels. The level that currently is selected is highlighted in a different color from the other levels and appears in the Effective Level column.

5. Click the level to which you want to make the change.

   ■ To change an entire Category, click the category name.
   ■ To change the sub-component, click the sub-component name.

The change takes place immediately.
Monitoring and Analyzing Performance

This section describes several tools that are specific to OSM, as well as some general tools, that gather data when your system has performance-related issues.

Monitoring Performance Using WebLogic Server Administration Console

The WebLogic Server Administration Console provides a real-time view of system performance.

To access the performance monitor:

1. Start the WebLogic Server Administration Console.
2. Click Environment, then Servers, and then select a server from the list.
   
   This displays the General Configuration page for the selected server.
3. Click Monitoring.
4. Click the Health tab to view the health status for all OSM related sub-systems. If the status is not OK, review the reason and, if required, access the server log for more information.

   Health status severity levels are shown in the bottom left pane of the console under System Status.

5. Click the Performance tab to view JVM memory utilization statistics for the server.

   If the memory usage statistics are high, you must allocate more memory to the Java runtime by increasing the \(-Xms\) and \(-Xmx\) parameter values. The file that contains this parameter depends on the operating system you are using and the WebLogic Server to which the OSM server is deployed. An example is shown below.

   Example:

   Open the file $\{DOMAIN_HOME\}/bin/setDomainEnv.sh and provide the following parameter values:

   \[MEM_ARGS="-Xms2048m -Xmx2048m"\]

   The more memory that you allow for the Java runtime, the faster OSM will run under high loads.

6. Click the Threads tab to monitor thread activity for the server. Important columns to monitor are Queue Length and Pending User Request Count. A count of zero is optimal, meaning no user requests are stuck or waiting to be processed.

   If any of the counts are unusually high in the pool, go to the second table to troubleshoot the individual threads.
7. Click the **Workload** tab to monitor the Work Managers configured for the server. If the **Pending Requests** count is not zero, you should access the server log file for more information or perform a thread dump.

8. Click the **JDBC** tab to monitor the database pool connections configured for the server. The installer creates a maximum capacity of 15 connections for the connection pool.

   If the **Active Connections High Count** is 15 and the **Active Connections Average Count** is half of that, you may need to increase the number of connections. See "Managing Database Connections" for more information.

9. Click the **JTA** tab to monitor transaction activity on the server. There should be no Roll Back statistics in the summary; if so, view the server log file for more information.

### Monitoring the Managed Server

The high-level analytical tools and considerations for monitoring the managed server include the following:

- Monitor JVM using JConsole or JVisualVM, which are bundled with the JDK.
- Use the `jstackscript.sh` shell script to generate a series of thread dumps and run the results through the ThreadLogic tool. You can then review ThreadLogic advisories and follow the recommendations.
- In the Weblogic administration console, ensure all are servers are marked as **OK** in system status. If the server(s) listed have another status (for example, **Overloaded** or **Critical**) you must investigate immediately to determine the particular sub-system of the server that is under stress. Also, check the server logs.
- Use the **Reporting** page in OSM Task Web Client to monitor the orders that are in progress and task statistics. For information about the Reporting page, see OSM Task Web Client User’s Guide. You can also analyze OSM order completion rates using a script. For more information, see "Tools for Performance Testing, Tuning, and Troubleshooting”.

### Analyzing Garbage Collection

Garbage collection logs for the performance test runs include important information about the heap usage for the particular test. Use this data for later analysis and adjustment of the JVM heap and for tuning garbage collection. Use GCViewer or Garbage CAT to analyze garbage collection data that you have collected using Oracle JDK. For more information about GCViewer, see "GCViewer”.

### Analyzing Class Loading

Class loading issues result in "class not found" errors. WebLogic, from version 10.3.4 onward, has the Classloader Analysis Tool (CAT) available by default. For more information about the tool, see the section about using the Classloader Analysis Tool in Oracle Fusion Middleware Developing Applications for Oracle WebLogic Server document.

### Coherence Datagram Testing

OSM uses Coherence for caching, which works for clustered environments. You can determine, using the datagram test, whether Coherence issues are because of either network or configuration problems. For more information, see the section about running the datagram test utility in Oracle Coherence Administrator’s Guide.
Monitoring WebLogic Server
Monitor the WebLogic Server by using either WLST or JMX. Resources are always consumed when using monitoring tools and Oracle recommends that you begin by using WLST, because it uses fewer resources compared to JMX. Add other tools only when required.

There are open source projects available for monitoring by using JMX calls from the server. These monitor WebLogic server activities, such as JMS queues, JDBC connections, execute thread pool, and so on.

The following link provides tools that allow you to monitor by using JMX:
https://github.com/romix/JMX-server-monitoring/

Monitoring the Operating System
The most important aspects to consider when monitoring the operating system are: CPU, memory, and network use.

CPU use on the system should not exceed 90 percent. If you must keep latency to a minimum on your system, keep CPU use even lower. Additionally, you must account for times of peak use, as well as the possibility of increased load on the remaining servers in case of a cluster node failure.

For memory use, configure your OSM system in order to avoid paging and swapping. For more information, see the JVM section of the Weblogic Server documentation.

For network use, bandwidth and latency are the primary factors that have an impact on OSM performance and throughput. In particular, OSM is sensitive to latency between Oracle database servers and WebLogic servers. Monitor the amount of data being transferred across the network by checking the data transferred between WebLogic servers, and between WebLogic servers and Oracle Database servers. This amount must not exceed your network bandwidth. You can check for symptoms of insufficient bandwidth by monitoring for retransmission and duplicate packets.

Monitor the WebLogic servers to ensure that Input/Output (I/O) wait times are low.

Common operating system commands are available to measure and monitor CPU, memory, and network and storage performance and usage on Linux and Solaris systems. This includes top on Linux systems or prstat on Solaris operating systems, as well as sar, mpstat, iostat, netstat, vmstat, and ping on both platforms.

OSWatcher Black Box is a tool that monitors operating system statistics. For more information, see "OSW Black Box".

You can also use the Remote Diagnostics Agent (RDA) tool to generate a report that includes operating system measurements. For more information about this tool, see "Remote Diagnostics Agent".

Gathering OSM Execution Statistics
An execution statistics API allows you to gather execution statistics and provides a JMX interface for remote management and monitoring. The execution statistics MXBeans are registered with the Weblogic Server Runtime Bean Server.

To enable JConsole to access the MXBeans:
1. Enable the IIOP protocol for the WebLogic Server instance that hosts your MBeans.
2. Set the JAVA_HOME and WLS_HOME environment variables.
3. At the command prompt, enter the following commands:
Monitoring and Analyzing Performance

UNIX/Linux:

```
jconsole -Djava.class.path=$JAVA_HOME/lib/jconsole.jar:$JAVA_HOME/lib/tools.jar:$WLS_HOME/lib/wljmxclient.jar
-Djmx.remote.protocol.provider.pkgs=weblogic.management.remote
service:jmx:rmi:///jndi/iiop://hostname:port/weblogic.management.mbeanservers.runtime
```

Windows:

```
jconsole -Djava.class.path=%JAVA_HOME%\lib\jconsole.jar;%JAVA_HOME%\lib\tools.jar;%WLS_HOME%\lib\wljmxclient.jar
-Djmx.remote.protocol.provider.pkgs=weblogic.management.remote
service:jmx:rmi:///jndi/iiop://localhost:7001/weblogic.management.mbeanservers.runtime
```

The New Connection screen opens.

4. Enter the username and password for the WebLogic administrator.

5. Click the MBeans tab.

6. Expand the oracle.communications.ordermanagement node.

   The registered execution statistics managers for each application are displayed under the ExecutionStatsManagerMXBean node.

7. In Operation Invocation, click Start.

   **Note:** If you want to export the statistics to an XML file, use the dump method by clicking dump. You can leave the argument empty (which it is by default) and the file that is created is named using the application ID. The file is then saved under the stats sub-directory of the WebLogic domain.

Analyzing Heap Dumps

You analyze the heap so you can find out what objects are using memory. Use a tool such as Eclipse Memory Analyzer (MAT) to analyze the heap dump. For documentation and to download the tool, use the following link:

http://www.eclipse.org/mat/

   **Note:** Use the following command in Linux and UNIX to produce a heap dump:

   ```
jmap -heap:format=live pid
```

   where pid is the ID of the OSM WebLogic Server processor. For more information about heap dumps, see Oracle JMap documentation.

Analyzing Thread Dumps

A thread dump is a snapshot of the status of every thread within a JVM at a particular point. Analyzing thread dumps can be useful if your OSM system becomes sluggish or hangs. This type of analysis can reveal hot spots, in the form of frequently executed code, as well as threads that are stuck because of deadlocks or thread contention.
While a single thread dump can sometimes reveal the problem (for example, a thread deadlock), a series of thread dumps generated at regular intervals (for example, 10 thread dumps at 30-second intervals) is often required to troubleshoot more complex problems, like threads that are not progressing and are waiting for other processes.

You can use the VisualVM tool to generate and analyze thread dumps. The JStack tool can be used to generate thread dumps from a command line or using a shell script.

After you generate a series of thread dumps, Oracle recommends that you run the results through the ThreadLogic tool. This tool uses analysis algorithms and recommends solutions, or advisories, for common problems, including for WebLogic applications. This tool builds on the Thread Dump Analyzer (TDA) by adding logic for common patterns found in application servers. ThreadLogic is able to parse Sun, JRockit, and IBM thread dumps and provide advice based on predefined and externally defined patterns. To download the tool, use the following link:

https://java.net/projects/threadlogic/downloads

---

**Note:** Use the following method in Linux and UNIX to produce a thread dump:

```
kill -3 pid
```

where `pid` is the ID of the OSM WebLogic Server processor.

---

**Monitoring the Database**

When you monitor the operating system on the database server, make sure the system has enough idle CPU and I/O wait times are low (nothing more than few percent of IO waits are acceptable).

Recommendations for monitoring the database using Enterprise Manager include the following:

- In Enterprise Manager, go to the Database, Performance page. View SQL performance tables to determine whether any SQL statement is consuming too much CPU in relation to the others.

- A correctly tuned OSM schema has `insert`, `update`, and `delete` statements at the top of the SQL performance tables in Enterprise Manager. A `select` statement near the top indicates a problem with the SQL execution plan.

- You can run SQL Advisor on SQL statements and then follow any recommendation it provides.

- You must examine any spike in Session Activity in the area of CPU, CPU Wait, User I/O, Commit, Configuration, or Concurrency. Take action in accordance with your database administration practices.

- Periodically capture Active Session History (ASH) and Automatic Workload Repository (AWR) reports, which characterize the typical workloads. A database administrator who is familiar with these reports can analyze this data for performance issues.

- Use the Monitor Cluster, Performance page to ensure that, if you are running Active-Active Oracle RAC, both nodes have an equal load and show similar usage patterns.
AWR is a database monitoring process that collects information about overall database server performance statistics. Table 11–1 shows some of the measures that AWR collects in its report at periodic intervals.

<table>
<thead>
<tr>
<th>Measure</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Top 5 Waits</td>
<td>Shows bottlenecks in the database. Use this measure to guide further investigations.</td>
</tr>
<tr>
<td>Instance Efficiencies</td>
<td>Monitors various activities with the target of reaching 100 percent efficiency.</td>
</tr>
<tr>
<td>Load Profile</td>
<td>Reports the rates for reads, writes, and transactions.</td>
</tr>
<tr>
<td>Cache Sizes</td>
<td>Reports the size of each SGA region.</td>
</tr>
<tr>
<td>Shared Pool Stats</td>
<td>Provides a summary of the changes to shared pools.</td>
</tr>
<tr>
<td>Operating System Stats</td>
<td>Captures IOPS, CPU, memory, and network usage.</td>
</tr>
<tr>
<td>Input/Output Activity</td>
<td>Monitors input/output activity by tablespace, data file, and buffer pool statistics.</td>
</tr>
<tr>
<td>SQL Section</td>
<td>Provides execution metrics for SQL.</td>
</tr>
</tbody>
</table>

Managing Database Connections

In OSM, database connections are managed through database pools that are set up in WebLogic.

The database pool connections can be configured through the WebLogic Server Administration Console by clicking Services/JDBC/Data Sources/oms_pool/Connection Pool.

Use the parameters on the Connection Pool page to modify and fine tune your connection pool settings as the need arises.

Updating Database Schema Statistics

For production databases, Oracle recommends that you update your database schema statistics regularly during an off-peak time of day.

To update the database schema statistics:

1. Log into SQL*Plus as the OSM database user, using the primary database schema.
2. Determine the ESTIMATE_PERCENT parameter you would like to use for statistics gathering.

Oracle recommends setting the ESTIMATE_PERCENT parameter of GATHER_SCHEMA_STATS to DBMS_STATS.AUTO_SAMPLE_SIZE to maximize performance gains while achieving necessary statistical accuracy. With a larger ESTIMATE_PERCFENT value, statistics gathering for a large database could take several hours. If you prefer a faster, less accurate result, use a small ESTIMATE_PERCENT value such as 1.

3. Run the following procedure, substituting the ESTIMATE_PERCENT value that you determined for percent.

```sql
BEGIN
  DBMS_STATS.GATHER_SCHEMA_STATS ( 
    OWNNAME=>'USER', 
    ESTIMATE_PERCENT=>'percent', 
```
Using JMS Queues to Send Messages

OSM uses JMS Queues and Topics which are both JMS Destinations. Queues follow a point-to-point communication paradigm, while Topics follow the publish and subscribe paradigm.

Note: In an OSM clustered environment, you must use JMS queues as a JMS destination to receive JMS events. Do not use JMS topics in an OSM clustered environment.

When OSM sends data to an external system, such as UIM or ASAP, it does so by sending JMS messages to the appropriate JMS request queue of an external system.

If the external system is not processing the requests from OSM, the queues get backlogged. It is important to be able to monitor the size of the JMS queues in order to know whether or not they are backing up.

To monitor the JMS queues:

1. Login to WebLogic Administration Console
   Click Services/Messaging/JMS Servers/oms_jms_server.
   The General Configuration page is displayed.

2. Click the Monitoring tab then click Active Destinations.
   A list of active destinations targeted to the server is displayed.

   Note: The default view of the table does not contain the Consumers column. We recommend that you customize the table using Customize link to include this column, along with any other customizations you may want to make.

   The Consumers column defines the current number of listeners on the destination. If a destination does not have any listeners, then the external system will not receive the messages.

   The Messages Current column defines the current number of unprocessed messages in the JMS destination. A large number (for example, 10,000) in this destination is a problem. It means that the messages are not getting processed, or that the messages are getting processed but errors are occurring and the messages are getting put back on the destination.

When OSM is first installed, the following JMS destinations are present:

- oms_behavior_queue: Used for customizing task assignment
- oms_events: Internal destination used for events such as automation, notifications, and task state changes
- oms_order_events: Used for order state changes such as OrderCreateEvent, OrderStateChangeEvent, AmendmentStartedEvent, OrderCancelledEvent
Using Work Managers to Prioritize Work

- **oms_order_updates**: Internal destination used for processing amendments
- **oms_signal_topic**: Internal destination used to trigger a metadata refresh

### Monitoring the Event Queue

The destination `oms_events` is the JMS destination to which OSM events are sent. OSM events are sent when tasks change states, or when notifications occur.

The number of consumers for the `oms_events` is determined by which plug-ins are configured. If plug-ins are configured, the number of consumers must not be 0.

If there is a problem with automation plug-ins getting invoked, check the consumers queue and the messages queue.

If the consumers queue is less than the number of plug-ins, the plug-ins are not configured correctly. Check the `OSM_home\SDK\Samples\DatabasePlugin\map\automationmap.xml` file and make sure that all of the plug-ins have been deployed.

If the messages queue keeps getting larger, the plug-ins may not be committing the transactions during processing of the events. Verify the plug-in code and check the log files.

### Sending Data to External Systems Using Plug-Ins

If there are external systems deployed to the same WebLogic instance as OSM, when you monitor the JMS destinations, watch for the following.

**Note:** The important columns are Consumers, Messages, and Messages Received.

If the number in the messages column for these queues continues to grow, the external system may not be processing the messages sent by OSM. You must check to see if the external system is working properly.

If the number of consumers for the queues is 0, such as UIMrequestQueue, the external system may not have configured its listeners properly. Check to see if the external system is configured properly.

### About OSM and XA Support

The OSM database does not support XA transactions because the Oracle thin-client driver used for JDBC connections does not support XA. However, the OSM WebLogic Server configuration uses an XA emulation feature in order to get a two-phase commit across JMS/JDBC automation transactions.

Even though OSM uses a non-XA driver for database transactions, external XA resources can still participate in transactions. For example, JMS bridges can be XA-enabled for an outside application, but the OSM side of the transaction will still use the non-XA emulated two-phase commit. Note that this also applies to JMS queues which support Application Integration Architecture (AIA) cartridges.

### Using Work Managers to Prioritize Work

You use WebLogic Administration Console to configure work managers, which prioritize OSM work and manage threads. WebLogic Server uses a single thread pool.
in which all types of work are executed. Work managers allow you to define rules and run-time metrics that WebLogic Server uses to prioritize this work.

For more information about using work managers to optimize scheduled work, see Oracle Fusion Middleware Configuring Server Environments for Oracle WebLogic Server.

Creating and Configuring Work Managers

When you run the installer and restart the WebLogic Server, the system creates and configures default work managers and components. Table 11–2 shows the default work managers and components that are created and configured by the installer.

<table>
<thead>
<tr>
<th>Work Manager or Component</th>
<th>Configuration Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum Threads Constraint</td>
<td>Name: osmMaxThreadConstraint Count: 90% of the number of connections in the OSM datasource</td>
<td>Used by all OSM Work Managers to prioritize work.</td>
</tr>
<tr>
<td>Minimum Threads Constraint</td>
<td>Name: osmMinThreadConstraint Count: 10</td>
<td>Used by the work managers osmTaskClientWorkManager and osmOmClientWorkManager to guarantee that some threads are reserved for the web clients.</td>
</tr>
<tr>
<td>Work Manager</td>
<td>Name: osmAutomationWorkManager Maximum Threads Constraint: osmMaxThreadConstraint</td>
<td>Used to process work performed by automation tasks.</td>
</tr>
<tr>
<td>Work Manager</td>
<td>Name: osmXmlWorkManager Maximum Threads Constraint: osmMaxThreadConstraint</td>
<td>Used to process requests coming from external clients for the OSM XML API.</td>
</tr>
<tr>
<td>Work Manager</td>
<td>Name: osmTaskClientWorkManager Maximum Threads Constraint: osmMaxThreadConstraint Minimum Threads Constraint: osmMinThreadConstraint Request Class: osmFairShareReqClass</td>
<td>Used to process requests from manual users using the Task Web Client.</td>
</tr>
<tr>
<td>Work Manager</td>
<td>Name: osmWsJmsWorkManager Maximum Threads Constraint: osmMaxThreadConstraint</td>
<td>Used to process OSM JMS Web Service requests.</td>
</tr>
<tr>
<td>Work Manager</td>
<td>Name: osmWsHttpWorkManager Maximum Threads Constraint: osmMaxThreadConstraint</td>
<td>Used to process OSM HTTP Web Service requests.</td>
</tr>
<tr>
<td>Work Manager</td>
<td>Name: osmOmClientWorkManager Maximum Threads Constraint: osmMaxThreadConstraint Minimum Threads Constraint: osmMinThreadConstraint</td>
<td>Used to process requests from manual users using the Order Management Web Client.</td>
</tr>
</tbody>
</table>

In addition to the defaults, you can create and configure other work managers and components. For more information, see Oracle Fusion Middleware Configuring Server Environments for Oracle WebLogic Server.
To create a new work manager:

1. Log in to the WebLogic Server Administration Console.
2. In the left pane of the Console, expand Environment and select Work Managers. This page displays a summary of the work managers configured for the system.
3. Click New. The Create a New Work Manager Component page displays.
4. Select Work Manager, and then click Next.
5. Enter a name for the new Work Manager, and then click Next.
6. Select a deployment target from the list of available targets.
7. Click Finish. The new work manager is displayed in the summary of work managers list.

**Overriding the Internet Explorer Language in the OMS Web Clients**

If the Internet Explorer installation that is used to access the OMS web clients is set to a language other than English, and this language matches one of the properties files included in the oms.ear file, the Web client prompts appear in the non-English language.

The language used in the Web clients is controlled by the resources.properties file. Additional language property files in the oms.ear file include:

- resources_cs.properties (Czech language properties file)
- resources_zh.properties (Chinese language properties file)

To remove support for a non-English language, unpack the oms.ear file, remove the corresponding properties file, repack the oms.ear file and redeploy it. See "Working with the oms.ear File" for details.

For example, if the browser is set to use the Czech language, and the resources_cs.properties file exists in the oms.ear file, the Web client prompts appear in Czech. Removing the resources_cs.properties file causes the Web client prompts to appear in English, even though the browser language setting is still configured to the Czech language.

**Evaluating System Configuration Compliance**

The OSM compliance tool captures a snapshot of your system’s configuration and evaluates this configuration against established rules, which are based on best practices and guidelines. Using these rules, the compliance tool analyzes the system and produces an evaluation result that allows you to verify that your system is optimally configured for your environment.

The compliance tool is based on JMX technology and captures the configuration snapshot from target environments, such as JMX MBeans for the WebLogic domain, the OSM system and database parameters, and the OSM coherence cluster.

Several default compliance tool parameters are specified in the oms-config.xml file. Use these parameters to change the directory path where the system stores the following:

- The compliance snapshot output
The evaluation results

The snapshot files that will be evaluated

The evaluation rules

You can also specify the location of a compliance configuration override file in the oms-config.xml. For more information about setting values in the oms-config.xml file, see "Configuring OSM with oms-config.xml."

You can use the compliance configuration override file to change the behavior of built-in compliance rules or to stop built-in rules from executing altogether. The override file is an xml file that contains two configuration sections <profile-override> and <rule-filter-override>.

Use the <profile-override> section to change values in the default compliance profile. The compliance profile defines the context of a compliance evaluation. In general, minimum, maximum, and a default value are given for a rule configuration. Compliance rules check if the configuration value is within the boundary of minimum and maximum values. Compliance rules also check if the configuration value is still set to the default value which is an indication of lack of testing and tuning.

Use the <rule-filter-override> section to exclude built-in rules from executing during compliance evaluation. You can use the <exclude> element to exclude entire rules or the <exclude-result> element to selectively filter out rule results based on match criteria.

