# Oracle® Communications Order and Service Management

## System Administrator's Guide

Release 8.0
G38003-01
October 2025

Oracle Communications Order and Service Management System Administrator's Guide, Release 8.0

G38003-01

Copyright © 2009, 2025, Oracle and/or its affiliates.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software, software documentation, data (as defined in the Federal Acquisition Regulation), or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs) and Oracle computer documentation or other Oracle data delivered to or accessed by U.S. Government end users are "commercial computer software," "commercial computer software documentation," or "limited rights data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, reproduction, duplication, release, display, disclosure, modification, preparation of derivative works, and/or adaptation of i) Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs), ii) Oracle computer documentation and/or iii) other Oracle data, is subject to the rights and limitations specified in the license contained in the applicable contract. The terms governing the U.S. Government's use of Oracle cloud services are defined by the applicable contract for such services. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

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

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

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

# Contents


About This Content

1 OSM System Administration Overview

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

2 Starting and Stopping OSM

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

3 Changing GUI Application Appearance and Functionality

About Configuring the User Experience 1
- About Behaviors 1

Changing Web Client Appearance and Functionality 2
- Changing the Default Timeout Setting 2
- Setting Table Height 2
- Configuring the Data View Performance in the Order Management Web Client 3
- Configuring the Display Size of Text Fields 3
- Setting Default Action on Orders and Tasks in the Task Web Client 4
- Customizing the Appearance of Read-Only Text Fields 5

4 Setting Up OSM Security

About OSM Security 1
- Secure Solution Data Storage 1

Adding Users to OSM 2
- Adding Users to Groups in the WebLogic Remote Console 2




Copyright © 2009, 2025, Oracle and/or its affiliates.

Creating Workgroups as Roles in Design Studio 2
Assigning Users to Workgroups 4
Adding Users to WebLogic Server Security Realms 4
Using Secure Sockets Layer 5
Changing Secure Sockets Layer Configuration in OSM 5
Using WebLogic Server Authenticators with OSM 5
User-Level Authenticator Support 6
Group-Level Authenticator Support 6
Authenticator User and Group Assignment Considerations 6
Setting Up a Caching Realm 7
Setting Up OSM for Single Sign-On (SSO) and Single Logout (SLO) Using SAML2 7
Secure Credential Management 7
Using the EncryptPasswords Utility 8
About the EncryptPasswords Utility 8
Running the EncryptPasswords Utility 8
Removing an Encrypted Password 9
Using the CreateEncryptPasswords Utility 9
Before Running the CreateEncryptPasswords Utility 10
Running the CreateEncryptPasswords Utility 10
Using the Credential Store 11
About the Credential Store 11
How OSM Retrieves Credentials from the Credential Store 11
Configuring the Java Security Policy for the OSM Credential Store 12
Configuring the Java Security Policy Using the credStoreAdmin Command 13
Setting Up Email Notifications Using the credStoreAdmin Command 13
Managing Credentials in the Credential Store 14
Developing Cartridges to Use the Credential Store 15
Developing Automation Plug-ins to Use the Credential Store 15
Defining Data Providers in OSM Cartridges to Use the Credential Store 16
Using the Credential Store with Custom Data Providers 16
Using the Credential Store with Built-In Data Providers 17
Upgrading Existing Cartridge Code to Use the Credential Store 17
Using Built-in SOAP Adapter as a Data Provider Class 17
Using the XML Import/Export Application to Administer Users and Workgroups 18

## 5 Configuring OSM with oms-config.xml

Working with oms-config.xml 1
oms-config.xml Parameters 2
Configuring Operational Order Jeopardies with Configuration Files 24


Copyright © 2009, 2025, Oracle and/or its affiliates.

## 6 Configuring the Task Processor


About Configuring the Task Processor 1
Configuring the Task Processor for Performance 1


## 7 Managing the OSM Database Schema


Implementing a Strategy for OSM Information Lifecycle Management 1
Creating Tablespaces 2
Using Partitioning 4
- Benefits of Partitioning 5
  - Improved Manageability 5
  - Increased Availability 5
  - Increased Concurrency 5
  - Support for Active-Active Oracle RAC 6
  - Increased Query Performance 8
- Pitfalls of Partitioning 8
  - Order Search Performance 8
  - Purge Performance 9
  - Shared Pool 9
  - Development and Testing Environments 9

Order Purge Strategies 9
- Partition-Based Order Purge Strategy 10
  - Partition Purge Example 10
  - Advantages and Disadvantages of Partition-Based Purge 13
- Row-Based Order Purge Strategy 14
  - Row-Based Order Purge Example 15
  - Advantages and Disadvantages of Row-Based Order Purge 15
- Hybrid Purge Strategy 16

Partitioning Realms 16
- Partitioning Realms Configuration 17
  - Mapping Orders to Partitioning Realms 19
  - Enabling and Disabling Partitioning Realms 21
  - Renaming a Partitioning Realm 22
  - Refreshing Partitioning Realms Configuration 22
- Adding Partitions for New Partitioning Realms 22
  - partition_auto_creation Disabled 23
  - partition_auto_creation Enabled 23
- Purge Strategy for Partitioning Realms 23
- Default Partitioning Realm 24
- Non-Partitioned Schemas 24
- Order ID Blocks 24




Copyright © 2009, 2025, Oracle and/or its affiliates.