For more detailed information about the built-in rules, adding custom rules, and modifying the compliance configuration override file, see the compliance tool documentation, provided in the SDK at the following location:

OSM_home/SDK/Compliance/doc/index.html

Running the Compliance Tool

Run the compliance tool using WebLogic Scripting Tool (WLST) scripts. For information about the post-installation tasks required for OSM to interact with the compliance tool, see the post-installation section of OSM Installation Guide. For more information about WLST, see WebLogic Scripting Tool documentation:

http://docs.oracle.com/cd/E29542_01/web.1111/e13715/using_wlst.htm

Note: If you are running the compliance tool in a cluster, there are additional factors to consider. For more information, see "Cluster Considerations".

To run the compliance tool:

1. Enter WLST interactive mode and connect to an OSM managed server.
2. Do one of the following:
   - If you want to create the snapshot then evaluate it do the following:
     a. Run the following command, which creates a snapshot of the system configuration:
        
osmSnapshot("snapshot_directory")

        where snapshot_directory is the directory in which you want to put the snapshot of the system configuration. If you leave this blank, the system
puts the snapshot in the default directory specified by the `oms-config.xml` file.

b. Run the following command, which verifies configuration compliance by evaluating the snapshot:

```
osmEvaluate("evaluate_directory")
```

where `evaluate_directory` is the directory in which you want to put the evaluation of the snapshot against the compliance rules. If you leave this blank, the system puts the evaluation in the default directory specified by the `oms-config.xml` file.

- If you want to take a snapshot and immediately evaluate the snapshot do the following:

  a. Run the following command, which creates the compliance results file:

```
osmCompliance("compliance_directory")
```

where `compliance_directory` is the directory in which you want to put the compliance results. If you leave this blank, the system puts the compliance results in the default directory specified by the `oms-config.xml` file.

### Cluster Considerations

You can run the compliance tool in a clustered environment. Keep in mind that the `osmSnapshot` command collects the following types of configuration:

- **common**: the same on every managed server
- **coherence**: available only on the managed server that has the coherence MBean server
- **server**: specific to the managed server

It is typical to run the compliance tool on the managed server that has the coherence MBean server because doing so will evaluate the broadest set of configuration (common, coherence, and server).

### Evaluating Compliance Results

The compliance tool saves the evaluation results in the directory specified in the WLST command or in the `oms-config.xml` parameter. The evaluation results are saved in two formats: HTML and XML.

Note: The names of the evaluation result files are in the following formats: `yyyymmdd-hhmss.xml` and `yyyymmdd-hhmss.html`.

Table 11–3 lists the columns in the HTML evaluation results file and provides a description for the information contained in each column.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Index</td>
<td>Specifies rule index in the table.</td>
</tr>
<tr>
<td>Rule Name</td>
<td>Displays the name of the compliance rule.</td>
</tr>
</tbody>
</table>
Table 11–4 lists the elements in the XML evaluation results file and provides a description for the information stored in each element.

### Table 11–4 Compliance Tool Evaluation Results File: XML

<table>
<thead>
<tr>
<th>Column Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>compliantResult</td>
<td>Contains the evaluation result of a compliance rule. Each compliant result element stores evaluation result for one rule.</td>
</tr>
<tr>
<td>compliantMessage</td>
<td>Appears if a compliance rule passes. The message is not displayed when the rule fails. This message is copied directly from the rule file.</td>
</tr>
<tr>
<td>description</td>
<td>Specifies a general description of the rule.</td>
</tr>
<tr>
<td>keywordlist</td>
<td>Displays keywords related to targeted configurations.</td>
</tr>
<tr>
<td>nonCompliantMessage</td>
<td>Appears if a compliance rule fails. The message is not displayed when the rule passes. This message is copied directly from the rule file.</td>
</tr>
<tr>
<td>nonCompliantObjects</td>
<td>Contains a list of objects that failed the compliance rule. Non-compliant objects are not displayed when the rule passes.</td>
</tr>
<tr>
<td>nonCompliantObject</td>
<td>Contains an object that failed the compliance rule.</td>
</tr>
<tr>
<td>rationale</td>
<td>Displays the rationale behind a compliance rule.</td>
</tr>
<tr>
<td>referenceUrl</td>
<td>Specifies the link to a document that describes the compliance rule in more detail.</td>
</tr>
<tr>
<td>ruleName</td>
<td>Displays the name of the compliance rule.</td>
</tr>
<tr>
<td>severity</td>
<td>Defines the severity of this rule. Possible values are:</td>
</tr>
<tr>
<td></td>
<td>- Error</td>
</tr>
<tr>
<td></td>
<td>- Warning</td>
</tr>
<tr>
<td></td>
<td>- Information</td>
</tr>
<tr>
<td>filterReason</td>
<td>The reason a rule is excluded from compliance evaluation.</td>
</tr>
</tbody>
</table>
About Metrics Data

The OSM Order Metrics Manager measures and reports managed server and domain-wide performance metrics, and traces performance for your OSM environment. You view metric data using an interface called Oracle Dynamic Monitoring Service (DMS). For more information about DMS, see Oracle Fusion Middleware Performance and Tuning Guide.

Viewing Metrics Data

Metrics data is gathered and displayed using tables. You view OSM-specific metric data, which is displayed in the DMS Metrics tables and Aggregated Metrics tables. You can also use the DMS interface to view non-OSM metric data, for example, WebLogic and non-J2EE metrics.

You can also view metrics data by using Oracle Enterprise Manager. For information about viewing metrics using Enterprise Manager, see Oracle Fusion Middleware Performance and Tuning Guide. For more information about viewing and analyzing metric data using Oracle Application Management Pack (AMP) for Enterprise Manager, see "Analyzing Metric Data".

For information about installing metrics rules (ADML) files, see OSM Installation Guide.

Figure 11–1 shows the DMS user interface for an environment that has Oracle HTTP server configured. A list of links to metrics tables is on the left-hand side. When you click the link to a metrics table, the table opens in the pane on the right.

To view metrics tables using DMS:

1. In a browser, enter one of the following URLs:

   http://hostname:port/dms

   https://hostname.dnsdomainname.tld:sslport/dms

   where hostname is the name of the admin server, port is the port for the admin server domain, and sslport is the secure port for the admin server domain.

   The DMS Spy login screen is displayed.
2. Log in using your WebLogic username and password.
   A list of all metric table names is displayed.
3. Click the metric table name for the table that you want to view.
   The details of the metric table are displayed in the right-hand pane.

   **Note:** For any table, you can display a list of the metric column names, a description of each column name, and the derivatives and units for the metric. To view this information, click the **Metric Definition** link at the bottom of the table.

---

**About DMS Metrics Tables**

The DMS metrics tables display OSM metrics. You can view these metrics using the DSM Spy servlet, or using other systems, such as Enterprise Manager using MBeans.

DMS metric tables are named using the following convention: *industry_application_* _MetricName*. For example, *Comms_OrderManagement_OrdersRuntime*.

**About WebLogic Metrics Tables**

Using the links to WebLogic Metrics tables, you can view data about the WebLogic server and JMX. The tables include information about management of Java memory and WebLogic diagnostics, configuration, and runtime data.

**About Non-J2EE Metrics Tables**

The data for these metrics tables comes from an Oracle HTTP server. The tables include data about the PL/SQL API, the Oracle HTTP server, and the Oracle Process Manager and Notification Server (OPMN).

If you do not see a link to these tables, your domain is not connected to an Oracle HTTP server for load balancing. For information about adding and configuring an Oracle HTTP server, see the topic about configuring the WebLogic server domain for an OSM cluster in *OSM Installation Guide*.

**About Aggregated Metrics Tables**

Because OSM aggregated metrics are based on existing metric tables, aggregation allows you to specify the new aggregated metric tables and the steps by which to compute their data. The rules are written in XML configuration files, which are also called metric rules files or ADML files. For more information about updating metric rules, see the topic about manually loading metric rules files in *OSM Installation Guide*.

Aggregated metric tables are named using the following convention: *industry_* _application:MetricName*. For example, *Comms_OrderManagement:Cluster_all_orders*.

**About Metric Rules Files**

Metric rules files, or ADML files, contain the aggregate metric configurations. All aggregate metrics are defined in these files. For OSM, there are the following aggregate files:

- **Server based:** for managed server based aggregation. The file name is prefixed with **server-**, which indicates the metric rules contained in the XML file are for
aggregating OSM metrics across the managed server. The full file name is: server-oracle-comms_osm-11.0.xml.

- **Domain wide**: for aggregating OSM metrics across the domain. The file name is prefixed with domain-. The full file name is: domain-oracle-comms_osm-11.0.xml.

### Analyzing Metric Data

You can view the metric data that is gathered by the system about your environment by accessing the DMS interface that is provided by default with your OSM installation. The tables provide information about various aspects of your OSM environment, and you can compare the pieces of data over time to determine whether there is a problem. For example, if you want to make sure the system is properly reallocating resources after cluster resizing, you view the related aggregated metric data in the WebLogic cluster tables.

You can also view that same data using the Application Management Pack, which is a plug-in that is available in Enterprise Manager. This view provides a graphical representation of metric data, which allows you to analyze and interpret data over time. For information about viewing metrics using Enterprise Manager, see *Oracle Fusion Middleware Performance and Tuning Guide*. For information about using Application Management Pack, see *Oracle Application Management Pack for Oracle E-Business Suite User’s Guide*.

Figure 11–2 shows the Application Management Pack dashboard for an OSM system. The interface also displays metrics by server, order type, and cartridge.
Figure 11–2  OSM System Data in the AMP Dashboard
This chapter describes how to configure time zone settings in Oracle Communications Order and Service Management (OSM). This is an optional configuration task.

Configuring Time Zone Settings

The oms_timezone parameter, in the om_parameter table of the database, must be set to the non-dst timezone. This setting must be defined in seconds offset to GMT for the time zone where your OSM database resides. For example, if you are setting this parameter for Eastern Standard Time (EST), use the value -18000 (60 seconds x 60 minutes = 3600 seconds—or one hour—x 5 for five hours offset from GMT). Time zones east of GMT use positive numbers, and time zones west of GMT use negative numbers.

This setting must be done to the OSM database/schema at creation time. The parameter is called "oms server tz offset seconds" in the Scripter. This value is used by both the OSM database and OSM Administrator.

The database_timezone_offset, in the oms-config.xml file of the OSM server, must be set to be exactly the same as the value set in 1. This value is used by the OSM server.
This chapter provides guidelines to help you improve the performance of your Oracle Communications Order and Service Management (OSM) system.

**Guidelines for Performance Testing**

In working toward achieving the optimum performance for your OSM system, keep the following high-level goals in mind:

- Maximizing the rate by which orders are processed by the system (order throughput)
- Minimizing the time it takes for each order to complete

During performance testing and tuning, you must find a balance between these considerations, which often affect one another. For example, when you achieve maximum throughput, orders might not have the fastest completion time. It is also sometimes required that you configure OSM to respond faster but at the expense of order throughput.

The performance goal might also need to consider a set of technical safety boundaries for the system, such as a hardware resource utility (for example, heap size), or the ability to process a certain number of orders and manual users at the same time. There might be other secondary technical goals in order to ensure the system can manage under stress, such as handling a large burst of orders, outages of other systems, or failover and recovery of OSM hardware. These additional considerations also need to be verified during performance testing.

**Measuring Order Throughput**

This section outlines some of the factors you must consider when planning and implementing a performance testing strategy.

The most common measure of OSM performance is order throughput. OSM must fulfill orders at the rate that is determined by business need. OSM throughput is measured in task transitions per second (TPS).

Although the TPS metric varies for each deployment, it is useful for you to consider the following approximate guidelines and adjust them as your circumstances require:

- **Simple orders**, which typically complete less than 10 tasks per order.
- **Moderate orders**, which complete approximately 25 tasks per order.
- **Complex orders**, which complete approximately 100 tasks per order.
- **Very complex orders**, which complete approximately 1000 tasks per order.
Based on the above guidelines, you can calculate order throughput per second using the following formula:

\[
\frac{\text{throughput in task transitions per second}}{\text{average number of tasks per order}}
\]

Throughput can then be calculated hourly, by multiplying by 3600 seconds per hour; or daily, by multiplying by 3600 seconds per hour plus the number of operating hours per day.

Every release of OSM includes a product benchmark report, called a Performance Summary Report (PSR), which you can use as a reference when planning your performance testing. The PSR is an internal artifact. TPS is the most common performance metric reported in the OSM PSR.

Performance Considerations

Although many factors impact OSM performance, you can classify them into the following categories:

- **Hardware**: OSM performance is bounded by the limitations of the hardware on which it runs. For example, when maximum CPU, memory, or other resources are reached.
- **Software**: OSM relies on the other software within the solution, so properly tuning the operating system, JVM, database, WebLogic server, and so on is important in ensuring optimal performance for OSM.
- **Solution**: OSM cartridges provide the metadata instructions for the OSM server to fulfill orders according to business requirements. The level of complexity defined in OSM cartridges has an impact on how many orders OSM processes and how quickly.

Performance Considerations for Solutions

The following design-time factors have an impact on the performance of a solution:

- Number of process steps (tasks).
- Number of concurrently executing automation plug-in instances.
- Number, size, and depth of data elements in the order.
- Order view sizes and complexity.
- Incoming XML size (number of order line items) and complexity.
- Degree of XSLT and XQuery transformation.
- Complexity of the generated orchestration plan.
- Average number of revision orders per base order.

Performance Considerations for Large Orders

In some cases, you might want to model large orders for OSM. A large order typically contains a sizeable payload with more than a hundred order items, and where each order item may contain many data elements. OSM provides the following features that can help you manage these large orders:

- Order automation concurrency control (OACC) is a policy-driven OSM function that you can use to limit the number of concurrent automations plug-in instances that OSM can process at one time. For large orders, this ability can significantly
reduce contention caused by an excessive number of automation plug-ins processing at the same time. High levels of automation plug-in contention can create performance issues because of the number of message retries and timeouts on the JMS queues. You can specify a policy using a parameter in the oms-config.xml file (see “oms-config.xml Parameters”) or you can include an OACC policy in solution cartridge. See OSM Developer’s Guide for information about creating OACC policies.

- When you design automation plug-ins, you can implement GetOrder operations with the OrderDataFilter element that explicitly specifies which parts of the order to return data from. GetOrder operations without the filter returns the entire order and in large orders, this can impact performance. See OSM Developer’s Guide for information the OrderDataFilter.

- When you design automation plug-ins, you can implement updateOrder with the ResponseView filter that can request that order data be returned in an updateOrder response. If the response includes a fulfillment state update, then OSM automatically filters the response so that only order items and order components impacted by the fulfillment state update are included. See OSM Developer’s Guide for information the ResponseView.

- Use the oracle.communications.ordermanagement.table-layout.size and the oracle.communications.ordermanagement.table-layout.fetch-size oms-config.xml parameter to create a threshold that limit the number of order rows that OSM can retrieve at one time from the database when using Data tab in the Order Management Web client. See “oms-config.xml Parameters” for more information.

- Use the oracle.communications.ordermanagement.table-layout.threshold oms-config.xml parameter to specify a threshold that automatically applies the style behavior table layout if a multi-instance node exceeds the threshold when using Data tab in the Order Management Web client. See “oms-config.xml Parameters” for more information.

- Ensure that the show_all_data_history_logs_for_orderdetails is set to false to reduce the number of logs that OSM generates. See “oms-config.xml Parameters” for more information.

- For large orders where there are manual tasks that take a long time to accomplish, you can modify the order cash ejection logic. The default logic specifies that orders are ejected from the cash after 60 seconds. You can modify this ejection logic to manual tasks with a longer duration using the following oms-config.xml parameters:
  - OrderCacheMaxEntries
  - OrderCacheInactivityTimeout
  - ClosedOrderCacheMaxEntries
  - ClosedOrderCacheTimeout

These configuration parameters control all order cache settings. All other forms of controlling order cache size and order retention periods are no longer supported. See “oms-config.xml Parameters” for more information.

Planning for OSM Performance Testing
Before you test OSM performance in your system, you must plan and prepare for performance testing.
Ensure that project planning allows you sufficient time and resources to complete the following high-level steps:

1. Identify the solution architecture surrounding OSM, and identify performance acceptance criteria.
2. Secure hardware for performance testing.
3. Set up a performance test environment.
4. Design performance test cases.
5. Secure or develop test harnesses, such as external systems emulators, order creation scripts, system monitoring, and test data extraction scripts.
6. Run a test plan, analyze the outcome, tune the system, and report the results.
7. Improve and tune the system based on the analysis.
8. Repeat steps 6 and 7 until you have achieved your performance goals.

Guidelines for a Performance Test Environment

This section introduces general guidelines to conduct performance testing on OSM. The method involves testing and tuning various layers until you reach the performance goals.

Setting Goals for Testing

The goals that you set in your performance test plan, including the expected results for each test run, are typically based on business objectives for performance.

As part of your planning, document the characteristics of order complexity and other performance factors, as suggested in "Performance Considerations".

---

**Note:** Even if you cannot define specific performance requirements, it is valuable to conduct performance testing in order to get benchmark numbers that you can compare with future releases of the solution. You can also compare this information to the product benchmark numbers that are detailed in the PSR for the release.

---

Selecting a Performance Test Environment

The hardware sizing for the performance test environment must be comparable to that of the production environment. If the test environment is, for example, less than half of the capacity of the production environment, you are not adequately testing the performance capability of the solution.

When you are planning the performance test environment, use the following high-level list to include in the test plan:

- Versions of the following software: OSM, WebLogic Server, JDK, Oracle Database, Design Studio, Design Studio plug-ins, operating system.
- Solution and deployment architecture.
- Latest cartridges for performance testing.
- Configuration of WebLogic Server (all files in WLS_domain/config folder), JVM heap, Coherence configuration XML files, OSM oms-config.xml file.
Guidelines for a Performance Test Environment

- OSM database configuration: memory size, tablespaces and redo log sizes, layout, and so on. Also, for example, whether you plan to use partitioning and, if so, the number of orders per partition.

Establishing Warm-up Criteria

Establishing warm-up criteria is an important step in performance test planning. Before each OSM performance test, you must start and run the WebLogic server and the database in order to allow all Java classes to compile and the cache of the database to become populated with data. Typically, running the system for 5 to 10 minutes at 25 percent of its maximum speed is sufficient, at which point WebLogic server CPU usage and database input/output have stabilized.

Allowing the system to warm up ensures that these resources do not need to be loaded or initialized during a test run. Without warming up the system, any statistics that are gathered during the warm-up period can skew the results.

Synchronizing Time Across Servers

It is important that you synchronize the date and time across all machines that are part of the testing effort, including client test drivers. Do this using either Network Time Protocol (NTP) or manual synchronization. Synchronization is important in capturing accurate run-time statistics.

Note: The database must be installed on a server that runs in a time zone that does not use daylight-savings time. WebLogic servers can run in any time zone, as long as the oms-config.xml file parameter database_timezone_offset is adjusted accordingly. For more information about this parameter, see "Configuring OSM with oms-config.xml".

Determining Database Size

The size of the database has an impact on performance, so Oracle recommends that you run tests using a database that is the size that is planned for the production environment.

Note: Oracle recommends that you back up the OSM schema before running performance testing. After the test, you can restore the schema so that you do not need to purge orders. Keep in mind that exporting and importing the OSM schema can be time-consuming.

Setting Up a Test Client for Load Generation

You can use a test client to provide the orders for performance testing. The test client can be a custom application or any third-party tool, such as JMeter or LoadUI.

Keep the following in mind when setting up a test client:

- Ensure the test client does not impact OSM hardware resources. It is best to run test clients on different hardware from the hardware where OSM is deployed.
Ensure the number of test client threads is configurable and supported in the test client machine. This is essential for load and scalability testing because the number of concurrent users that the system can support is based on the number of test client threads. If high loading is required, you might need to use multiple test client machines to generate sufficiently high load.

Exclude start-up time and warm-up time from time measurements. For more information, see "Establishing Warm-up Criteria".

Ensure the test client can provide vital statistics on performance data, such as average, maximum, standard deviation, and 90th percentile performance numbers.

Ensure the test client can complete long running tests with a steady load.

For third-party tools like JMeter, there are plug-ins available to add monitoring capabilities for server and database calls. Access the plug-ins using the following link:

http://jmeter-plugins.org/

Setting Up Emulators

If the entire system is not ready for testing, you can set up emulators to simulate external requests and responses. For example, if you need to do OSM performance testing before the billing or inventory system is ready, you can use emulators.

If you are using the Order-to-Activate cartridges, OSM provides an Oracle Application Integration Architecture (AIA) Emulator, which you can use to emulate an order. Run emulators on separate hardware from the OSM server so they do not consume OSM resources during performance testing. For more information about setting up and using Oracle AIA emulators, see OSM Cartridge Guide for Oracle Application Integration Architecture.

Setting Up Monitoring Tools, Scripts, and Log Levels

To capture statistics while the application is running, you must set up certain tools and scripts. For information about the tools that you can use to monitor performance in OSM, see "Monitoring and Analyzing Performance".

Set logs to a level that captures the monitoring information that you need. Perform heap and thread dumps at the following intervals:

- Before the test run.
- During the middle or peak time of the run.
- After the test is done.

---

**Note:** Capturing monitoring dumps has a moderate impact on performance. For more information about the procedures and tools for monitoring, see "Monitoring and Analyzing Performance".

---

Running Tests and Analyzing Test Performance

In order to analyze the results of the test runs, you must make sure to do the following during testing:

- Gather operating system information for OSM application machines.
Gather Automatic Workload Repository (AWR) and AWR for Oracle RAC, ADDM, and (Active Session History) ASH reports from the database for the exact duration of the test.

Monitor information for WebLogic Server CPU, heap, and threads using tools like VisualVM or JConsole.

Gather garbage collection logs and server logs.

Gather multiple thread dumps regularly, especially during issues.

Gather heap dumps, if necessary.

Monitor WebLogic server activities, such as JMS queues, JDBC connections, execute thread pool, and so on, using WLST.

Monitor network and storage for less latency and consistent throughput based on the documented service times for the hardware.

For more information about monitoring and gathering information during performance testing, see "Monitoring and Analyzing Performance".

**Improving Performance**

Performance testing and tuning is an iterative process with cycles of testing, tuning, analyzing, and retesting. OSM includes a number of components and technologies in its architecture, including:

- Operating systems
- Oracle WebLogic Server
  - Java Message Service
  - Oracle Coherence
- Oracle Database
- Oracle Communications Design Studio

Achieving optimal OSM performance depends on proper configuration and tuning of OSM components and technologies.

**Storage and System Considerations**

Keep the following in mind when considering the storage requirements for your system:

- Ensure you have a minimum of 20 GB of available disk space for installing and deploying all OSM required packages, creating a domain, and deploying the OSM application.