Cartridge Management Strategy 25
Sizing Partitions 26
Sizing Hash Sub-Partitions 26
Sizing Range Partitions for Partition-Based Order Purge 27
Purge Performance 27
Estimating Storage 28
"All-In" Order Volume 30
Partition Size Restrictions 30
Retention Policy 31
Time-to-Close Wait 31
Oracle RAC 33
Purge Frequency 34
Sizing Range Partitions for Row-Based Order Purge 35
Sizing Range Partitions for Zero Downtime 36
Sizing Range Partitions for Infrequent Maintenance 37
Online vs. Offline Maintenance 38
Managing Order Data 39
Adding Partitions (Online or Offline) 39
Using Row-Based Order Purge 40
Purging a Single Order by Order ID 42
Purging Orders that Satisfy Given Criteria 42
Scheduling Order Purge 43
Stopping and Resuming an Order Purge 43
Using Partition-Based Order Purge 43
Differences Between Purging and Dropping Partitions 43
Purging Partitions (Online or Offline) 44
Purging Entire Partitions That Do Not Contain Open Orders (Online or Offline) 45
Purging Partitions Partially (Offline Only) 46
Dropping Partitions (Offline Only) 50
Dropping Empty Partitions (Online or Offline) 51
Reclaiming Unused Space in Volatile Tables 51
Order Purge Policies 52
Purging Related Orders Independently 52
Auditing and Monitoring Order Purges 53
Audit Tables 54
Managing Exchange Tables for Partition-Based Order Purge 55
About OSM Purge Tables 55
About OSM Backup Tables 56
Creating Exchange Tables (Online or Offline) 57
Purging Exchange Tables (Online or Offline) 57
Dropping Exchange Tables (Online or Offline) 57
Estimating Partition Disk Space (Online or Offline) 57


Copyright © 2009, 2025, Oracle and/or its affiliates.

Managing Cartridges 58
Using Fast Undeploy 59
Purging Metadata of Undeployed Cartridges 59
Configuration Parameters 60
range_partition_size 61
subpartitions_number 61
default_xchg_capacity 61
xchg_retained_orders_thres 61
degree_of_parallelism 62
degree_of_parallelism_rebuild_indexes 62
degree_of_parallelism_rebuild_xchg_indexes 62
purge_job_class 62
parallel_execute_chunk_size 63
partition_auto_creation 63
purge_policy_rebuild_unusable_indexes 63
purge_policy_purge_related_orders_independently 64
purge_policy_consolidate_partitions 64
purge_policy_time_to_close_wait 64
purge_audit_retention_days 66
deferred_segment_creation 66
purge_commit_count 66
About PL/SQL API 66
DBMS Output 66
Specifying Purge Criteria 67
Parallel Execution 69
Concurrency Restrictions 69
PL/SQL API Reference 70
Setup and Tuning Procedures 70
om_part_maintain.setup_xchg_tables (Online or Offline) 70
om_part_maintain.drop_xchg_tables (Online or Offline) 71
om_part_maintain.set_dop (Online or Offline) 71
om_part_maintain.set_dop_rebuild_indexes (Online or Offline) 71
om_part_maintain.set_dop_rebuild_xchg_indexes (Online or Offline) 72
Maintenance Procedures and Functions 72
om_part_maintain.add_partition (Offline Only) 72
om_part_maintain.add_partitions (Offline Only) 72
om_part_maintain.drop_partitions (Offline only) 73
om_part_maintain.drop_empty_partitions (Online or Offline) 74
om_part_maintain.purge_partitions (Online or Offline) 75
om_part_maintain.purge_partitions_2 (Online or Offline) 79
om_part_maintain.purge_entire_partition (Online or Offline) 81
om_part_maintain.estimate_ptn_purged_space (Online or Offline) 81


Copyright © 2009, 2025, Oracle and/or its affiliates.