- Ensure the OSM domain that you plan to use for setting up a cluster environment is hosted on high-performance shared storage.

- For the database, plan to have at least 500 GB of free disk space for the OSM schema. This requirement might be higher, depending on the design and the order volume, and you must plan for this during hardware sizing.

- Use RAID 1+0 (normal redundancy) backed shared storage for installing Fusion Middleware packages, creating the OSM domain, and deploying applications and server log files.
Ensure you have reliable and high performance storage of 5 GB for JMS persistent file stores. This requirement might be higher, depending on the design and the order volume.

The usual latency requirement from storage is service time of less than 5 ms. You must decide the IOPS (input/output per second) requirement during hardware sizing.

The software technologies that run with OSM include JDK, Fusion Middleware applications, WebLogic servers, and Oracle Database servers. The best practices for installing these technologies on your system include the following:

- Ensure that the 64-bit version of the JDK is installed and available. Ensure that a 64-bit JDK is also used to run the installers. Oracle does not recommend using a 32-bit JDK for any sized production system or for performance testing because it does not support adequate heap sizes for performance and scalability demands.

- Plan to use 64-bit versions of Fusion Middleware components. When installing and creating the Fusion Middleware home directory, use 64-bit JDK to run the installers. Doing so ensures that Fusion Middleware installations have 64-bit support, including native libraries support (these are also called WebLogic Server Native IO Performance Packs).

- Ensure JDK and Fusion Middleware installations are updated with all recommended patches for the OSM release.

**Tuning Application System Components**

You can run multiple WebLogic servers in a cluster on the same system so as to maximize the use of available resources while keeping the heap size of the associated JVM instances to a reasonable level. Ensure that you limit the number of JVM instances to the number of available processors.

**Server and Operating System**

Incorrectly configuring the operating system, or configuring it to a suboptimal level, has a significant impact on OSM performance.

System and user limit recommendations include the following:

- **Core file size**: Limit core file size to zero. If there is a core dump or the JVM crashes, very large memory and data heaps might be written to the disk. Oracle recommends setting a positive value for core file size only if a crash occurs and must be debugged on the next occurrence.

- **Number of open files**: OSM typically references and loads large numbers of internal and third-party JAR archives. Also, each of the applications opens and maintains several configuration files, log files, and numerous network socket and JDBC connections. All these activities use a large number of open files, both during start up as well as potential (re)deployments of those applications. Oracle recommends increasing the number of open file limits for OSM.

- **Number of user processes**: For the same reasons as number of open files, Oracle recommends increasing the limit for user processes.

- **Socket buffers**: To help minimize packet loss for the Coherence cluster, the socket buffers of the operating system must be large enough to handle the traffic. Oracle recommends at least 2 MB. See external link at section 4.3.8 for details. For more information, see “Tuning Coherence”.

---

**Improving Performance**

13-8 OSM System Administrator's Guide
Table 13–1 includes a summary of the recommendations for configuring the operating system.

**Table 13–1  Configuration Recommendations for the Operating System**

<table>
<thead>
<tr>
<th>Configuration Item</th>
<th>Solaris</th>
<th>Linux</th>
</tr>
</thead>
<tbody>
<tr>
<td>ULIMIT PARAMETERS</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>core file size (soft)</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>core file size (hard)</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>open files (soft)</td>
<td>8192</td>
<td>65536</td>
</tr>
<tr>
<td>open files (hard)</td>
<td>65536</td>
<td>65536</td>
</tr>
<tr>
<td>max_user_processes (soft)</td>
<td>29995</td>
<td>774889</td>
</tr>
<tr>
<td>max_user_processes (hard)</td>
<td>29995</td>
<td>774889</td>
</tr>
</tbody>
</table>

**Note:** With engineered systems, such as Sparc SuperCluster or Exalogic, most of the operating system tuning is done by default.

**WebLogic Domains**

Change WebLogic domain values using WebLogic Server Administration Console. For more information, see "Accessing the WebLogic Server Administration Console".

Recommendations for creating and configuring WebLogic domains for OSM include the following:

- **Production mode**: Create the domain in production mode so that the default optimization settings are applied. Verify that all the servers in a domain are in production mode using the Domain, Configuration tab, General sub-tab in the WebLogic Server Administration Console.

- **Transaction timeouts**: Set timeouts for correct rollback handling, keeping in mind that global transactions span multiple transaction sources, such as JMS and JDBC.

- **Longest transaction time**: Set the longest time a transaction can be active. The global transaction timeout, which is the longest time that a transaction can be active, is determined by the Java Transaction API (JTA) timeout. To avoid premature rollbacks, increase the JTA timeout to a value of 9000 seconds, which avoids most problems. You can determine the optimal level for your system based on performance and stress testing. Change the JTA timeout value using the Domain, Configuration tab, JTA sub-tab in the WebLogic Server Administration Console.

**Note:** If you are using Oracle Exalogic Elastic Cloud, enable optimizations in order to improve thread count management and request processing, and to reduce lock contention.

Table 13–2 includes a summary of the recommended WebLogic domain configurations.
Table 13–2 Configuration Recommendations for the Domain

<table>
<thead>
<tr>
<th>Configuration Item</th>
<th>Value</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Production mode</td>
<td>true</td>
<td>--</td>
</tr>
<tr>
<td>JTA Timeout</td>
<td>9000</td>
<td>Set this value for better deployment tolerance rather than performance. All transactions typically complete within only a few seconds (excluding the first transaction after server startup).</td>
</tr>
<tr>
<td>Enable Exalogic Optimizations</td>
<td>true</td>
<td>When you are using an Exalogic server.</td>
</tr>
<tr>
<td>Log: Rotate log on startup</td>
<td>true</td>
<td>--</td>
</tr>
<tr>
<td>Log: Number of files limited</td>
<td>true</td>
<td>--</td>
</tr>
<tr>
<td>Log: File count</td>
<td>10</td>
<td>--</td>
</tr>
</tbody>
</table>

WebLogic Servers

Change WebLogic server values using WebLogic Server Administration Console. For more information, see “Accessing the WebLogic Server Administration Console”.

Recommendations for creating and configuring managed servers for OSM include the following:

- **64-bit flags**: Ensure to use the `-d64` option for each server so that available native libraries or performance packs are used when you start WebLogic servers.

- **Logging**:
  - Consider limiting the number of log files, the log file maximum size, and rotation policy. Limit the number of log files to 10 and ensure log files are rotated at start up so that a fresh log file is available.
  - Consider turning off logging messages to the domain for each managed server. WebLogic server instances send messages to a number of destinations, including the local server log file and the centralized domain log, which can affect performance. For more information, see the chapter about understanding WebLogic logging services in the document *Oracle Fusion Middleware Configuring Log Files and Filtering Log Messages for Oracle WebLogic Server*.

- **JDBC logging**: Disable JDBC logging in production systems because it has a substantial impact on performance.

- **WebLogic networking**: Enable native input/output (IO). WebLogic uses software modules called muxers (multiplexers) to read incoming requests on the server and incoming responses on the client. To maximize network socket performance, make sure to use native, platform-optimized muxers.

- **Stuck thread maximum time**: WebLogic considers a thread to be stuck when the thread takes more than a specified amount of time to process a single request. The default value of 600 seconds might be too high. Oracle recommends that you set this value to an optimal level based on performance and stress testing.

Oracle recommends that you do not leave the WebLogic Server listen address undefined on a computer that uses multiple IP addresses. In this case, the server binds to all available IP addresses, which slows down server startup time. Bind a WebLogic server to a fully qualified hostname, rather than an IP address. This ensures that SSL server-to-server communication works correctly without requiring hostname
verification. It also allows administrators to change IP addresses without reconfiguring WebLogic.

For more information and answers to questions, such as how to use JDBC caches and how to tune session persistence, see the topic titled Top Tuning Recommendations for WebLogic Server in WebLogic Server documentation.

Table 13–3 includes a summary of the recommended WebLogic server configurations.

**Table 13–3 Configuration Recommendations for the Server**

<table>
<thead>
<tr>
<th>Configuration Item</th>
<th>Value</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Production mode</td>
<td>true</td>
<td>--</td>
</tr>
<tr>
<td>-d64</td>
<td>enabled</td>
<td>Make sure to use 64-bit native libraries in the startup paths.</td>
</tr>
<tr>
<td>Native IO enabled</td>
<td>true</td>
<td>--</td>
</tr>
<tr>
<td>Log: Number of files limited</td>
<td>true</td>
<td>--</td>
</tr>
<tr>
<td>Log: File count</td>
<td>10</td>
<td>--</td>
</tr>
<tr>
<td>All file-based persistent stores use synchronous write policy</td>
<td>Direct-Write-Without-Cache</td>
<td>--</td>
</tr>
<tr>
<td>All JMS Servers &gt; Persistent Stores</td>
<td>enabled</td>
<td>File-based persistence has better performance that a JDBC store, but JDBC offers consistent backup snapshots of OSM data and JMS messages. For more information, see the topic about persistent store, JMS file store, and JDBC store in OSM Installation Guide.</td>
</tr>
</tbody>
</table>

**Configuring a Java HotSpot Virtual Machine**

Configuring a Java HotSpot Virtual Machine (VM) must take into account how the JVM does the following:

- Generates the byte code for the Java code that it executes.
- Performs garbage collection.
- Sizes and manages heap space between old and new objects.

The JVM dynamically allocates memory to Java objects from the heap. The size of the heap affects the frequency and duration of garbage collection and overall system throughput. Tuning the heap size involves finding a balance between garbage collection time and the amount of memory required to store the Java objects created by your application, including objects created by the underlying platform.

Oracle recommends that you specify the same settings for both maximum heap and minimum size settings for various memory categories, to avoid the JVM frequently growing and shrinking the heap. The Java HotSpot VM parameters that you configure are `-Xmssize` and `-Xms`size`. This is the standard recommendation for JVM performance tuning.

In addition, set the following Java HotSpot VM parameters:

- `-server`: Set the server version of the Java HotSpot VM. Even though this setting causes the JVM to perform more slowly at first, performance improves over time.
- `-Xms<hh>g -Xmx<hh>g`: Specifies the same <hh> size area for JVM total heap for objects. This helps performance by avoiding constant resizes of the heap.
-XX:PermSize=<pp>m -XX:MaxPermSize=<pp>m: Specifies same <pp> size area for JVM permanent generation area. Primarily, this area is used for keeping loaded classes. The default value for 64-bit JVMs is 256m, however this is inadequate for managed servers that are hosting an OSM application. Set a higher value (512m) to avoid JVM thread hang ups on managed servers. This setting is optional for administration and proxy servers.

-XX:+UseCompressedOops: An oop, or ordinary object pointer, is a managed pointer to an object. An oop is normally the same size as a native machine pointer, which means 64 bits on an LP64 system. For more information, see the knowledge article about technical performance [Doc ID 1552842.1], available from the Oracle Support Web site:

https://support.oracle.com

-XX:+HeapDumpOnOutOfMemoryError and -XX:HeapDumpPath=path: A heap dump provides a snapshot of what is happening in the heap and is useful in troubleshooting memory issues. For example, your OSM system might stop working because it is out of memory. Specify the -XX:+HeapDumpOnOutOfMemoryError and -XX:HeapDumpPath=path parameters to automatically generate a heap dump in the specified directory when an out of memory condition is reached.

The JVM also manages the dynamic allocation of memory to Java programs. This includes detecting and de-allocating unused memory, which is known as garbage collection (GC). Garbage collection algorithms are either parallel or concurrent. With parallel algorithms, the running of application logic stops at regular intervals. With concurrent algorithms, the JVM performs garbage collection logic concurrently with application logic.

For many OSM deployment scenarios, using parallel algorithms is the best option, given that throughput is often the most important consideration. For a Java HotSpot VM, set the -XX:+UseParallelGC parameter.

Specify and tune number of parallel GC threads when you deploy the solution to multi-core systems. The maximum effective value for this parameter is equal to the number of processor cores on the system. General guidance is that on an n-core system, allocate anywhere from four to 50 percent of n ParallelGC threads. Adjust the value of n downward if there are additional JVM instances running on the same system.

Further recommendations are the following:

- Turn on the -XX:+ParallelRefProcEnabled setting to reduce the reference collection pause time. However, switch to CMS collector, using the -XX:+UseConcMarkSweepGC parameter, if there are long GC pauses causing OSM Coherence cluster to become unstable. This might be the case if your deployment must process large orders.

- Use the -XX:+DisableExplicitGC parameter to disable garbage collection that is triggered by application calls to System.gc(). Otherwise, garbage collection might occur too frequently.

- Determine optimal values by analyzing garbage collection efficiency at different settings. Enable verbose garbage collection during your tuning exercise. Verbose garbage collection generates logs that detail the timing and duration of garbage collection activities. Use the -verbose:gc Java HotSpot VM parameter. The -Xloggc:logfilename parameter redirects the associated logs to a file. Also set the -XX:+PrintGCDetails, -XX:+PrintGCTimeStamps and -XX:+PrintTenuringDistribution parameters during analysis and tuning.
Use the GCViewer tool to view the data produced by verbose garbage collection. For information, see "GCViewer". Pay particular attention to garbage collection duration and frequency. While keeping the number of garbage collections threads and garbage collection frequency to a reasonable level, consider adding threads or reducing heap size, or both, if garbage collection takes more than a few seconds. If you reduce heap size, you can consider deploying additional WebLogic instances on the same server. Using verbose garbage collection is the preferred approach for a production system. A more involved approach is the live profiling of running code. The VisualVM tool can be used when a more detailed analysis is required. For information about GCViewer and the VisualVM tool, see "GCViewer" and "Java VisualVM".

For more information about garbage collection tuning, see the following Web site:
http://www.oracle.com/technetwork/java/javase/gc-tuning-6-140523.html

Table 13–4 includes a summary of the recommendations for configuring the WebLogic Server JVM.

<table>
<thead>
<tr>
<th>Configuration Item</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>-server on</td>
<td></td>
</tr>
<tr>
<td>-Xms&lt;nn&gt;g -Xmx&lt;nn&gt;g</td>
<td>24</td>
</tr>
<tr>
<td>-XX:PermSize&lt;nn&gt;m</td>
<td>512</td>
</tr>
<tr>
<td>-XX:+UseCompressedOops on</td>
<td></td>
</tr>
<tr>
<td>-XX:+HeapDumpOnOutOfMemoryError on</td>
<td></td>
</tr>
<tr>
<td>-XX:HeapDumpPath=path</td>
<td></td>
</tr>
<tr>
<td>-XX:+UseParallelGC on</td>
<td></td>
</tr>
<tr>
<td>-XX:+UseParallelOldGC on</td>
<td></td>
</tr>
<tr>
<td>-XX:+ParallelRefProcEnabled on</td>
<td></td>
</tr>
<tr>
<td>-verbose:gc on</td>
<td></td>
</tr>
</tbody>
</table>

Cluster Configuration Recommendations
Recommendations for OSM cluster include the following:

- **Messaging mode**: Oracle recommends that you use multicast messaging mode when setting up OSM. For more information about using multicast or unicast, see the topic about configuring the WebLogic server domain for an OSM cluster in OSM Installation Guide.

- **Load balancing**: Set up clusters to use a round-robin algorithm.

Table 13–5 includes a summary of the recommendations for OSM cluster:

<table>
<thead>
<tr>
<th>Configuration Item</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cluster Messaging Mode</td>
<td>multicast</td>
</tr>
<tr>
<td>Default Load Algorithm</td>
<td>round-robin</td>
</tr>
</tbody>
</table>

JMS Messaging Recommendations
Recommendations for JMS Messaging include the following:
JMS message persistence needs to use a file store rather than a JDBC store, which is the default in the OSM installer. For optimal performance, Oracle recommends that you use Direct-Write-With-Cache file stores, if this option is supported in your environment. For information about the best practices for configuring a WebLogic file store, see the chapter about using the WebLogic persistent store in Oracle Fusion Middleware Configuring Server Environments for Oracle WebLogic Server.

In a clustered environment, WebLogic uses load balancing to distribute the workload across clusters. For OSM, set the load balancing policy for distributed JMS queues to Random. For more information about WebLogic JMS Random distribution, see the chapter about configuring advanced JMS system resources in Oracle Fusion Middleware Configuring and Managing JMS for Oracle WebLogic Server.

OSM uses messaging bridges with Asynchronous Mode Enabled and Exactly-once quality of service. For this type of messaging bridge, you can improve throughput by increasing the Batch Size attribute, which reduces the number of transaction commits.

WebLogic supports the Store and Forward (SAF) service for reliable delivery of messages between distributed applications running on different WebLogic Server instances. It is recommended that you set the Conversation Idle Time Maximum on SAF agents to a positive value to allow messages to be forwarded to other active members when the original target is down or unavailable. For more information about the WebLogic SAF service, see the chapter about understanding the SAF service in Oracle Fusion Middleware Configuring and Managing Store-and-Forward for Oracle WebLogic Server.

**JDBC Recommendations**

This section includes recommendations for JDBC.

For engineered systems, Oracle recommends that you use SDP protocol over Infiniband (IB). This protocol enables multiple performance enhancements, such as input/output, thread management, and request handling efficiency. Typical steps to enable SDP protocol include:

- Ensure the physical Infiniband connection exists and is operational between the WebLogic server host and the database host.
- Set up an SDP listener on the Infiniband network.
- In the JDBC URL, replace TCP protocol with SDP protocol.
- If necessary, change the port number to match the SDP listener's port in the JDBC URL.
- Manually add the system properties `-Djava.net.preferIPv4Stack=true` and `-Doracle.net.SDP=true` to the `startWebLogic.sh` script (or run `startManagedServer_XX.sh`).
- **Initial Capacity**: Oracle recommends setting this value to Max Capacity for production deployments to avoid shrinking and growing of the JDBC pool size. Setting the Initial Capacity value impacts available resources on both WebLogic server and database server by consuming additional resources at the start up time, and also lengthens the server start up time.
- **Max Capacity**: Set this to a peak and sustainable value, which can be supported by both by WebLogic server (additional memory and processing) and the database server (session resources and concurrency). In uniform cluster deployments of JDBC connection pools to a cluster, the Max Capacity value applies to each WebLogic server node individually, not for the whole cluster. Select the Max
Improving Performance

Capacity such that \((\text{number of nodes in WebLogic server cluster } \times \text{Max Capacity}) \leq \text{number of peak, concurrent and safely sustainable sessions to the database server})\).

**Note:** Max Capacity is an important parameter that requires iterative tuning based on scenario and workload. One approach is to set it to a high value for peak load tests, and monitor what percentage of it has been used, and then adjust the MaxCapacity to at least that high.

- **Statement Cache Size:** The prepared statement cache size is used to keep a cache of compiled SQL statements in memory, thus improving the performance by avoiding repeated processing in the WebLogic server and database. For lightly used DataSources, the default value of 10 is sufficient.

**Note:** Each JDBC connection in the Connection Pool creates its own prepared statement cache. When you tune this parameter, consider the additional memory consumption demand caused by \((\text{steady size of Connection Pool} \times \text{Prepared Statement Cache Size})\). Too high may cause Out of Memory exceptions on WebLogic server and may disable the connection pool altogether, rendering the server useless. Tuning Statement Cache Size is achieved by an iterative process, influenced by factors of the scenario, workload, and steady state size of the connection pool for the given data source.

**Tuning Coherence**
For better reliability, especially when OSM nodes reside on different systems, nodes in OSM cluster must be set up with Coherence Well Known Address (WKA) (a point-to-point unicast) configuration. Each OSM node then has a unique Coherence WKA configuration, in which addresses of all other nodes must also be listed. For more information about Coherence, see **OSM Installation Guide**.

For information about tuning Coherence, see the performance tuning chapter in **Oracle Coherence Developer’s Guide**.

**Using Work Managers**
OSM uses work managers to prioritize work. Properly tuned work managers are an effective way to defend against system overload. By limiting the threads configured in OSM work managers, you ensure that your system does not accept more load than you have tested for.

For information about configuring work managers in WebLogic for OSM, see "Using Work Managers to Prioritize Work".

**Tuning Oracle Database**
For every Oracle Database installation, there are a number of init.ora (or spfile.ora) parameters that affect performance. The following table lists important init parameters for use with OSM:

<table>
<thead>
<tr>
<th>Parameter Name</th>
<th>Sparc SuperCluster (SSC)</th>
<th>Exadata</th>
</tr>
</thead>
<tbody>
<tr>
<td>commit_wait</td>
<td>transaction commit</td>
<td>transaction commit</td>
</tr>
</tbody>
</table>
For information about collecting more accurate database statistics, see the knowledge article about best practices for managing optimizer statistics [Doc ID 1662447.1], available from the Oracle Support Web site:

https://support.oracle.com

Configuring Database Schema Partitioning
For detailed information about partitioning the database schema, see "Managing the OSM Database Schema".

Multi-database Source Configuration Using N-Oracle RAC Nodes
Oracle recommends that you use an Active-Active Oracle RAC database configuration to provide scalability and high availability for the database. For more information about high availability and its impact on the database Oracle RAC, see Appendix B of OSM Installation Guide.

Note: OSM installer sets up only two active Oracle RAC nodes, by default. For information about adding more nodes, see the post-installation chapter of OSM Installation Guide.

Database Storage
Recommendations for setting up database storage include the following:

Table 13–6 (Cont.) Recommendations for init Parameters for Oracle Database

<table>
<thead>
<tr>
<th>Parameter Name</th>
<th>Sparc SuperCluster (SSC)</th>
<th>Exadata</th>
</tr>
</thead>
<tbody>
<tr>
<td>db_block_size</td>
<td>8192</td>
<td>8192</td>
</tr>
<tr>
<td>db_writer_processes</td>
<td>8</td>
<td>3</td>
</tr>
<tr>
<td>filesystemio_options</td>
<td>'setall'</td>
<td>'setall'</td>
</tr>
<tr>
<td>memory_max_target</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>memory_target</td>
<td>134217728000</td>
<td>0</td>
</tr>
<tr>
<td>O7_DICTIONARY_ACCESSIBILITY</td>
<td>TRUE</td>
<td>TRUE</td>
</tr>
<tr>
<td>open_cursors</td>
<td>15000</td>
<td>1000</td>
</tr>
<tr>
<td>pga_aggregate_target</td>
<td>0</td>
<td>8589934592</td>
</tr>
<tr>
<td>processes</td>
<td>5000</td>
<td>5000</td>
</tr>
<tr>
<td>sessions</td>
<td>7680</td>
<td>7536</td>
</tr>
<tr>
<td>sga_max_size</td>
<td>77846282240</td>
<td>34359738368</td>
</tr>
<tr>
<td>sga_target</td>
<td>77846282240</td>
<td>34359738368</td>
</tr>
<tr>
<td>undo_retention</td>
<td>1800</td>
<td>1800</td>
</tr>
<tr>
<td>deferred_segment_creation</td>
<td>true</td>
<td>true</td>
</tr>
</tbody>
</table>

Note: Oracle recommends that you use the default value for commit_wait and ensure that values for sga_max_size and sga_target for SSC are equal. This is to avoid major issues with the Solaris kernel.
Use Automatic Storage Management (ASM) for managing data disk storage.

For storage reliability, use Normal (two-way mirrored) Redundancy and ensure that the tablespace, data files, and redo logs are on this storage.

Place redo logs, which are sensitive to storage response time, should be put on storage with a service time of less than 5 ms.

Specify large redo log files to avoid frequent redo log switching. For redundancy, use a mirrored configuration for redo logs.

Finalize the requirements for latency and IOPS during your hardware sizing exercise.

Recommendations for configuring tablespaces include the following:

- Use Automatic Segment Space Management (ASSM) for each tablespace.
- Whenever disk space permits, use BIGFILE for tablespace creation. This simplifies the management of tablespace allocation by using a single data file for tablespace.
- If you must use a SMALLFILE tablespace, plan for the possibility that a large number of data files might be created for OSM schema. Plan to implement a naming convention for data files and find an ideal location for data files that allows for future growth.

Preventing Timeout Issues During Cartridge Deployment

To avoid timeout issues during cartridge deployment, in the `oms-config.xml` file, set the `CartridgeDeploymentTransactionTimeout` parameter to 2400. You can also specify this value in the shell script used to start WebLogic using the `-Dcom.mslv.oms.cartridgemgmt.DeployCartridgeMDB.CartridgeDeploymentTransactionTimeout=2400` parameter.

Using the Compliance Tool

The OSM compliance tool captures configuration snapshots and evaluates the compliance of captured configuration snapshots against a set of rules, which are based on product usage, and best practices and guidelines.

For information about evaluating system compliance using this tool, see "Evaluating System Configuration Compliance".

Analyzing Performance Data

For information about monitoring performance and analyzing the results of the data that you gather, see "Monitoring and Analyzing Performance".

Maintenance and Prevention

Some activities require that you take the system offline, so you must plan for these; you can do other activities while the system is running.

In order to ensure that your system maintains a good level of performance, consider the following:

- Validate that there is always adequate database storage in advance of system growth. Notify your database administrator if you see a database space shortage.

- Make sure there is enough file space storage for generated logs. Do so by cleaning and archiving unused logs and output files. Devise an archiving strategy for how...
long you want to store new logs (number of days) and when to purge older logs, keeping in mind that older logs might be critical when troubleshooting and resolving issues.

- Keep in mind that WebLogic servers use persistent file stores, which are physical files that can grow quite large. Plan to regularly shrink the file stores (requires server down time).

- Keep old order data only for as long as you need to meet your business and legal requirements. Keeping order data past when it is no longer required can take up large amounts of storage and cause performance issues. For information about managing and partitioning your database, see "Managing the OSM Database Schema".

- Do not keep multiple versions of a cartridge when they are not required. Undeploy old cartridges and their related completed orders.

- Inspect JMS error queues, which can become very large, in order to find repeated error patterns. For more information, see "Using JMS Queues to Send Messages".

- Size OSM partitions properly in order to reduce the frequency of maintenance for creating partitions, and dropping or purging older partitions. For information about managing and partitioning your database, see "Managing the OSM Database Schema".
This chapter provides guidelines to help you troubleshoot problems with your Oracle Communications Order and Service Management (OSM) system.

**Diagnosing and Resolving Problems**

This section presents a general checklist that you can use for resolving problems, lists some common problems that you might encounter while using OSM, and tells you how to get help to resolve problems.

When problems do occur, it is important to diagnose and resolve them as early as possible. When you are diagnosing and resolving problems, you must be able to obtain the following information:

- Database AWR report for a particular period of time.
- Database ASH report for a particular period of time.
- WebLogic administration server logs and output files.
- WebLogic managed server logs and output files.
- WebLogic node manager’s logs and output files (if configured).
- JVM garbage collector logs (if collected).
- JVM heap dumps, which provide warnings about the size of the files.
- JVM thread dumps (several in succession).
- OSM model and a single order extracted from the database schema. For more information, see “Exporting and Importing the OSM Model and a Single Order”.

**General Checklist for Resolving Problems**

If you have a problem with your OSM system, go through the following checklist before you contact Oracle Technical Support:

- What exactly is the problem? Can you isolate it? For example, if an order causes a problem on one computer, does it give the same result on another computer?
  Oracle Technical Support needs a clear and concise description of the problem, including when it began to occur.

- What do the log files say?
  This is the first thing that Oracle Technical Support asks for. Check the error log for the OSM component you are having problems with.

- Have you read the documentation?
Look through the list of common problems and their solutions in "Diagnosing Some Common Problems with OSM".

- Has anything changed in the system? Did you install any new hardware or new software? Did the network change in any way? Does the problem resemble another one you had previously? Has your system usage recently jumped significantly?

- Is the system otherwise operating normally? Has response time or the level of system resources changed? Are users complaining about additional or different problems?

Diagnosing Some Common Problems with OSM

This section describes common problems and their solutions.

**Cannot Log in or Access Certain Functionality**

If you cannot log in or access certain functionality, check the following possible causes:

- Are you a valid user in the Oracle WebLogic Server security realm?
- Is the OSM Web application deployed?
- Are all OSM Enterprise Java Beans (EJB) deployed?
- Are the OSM database resources deployed?
- Do you belong to the correct groups in the WebLogic Server security realm?
- Do you belong to any OSM workgroup?

**System Appears Slow**

If the functionality of OSM appears to be present, but performance is slow, check the following possible causes:

- The amount of memory being used (check the max memory configuration in the WebLogic server startup script on the workstation where you have deployed OSM)
- The CPU and disk usage on the machine hosting the OSM database
- The database connections
- For slow worklist access, check the number of flexible headers on your worklist. The number of flexible headers has a direct negative effect on worklist performance.

**Avoiding Timeout Issues in Design Studio**

To avoid timeout issues from Oracle Communications Design Studio during cartridge deployment, set the following parameter in `Domain_Home/bin/startWebLogic.sh` after the line that reads `umask 037`:

```
export JAVA_OPTIONS="$JAVA_OPTIONS -Dcom.mslv.oms.cartridgemgmt.DeployCartridgeMDB.CartridgeDeploymentTransaction.Timeout=2400"
```

1. Log in to the WebLogic Administration Console.
2. In the Domain Configurations section, click Domain.

   The settings for your domain are displayed on a tabbed page.
3. Click the JTA tab and set the value of the **Timeout Seconds** parameter to 9000.
4. Click the Security tab and select **Anonymous Admin Lookup Enabled**.
5. Click **Save**.
6. Restart servers.

---

**Note:** If an order has stopped processing due to a timeout, the order must be aborted. It will not resume processing even after the timeout value has been increased.

---

**Error: "Login failed. Please try again."**

If the error "Login failed. Please try again" is displayed when trying to log in through the Web client and you have entered the correct user name and password, you probably do not belong to the correct groups in the WebLogic Server security realm.

**Solution**

Log in to the WebLogic Administration Console using the administrator account. Make sure you have been added to the group **OMS_client**. Try to log in again.

**No Users or Groups Are Displayed**

After OSM installation, you do not see any users or groups on the Users and Groups tab. This is because non-dynamic changes have been made, and the Admin Server (and managed server, if applicable) requires a restart.

**Solution**

1. Restart the Admin/Managed Server to clear the condition.
   
   If the condition does not clear, proceed with the steps below.
2. Log in to the WebLogic Administration Console and select **Domain**.
4. Select Advanced. If necessary, scroll down the page to find Advanced.
5. Select the **Allow Security Management Operations if Non-dynamic Changes have been Made** check box.
6. Click **Save**.
7. Navigate to the Users and Groups tab.
   
   Your users and groups appear.

**Automation Plug-ins Are Not Getting Called**

If the custom automation plug-ins are not getting called, check the following possible causes:

- Is the Automation configuration deployed properly?
- Are the JMS resources deployed?
- Are the JMS destinations, queues, and topics configured properly?
Too Many Open Files
If you have a large number of external clients connected to OSM and receive the error: "java.net.SocketException: Too many open files", do one of the following:

■ From the WebLogic Administration Console, select Servers, then Server, then Protocols, and then HTTP. Reduce the value in Duration from the default 30 seconds to 15 or even 5 seconds. This will allow the WebLogic server to close idle HTTP connections and release more sockets.

Proxy Fails on a Clustered System
If a proxy fails on a clustered OSM system, all HTTP requests that would normally go through the proxy can no longer get to the OSM server. The problem could be with the physical host the server is running on, or it could be a problem with a standalone managing server that is not part of the cluster but is part of the domain.

To recover, restart the proxy.

Orders Are Not Being Created on a Clustered System
If messages are being successfully added to the JMS queue but the corresponding orders are not being created in OSM, first ensure that the servers are running. If they are running, check to see whether the address has been set for your cluster. The Cluster Address field is located in the General tab of the settings for your cluster. If the cluster address is not set, or does not contain the correct values for your managed servers, OSM will not pick up orders from the JMS queue. Generally, this value is set when the domain is created, but it can be changed or removed manually, which can cause this problem to occur.

For more information about the correct value for a cluster address, see the discussion about configuring the WebLogic Server Domain in the OSM Installation Guide chapter on installing OSM in a clustered environment.

Problems Displaying Gantt Charts on Solaris 5.10 Hosted Systems
When running on Solaris 5.10 only, to use X server to display Gantt charts in the Task Web client, you must configure the Java settings for the Oracle WebLogic Server to avoid display problems and system instability and performance problems.

See the discussion about enabling graphical displays in the post-installation section of the OSM Installation Guide.

OSM Fails to Process Orders Because of Metadata Errors
Metadata errors can occur in any cartridge with orchestration model entities and can cause order processing failures. Search for the string Metadata Errors in the Console view of the Cartridge Management editor in Design Studio. If you are not using Design Studio to deploy cartridges, look in the WebLogic Server logs for the same string.

For more information, see the discussion of metadata errors in OSM Developer’s Guide.

Error: "Not Backend Servers Available"
If the error "No Backend Servers Available" is displayed, you are likely disconnected from your servers. Ensure your servers are connected and functional before continuing with OSM operations.
Quick Fix Button Active During Order Template Conflicts in Design Studio
Conflicts can occur when order templates are created in Design Studio. Presently, Quick Fix does not work for order template conflicts, even if the Quick Fix button is active. All order template conflicts must be resolved manually.

Cannot Create New Orders on a New Cartridge Version
Order creation can fail on a new version of an existing cartridge, even after you have updated all required entities, and built and deployed the cartridge.

When the createOrder request fails, you receive a response like the following example:

```xml
<env:Envelope xmlns:env="http://schemas/soap/envelope/">
  <env:Header/>
  <env:Body>
    <env:Fault xmlns:ord="http://URL/communications/ordermanagement">
      <faultcode>ord:Fault</faultcode>
      <faultstring>Failed to create and start the order due to java.lang.RuntimeException: OMSException: encountered error starting orchestration caused by:Cannot find task for notification id</faultstring>
      <faultactor>unknown</faultactor>
      <detail>
        <InvalidOrderSpecificationFault xmlns="http://URL/communications/ordermanagement">
          <Description>Failed to create and start the order due to java.lang.RuntimeException: OMSException: encountered error starting orchestration caused by:Cannot find task for notification id</Description>
        </InvalidOrderSpecificationFault>
      </detail>
    </env:Fault>
  </env:Body>
</env:Envelope>
```

Solution
1. Open the solution cartridge.
2. Click the Dependency tab of the model project.
3. Remove all the dependencies that are displayed for the project.
4. Re-add all the dependencies.
5. Restart Design Studio.

Getting Help with OSM Problems
If you cannot resolve your problems with OSM, contact Oracle Technical Support.

Before You Contact Support
Problems can often be fixed by shutting down OSM and restarting the computer that OSM runs on.

If that does not solve the problem, the first troubleshooting step is to look at the error log for the application or process that reported the problem. Consult "General Checklist for Resolving Problems" before reporting the problem to Oracle.
Reporting Problems
If "General Checklist for Resolving Problems" does not help you to resolve the problem, write down the pertinent information:

- A clear and concise description of the problem, including when it began to occur.
- Relevant portions of the log files.
- Relevant configuration files, such as oms-config.xml.
- Recent changes in your system, even if you do not think they are relevant.
- List of all the OSM components and patches installed on your system.

When you are ready, report the problem to Oracle.
This chapter details the Oracle Communications Order and Service Management (OSM) log messages. The sections included in this chapter are:

- OSM Catalog Messages
- Automation Catalog Messages

**OSM Catalog Messages**

Table 15–1 shows OSM Catalog messages.
## Table 15–1  OSM Catalog Messages

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Severity</th>
<th>Description</th>
</tr>
</thead>
</table>
| 600000     | error    | **Message Body**  
Order type/source is not found. The order type={1} /source={0} either does not exist, or is not available to the user.  
**Message Details**  
The order type={1}/ source={0} either does not exist, or is not available to the user.  
**Method**  
logSourceTypeNotFound(String source, String type) |
| 600001     | error    | **Message Body**  
Order is not found. The order with orderid={0} and orderHistId={1} does not exist, or is not available to the user.  
**Message Details**  
The order with orderid={0} and orderHistId={1} does not exist, or is not available to the user.  
**Cause**  
The orderHistId might not be up to date.  
**Action**  
Refresh server.  
**Method**  
logOrderNotFound(String orderId, String orderHistId) |
| 600002     | error    | **Message Body**  
Order template is not found. The order template does not exist, or is not available to the user.  
**Message Details**  
The order template does not exist, or is not available to the user.  
**Method**  
logOrderTemplateNotFound() |
| 600003     | error    | **Message Body**  
Remark is not found. The given remark (orderid={0}, histid={1} remarkid={2}) does not match a remark in OMS.  
**Message Details**  
The given remark (orderid={0}, histid={1} remarkid={2}) does not match a remark in OMS.  
**Cause**  
Remark might have been deleted from the server or cannot be found in a specified location.  
**Action**  
Contact your local administrator.  
**Method**  
logRemarkNotFound(String orderId, String orderHistId, String remarkId) |
# Table 15–1 (Cont.) OSM Catalog Messages

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Severity</th>
<th>Description</th>
</tr>
</thead>
</table>
| 600004     | error    | **Message Body**  
Header for mnemonic path is not found. The header for mnemonic path={0} does not exist, or is not available to the user.  
**Message Details**  
The header for mnemonic path={0} does not exist, or is not available to the user.  
**Method**  
logHeaderNotFound(String mnemonicPath) |
| 600005     | error    | **Message Body**  
The format of the order data is not correct. The message details the error location. Invalid data = [0]  
**Message Details**  
Invalid data = [0]  
**Method**  
logOrderDataInvalid(String orderdata) |
| 600006     | error    | **Message Body**  
An attempt to update an order was made without first retrieving the order with an Accept parameter of true. Order (orderid={0} histid={1}) has not been accepted by user={2}.  
**Message Details**  
Order (orderid={0} histid={1}) has not been accepted by user={2}  
**Cause**  
An attempt to update an order was made without accepting the order.  
**Action**  
You should accept the order first, then update the order.  
**Method**  
logOrderNotAcceptedByUser(String orderId, String orderHistId, String userid) |
| 600007     | error    | **Message Body**  
Order update failed. The order (orderid={0} histid={1}) could not be updated due to a data format error. The message details the reason for failure. Data={2}  
**Message Details**  
The order (orderid={0} histid={1}) could not be updated due to a data format error. The message details the reason for failure. Data={2}  
**Cause**  
Data format error.  
**Action**  
Make sure all your data are in correct format and comply with their masks.  
**Method**  
logOrderUpdateFailed(String orderId, String histid, String data) |
### Table 15–1 (Cont.) OSM Catalog Messages

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Severity</th>
<th>Description</th>
</tr>
</thead>
</table>
| 600008     | error    | **Message Body**  
Mandatory check failed. A mandatory field was not given a value when attempting to create, assign, complete, or suspend an order. [2] number of data was missing for order with order id={0}, order history id={1}. The first missing/extra node is node id={3} and order type={4}.  
**Message Details**  
Mandatory check failed. A mandatory field was not given a value when attempting to create, assign, complete, or suspend an order. [2] number of data was missing for order with order id={0}, order history id={1}. The first missing/extra node is node id={3} and order type={4}  
**Cause**  
Not all mandatory fields are filled.  
**Action**  
Fill in data for all mandatory fields.  
**Method**  
logMandatoryCheckFailed(String orderID, String orderHistID, String num, String firstNodeID, String firstNodeType) |
| 600009     | error    | **Message Body**  
Transition is invalid. The order (orderid={0} histid={1}) cannot be transitioned to state={2}. Use ListStatesNStatuses.Request to get a list of valid states.  
**Message Details**  
The order (orderid={0} histid={1}) cannot be transitioned to state={2}. Use ListStatesNStatuses.Request to get a list of valid states.  
**Cause**  
Cannot transition to the selected state.  
**Action**  
Use the ListStatesNStatuses XML API request to get a list of valid states. See OSM Developer’s Guide for information.  
**Method**  
logTransitionInvalid(String orderid, String histid, String state) |
| 600010     | error    | **Message Body**  
Unable to accept order. When retrieving an order for update, the order (orderid={0} histid={1}) cannot be accepted by the current user={2}  
**Message Details**  
When retrieving an order for update, the order (orderid={0} histid={1}) cannot be accepted by the current user={2}.  
**Cause**  
The orderid and histid are not up to date, or the order has been accepted by another user.  
**Action**  
Refresh the server to get new orderid and histid. If the order is currently accepted by another user, you cannot perform Accept operation.  
**Method**  
logUnableToAccept(long orderid, long histid, String userid) |
### Table 15–1 (Cont.) OSM Catalog Messages

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Severity</th>
<th>Description</th>
</tr>
</thead>
</table>
| 600011     | error    | **Message Body**  
User is not found. The order orderid=0 histid=1 cannot be assigned to userid=2.  
**Message Details**  
The order orderid=0 histid=1 cannot be assigned to userid=2.  
**Cause**  
User is not found.  
**Action**  
Try to assign the order to another user.  
**Method**  
logUserNotFound(String orderid, String histid, String userid) |
| 600012     | error    | **Message Body**  
Invalid state mnemonic. The order (orderid=0 histid=1) cannot be suspended with the given state (state=2) mnemonic. Note: Only user-defined states are valid. If you want to complete or assign an order, you must use the appropriate request.  
**Message Details**  
The order (orderid=0 histid=1) cannot be suspended with the given state (state=2) mnemonic. Note: Only user-defined states are valid. If you want to complete or assign an order, you must use the appropriate request.  
**Cause**  
You are calling suspendOrder with an invalid user defined state.  
**Action**  
Supply a valid user defined state, or try to use other appropriate requests.  
**Method**  
logInvalidStateMnemonic(String orderid, String histid, String state) |
| 600013     | error    | **Message Body**  
Invalid status mnemonic. The order (orderid=0 histid=1) cannot be completed with the given status mnemonic (status=2).  
**Message Details**  
The order (orderid=0 histid=1) cannot be completed with the given status mnemonic (status=2).  
**Cause**  
The status might not be valid for the current task.  
**Action**  
Supply a valid status.  
**Method**  
logInvalidStatusMnemonic(String orderid, String histid, String status) |
<table>
<thead>
<tr>
<th>Error Code</th>
<th>Severity</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>600014</td>
<td>error</td>
<td>Remark cannot be modified. The time interval in which a created remark can be modified has elapsed. The remark can no longer be modified. (orderid={0} histid={1} remarkid={2} userid={3})</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Message Details</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Remark cannot be modified. The time interval in which a created remark can be modified has elapsed. The remark can no longer be modified. (orderid={0} histid={1} remarkid={2} userid={3})</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Method</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>logRemarkCannotBeModified(String orderid, String histid, String remarkid, String userid)</td>
</tr>
<tr>
<td>600015</td>
<td>error</td>
<td>Request Unknown. The request type could not be identified. Type given={0}</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Message Details</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>The request type could not be identified. Type given={0}</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Method</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>logRequestUnknown(String type)</td>
</tr>
<tr>
<td>600016</td>
<td>error</td>
<td>Request parameter error. A parameter for the request is missing or invalid. The message details the parameter in question Parameter = {0}, request type = {1}.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Message Details</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>A parameter for the request is missing or invalid. The message details the parameter in question Parameter = {0}, request type = {1}.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Method</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>logRequestParameterError(String parameter, String requestType)</td>
</tr>
<tr>
<td>600017</td>
<td>error</td>
<td>Not authorized. The user={0} is not authorized to make the request={1}.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Message Details</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>The user={0} is not authorized to make the request={1}.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Cause</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>The user is not authorized to perform the operation.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Method</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>logNotAuthorized(String userid, String request)</td>
</tr>
<tr>
<td>600018</td>
<td>error</td>
<td>Database connection failed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Method</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>logDatabaseConnectionFailed()</td>
</tr>
<tr>
<td>600019</td>
<td>error</td>
<td>Security violation by user = {0}</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Cause</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>User is not authorized to perform this operation.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Method</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>logSecurityViolation(String userid)</td>
</tr>
</tbody>
</table>
### Table 15–1 (Cont.) OSM Catalog Messages

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Severity</th>
<th>Description</th>
</tr>
</thead>
</table>
| 600020     | error    | **Message Body**  
Naming exception was thrown while looking up JNDI name={0}.  
**Cause**  
JNDI name might not exist.  
**Action**  
Contact your local administrator.  
**Method**  
logNamingException(String name) |
| 600021     | error    | **Message Body**  
Remote exception thrown, while access object = {0}  
**Message Details**  
Remote exception was thrown while working with an EJB.  
**Method**  
logRemoteException(String name) |
| 600022     | error    | **Message Body**  
EJB Create exception thrown while creating object={0}.  
**Message Details**  
EJB Create exception thrown.  
**Method**  
logEJBCreateException(String name) |
| 600023     | error    | **Message Body**  
Unknown exception thrown, message={0}.  
**Message Details**  
Unknown exception thrown, message={0}.  
**Method**  
logUnknownException(String message) |
| 600024     | error    | **Message Body**  
Cannot deliver JMS message to queue = {0}  
**Message Details**  
Cannot deliver JMS message.  
**Cause**  
JMS Queue might be down.  
**Method**  
logCannotDeliverMessage(String arg) |
### Table 15–1 (Cont.) OSM Catalog Messages

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Severity</th>
<th>Description</th>
</tr>
</thead>
</table>
| 600025     | error    | **Message Body**  
Invalid XML document, doc={0}.  
**Message Details**  
Invalid XML document, doc={0}.  
**Cause**  
XML document has syntax errors.  
**Action**  
Fix the syntax errors.  
**Method**  
logInvalidXMLDocument(String doc) |
| 600028     | debug    | **Message Body**  
SQL: {0}  
**Message Details**  
SQL statement execution  
**Method**  
logSQL(String sql) |
| 600029     | debug    | **Message Body**  
[0]  
**Message Details**  
Log of a request with all its parameters, based on a toString()  
**Method**  
logRequestWithParameters(com.nortel.oms.request.Request request) |
| 600030     | debug    | **Message Body**  
Creating order node: mnemonic path {0}, node id {1}, node type {2}, node index {3},  
parent index {4}, double value {5}, value {6}, parent new? {7}  
**Message Details**  
Node information was sent to the database  
**Cause**  
An order node has been created  
**Action**  
None  
**Method**  
logNodeCreate(String mnemonicPath, long nodeID, String nodeType, long  
nodeIndex, long parentIndex, double doubleValue, String value, String parentNew) |
### Table 15–1 (Cont.) OSM Catalog Messages

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Severity</th>
<th>Description</th>
</tr>
</thead>
</table>
| 600031     | debug    | **Message Body**  
Deleting Order Node: node ID {0}, node type {1}, node index {2}, old double value [3], old text value {4}  
**Message Details**  
Node information was sent to the database  
**Cause**  
An order node has been deleted  
**Action**  
None  
**Method**  
logNodeDelete(long nodeID, String nodeTypeCode, long nodeIndex, double oldDoubleValue, String oldValue) |
| 600032     |          | **Message Body**  
Updating Order Node: mnemonic path {0}, node ID {1}, node type {2}, node index {3}, old double value {4}, old text value {5}, new double value {6}, new text value {7}  
**Message Details**  
Node information was sent to the database  
**Action**  
None  
**Method**  
logNodeUpdate(String mnemonicPath, long nodeID, String nodeTypeCode, long nodeIndex, double oldDouble, String oldText, double newDouble, String newText) |
| 600035     | debug    | **Message Body**  
Data validation failed in order editor. Node (name={0} nodeid={1} nodetype={2} nodeIndex={3} nodeDataType={4} mask={5}) does not comply with mask. Order ID={6} and user={7}  
**Message Details**  
Node (name={0} nodeid={1} nodetype={2} nodeIndex={3} nodeDataType={4} mask={5}) does not comply with mask. Order ID={6} and user={7}  
**Cause**  
Data supplied do not comply with their masks.  
**Action**  
Must supply data with correct format.  
**Method**  
logOrderEditorDataValidationFailed(String name, String nodeId, String nodeType, String nodeIndex, String nodeDataType, String mask, String orderID, String user) |
| 600036     | debug    | **Message Body**  
Order editor - Create a node.  
**Message Details**  
Order editor: create a node (nodeId={0} nodeType={1} parentWebID={2})  
**Method**  
logOrderEditorCreateNode(String nodeId, String nodeType, String parentWebID) |
### Table 15–1  (Cont.) OSM Catalog Messages

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Severity</th>
<th>Description</th>
</tr>
</thead>
</table>
| 600037     | debug    | **Message Body**  
Order editor - delete a node  
**Message Details**  
Order editor - delete a node (webID={0})  
**Method**  
logOrderEditorDeleteNode(String webID) |
| 600038     | debug    | **Message Body**  
Notification Engine - create message.  
**Message Details**  
Creation of event message  
**Method**  
logNotificationEngineCreateMsg(String arg) |
| 600039     | debug    | **Message Body**  
Could remove an event from the DB. Event Id = {0}  
**Method**  
logEventEngineRemoveEvent(long eventID) |
| 600055     | warning  | **Message Body**  
An exception occurred while removing the session for user [0]. Reason: {1}.  
**Message Details**  
While logging out user [0], an exception was thrown when calling EJBOBJECT.remove().  
**Cause**  
The server does not allow the session to be removed, or a communication error occurred.  
**Method**  
logEJBRemoveException(String username, Throwable th) |
| 600063     | error    | **Message Body**  
Error loading screen definitions from [0]. Reason: {1}.  
**Message Details**  
The screen definition file [0] could not be loaded. The screen definition file is required to construct the Web pages.  
**Cause**  
The OMS application has not been deployed or built properly.  
**Action**  
Make sure that the screendefinitions.xml file is in the oms.ear file: oms.ear/oms.war/WEB-INF/conf/  
**Method**  
logCouldNotLoadScreenDefinitions(String url, Throwable th) |
<table>
<thead>
<tr>
<th>Error Code</th>
<th>Severity</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>600066</td>
<td>warning</td>
<td>Poller cannot locate listener {0}. Reason: {1}.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Message Details</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Poller Servlet was deployed before listener {0}.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Cause</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Poller Servlet was deployed before listener {0}.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Action</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>None</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Method</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>logGetPollerListener(String listener, Throwable th)</td>
</tr>
<tr>
<td>600069</td>
<td>error</td>
<td>Could not unsubscribe for {0} event.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Message Details</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Event unsubscription for {0} event has failed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Cause</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>JMS queue might be down.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Action</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Use the WebLogic Server Console to verify JMS deployment.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Method</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>logEventTypeUnsubscribeException(String eventType)</td>
</tr>
<tr>
<td>600071</td>
<td>debug</td>
<td>Filter value {1} does not have the proper format ({3}) for {0}.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Message Details</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>A filter could not be generated using the value {0}, formatter for {1} and mask {3}.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Cause</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>User error</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Action</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Expected. You must retry.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Method</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>logFilterFormatError(String headerName, String filterValue, String mask)</td>
</tr>
<tr>
<td>600072</td>
<td>debug</td>
<td>{3} invalid. Wildcards are not permitted for operation {1} for header {0}.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Message Details</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>{3} invalid. Wildcards are not permitted for operation {1} for header {0}.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Cause</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>User input. Expected error.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Action</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>You must retry.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Method</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>logFilterWildcardError(String headerName, String operationType, String value)</td>
</tr>
</tbody>
</table>
### Table 15–1 (Cont.) OSM Catalog Messages

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Severity</th>
<th>Description</th>
</tr>
</thead>
</table>
| 600073     | error    | Message Body
SSL is not enabled.
Message Details
SSL is not enabled.
Cause
SSL is not enabled.
Action
You must have SSL enabled for your server through WebLogic console.
Method
logSSLDisabled() |
| 600076     | error    | Message Body
Unable to connect to remote file system for accessing attachments.
Message Details
Unable to connect to T3 remote file system for accessing attachments.
Cause
Targeted server might be down.
Action
Contact your local administrator.
Method
logT3ConnectionException() |
| 600077     | error    | Message Body
Unable to add attachment with id={0} in remote file system.
Message Details
Unable to add attachment with id={0} in remote file system using T3 file services.
Cause
IO Exception.
Method
logT3AddAttachmentException(String attachmentID) |
| 600078     | error    | Message Body
Unable to delete attachment with id={0} in remote file system. Attachment with id={0} does not exist.
Message Details
Unable to delete attachment with id={0} in remote file system using T3 file services. Attachment with id={0} does not exist.
Cause
The attachment does not exist.
Method
logT3DeleteAttachmentNotFoundException(String attachmentID) |
### Table 15–1 (Cont.) OSM Catalog Messages

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Severity</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>600079</td>
<td>error</td>
<td>Message Body&lt;br&gt;Unable to read attachment with id={0} in remote file system.&lt;br&gt;Message Details&lt;br&gt;Unable to read attachment with id={0} in remote file system using T3 file services.&lt;br&gt;Method&lt;br&gt;logT3ReadAttachmentException(String attachmentID)</td>
</tr>
<tr>
<td>600086</td>
<td></td>
<td>Message Body&lt;br&gt;Retrieve PendingOrdersReport&lt;br&gt;Message Details&lt;br&gt;Retrieve PendingOrdersReport with orderTypesID={0}, orderSourceID={1}, summaryLevel={2}&lt;br&gt;Method&lt;br&gt;logPendingOrdersReport(String orderTypesID, String orderSourceID, String summaryLevel)</td>
</tr>
<tr>
<td>600088</td>
<td>error</td>
<td>Message Body&lt;br&gt;Unable to add attachment with id={0} in remote file system. Attachment with id={0} already exists.&lt;br&gt;Message Details&lt;br&gt;Unable to add attachment with id={0} in remote file system using T3 file services. Attachment with id={0} already exists.&lt;br&gt;Cause&lt;br&gt;The attachment already exists.&lt;br&gt;Action&lt;br&gt;None&lt;br&gt;Method&lt;br&gt;logT3AddAttachmentAlreadyExists(String attachmentID)</td>
</tr>
<tr>
<td>600089</td>
<td>error</td>
<td>Message Body&lt;br&gt;Unable to add attachment with id={0} in remote file system. Attachment with id={0} exceeds maximum file size specified in the configuration file.&lt;br&gt;Message Details&lt;br&gt;Unable to add attachment with id={0} in remote file system using T3 file services. Attachment with id={0} exceeds maximum file size specified in the configuration file.&lt;br&gt;Cause&lt;br&gt;Your attachment size is too big.&lt;br&gt;Action&lt;br&gt;Increase the size of the max_attachment_size parameter in the oms-config.xml file. See “Configuring OSM with oms-config.xml” for more information about editing the oms-config.xml file.&lt;br&gt;Method&lt;br&gt;logT3AddAttachmentExceedMaxFileSize(String attachmentID)</td>
</tr>
</tbody>
</table>
### Table 15–1 (Cont.) OSM Catalog Messages

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Severity</th>
<th>Description</th>
</tr>
</thead>
</table>
| 600090     | error    | Message Body
Unable to read attachment with id={0} in remote file system. Attachment with id={0} does not exist.

Message Details
Unable to read attachment with id={0} in remote file system using T3 file services. Attachment with id={0} does not exist.

Cause
The attachment does not exist.

Action
None

Method
logT3ReadAttachmentNotFound(String attachmentID)

| 600091     |          | Message Body
SSL port is not found.

Message Details
SSL port is not found.

Cause
Either server is not found, or there is no SSL port set up for the current server.

Action
Make sure SSL port was set up through WebLogic console.

Method
logSSLPortNotFound()

| 600092     | error    | Message Body
MBean home is not found.

Message Details
MBeanHome.ADMIN_JNDI_NAME cannot be found.

Cause
MBeanHome.ADMIN_JNDI_NAME cannot be found.

Action
None

Method
logMBeanHomeDisabled()
### Table 15–1 (Cont.) OSM Catalog Messages

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Severity</th>
<th>Description</th>
</tr>
</thead>
</table>
| 600093     | error    | **Message Body**  
DataSource connection to database could not be found.  
**Message Details**  
The EJB requesting a connection to the database does not have a DataSource configured. This caused the data access object to fail.  
**Cause**  
The database could not be located on server startup, or the EJB deployment descriptor ejb-jar.xml is missing a resource-ref to jdbc/DataSource, or WebLogic-ejb-jar.xml is missing a resource-descriptor tag to ordermanager/oms1/internal/jdbc/DataSource  
**Action**  
Determine if the database is available. Determine if EJB has correct deployment descriptor  
**Method**  
logDataSourceNotFound() |
| 600098     | error    | **Message Body**  
The current server name cannot be found  
**Message Details**  
The current server name cannot be found  
**Cause**  
MBean cannot find current run time server  
**Action**  
None  
**Method**  
logserverNameNotFound() |
| 600099     | error    | **Message Body**  
Get max_read_only_field_length property from oms-config.xml has failed.  
**Message Details**  
Get max_read_only_field_length property from oms-config.xml has failed.  
**Cause**  
max_read_only_field_length entry does not exist in oms-config.xml.  
**Action**  
Make sure max_read_only_field_length entry exists in oms-config.xml.  
**Method**  
logGetMaxReadOnlyLengthError() |
| 600103     | error    | **Message Body**  
OMS is not enabled to send {0} event to the automated agent.  
**Cause**  
Either it is disabled in oms-config.xml file or the [1] key is not found in the file.  
**Action**  
Make sure it is [1] key in the oms-config.xml file is set to true.  
**Method**  
logDisabledJMSEvent(String event, String key) |
Table 15–1  (Cont.) OSM Catalog Messages

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Severity</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>600104</td>
<td>error</td>
<td><strong>Message Body</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Exception thrown while trying to enable or disable {0} event.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Method</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>logExceptionFindListenerForJMSEvent(String event)</td>
</tr>
<tr>
<td>600105</td>
<td>warning</td>
<td><strong>Message Body</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Cannot send messages. JMS connection is down.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Cause</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>JMS connection is down.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Action</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Use the WebLogic Server Console to verify JMS deployment.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Method</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>logJMSTopicConnectionDown()</td>
</tr>
<tr>
<td>600108</td>
<td>error</td>
<td><strong>Message Body</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>An OMS Exception has been thrown. Reason: {0}</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Method</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>logOMSException(throwable th)</td>
</tr>
<tr>
<td>600109</td>
<td>debug</td>
<td><strong>Message Body</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Lock successful</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Message Details</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Exclusive access to events table granted.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Method</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>logEventTableLockSuccess()</td>
</tr>
<tr>
<td>600110</td>
<td>debug</td>
<td><strong>Message Body</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Lock failed</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Message Details</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Access to events table denied</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Cause</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Other instance of Poller servlet holds the lock</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Action</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Wait for the current operation to end, and try again.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Method</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>logEventTableLockFailure()</td>
</tr>
<tr>
<td>Error Code</td>
<td>Severity</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>----------</td>
<td>-------------</td>
</tr>
<tr>
<td>600111</td>
<td>error</td>
<td><strong>Message Body</strong>&lt;br&gt;Invalid value ((1)) is specified for ((0)) in configuration file. Defaulting to ((2)).&lt;br&gt;<strong>Message Details</strong>&lt;br&gt;Provided value for a parameter is missing or is invalid. A default value is found in oms-config-defaults.xml file and is defaulted to.&lt;br&gt;<strong>Cause</strong>&lt;br&gt;Value out of range - nonexisting path - invalid timezone id - incorrect boolean value&lt;br&gt;<strong>Action</strong>&lt;br&gt;Check max and min values in oms-config-defaults.xml file - correct the path string - find the correct id for your timezone - check how boolean values are represented in oms-config-defaults.xml file&lt;br&gt;<strong>Method</strong>&lt;br&gt;logInvalidValueForParameter(String property, String value, String default)</td>
</tr>
<tr>
<td>600112</td>
<td>error</td>
<td><strong>Message Body</strong>&lt;br&gt;Invalid value ((1)) is specified for ((0)) in configuration file and no default value found for it in oms-config-defaults.xml file.&lt;br&gt;<strong>Message Details</strong>&lt;br&gt;Provided value for parameter is missing or is invalid. A default value is not found in oms-config-defaults.xml to default to.&lt;br&gt;<strong>Cause</strong>&lt;br&gt;oms-config-default.xml file cannot be accessed or it does not provide necessary information for the property in question&lt;br&gt;<strong>Action</strong>&lt;br&gt;verify oms-config.xml and oms-config-defaults.xml files. Check the installation. check the spelling of parameters in oms-config.xml file&lt;br&gt;<strong>Method</strong>&lt;br&gt;logInvalidValueAndNoDefault(String property, String value)</td>
</tr>
<tr>
<td>600114</td>
<td>error</td>
<td><strong>Message Body</strong>&lt;br&gt;Error reading configuration parameters XML file(URL: (0)) - (1) - (2)&lt;br&gt;<strong>Message Details</strong>&lt;br&gt;An error happened while accessing or reading the indicated configuration file.&lt;br&gt;<strong>Cause</strong>&lt;br&gt;passing error - a malformed URL F119G120E120-G120I/O exception&lt;br&gt;<strong>Method</strong>&lt;br&gt;logErrorLoadingConfigXMLFile(String url, String cause, String msg)</td>
</tr>
<tr>
<td>600115</td>
<td>error</td>
<td><strong>Message Body</strong>&lt;br&gt;In class (0), an SQL Statement was not closed after use. It will now be closed.&lt;br&gt;<strong>Cause</strong>&lt;br&gt;AbstractProxy.close() was not called after using SQL connection&lt;br&gt;<strong>Action</strong>&lt;br&gt;Report problem to development&lt;br&gt;<strong>Method</strong>&lt;br&gt;logStatementNotClosed(String classname)</td>
</tr>
</tbody>
</table>
### Table 15–1 (Cont.) OSM Catalog Messages

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Severity</th>
<th>Description</th>
</tr>
</thead>
</table>
| 600116     | error    | **Message Body**  
In class [0], an SQL Connection was not closed after use. It will now be closed.  
**Cause**  
AbstractProxy.close() was not called after using SQL connection  
**Action**  
Report problem to development  
**Method**  
logConnectionNotClosed(String classname) |
| 600117     | warning  | **Message Body**  
The view node [0] of type [1] has an invalid default value of [2]. The value has been ignored.  
**Cause**  
The default value is not entered correctly, or is incorrect for the data type.  
**Action**  
Change default value.  
**Method**  
logInvalidDefaultOrderNodeValue(String mnemonicPath, String nodeType, String default) |
| 600118     | error    | **Message Body**  
Unable to locate resource bundle.  
**Cause**  
Resource does not exist.  
**Action**  
Make sure there exist a resource file named [0]_[1].properties with key(s) [2].  
**Method**  
logI18NMissingResourceException(String baseName, String locale, String key) |
| 600119     | error    | **Message Body**  
Poller failed in init, message=[0].  
**Method**  
logPollerInitFailure(String msg) |
| 600120     | error    | **Message Body**  
Processing of timeout event failed, message=[0].  
**Method**  
logPollerProcessTimeoutFailed(String msg) |
| 600121     | error    | **Message Body**  
Unexpected system exception processing XML request: [0]  
**Cause**  
Unexpected system exception processing XML request: [0]  
**Action**  
Report to development  
**Method**  
logXMLAPIProcessorError(Throwable th) |
### Table 15–1 (Cont.) OSM Catalog Messages

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Severity</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>600122</td>
<td>debug</td>
<td>Message Body&lt;br&gt;XMLAPI Servlet, message={0}.&lt;br&gt;Method&lt;br&gt;logXMLAPIServletTrace(String msg)</td>
</tr>
<tr>
<td>600123</td>
<td>error</td>
<td>Message Body&lt;br&gt;Could not load XMLAPI Servlet request mappings. Servlet will be unavailable.&lt;br&gt;Cause&lt;br&gt;xml-request-mappings.xml file is missing.&lt;br&gt;Action&lt;br&gt;check web application WEB-INF/conf directory.&lt;br&gt;Method&lt;br&gt;logRequestMappingUnavailable(Exception reason)</td>
</tr>
<tr>
<td>600127</td>
<td>warning</td>
<td>Message Body&lt;br&gt;Error reading XML Document: {0}&lt;br&gt;Method&lt;br&gt;logErrorReadingXML(Excetion cause)</td>
</tr>
<tr>
<td>600128</td>
<td>warning</td>
<td>Message Body&lt;br&gt;An error occurred processing XML request: {0}</td>
</tr>
<tr>
<td>600129</td>
<td>warning</td>
<td>Message Body&lt;br&gt;SQL Exception {0} : {1}&lt;br&gt;Method&lt;br&gt;logSQLException(int errorCode, String message, Throwable th)</td>
</tr>
<tr>
<td>600132</td>
<td>error</td>
<td>Message Body&lt;br&gt;Unable to add attachment. Attachment exceeds maximum file size={0} specified in the configuration file.&lt;br&gt;Message Details&lt;br&gt;Unable to add attachment. Attachment exceeds maximum file size={0} specified in the configuration file.&lt;br&gt;Method&lt;br&gt;logAddAttachmentExceedMaxFileSize(String maxSize)</td>
</tr>
<tr>
<td>600133</td>
<td>error</td>
<td>Message Body&lt;br&gt;Invalid oms configuration on attachments. Attachment file system must not be null or empty. Maximum attachment size must be bigger than zero.&lt;br&gt;Message Details&lt;br&gt;Invalid oms configuration on attachments. Attachment file system must not be null or empty. Maximum attachment size must be bigger than zero.&lt;br&gt;Method&lt;br&gt;logInvalidOmsConfigurationOnAttachments()</td>
</tr>
<tr>
<td>Error Code</td>
<td>Severity</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>----------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| 600134     | error    | **Message Body**  
|            |          | Missing T3 file service with name={0}.  
|            |          | **Message Details**  
|            |          | Missing T3 file service with name={0}.  
|            |          | **Cause**  
|            |          | Missing configuration of T3 file server name in WebLogic console.  
|            |          | **Action**  
|            |          | Contact your local administrator to set T3 file server.  
|            |          | **Method**  
|            |          | logMissingT3FileServiceName(String fileSystemName) |
| 600135     |          | **Message Body**  
|            |          | T3 file service with name={0} must target exactly to one server.  
|            |          | **Message Details**  
|            |          | T3 file service with name={0} must target exactly to one server.  
|            |          | **Cause**  
|            |          | T3 file service was targeted to more than one server.  
|            |          | **Action**  
|            |          | Contact your local administrator to set up T3 file server properly.  
|            |          | **Method**  
|            |          | logDuplicateT3FileServiceTarget(String fileSystemName) |
| 600136     | error    | **Message Body**  
|            |          | Invalid T3 file service path.  
|            |          | **Message Details**  
|            |          | Invalid T3 file service path.  
|            |          | **Cause**  
|            |          | Invalid T3 file service path.  
|            |          | **Action**  
|            |          | Make sure T3 file service path is valid.  
|            |          | **Method**  
|            |          | logInvalidT3FileServicePath() |
| 600138     | error    | **Message Body**  
|            |          | Targeted server = [[0]] for the T3 File Service with name = [[1]] is not running.  
|            |          | **Message Details**  
|            |          | Targeted server = [[0]] for the T3 File Service with name = [[1]] is not running.  
|            |          | **Cause**  
|            |          | Targeted server = [[0]] for the T3 File Service with name = [[1]] is not running.  
|            |          | **Action**  
|            |          | Make sure the targeted server is running.  
|            |          | **Method**  
|            |          | logTargetedServerForT3FileServiceNotRunning(String t3TargetServerName, String fileSystemName) |
| 600139     | debug    | **Message Body**(0) |
## Automation Catalog Messages

Table 15–2 shows Automation Catalog messages.

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Severity</th>
<th>Description</th>
</tr>
</thead>
</table>
| 700002     | debug    | **Message Body**
|            |          | `parseXmlData(String xmlData)`
|            |          | **Method**
|            |          | Parse the XML data [0] |
| 700004     | debug    | **Message Body**
|            |          | `lookupOMSBean(String jndiName)`
|            |          | **Method**
|            |          | Lookup the OMSSession Bean with jndiName=[0] |
| 700007     | debug    | **Message Body**
|            |          | `receiveTask(long orderId, long orderHistoryId, String taskMnemonic)`
|            |          | **Method**
|            |          | Receive Task with orderId=[0] orderHistoryId=[1] and task mnemonic=[2] |
| 700008     | debug    | **Message Body**
|            |          | `completeTaskOnExit(long orderId, long orderHistId, String taskMnemonic, String completStat)`
|            |          | **Method**
|            |          | Complete task On Exit with orderId=[0], orderHistoryId=[1], taskMnemonic=[2] and Completion Status=[3] |
| 700009     | debug    | **Message Body**
|            |          | `assignTask(long orderId, long orderHistoryId, String taskMnemonic, String userId)`
|            |          | **Method**
| 700010     | debug    | **Message Body**
|            |          | `suspendTask(long orderId, long orderHistoryId, String taskMnemonic, String suspendState)`
|            |          | **Method**
| 700011     | debug    | **Message Body**
|            |          | `acceptTask(long orderId, long orderHistoryId, String taskMnemonic)`
|            |          | **Method**
|            |          | Accept Task with OrderId=[0] orderHistryId=[1] and taskMnemonic=[2] |
| 700012     | debug    | **Message Body**
|            |          | `updateOrderData1(long orderId, long orderHistId, String TaskNmem, String updateData)`
|            |          | **Method**
|            |          | UpdateOrderData with orderId=[0] orderHistoryId=[1], TaskMnemonic=[2] and XML Data=[3] |
### Table 15–2 (Cont.) Automation Catalog Messages

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Severity</th>
<th>Description</th>
</tr>
</thead>
</table>
| 700014     | debug    | Message Body  
The orderData id={0} and Data:{1}  
Method  
printOrderDataAsXML(long OrderId, String OrderData) |
| 700016     | debug    | Message Body  
Lookup the Automator: type={0} and jndiName={1}  
Method  
lookupAutomator(String type, String jndiName) |
| 700019     | debug    | Message Body  
Get a JMS Message with EventType={0}, Mnemonic={1}  
Method  
getAMessage(String type, String mnemonic) |
| 700020     | debug    | Message Body  
Set the outMessage Correlation to={0} the correlationId={1}  
Method  
setCorrelationId(String correlation, String correlationId) |
| 700021     | debug    | Message Body  
Send Notification Email message to :{0}, subject:{1}, MessageBody:{2}  
Method  
sendEmailNotification(String to, String subject, String msgBody) |
| 700023     | error    | Message Body  
An automation exception has occurred At {0}, the reason is :{1}  
Cause  
See message body.  
Method  
AutomationException(String msg, Throwable th) |
| 700024     | error    | Message Body  
Naming Exception has occurred at {0} reason {1}  
Cause  
Cannot find user plug-in from JNDI tree.  
Method  
namingException(String name, Throwable ex) |
| 700033     | error    | Message Body  
{0} {1}  
Cause  
See message body for details.  
Method  
logAutomationException(String desc, Throwable exception) |
Using the XML Import/Export Application

This chapter provides information about the XML Import/Export application (XMLIE), which is used to manage data and metadata in the Oracle Communications Order and Service Management (OSM) database schema.

About the XML Import/Export Application

XMLIE is included with the OSM SDK Tools component that can be installed with the OSM installer. For more information about installing the OSM SDK Tools component, see OSM Installation Guide.

There are two types of information in an OSM database schema:

- Metadata: Information that defines the order model. For example, the definitions of processes, orders, and tasks.
- Data: Information that represents orders. For example, order nodes, attributes, and values.

Using XMLIE, you can perform actions such as import and export metadata, purge metadata and data, and migrate data. You can also use XMLIE to validate the metadata to create a graphical representation of the metadata.

**Note:** Although actions such as importing and exporting metadata, purging both metadata and data, and migrating data can be done using XMLIE, Oracle Communications Design Studio is the preferred application for performing these functions.

**Note:** XMLIE can work with a localized database, but the application must also be localized. See OSM Developer’s Guide for information on localizing OSM, including localizing XMLIE.

If you are running the OSM application on a UNIX or Oracle Linux platform, you must run XMLIE by using a set of Ant scripts. If you are running the OSM application on a Windows platform, you must run XMLIE by using a set of batch scripts.

About Using the XML Import/Export Application

The following steps provide a high-level overview of using XMLIE:

1. Configure the XMLIE environment files:
For Ant commands, configure the OSM_home/SDK/XMLImportExport/build.properties file (where OSM_home is the directory in which OSM is installed). See "Configuring the build.properties File for Ant Commands".

For batch scripts, configure the OSM_home/SDK/XMLImportExport/config.bat script. See "Configuring the config.bat Script for a Batch Scripts".

2. Copy the OSM_home/SDK/XMLImportExport/config/config_sample.xml file and rename it to OSM_home/SDK/XMLImportExport/config/config.xml in the same directory.

   **Note:** OSM creates the OSM_home/SDK/XMLImportExport/config/config_sample.xml file when you install the OSM SDK Tools component. This file contains a sample XMLIE configuration file that can be used as a template for the config.xml configuration file.

   The config.xml file name is arbitrary. If you customize the name of the config.xml file, ensure that you substitute the customized name wherever you must specify the config.xml file (for example, when using the import and export commands in the import.bat and export.bat scripts).

   This chapter uses the default config.xml file name in all examples.

3. Configure the OSM_home/SDK/XMLImportExport/config/config.xml file.

   Both the XMLIE Ant commands and batch scripts use this file to define:

   - Connections to other components
     For example, database connection XML node that provides the OSM schema user name, password, and connection details.

   - How the Ant commands and batch scripts work
     For example, the import.bat script is configured using the import node in the config.xml file. This node contains elements that specify whether the imported data is validated, what actions to take if the database is not empty, and what actions to take if the XML model you are importing already exists in the database.

   - The data or metadata on which the commands are to act
     For example, you can configure a selective import that only imports one specific element into an existing OSM cartridge using the selection element.

   See “Configuring the config.xml File XML Import/Export Nodes and Elements”.

4. You can create an empty text file with a .xml extension and then use the export.bat script or the Ant export command to populate the file. Then you can edit it for use with other batch scripts or Ant commands.

   **Note:** This chapter uses xmlModelFile as the documentation placeholder name for this file.

5. Run the Ant command or batch script.
About XML Import/Export Batch Scripts and Ant Commands

The following sections describe the XMLIE batch scripts and Ant commands.

About XML Import/Export Ant Commands and Syntax

OSM supports the following Ant commands and syntax for UNIX and Linux systems:

- **ant export**: exports all cartridges, all cartridges within a given namespace, or one specific cartridge.
- **ant import**: imports the XML model.
- **ant purge**: completely purges the target database.
- **ant undeploy**: undeploys the given cartridge if no pending orders exist.
- **ant forceUndeploy**: forces to undeploy the given cartridge including pending orders.
- **ant convert**: upgrades the old XML model to latest version.

**Note:** This command is deprecated. Use the `ant import` command instead because it automatically upgrades models during an import.

- **ant refresh**: refreshes the metadata in the WebLogic Server.
- **ant htmlModel**: converts an XML metadata document to HTML and graphical format.
- **ant userAdmin**: adds users from WebLogic groups and OSM workgroups.

For more information about the **ant userAdmin** command, see "Using the XML Import/Export Application to Administer Users and Workgroups".

Before you can run these Ant commands, you must configure the Ant environment `build.properties` file and the `config.xml` file. For more information see "Configuring the XML Import/Export Environment Files" and "Configuring the config.xml File XML Import/Export Nodes and Elements".

You can also use the following script to encrypt passwords contained in the `config.xml` file. See "Using the EncryptPasswords Utility" for more information.

- **EncryptPasswords.sh**

About XML Import/Export Batch Scripts and Syntax

OSM supports the following batch scripts for Windows systems in the `OSM_home/SDK/XMLImportExport` folder:

- **import.bat**: Imports an XML metadata document into an OSM database.
- **export.bat**: Exports an OSM database to an XML metadata document.
- **migrate.bat**: Migrates order data from one version of a cartridge to another version of the same cartridge.
- **purge.bat**: Purges the entire OSM schema (metadata and orders) or undeploys a specific cartridge.
- **orderPurge.bat**: Purges orders that satisfy purge criteria; can be run on an immediate or scheduled basis.
Configuring the XML Import/Export Environment Files

The following sections describe the files you need to edit to configure the environment for the XMLIE Ant commands and batch scripts.

Configuring the build.properties File for Ant Commands

The paths to the config.xml file and the XML model document, along with cartridge and environment properties used by the XMLIE Ant commands, must be specified in the $OSM_HOME/SDK/XMLImportExport/build.properties file. Unlike the XMLIE batch scripts, you do not have to specify these paths in the command line.

To configure the build.properties file for Ant commands:

1. Open the $OSM_HOME/SDK/XMLImportExport/build.properties file.
2. Update the following variables:
Configuring the XML Import/Export Environment Files

Using the XML Import/Export Application

```java
java.maxmemory=root.dir
xmlie.root.dir=./
xmlie.root.modelDocument=./data.xml
xmlie.root.configDocument=./config/config.xml
xmlie.root.namespace=test
xmlie.root.version=4.0
xmlie.root.htmlDir=./htmlModel
```

where:
- `java.maxmemory`: The maximum heap to be used by JVM
- `root.dir`: The path of XMLIE directory
- `modelDocument`: The path for XML model document
- `configDocument`: The path for `config.xml`
- `namespace`: The OSM cartridge namespace
- `version`: The cartridge version
- `htmlDir`: The path for HTML model directory

**Note:** You can also refer to the README.txt file found under SDK/XMLImportExport for information about configuring the `build.properties` file for Ant scripts.

For example:

```java
java.maxmemory=512m
xmlie.root.dir=./
xmlie.root.modelDocument=./data.xml
xmlie.root.configDocument=./config/config.xml
xmlie.root.namespace=test
xmlie.root.version=4.0
xmlie.root.htmlDir=./htmlModel
```

3. To configure order purge parameters in the `build.properties` file, see "Running Ant with the orderPurge.xml file On UNIX or Linux to Purge Orders".

4. To configure cartridge migration parameters in the `build.properties` file, see "Configuring and Running an Order Migration".

**Configuring the config.bat Script for a Batch Scripts**

To configure the batch script environment:

1. Update the `OSM_home\SDK\XMLImportExport\config.bat` script to specify the following paths and configuration options:

   **Note:** No other configuration values in the `config.bat` script should be edited.

   - `JAVA_HOME`: The directory in which the Java SDK or Java Runtime Environment resides.
   - `APP_ROOT`: The XML handler root directory. If the XMLIE application is installed in the default location, you must enclose the path in double quotation marks, as in: "C:\Program Files\OSM\SDK\XMLImportExport".
■ **JAVA_OPTS**: The minimum and maximum memory requirements.

For example:

```
set JAVA_HOME=C:\Oracle\Middleware\jdk1.7.0_51
set APP_ROOT="D:\osm\SDK\XMLImportExport"
set JAVA_OPTS=-ms512m -mx512m
```

---

### Configuring the config.xml File XML Import/Export Nodes and Elements

To configure the `config.xml` file:

1. Open the `OSM_home/XMLImportExport/config/config_sample.xml` file.
2. Create a `config.xml` file by copying `config_sample.xml` to `config.xml`.

**Note:** The `sample_config.xml` file contains references to absolute paths that start with "C:". Be sure to configure these paths to reflect your environment.

3. Configure the database connection node in the `config.xml` file. This node is required for most Ant commands and batch scripts.

```xml
<databaseConnection>
  <user>osm_schema</user>
  <password>osm_schema_pwd</password>
  <dataSource>jdbc:oracle:thin:@ip_address:osm_database_port:osm_database_sid</dataSource>
</databaseConnection>
```

where:

- **osm_schema** and **osm_schema_pwd** are the OSM schema user name and password.
- **ip_address**, **port**, and **sid** are the OSM database IP address, port number, and SID.

**Note:** To export / import the symbols change the character set and db connection to:

```xml
<encoding>UTF-8</encoding>
<dataSource>jdbc:oracle:oci1: (description= (address=(host=kanlddb698.ca.oracle.com) (protocol=tcp) (port=1521)) (connect_data=(SID=ORCH10G)))</dataSource>
```

---

4. Configure the XML API connection node to specify the connection information to be used by operations that utilize the XML API. For example, migration Ant commands or batch scripts require this node.

```xml
<xmlAPIConnection>
  <user>weblogid_user</user>
  <password>weblogic_pwd</password>
  <url>http://ip_address:port</url>
</xmlAPIConnection>
```

where
weblogic_user and weblogic_pwd are the user name and password of the user performing the migration. This user must have access to all source orders being migrated and to target order's type/source order entry task (if closeSource set to true, see "About Migrating Orders" for details).

- ip_address, and port are the WebLogic Administration server IP address and port number.

5. Configure the WebLogic Administrator credentials and connection information node. This node is required for Ant commands and batch scripts that modify OSM user credentials:

   ```xml
   <j2eeAdminConnection>
     <j2eeServiceName>weblogic</j2eeServiceName>
     <user>weblogic_user</user>
     <password>weblogic_pwd</password>
     <hostname>ip_address</hostname>
     <port>port</port>
   </j2eeAdminConnection>
   ```

where

- weblogic_user and weblogic_pwd are the user name and password of the WebLogic with Administrator privileges.
- ip_address, and port are the WebLogic Administration server IP address and port number.

6. Configure the XMLIE log file location node:

   ```xml
   <log logFileUrl="file:/filename_path" overwrite="boolean"/>
   ```

where:

- filename_path is the path to the log file that includes the file name of the log file.
- boolean can be true or false. If set to true (default), XMLIE overwrites the log file every time the application starts. If set to false, XMLIE creates a cumulative log by saving the log from session to session and adding new messages to it.

7. Do one of the following:

   - For importing metadata to the OSM database, configure the import node. See "About Importing Metadata".
   - For exporting metadata from an OSM database, configure the export node. See "About Exporting Metadata".
   - For purging metadata and order data in an OSM database, configure the purge node. See "About Purging MetaData and Data".
   - For migrating orders between OSM cartridges, configure the migrate node. See "About Migrating Orders".
   - For validation Metadata from an existing XML file, configure the validation node. See "About Validating the Metadata Model and Data".
   - For creating a graphical HTML representation see "About Creating a Graphical Representation of the Metadata Model".

8. (Optional) If XMLIE is to be run unattended, secure the EncryptPassword utility and the configuration file that contains user credentials.
For enhanced security, each of the XMLIE operations that require user passwords prompts you for those passwords during invocation. If XMLIE is to be run unattended, you can alternatively encrypt those passwords and store them in the XMLIE configuration file (typically `config.xml`).

If passwords are to be stored in the XMLIE configuration file, do the following:

a. Set the permissions of the configuration file to be readable only by select administrative users. Refer to your OS documentation for instruction.

b. Run the EncryptPassword utility so that user name and password credentials for all XMLIE users are encrypted for safe storage. See “Using the EncryptPasswords Utility” for more information.

**Note:** If you plan to run XMLIE in an unattended mode, you must first run the EncryptPasswords utility; otherwise, you cannot perform many of the application functions and OSM gives an error indicating that you must run the EncryptPasswords utility.

---

**About Importing and Exporting Metadata**

You can transfer metadata from one OSM database to another when setting up a new OSM environment. For example, you may want to set up multiple OSM environments of the same version, such as development, test, and production environments. Or, you may want to set up a new version of an OSM production environment based on a previous version of an OSM production environment.

Transferring metadata from one OSM database to another is a combination of an export from the one OSM database followed by an import to another. You can also import and export selected parts of the metadata, as opposed to all of the metadata.

**Note:** The actions that XMLIE runs are supported in Design Studio. Any metadata changes made using XMLIE will be overwritten when deploying the same cartridge using Design Studio.

The following sections describe import and export commands and configurations.

---

**About Exporting Metadata**

XMLIE provides the `export.bat` script or the `export` Ant command, which is used to export metadata from an OSM database. Exported metadata is stored in an XML file (the `xmlModelFile`). You can export metadata using Design Studio; however, this functionality remains available through XMLIE as well.

You can export the entire metadata database, or you can use the export `selection` element to specify the metadata to export based on:

- All entities or selected entities from a specific cartridge
- All cartridges in a namespace

Each OSM cartridge is uniquely identified by namespace and version, so exporting all of the cartridges in a namespace includes all versions of the cartridge within the namespace.
Note: A selective export exports all of the data, then it applies a filter according to the selection. Consequently, selective exports take the same amount of time as full exports.

About the Order of Exported Metadata
When the order of the resulting list makes no logical or significant difference, XMLIE places the metadata files in ascending alphabetical order. This ensures that when you do repeated exports of the same database, the metadata files will always appear in the same order, which makes it easier to merge metadata changes with future development.

In Example 16–1, the definition of entities (for example, each state) is now sorted alphabetically by name:

**Example 16–1 Definition of Entities (each state) Sorted Alphabetically by Name**

```xml
<state name="accepted">
<description>Accepted</description>
</state>
<state name="assigned">
<description>Assigned</description>
</state>
<state name="completed">
<description>Completed</description>
</state>
<state name="received">
<description>Received</description>
</state>
...
```

In Example 16–2, the references to entities (for example, each state) is now sorted alphabetically by name:

**Example 16–2 References to Entities (states or statuses) Sorted Alphabetically by Name**

```xml
<task name="enter_payment_information" xsi:type="genericTaskType">
<description>Enter Payment Information</description>
<state>accepted</state>
<state>completed</state>
<state>received</state>
<status>back</status>
<status>next</status>
</task>
```

In Example 16–3, the Import/Export application does not sort by name because the order matters (that is, there is a logical difference). Country appears after last_name because the designer specified country to appear after last_name:

**Example 16–3 Not Sorted By Name**

```xml
<masterOrderTemplate>
<dataNode element="account_information"/>
<dataNode element="first_name"/>
<dataNode element="last_name"/>
<dataNode element="country"/>
&viewRule xsi:type="eventRuleType">
<event>value-changed</event>
```
About Importing and Exporting Metadata

About Export Layout Options
When you perform a selective export, you can specify different export layouts by using the export command to create one or multiple files, and where to put the files. You have the following options:

- Use the `singleDocument` layout to export to a single file that contains all of the exported entities.
- Use the `cartridge` layout to export to one main document and a single folder for each exported cartridge. Each cartridge folder contains a version folder.

Figure 16–1 shows the `cartridge` layout.

Figure 16–1 Cartridge Layout

- You can use the `entity` layout to export to:
  - One main document
  - A folder for each cartridge included in the export (you can specify that the export include entities from more than one cartridge)
  - A folder for the cartridge version
  - A single folder for each entity type exported. Under the entity type folders are the individual entity XML files

Figure 16–2 shows the `entity` layout.
Keeping the ID Integrity in SQL Rules

**Note:** This section is applicable only if you are upgrading up to OSM 7.0 from a previous release. SQL rules and text rules were replaced in OSM 7.0. SQL rules are supported in Design Studio. The SQL rule is imported as a separate file that can be edited as a text document.

The SQL based rule type in OSM can contain entity IDs (mostly node IDs), which must be replaced with new IDs during data migration. IDs must be exposed as entity attributes, helping the application find them in SQL based rules, and replace them with new ones while importing them into a fresh environment. IDs in a document otherwise serve no purpose and should be ignored. You can set this feature to false by setting the `exposeEntityID` in the `config.xml` file for XMLE, to false, thereby reducing the model document size.

Text rules can reference any entity ID. Generally, however, node IDs used in these known patterns for other possible entity ID conversion routines require user assistance to tokenize the rule text for proper parsing and conversion. A token suggested before the ID must be tokenized for IDs except known patterns (all node functions and stored procedures in `om_ordinst_value_pkg` package):

```sql
/*$entityType*/
```

**Example 16–4 Original Rule**

```sql
declare val1 date;
delay_flag varchar2(10);
begin
select timestamp_out into val1
from om_hist$order_header hist, om_task task, om_state st
where hist.task_id = task.task_id
and node_id = /*$dataNode*/76983
and hist.hist_order_state_id = st.state_id and st.state_mnemonic = 'completed'
```
and hist.order_seq_id = :order_seq_id;

delay_flag := om_ordinst_value_pkg.get_node_value_like(:order_seq_id,76983,:coord_set_id);
if  ( rtrim(delay_flag)='yes' ) or (val1 <= (sysdate - 2/24)) then
  :rule_result := 'true';
else
  :rule_result := 'false';
end if;
end;

Example 16–5  Modified Rule
The import operation detects IDs and replaces them with new IDs.

declare val1 date;
delay_flag    varchar2(10);
begin
select timestamp_out into val1
from om_hist$order_header hist, om_task task, om_state st
where hist.task_id = task.task_id
and node_id = /*$dataNode*/new_id
and hist.hist_order_state_id = st.state_id and st.state_mnemonic = 'completed'
and hist.order_seq_id = :order_seq_id;

delay_flag := om_ordinst_value_pkg.get_node_value_like(:order_seq_id, new_id,:coord_set_id);
if  ( rtrim(delay_flag)='yes' ) or (val1 <= (sysdate - 2/24)) then
  :rule_result := 'true';
else
  :rule_result := 'false';
end if;
end

Configuring and Running an Export
To configure and run an export:

1. Configure the config.xml file for use with the export Ant command.

```xml
<export validateModel="validation" exposeEntityID="entityID" layout="layout_action" readOnlyFileAction="readOnly_action"/>
</export>
```

where:

- **validation**: Options are:
  - true: Validates the XML model before performing the export.
  - false: Does not validates the XML model before performing the export. (default).

Caution: Oracle recommends you do not skip the model validation, so this parameter should always be set to true.

- **entityID**: Options are:
  - true: Exposes the EntityID in the exported file. (default)
  - false: Conceals the Entity ID in the exported file.
About Importing and Exporting Metadata

- **layout_action**: Options are:
  - **singleDocument**: Exports everything to a single file. (default)
  - **cartridge**: Creates a single file for each cartridge exported.
  - **entity**: Creates a single file for each entity exported.

- **readOnly_action**: Options are:
  - **ignore**: The Export operation will not overwrite read-only files. (default)
  - **replace**: The Export operation will overwrite read-only files.

2. (Optional) Add the **selection** element within the **export** node to export targeted entities to OSM metadata using an XPath expression. Use the following syntax to define the selection element:

```xml
<export validateModel="validation" exposeEntityID="entityID" layout="layout_action" readOnlyFileAction="readOnly_action">
  <selection>/oms:model/oms:cartridge[@namespace="namespace" and @version="version"]/oms:entity</selection>
  <selection>/oms:model/oms:entity</selection>
</export>
```

where
- **namespace**: The namespace for the cartridge.
- **version**: The cartridge version.
- **entity**: The entity you are targeting. For example, workgroup, region, and schedule.

3. Do the following:
   a. If you using Ant, run the following command:
      ```
      ant export
      ```
   b. If you using a batch script, run the following command:
      ```
      export.bat xmlModelFile config/config.xml
      ```

**About Importing Metadata**

XMLIE provides the **import** command, which is used to import metadata into an OSM database. If you import metadata, make sure that the elements you import do not conflict with existing metadata that is part of a Design Studio OSM cartridge. Otherwise you may encounter version conflict, overwrite existing elements, and create other discrepancies.

You can import the entire metadata database using the import node or you can use the import **selection** element within an **import** node to specify the metadata to import based on:

- All entities or selected entities from a specific cartridge
- All cartridges in a namespace

Each OSM cartridge is uniquely identified by namespace and version, so exporting all of the cartridges in a namespace includes all versions of the cartridge within the namespace.
- System level parameters
By enabling selective imports, you can grant concurrent access to a single model or cartridge for multiple developers. Using this method, developers can import just the entities on which they are working at that moment, which gives other developers access to other entities within the cartridge.

The import operation is performed in one transaction. Consult your Oracle Database Administrator (DBA) for the appropriate setup for the rollback segment.

---

**Note:** User workgroups are not part of the metadata model, so they must be re-entered after an import.

---

**Note:** After importing or exporting a cartridge, you must remap the e-mail notifications that were associated with individual users. For best results, associate notifications only with workgroups because user names differ between environments.

---

**Configuring and Running an Import**

To configure and run an import:

1. Configure the `config.xml` file:

   ```xml
   <import validateModel="validation" nonEmptyDatabaseAction="database_action" entityConflictAction="entity_action"/>
   </import>
   ```

   where:

   - **validation**: Options are:
     - `true`: Validates the XML model before performing the import.
     - `false`: Does not validates the XML model before performing the import. (default)

   **Caution:** Oracle recommends you do not skip the model validation, so this parameter should always be set to `true`.

   - **database_action**: Options are:
     - `ignore`: The import completes even if it detects that the database is non-empty. (default)
     - `abort`: The import terminates if it detects that the database is non-empty.
     - `purge`: The import purges the existing database if it detects that the database is non-empty.

   If a model includes some changes for existing entities in the database, the import checks for order dependency. If those changes do not violate database constraints for pending orders, the import completes successfully. If the modified entities violate the existing orders, the violation is reported as known application exceptions with descriptive messages.

   - **entity_action**: The `entityConflictAction` import parameter value specifies the import behavior when the application encounters a conflict for an existing entity in the database. For example, a conflict occurs if you import a new
model that contains one or more entities that already exist in the database. Options are:

- **abort**: If an entity conflict exists, the import process stops.
- **replace**: The import replaces conflicted entities, that is, entities that already exist. (default)
- **ignore**: The import does not replace conflicted entities.

2. (Optional) Add the **selection** element within the **import** node to import targeted entities to an OSM cartridge using an XPath expression. Use the following syntax to define the selection element:

   ```xml
   <import validateModel="validation" nonEmptyDatabaseAction="database_action" entityConflictAction="entity_action">
     <selection>/oms:model/oms:cartridge[@namespace="namespace" and @version="version"]/@oms:entity</selection>
   </import>
   ```

   where
   - **namespace**: The namespace for the cartridge.
   - **version**: The cartridge version.
   - **entity**: The entity you are targeting. For example, workgroup, region, and schedule.

3. Do the following:
   a. If you are using Ant, run the following command:
      ```bash
      ant import
      ```
   b. If you using a batch script, run the following command:
      ```bash
      import.bat xmlModelFile config/config.xml
      ```

**Sample Procedure for Adding a New Workgroup Definition (Role)**

You can add a new workgroup definition (a workgroup is called a role in Design Studio) without having to redeploy a cartridge by using XMLIE.

**Note:** The workgroup definition is a system level entity that can be applied to multiple different cartridges. If you add a new workgroup using XMLIE, make sure you do not overwrite existing workgroup definitions.

To add a new workgroup definition using XMLIE:

1. In the `OSM_home\SDK\XMLImportExport\config\config.xml` file, add a selection element to an export node that targets the `workgroupDefinition` entity.

   For example:
   ```xml
   <export validateModel="false" exposeEntityID="false" layout="singleDocument" readOnlyFileAction="ignore">
     <selection>/oms:model/oms:workgroupDefinition</selection>
   </export>
   ```

2. Save and exit the file.
3. In the OSM_home\SDK\XMLImportExport\ folder, create an XML document to store the XML workgroup information from the cartridge you are targeting. For example: worgrkoupDefinition.xml.

4. Run the OSM_home\SDK\XMLImportExport\export.bat script:
   For example:
   ```
   export.bat C:\osminstall\SDK\XMLImportExport\worgroupDefinition.xml
   C:\osminstall\SDK\XMLImportExport\config\config.xml
   ```

5. When the export is complete, open the XML file containing the workgroup information and add a new workgroupDefinition element and all child elements. For example:
   ```
   <?xml version='1.0' encoding='ISO-8859-2'?>
   xmlns:osm="http://xmlns.oracle.com/communications/ordermanagement/model"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   file:///D:/OSMSTA~1/SDK/XMLIMP~1//models/OmsModel.xsd">
   <workgroupDefinition name="newWorkgroup">
   <description>newWorkgroup</description>
   <permissions>
   <createdVersionedOrders />
   <exceptionProcessing />
   <onlineReports />
   <priorityModification />
   <referenceNumberModification />
   <searchView />
   <taskAssignment />
   <worklistViewer />
   </permissions>
   <calendar>
   <weeklyWorkHours>no_schedule</weeklyWorkHours>
   <region>no_region</region>
   </calendar>
   </workgroupDefinition>
   </model>
   ```

6. Save and exit the file.

7. Run the OSM_home\SDK\XMLImportExport\import.bat script:
   For example:
   ```
   import.bat C:\osminstall\SDK\XMLImportExport\worgroupDefinition.xml
   C:\osminstall\SDK\XMLImportExport\config\config.xml
   ```
Note: In this scenario, the XML model file contains only those elements that need to be imported. If the model file contained other elements that did not need to be imported, you can add a selection element to the import node in the OSM_home\SDK\XMLImportExport\config\config.xml file that targets the workgroupDefinition entity in the model file.

For example:

```
<import validateModel="false"
    nonEmptyDatabaseAction="ignore" entityConflictAction="replace">
    <selection>/oms:model/oms:workgroupDefinition</selection>
</import>
```

Sample Procedure for Adding a Task to a Workgroup (Role)

You can add a task to a workgroup (a workgroup is called a role in Design Studio) without having to redeploy a cartridge by using XMLIE.

Note: This procedure assumes that you have already created the task that you want to assign to a workgroup.

To add a task to a workgroup using XMLIE:

1. In the OSM_home\SDK\XMLImportExport\config\config.xml file, add a selection element to an export node that targets the workgroup entity in the cartridge you want to add a task to.

   For example:

   ```xml
   <export validateModel="false" exposeEntityID="false" layout="singleDocument"
           readOnlyFileAction="ignore">
      <selection>/oms:model/oms:cartridge[@namespace="bb_ocm_demo" and
                      @version="1.0.0.0.1"]/oms:workgroup</selection>
   </export>
   ``

2. Save and exit the file.

3. In the OSM_home\SDK\XMLImportExport folder, create an XML document to store the XML workgroup information from the cartridge you are targeting. For example: worgroup.xml.

4. Run the OSM_home\SDK\XMLImportExport\export.bat script:

   For example:

   ```
   export.bat C:\osminstall\SDK\XMLImportExport\worgroup.xml
   C:\osminstall\SDK\XMLImportExport\config\config.xml
   ```

5. When the export is complete, open the XML file containing the workgroup information and add a new task element. For example:

   ```xml
   <?xml version='1.0' encoding='ISO-8859-2'?>
       xmlns:osm="http://xmlns.oracle.com/communications/ordermanagement/model"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       file:///D:/OSMSTA~1/SDK/XMLIMP~1//models/OmsModel.xsd">
      <schemaVersion>7.2.0</schemaVersion>
   ```
<version>
  <label>7.2.0.0.366</label>
  <majorVersion>1.0</majorVersion>
</version>
<cartridge namespace="bb_ocm_demo" version="1.0.0.0.1">  
  <description>BB OCM Demo</description>  
  <default>true</default>  
  <timestamp>2012-05-28T13:05:12</timestamp>
  <workgroup name="demo">
    <column name="Phone ">#
      <path>/subscriber_info/primary_phone_number</path>
    </column>
    <column name="Name">
      <path>/subscriber_info/name</path>
    </column>
    <permissions>
      <orderEntry>
        <orderType>add_adsl_siebel</orderType>
        <orderSource>add_adsl_siebel</orderSource>
      </orderEntry>
      <task>activate_dslam</task>
      <task executionModes="do">add_adsl_siebel_creation</task>
      <task>add_capacity</task>
      <task>assign_port</task>
      <task>demo_query</task>
      <task>send_customer_survey</task>
      <task>ship_modem_self_install_pkg</task>
      <task>verify_adsl_service_availability</task>
      <task>verify_order</task>
      <task>new_task</task>
    </permissions>
  </workgroup>
</cartridge>
</model>

6. Save and exit the file.

7. Run the OSM_home\SDK\XMLImportExport\import.bat script:

For example:

import.bat C:\osminstall\SDK\XMLImportExport\workgroup.xml
C:\osminstall\SDK\XMLImportExport\config\config.xml

**Note:** In this scenario, the XML model file contains only those elements that need to be imported. If the model file contained other elements that did not need to be imported, you can add a selection element to the import node in the OSM_home\SDK\XMLImportExport\config\config.xml file that targets the workgroup entity in the model file.

For example:

```xml
<import validateModel="false"
  nonEmptyDatabaseAction="ignore" entityConflictAction="replace">
  <selection><oms:model><oms:cartridge[@namespace="bb_ocm_demo" and @version="1.0.0.0.1"]/oms:workgroup></selection>
</import>
```
About Purging MetaData and Data

XMLIE provides the `purge.bat`, `purgeOrder.bat` scripts and Ant `purge`, `undeploy`, and `forceundeploy` commands, which are used to purge metadata and data from an OSM schema. For example, you may need to purge an existing schema prior to importing a new model, or you may need to purge the data from a test environment at the beginning of each new phase of testing.

Using the `purge.bat` script or purge Ant command, you can remove everything (all metadata and data) from the schema, or you can remove data from a specified cartridge. Using the `purgeOrder.bat` script or purge Ant command, you can remove all the data. The `purgeOrder.bat` script and purge Ant command do not affect metadata.

Oracle recommends that you purge all order data related to a cartridge before purging a cartridge. Purging large amounts of order data or cartridges should only be done during off-peak hours.

**Note:** You must shut down the WebLogic Server before executing the `purge` command or an exception is thrown.

The `purge.bat` script and purge Ant command are not transactional so any unexpected failure may leave the schema in an invalid state. If this occurs, repeat the `purge.bat` script or purge Ant command until it completes successfully.

Undeploying Cartridges and Purging the Database Schema

You can undeploy a cartridge using either `fast undeploy` or `undeploy`. When you use fast undeploy, which is the default functionality, the cartridge is undeployed from OSM, but cartridge metadata and order data are not purged from the database. The fast undeploy functionality is useful in development and test environments where you need to deploy and undeploy cartridges frequently.

When you use the undeploy functionality (where the `fast_cartridge_undeploy` parameter is set to `false`), the cartridge is undeployed and its metadata and order data are purged from the database, which can be time-consuming depending on how complex the cartridge is and whether it has a significant number of orders.

For information about changing the `fast_cartridge_undeploy` parameter in the `oms-config.xml` file, see "Configuring OSM with oms-config.xml".

To purge the entire schema (metadata and orders) or undeploy a specific cartridge using Ant commands:

1. Do one of the following:

   **Note:** You can target an entire schema for the `undeploy`, `forceundeploy`, and `purge` commands, or you can specify a namespace or a namespace and version by configuring the following `build.properties` parameters:

   - `xmlie.root.namespace`
   - `xmlie.root.version`

   For more information about these parameters, see "Configuring the `build.properties` File for Ant Commands".
To purge a target schema, run the following command:

```
ant purge
```

To undeploy a cartridge using fast undeploy, but only if no pending orders exist, run the following command:

```
ant fastUndeploy
```

To undeploy a cartridge, but only if no pending orders exist, run the following command:

```
ant undeploy
```

To undeploy a cartridge using fast undeploy, even if pending orders exist, run the following command:

```
ant forceFastUndeploy
```

To undeploy a cartridge, even if pending orders exist, run the following command:

```
ant forceUndeploy
```

To purge the entire schema (metadata and orders) or undeploy a specific cartridge using batch scripts:

1. Do one of the following:

   - To completely purge a target schema, use the following batch script:
     
     ```
purge.bat config\config.xml force
     ```
     
     If you run this script without the `force` attribute, the `purge.bat` script fails if any pending orders exist on the cartridge you are purging. If you run this script with the `force` attribute, the script purges the cartridge and all pending orders.

   - To undeploy every version of a cartridge with a specified namespace, but only if no pending orders exist, use the following batch script:
     
     ```
purge.bat config\config.xml -n namespace
     ```
     
     where `namespace` is the OSM cartridge namespace.

   - To undeploy a specified version of a cartridge, but only if no pending orders exist, use the following batch script:
     
     ```
purge.bat config\config.xml -n namespace -v version
     ```
     
     where `namespace` and `version` are the OSM namespace and cartridge version.

   - To undeploy every version of a cartridge, run the following batch script:
     
     ```
purge.bat config\config.xml force -n namespace
     ```
     
     where `namespace` is the OSM cartridge namespace.

   - To undeploy a specific cartridge version, use the following batch script:
     
     ```
purge.bat config\config.xml force -n namespace -v version
     ```
     
     where `namespace` and `version` are the OSM namespace and cartridge version.
To undeploy every version of a cartridge using fast undeploy, but only if no pending orders exist, use the following batch script:

```
fastUndeploy.bat config\config.xml -n namespace
```

where *namespace* is the OSM cartridge namespace.

To undeploy a specific cartridge version using fast undeploy, but only if no pending orders exist, use the following batch script:

```
fastUndeploy.bat config\config.xml -n namespace -v version
```

where *namespace* and *version* are the OSM namespace and cartridge version.

To undeploy every version of a cartridge using fast undeploy, even if pending orders exist, use the following batch script:

```
fastUndeploy.bat config\config.xml force -n namespace
```

where *namespace* is the OSM cartridge namespace.

To undeploy a specific cartridge version using fast undeploy, even if pending orders exist, use the following batch script:

```
fastUndeploy.bat config\config.xml force -n namespace -v version
```

where *namespace* and *version* are the OSM namespace and cartridge version.

### About Purging Orders

OSM provides the following ways to remove orders:

- XMLIE order purge - You can purge orders using criteria such as one or more order states, purge before date, order source, order type, namespace, version, start date, and stop date using XMLIE order purge.
- Dropping old partitions that contain completed orders.

---

**Note:** Oracle recommends dropping old partitions that contain completed orders as the best way to purge orders (see "Partition-Based Order Purge Strategy" for more information). If you cannot use this method because of pending orders, you can use the XMLIE order purge script as a slower alternative.

---

### Purging Orders with the orderPurge.bat Script on Windows

You can configure the `orderPurge.bat` script to:

- purge orders based on one or more order states
- purge orders from a specific cartridge (namespace/version) or all cartridges
- purge orders in a specified range of IDs, such as a particular partition
- schedule the purge to run during off-peak hours or run it immediately

Use the following commands to perform the indicated order purge operation.

1. Use the following commands to run an immediate order purge:

```
orderPurge.bat xmlConfigFile doPurge "purge_before=before_purge" "order_states=order_states" "namespace=namespace" "version=version" "order_type=order_type" "order_source=order_source" "commitCount=commitCount" "orderIdLessThan=orderIdLessThan" "orderIdGreaterOrEq=orderIdGreaterOrEq"
```
About Purging MetaData and Data

"parallelism=parallelism"

---

**Note:** All date parameters must be specified in the format:  
yyyy-mm-ddThh24:mi:ss

For example: 2011-06-28T13:39:00 EST

---

where:

- **before_purge:** Use this data parameter with the **order_state** parameter. For example, to purge orders completed 30 days ago, specify **order_state="COMPLETED"** and **purge_before=2011-06-28T13:39:00 EST** (or a date that is 30 days before the current date). Options are:
  - **all:** All orders that were created before this date are considered for the purge
  - **any closed state:** All orders whose completion date is before this date are considered for the purge.
  - **any open state:** All orders that were created and transitioned to the state specified before this date are considered for the purge.

If no **purge_before** date is specified, the date is set to 5 seconds before the purge starts.

- **order_states:** An order state must be specified. Options are one or more of the following comma separated values:
  - **all**
  - **open**
  - **not_running**
  - **running**
  - **not_started**
  - **suspended**
  - **cancelled**
  - **compensating**
  - **amending**
  - **cancelling**
  - **closed**
  - **completed**

For example: "not_started,completed"

- **namespace:** Must be specified. Valid values are either a namespace mnemonic or ALL (applies to all cartridges). For example, to purge all orders regardless of other conditions, specify **order_state="ALL"** and **namespace="ALL"**.

- **version:** If namespace is ALL, version is ignored. If namespace is specified but no version is specified, the purge applies to all versions of the namespace.

- **order_type:** The order type mnemonic. If specified, only orders with this type are purged.
About Purging MetaData and Data

Using the XML Import/Export Application

16-23

- **order_source**: The order source mnemonic. If specified, only orders with this source are purged.

- **commitCount**: The number of orders to purge before committing the transaction to the database. If specified, this can improve the performance of the order purge by breaking it up into smaller transactions.

- **orderIdLessThan**: (Optional) Limits the purge to the range of order IDs that are less than the specified order ID, such as a partition upper bound.

- **orderIdGreaterOrEq**: (Optional) Limits the purge to the range of order IDs that are greater than or equal to the specified order ID. For example, you can use it in conjunction with **orderIdLessThan** to limit the purge to a specific partition.

- **parallelism**: (Optional) The degree of parallelism. If the value is 1, orders are purged serially. If the value is greater than 1, orders are purged in parallel using database jobs.

For example:

```
orderPurge.bat config/config.xml doPurge "purge_before=2011-01-01T23:59:59 EST" "order_states=COMPLETED,NOT_STARTED" "namespace=abc" "version=1.0" "order_type=x" "order_source=y" commitCount=10 "orderIdLessThan=5400" "orderIdGreaterOrEq=1560" "parallelism=4"
```

2. Use the following syntax to run a scheduled order purge:

---

**Note:** The syntax for the scheduled order purge is identical to the immediate order purge except for the **start_date** and **stop_date** attributes.

---

```
orderPurge.bat xmlConfigFile doPurge "purge_before=before_purge" "order_states=order_states" "namespace=namespace" "version=version" "order_type=order_type" "order_source=order_source" commitCount=commitCount "orderIdLessThan=orderIdLessThan" "orderIdGreaterOrEq=orderIdGreaterOrEq" "parallelism=parallelism" "stop_date=stop_date" "start_date=start_date"
```

where:

- **stop_date**: The time when the purge should stop, even if all orders satisfying the purge criteria have been purged (e.g., stop the purge before peak hours). If no stop_date is specified, the purge stops when all orders satisfying the purge criteria have been purged.

- **start_date**: For scheduled purges only - the time when the purge should start (must be later than the current time). When the start_date is reached, the purge starts automatically. If no start_date is specified, the purge is immediate.

For example:

```
orderPurge ./config/config.xml schedulePurge "purge_before=2011-01-01T23:59:59 EST" "order_states=COMPLETED,NOT_STARTED" "namespace=abc" "version=1.0" "order_type=x" "order_source=y" commitCount=10 "orderIdLessThan=5400" "orderIdGreaterOrEq=1560" "parallelism=4" "start_date=2007-01-01T23:59:59 EST"
```

3. Use the following syntax to list all scheduled order purges that have not started:

```
orderPurge.bat xmlConfigFile listPurges
```

For example:

```
orderPurge.bat ./config/config.xml listPurges
```
4. Use the following syntax to remove an order purge that has not started:

```
orderPurge.bat xmlConfigFile removePurge *job_id*
```

where `job_id` is the job ID of the scheduled purge.

For example:

```
orderPurge.bat config/config.xml removePurge *job_id=12345*
```

Running Ant with the orderPurge.xml file On UNIX or Linux to Purge Orders

To purge orders from an OSM schema using Ant:

1. Open the `OSM_home/SDK/XMLImportExport/build.properties` file.

2. To perform an immediate purge of some or all orders before a certain date using the `immediateOrderPurge` Ant `purge` command attribute, edit the following arguments:

```
xmli.orderPurge.purgeBefore=before_purge
xmli.orderPurge.orderStates=status
xmli.orderPurge.namespace=namespace
xmli.orderPurge.version=version
xmli.orderPurge.orderType=order_type
xmli.orderPurge.orderSource=order_source
xmli.orderPurge.commitCount=commitCount
xmli.orderPurge.orderIdLessThan=orderIdLessThan
xmli.orderPurge.orderIdGreaterOrEq=orderIdGreaterOrEq
xmli.orderPurge.parallelism=parallelism
```

where

- `before_purge`: Use this data parameter with the `order_state` parameter. For example, to purge orders completed 30 days ago, specify `order_state="COMPLETED"` and `purge_before=2011-06-28T13:39:00 EST` (or a date that is 30 days before the current date). Options are:
  - **all**: All orders that were created before this date are considered for the purge
  - **any closed state**: All orders whose completion date is before this date are considered for the purge.
  - **any open state**: All orders that were created and transitioned to the state specified before this date are considered for the purge.

If no `purge_before` date is specified, the date is set to 5 seconds before the purge starts.

- `order_states`: An order state must be specified. Options are one or more of the following comma separated values:
  - all
  - open
  - not_running
  - running
  - not_started
  - suspended
About Purging MetaData and Data Using the XML Import/Export Application

- cancelled
- compensating
- amending
- cancelling
- closed
- completed

For example: "not_started,completed"

- **namespace**: Must be specified. Valid values are either a namespace mnemonic or ALL (applies to all cartridges). For example, to purge all orders regardless of other conditions, specify `order_state="ALL"` and `namespace="ALL"`.

- **version**: If namespace is ALL, version is ignored. If namespace is specified but no version is specified, the purge applies to all versions of the namespace.

- **order_type**: The order type mnemonic. If specified, only orders with this type are purged.

- **order_source**: The order source mnemonic. If specified, only orders with this source are purged.

- **commitCount**: The number of orders to purge before committing the transaction to the database. If specified, this can improve the performance of the order purge by breaking it up into smaller transactions.

- **orderIdLessThan**: (Optional) Limits the purge to the range of order IDs that are less than the specified order ID, such as a partition upper bound.

- **orderIdGreaterOrEq**: (Optional) Limits the purge to the range of order IDs that are greater than or equal to the specified order ID. For example, you can use it in conjunction with `orderIdLessThan` to limit the purge to a specific partition.

- **parallelism**: (Optional) The degree of parallelism. If the value is 1, orders are purged serially. If the value is greater than 1, orders are purged in parallel using database jobs.

For example:

```xml
 orderStates=COMPLETED
 namespace=default
 version=1.0.0.0.0
 orderType=ot
 orderSource=os
 commitCount=10
 orderIdLessThan=5400
 orderIdGreaterOrEq=1560
 parallelism=4
 />
```

3. To schedule a purge of some or all orders on a certain date using the `scheduleOrderPurge` Ant purge command attribute, edit the following additional arguments:

```xml
<xmlie.orderPurge startDate=start_date
 stopDate=stop_date
 />
```

where:
About Purging MetaData and Data

- **stop_date**: The time when the purge should stop, even if all orders satisfying the purge criteria have been purged (e.g., stop the purge before peak hours). If no stop_date is specified, the purge stops when all orders satisfying the purge criteria have been purged.

- **start_date**: For scheduled purges only - the time when the purge should start (must be later than the current time). When the start_date is reached, the purge starts automatically. If no start_date is specified, the purge is immediate.

For example:

```xml
orderStates=COMPLETED
namespace=default
version=1.0.0.0.0
orderType=ot
orderSource=os
commitCount=10
eventIdLessThan=5400
eventIdGreaterOrEq=1560
parallelism=4
startDate=2007-01-01T00:01:01 EST
stopDate=2007-12-31T23:59:59 EST
```

4. To remove an scheduled order purge that has not started using the **removeOrderPurge** Ant purge command attribute, edit the following arguments:

```xml
<xmlie.orderPurge jobId=job_id
```

where **job_id** is the job ID of the scheduled purge.

For example:

```xml
<xmlie.orderPurge jobId=0
```

---

**Note**: Orders that satisfy the purge criteria are purged and related details such as the number of orders purged are logged in the XMLIE log file (as configured in config.xml). If an error occurs, the purge stops. Errors and exceptions are output to the command line and are logged in the log files.

5. Use the following syntax to run an immediate order purge:

```bash
ant immediateOrderPurge
```

6. Use the following syntax to run a scheduled order purge:

```bash
ant scheduleOrderPurge
```

7. Use the following syntax to list all scheduled order purges that have not started:

```bash
ant listOrderPurges
```

8. Use the following syntax to remove an order purge that has not started:

```bash
ant removeOrderPurge
```
About Migrating Orders

XMLIE provides the **migrate** Ant command and a **migrate.bat** script to migrate (copy) orders from one version of a cartridge to another. You can only migrate orders between versions of the same namespace, not between different namespaces. You can also migrate order data, reference numbers, remarks, attachments and specify a single order type/source to migrate, or all order types/sources within the cartridge.

For example, you may need to migrate orders within a cartridge when upgrading to a new version of OSM. Migrated orders are placed at the first process task, regardless of where the original order was in the process flow.

You cannot migrate orders across environments, for example, from a production environment to a test environment.

XMLIE migrates orders in three steps:

1. Create a new order and copy data from the source order to the target order.

   All order field data is copied provided the field has a value in the source order and the field is defined in the target order’s creation task data. Order reference numbers and remarks/attachments are also copied if **copyReference** and **copyRemarks** are set to true. The migrated order is placed at the first process task, not at the equivalent task in the original order.

2. Once the copy is complete, close the source order (**closeSource** set to true).

   **Note:** If you choose to close the source order, the Exception Processing function must be associated with your workgroup.

3. Submit the target order (**submitTarget** set to true).

   Orders are migrated individually. If an error occurs in any of these steps and **errorAction** is set to abort, processing stops immediately. Prior steps are not rolled back. If **errorAction** is set to ignore, any remaining steps are skipped and the process starts over at step 1 with the next available order to be migrated.

   **Note:** Order migration should only be done within a window where no other order processing will occur.

It is extremely important that the target order creation task data contain all of the fields in the source order. The fields must be the same data type and have the same mnemonic and length to be considered equal. Any field that exists in the source order but not in the target creation task data is ignored. Fields that are defined in the target creation task data that have no associated data in the source order remain blank.

The most common failure scenarios are:

- The specified target version or type/source combination does not exist. This is a fatal exception because no orders can be migrated. Processing halts immediately regardless of the **errorAction** setting.

- The target order does not submit because the target creation task data contains mandatory fields that were not set in the source order.

### Configuring and Running an Order Migration

To configure and run an order migration:
1. Configure the config.xml file migration and userInteraction elements.

```xml
<migrate submitTarget="submitTarget" closeSource="closeSource"
copyReference="copyReference" copyRemarks="copyRemarks"
errorAction="errorAction"/>
<userInteraction confirmation="confirmation"/>
```

where:
- **submitTarget**: Options are:
  - `true`: Submits the target order following migration. (default)
  - `false`: Leaves the target order in the creation task.
- **closeSource**: Options are:
  - `true`: Closes the source order following migration. If you choose to close the source order, the Exception Processing function must be associated with your workgroup. (default)
  - `false`: Leaves the source order unchanged by the migration operation.
- **copyReference**: Options are:
  - `true`: Sets the order reference number of the target order to that of the source order. (default)
  - `false`: Leaves the target order reference number empty.
- **copyRemarks**: Options are:
  - `true`: Copies the source order remarks and attachments to the target order. (default)
  - `false`: Does not copy remarks and attachments.
- **errorAction**: Options are:
  - `abort`: Stops processing immediately. (default)
  - `ignore`: Attempts to migrate the next available order.
- **confirmation**: Used for validation while migrating, so if there are any warnings/errors it might ask user for confirmation. Options are:
  - `true`: Confirms the warning/error message if any.
  - `false`: Does not confirm the warning/error messages if any.

2. If you are using batch scripts, use the following syntax to migrate an order from one version of a cartridge to another version of the same cartridge:

```bash
migrate.bat config/config.xml -sourcenamespace namespace -sourceversion version -sourceordertype type -sourceordersource source -targetversion version
```

where:
- **namespace**: Must be specified. Valid values are the namespace mnemonic.
- **version**: Must be specified. The versions of the source namespace and the target namespace.
- **type**: The order type mnemonic. If specified, only orders with this type are migrated.
- **source**: The order source name. If specified, only orders with this source are migrated.
For example:

migrate.bat config/config.xml -sourcenamespace default -sourceversion 1.0 -targetversion 2.0

and

migrate.bat config/config.xml -sourcenamespace default -sourceversion 1.0 -sourceordertype request for long distance -sourceordersource client care -targetversion 2.0

3. If you are using the Ant migration command, do the following:
   a. Open the OSM_home/SDK/XMLImportExport/build.properties file.
   b. Edit the following arguments:
      
      xmlie.root.namespace=namespace
      xmlie.root.version=version
      xmlie.root.sourceordertype=type
      xmlie.root.sourceordersource=source
      xmlie.root.targetorderversion=version

      where:
      
      – namespace: Must be specified. Valid values are the namespace mnemonic.
      – version: Must be specified. The versions of the source namespace and the target namespace.
      – type: The order type mnemonic. If specified, only orders with this type are migrated.
      – source: The order source name. If specified, only orders with this source are migrated.

      For example:

      xmlie.root.namespace=bb_ocm_demo
      xmlie.root.version=1.0.0.0.0
      xmlie.root.sourceordertype=Add Order
      xmlie.root.sourceordersource=Add Order
      xmlie.root.targetorderversion=1.0.0.0.1

   c. Use the following syntax start a migration from one version of a cartridge to another version of a cartridge:

      ant migrate

About Validating the Metadata Model and Data

XMLIE provides the validate command, which is used to assure metadata model validity for an OSM schema. Using the validate command, you can perform the following:

- **Model validation**: The XML schema model defines most entity relationships and constraints.

- **Pending order validation**: This validation is applied during the import process (import.bat). It performs validations that have not been covered by validate.bat.
Note: When you perform a validation, you must supply a well-formed model, otherwise you may encounter undefined exceptions.

The application applies an XML schema validation as well as some additional rules that could not be implemented in the XML schema. You can use any XML application to validate the model against the schema, but for additional validations, the explicit validation (using validate.bat) or implicit validation (before import or after export), assures model correctness and completeness.

You can validate the metadata model before the import takes place, or change config.xml so that the validation takes place during the import.

Configuring and Running an XML Document Validation

To configure and run a validation:

1. Configure the config.xml file.

   ```xml
   <validation validateAgainstDB="validateAgainstDB" validateDocument="validateDocument" validationReportURI="filename_path"/>
   ```

   where:

   - `validateAgainstDB`: Options are:
     - true: Validates the XML document against existing orders in the database schema to ensure it is compatible. (default)
     - false: Does not perform an XML model validation against the database schema.

   - `validateDocument`: Options are:
     - true: Validates the XML document against the OSM XML schema to ensure it is well formed. (default)
     - false: Does not perform the XML document validation.

   - `filename_path` is the path to the validation log file that includes the file name of the validation log file.

2. Do one of the following:

   a. If you are using Ant commands, do the following:
      ```
      ant validate
      ```

   b. If you are using batch scripts, do the following:
      ```
      validate.bat xmlModelFile config\config.xml
      ```

Validating an XML Document During the Import or Export Process

To validate an XML document during the import or export process, set the configuration file `validateModel` parameter to true. See "Configuring and Running an Import" and "Configuring and Running an Export" for syntax information and examples.
About Creating a Graphical Representation of the Metadata Model

XMLIE provides the `modeldoc` command, which is used to convert an OSM metadata XML document into an HTML presentation. The HTML presentation is a graphical representation of the metadata model which is much easier to understand and navigate than the metadata XML document. You can use the resulting HTML presentation to view the relationships between and dependencies of the various metadata elements using standard HTML presentation navigation methods.

If you are going to use the `modeldoc.bat` script or the Ant `htmlModel` command to create HTML representations of cartridge metadata, some configuration needs to be set up in order for GraphViz to support these commands and scripts.

Configuring and Creating a Graphical Representation of a Metadata Model

To configure and create a graphical representation of a metadata model:

1. Download and install GraphViz. For more information see OSM Installation Guide.
2. Update the `OSM_home\SDK\XMLImportExport\config.bat` script with an environment variable containing the path to the third-party GraphViz library.  
   For example:
   ```
   set GRAPHVIZ=C:\Applications\ATT\Graphviz\bin
   ```
3. If you are using the `OSM_home\SDK\XMLImportExport\modeldoc.bat` script, update the Java command line in the `modeldoc.bat` file with an entry referring to the environment variable containing the path to the third-party GraphViz library:
   ```
   %JAVA_HOME%/bin/java %JAVA_OPTS% -classpath %CLASSPATH% %APP_PROPERTIES% com.mslv.oms.metadatahandler.operation.ModelDocOperator %XML_MODEL% %GRAPHVIZ% ./modeldoc
   ```
4. If you are using the Ant `htmlModel` command, configure the `graphiz` property in the `OSM_home\SDK\XMLImportExport\build.xml` file to specify the directory where the third-party GraphViz software is installed. For example:
   ```
   <property name="graphiz" value="./bin/ATT/Graphviz/bin"/>
   ```
5. If you are using batch scripts, run the following script:
   ```
   modeldoc.bat xmlModelFile HTMLModelDirectoryPath
   ```
   where `HTMLModelDirectoryPath`: Specifies the path to the directory for the HTML model files for the `modeldoc.bat` script.
6. If you are using the Ant `htmlModel` command, run the following command:
   ```
   ant htmlModel
   ```

Viewing the Graphical Representation

To view the HTML presentation, open the resulting HTML index file and begin navigating through the metadata model using the automatically generated hyperlinks.

**Note:** To view the HTML presentation, your browser must support Adobe SVG Viewer.
This appendix presents information about the tools that are available for performance testing, tuning, and troubleshooting for your Oracle Communications Order and Service Management (OSM) system.

**WebLogic Server Administration Console**

Oracle WebLogic Server Administration Console is a web-based, graphical user interface that allows you to configure, manage, and monitor WebLogic server.

For more information, see "About Monitoring and Managing OSM" and WebLogic Server documentation.

**Java VisualVM**

Java VisualVM is a Java virtual machine (JVM) troubleshooting tool that integrates the functionality of several JDK tools, including JConsole, JStack, and JMap. VisualVM helps generate and analyze thread dumps and heap dumps, track down memory leaks, perform and monitor garbage collection, and do lightweight memory and CPU profiling.

Java VisualVM is part of the JDK. For example, on Linux systems, you can start VisualVM using the following command:

```
JDK_home/bin/jvisualvm
```

For more information about Java VisualVM, see the following website:

[http://visualvm.java.net/](http://visualvm.java.net/)

**JMap**

Use the JMap utility to generate a Java heap dump. You can use VisualVM to view the heap dumps that you generate using JMap. Ensure that there is adequate space in the directory where you want to save the heap dump.

JMap is part of the JDK. For example, on Linux systems, JMap can generate a heap dump to the file `heap_dump.out` using the following command:

```
JDK_home/bin/jmap -dump:format=b processid
```
**JStack**

Use the JStack utility to generate a thread dump. You can use VisualVM to view the thread dumps that you generate using JStack. Ensure that there is adequate space in the directory where you want to save the thread dump.

JStack is part of the JDK. For example, on Linux systems, you can start JStack using the following command:

```
JDK_home/bin/jstack -l processid
```

**Note:** The advantage of using JStack instead of VisualVM to generate thread dumps is that JStack can be scripted. For example, the attached `jstackscript.sh` shell script generates `count` thread dumps every `delay` seconds.

**OSM Task Web Client**

The OSM Task Web Client provides a web-based user interface for order tracking and reporting. For more information, see *OSM Task Web Client User’s Guide*.

**Enterprise Manager**

Oracle Enterprise Manager is a web-based management tool for database performance diagnostics and tuning.

There are two versions of Enterprise Manager that you can use to manage Oracle 11g databases:

- **Database Control**: Manages a single database, including Oracle RAC databases, and is installed with the Oracle database.

- **Grid Control**: Manages multiple databases and offers enhanced Oracle RAC support. This is a separately licensed product. You can also use Grid Control to manage WebLogic Server and coherence.

For more information, see Enterprise Manager documentation.

**SoapUI**

SoapUI is a tool that you can use to submit an XML order to a run-time OSM environment. Doing this confirms that OSM is able to receive and respond to order requests. In this case, you can submit test orders associated with a test cartridge.

When submitting sample orders to run-time environments, the root level of the sample order XML document must be either a `CreateOrder` or `CreateOrderBySpec` request.

For more information about submitting orders with SoapUI, see *OSM Developer’s Guide*.

For more information about SoapUI, and to download the software, see the following website:

**Compliance Tool**

The Oracle Communications Product Compliance Tool captures configuration data for OSM and its platform components. The tool then evaluates product compliance against documented configuration requirements, best practices, and guidelines.

The compliance tool uses a set of compliance rules to determine if a configuration value is properly set or, if it allows a range of valid values, whether the configured value falls within that range. The tool also verifies that required or recommended patches have been applied.

For every compliance rule, reports include a description of the rule, an indication of whether the rule passed or failed, and the rationale for the compliance rule. For non-compliant results, a severity level and the reason for the failure are also included.

For more information about the compliance tool, see "Evaluating System Configuration Compliance".

**Design Studio**

Design Studio is an Eclipse-based design environment for OSM cartridge development. The Cartridge Management view displays a list of the cartridges that are deployed to your OSM system. The Deployed Versions table lists which cartridge version and build combination is currently deployed in the target environment for the selected cartridge.

For more information, see Design Studio Concepts.

**Software Load Balancer**

Oracle HTTP Server is the recommended standalone software load balancer for OSM in production environment.

For more information about setting up an HTTP server for OSM, see the knowledge article [Doc ID 1618630.1], available from the Oracle Support Web site:

https://support.oracle.com

For a simpler alternative, you can install the WebLogic proxy plug-in for OSM managed servers and use it for load balancing for performance testing.

**GCViewer**

GCViewer is a free open source tool to view data that is produced by verbose garbage collection. For more information about GCViewer versions and to download, see the following websites:

http://www.tagtraum.com/gcviewer.html

https://github.com/chewiebug/GCViewer

http://sourceforge.net/projects/gcviewer/files/gcviewer-1.33.jar/download

https://github.com/chewiebug/GCViewer/wiki/Changelog

**ThreadLogic**

ThreadLogic is an open source visual thread dump analyzer that provides an in-depth analysis of WebLogic Server thread dumps. ThreadLogic can also merge and analyze
multiple thread dumps and provide enhanced reporting that lets you view whether threads are progressing across thread dumps.

For more information about ThreadLogic, and to download the software, see the following website:
https://java.net/projects/threadlogic

You run ThreadLogic, version 1.1.205, using the following command:
java -jar ThreadLogic-1.1.205.jar

You can open the thread dumps or merge thread dumps by selecting several and then right-clicking and choosing Diff Selection. For an overview of ThreadLogic, see the following website:

**OSW Black Box**

OSW Black Box is an Oracle utility that provides platform-independent, real-time analysis of a large volume of operating system data. OSW Black Box runs as a background process and collects data at 30-second intervals, logging the last 48 hours of data to archive files.

OSW Black Box has the following components:

- **OSWbb**: a shell script data collector.
- **OSWbba**: a Java utility that analyzes data collected by OSWbb.

You generate a text-based report that you can attach to service requests. This report also provides recommendations about resolving problems. The System Status section of the report gives a summary status for CPU, memory, input/output, and net subsystems. Statuses include: Critical, Warning, OK, or Unknown.

For more information about OSW Black Box and the analyzer tool, see the knowledge articles [Doc ID 301137.1 and Doc ID 461053.1], available from the Oracle Support Web site:
https://support.oracle.com

To run OSW Black Box as a background process:

1. Run the following command:
   ```bash
   nohup ./startOSWbb.sh &
   ```

To stop OSW Black Box processes:

1. Run the following command:
   ```bash
   ./stopOSWbb.sh
   ```

To generate an OSW Black Box Analyzer report:

1. Run the following command:
   ```bash
   java -jar oswbba.jar -i oswbb_archive_directory -A
   ```
Remote Diagnostics Agent

Oracle Remote Diagnostics Agent (RDA) provides a suite of data collection and diagnostic scripts for Oracle products. RDA reports are packaged as an archive, which you upload and attach to a service request.

While you can use RDA only when a problem occurs, Oracle recommends that you use RDA diagnostics regularly to help prevent problems. RDA reports include both system configuration and performance data. Taking regular RDA snapshots of the performance of your system under normal conditions might help you troubleshoot a faulty or degraded system.

RDA is installed with Oracle Fusion Middleware. Oracle recommends that you define the RDA_HOME environment variable as MW_home/oracle_common/rda. RDA is written in Perl and uses a command-line script.

Set up RDA to generate reports about the operating system, WebLogic server, database, Oracle RAC cluster, or OSM.

To verify that RDA is installed on your system:

1. On the command line, do one of the following:
   - For UNIX, run the following command:
     
     perl rda.pl -cv
   
   - For Linux, run the following command:
     
     rda.sh -cv

   For information about what to do if RDA is not installed in your environment, see the knowledge article [Doc ID 314422.1], available from the Oracle Support Web site:

     https://support.oracle.com

Operating System RDA Report

You set up RDA to generate an operating system report, and then you generate and package the report.

To set up RDA to generate an operating system report:

1. On the command line, run the following command:

   rda.sh -S CONFIG OS PROF PERF NET

   The setup files are saved to the current directory.

To generate and package the RDA operating system report:

1. On the command line, run the following command:

   rda.sh -vCRP

   The report is saved in the directory from which you ran the RDA command. An output folder and an output RDA_output_hostname.zip archive are created.

To view the RDA operating system report:

1. Go to the RDA output folder.

2. Double-click the RDA_start.htm file.

   The report opens in a web browser.
WebLogic RDA Report

You set up and generate a WebLogic RDA report for every WebLogic server in your OSM system.

To set up and generate WebLogic RDA report:

1. On the command line, run the following command:

   `$RDA_HOME/rda.sh -s wls_report -p WebLogicServer`

   The report is saved in the directory from which you ran the RDA command. A `wls_report` folder and an `RDA_wls_report_hostname.zip` archive are created.

Database RDA Report

To set up and generate a database RDA report:

1. On the command line, run the following command:

   `$RDA_HOME/rda.sh -s db_report -p DB11g`

   The report is saved in the directory from which you ran the RDA command. A `db_report` folder and an `RDA_db_report_hostname.zip` archive are created.

Oracle RAC Cluster RDA Report

The Oracle RAC Cluster RDA report includes log files, init files, and diagnostic files, as well as your network and cluster configurations.

For more information about Oracle RAC Cluster RDA reports, see the knowledge article [Doc ID 359395.1], available from the Oracle Support Web site:

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

To set up an Oracle RAC Cluster RDA report:

1. On the command line, run the following command:

   `$RDA_HOME/rda.sh -p Rac`

   The data collected from each Oracle RAC node is packaged in a separate ZIP file. The ZIP files are listed in the Remote Data Collection, Collected Data section of the RDA report.

OSM RDA Report

For a clustered environment, run this report on every host.

For more information about using RDA reports with OSM and other Oracle Communications applications, see the knowledge article [Doc ID 1057563.1], available from the Oracle Support Web site:

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

To set up and generate an OSM RDA report:

1. On the command line, run the following command:

   `$RDA_HOME/rda.sh -s osm_report -p Com_OSM`

2. Run the following command:

   `$RDA_HOME/rda.sh -s osm_report`
The report is saved in the directory from which you ran the RDA command. An osm_report folder and an RDA_osm_report_hostname.zip archive are created.