om_part_maintain.purge_xchg_bck_tables (Online or Offline) 82
om_part_maintain.purge_xchg_prg_tables (Online or Offline) 82
om_new_purge_pkg.delete_order (Online or Offline) 82
om_new_purge_pkg.purge_orders (Online or Offline) 82
om_new_purge_pkg.schedule_order_purge_job (Online or Offline) 84
om_new_purge_pkg.select_orders (Online or Offline) 85
om_new_purge_pkg.purge_selected_orders (Online or Offline) 86
om_new_purge_pkg.stop_purge (Online or Offline) 86
om_new_purge_pkg.resume_purge (Online or Offline) 86
om_handler_pkg.abort_order 87
Advanced Procedures 88
om_part_maintain.backup_selected_ords (Offline) 88
om_part_maintain.restore_orders (Offline) 89
Troubleshooting Functions 89
om_part_maintain.get_partitions (Online or Offline) 89
om_part_maintain.is_equipartitioned (Online or Offline) 89
Recovery Procedures 90
om_part_maintain.equipartition (Offline only) 90
om_part_maintain.purge_orphan_order_data (Online or Offline) 90
om_part_maintain.rebuild_unusable_indexes (Online or Offline) 90
om_part_maintain.rebuild_index (Online or Offline) 91
om_part_maintain.sys$undo_restore_table (Offline) 92
om_part_maintain.sys$undo_restore_orders (Offline) 92
Database Reference 92
Database Views 92
OM_AUDIT_PURGE_ALL 92
OM_AUDIT_PURGE_LATEST 94
Database Tables 94
OM_AUDIT_PURGE 94
OM_AUDIT_PURGE_ORDER 96
OM_AUDIT_PURGE_PARAM 96
Troubleshooting and Error Handling 97
Error Handling for add_partitions 98
Error Handling for drop_partitions 99
Error Handling for purge_partitions 99
Troubleshooting 99
Error Handling 101
Error Handling for rebuild_unusable_indexes 102
Error Handing for setup_xchg_tables 102
Performance Tuning 102
Tuning degree_of_parallelism 103
Tuning degree_of_parallelism_rebuild_indexes 103


Copyright © 2009, 2025, Oracle and/or its affiliates.

Tuning degree_of_parallelism_rebuild_xchg_indexes 103
Tuning Parallel Job Execution 103
Tuning parallel_execute_chunk_size 104
Tuning Row-Based Purge 105
Database Terms 105

## 8 Managing Optimizer Statistics

About Optimizer Statistics 1
Gathering Optimizer Statistics 1
Gathering Statistics Online 1
Automated Optimizer Statistics Collection 1
Gathering Fixed Object Statistics 2
Gathering System Statistics 3
Gathering Cartridge Metamodel Statistics 3
Gathering Order Statistics 3
High Volatility Order Tables 3
Low Volatility Order Tables 4
Medium Volatility Order Tables 4
Enabling Incremental Statistics 5
Gathering High-Volatility-Table Statistics 5
Gathering Low-Volatility-Table Statistics 6
Preparing a New Partition 6
Populating New Partition Statistics 6
Using Statistics from Another Partition 6
Using Statistics from a Statistics Table 8
Using Statistics from Another System 8
Using Default Statistics Package 8
Locking Partition Statistics 10
Configuring a Partition When It Is No Longer the Active Partition 10
Optimizer Statistics Error Handling Using Datapump 10
Optimizer Statistics Management Performance Tuning 11
Using Parallel Collection for Gathering Statistics 11
Cursor Invalidations 11
Optimizer Statistics Management PL/SQL API Reference 12
Setup and Tuning Procedures 12
om_db_stats_pkg.lock_volatile_order_stats 12
om_db_stats_pkg.unlock_volatile_order_stats 12
om_db_stats_pkg.set_table_prefs_incremental 12
om_db_stats_pkg.set_table_volatility 12
Maintenance Procedures 13
om_db_stats_pkg.gather_cartridge_stats 13


Copyright © 2009, 2025, Oracle and/or its affiliates.

om_db_stats_pkg.gather_order_stats 13
om_db_stats_pkg.gather_volatile_order_stats 13
om_db_stats_pkg.copy_order_ptn_stats 13
om_db_stats_pkg.lock_order_ptn_stats 14
om_db_stats_pkg.unlock_order_ptn_stats 14
Advanced Procedures 14
om_db_stats_pkg.export_order_ptn_stats 15
om_db_stats_pkg.import_order_ptn_stats 15
om_db_stats_pkg.expdp_order_ptn_stats 16
om_db_stats_pkg.impdp_order_ptn_stats 16
Troubleshooting Procedures 17
om_db_stats_pkg.lstj_copy_order_ptn_stats 17
om_db_stats_pkg.get_order_ptn_stats 17
om_db_stats_pkg.list_order_ptn_stats 17
om_db_stats_pkg.check_order_ptn_stats 17
Recovery Procedures 18
om_db_stats_pkg.remj_copy_order_ptn_stats 18

## 9 Backing Up and Restoring OSM Files and Data

About Backing Up and Restoring OSM Files and Data 1
Backup and Restore Overview 1
Backup and Restore Schedule 1
OSM_CFG_Home Directory 1
Oracle Database 1
WebLogic Server Files 1
Backup and Restore Considerations 2
Backing Up and Restoring the OSM Files 2
Backing Up the OSM Files 2
Restoring the OSM Files 2
Oracle Database Backup Considerations 2
RMAN Considerations 2
Oracle Flashback Technology Considerations 3
Mirroring Technology Considerations 3
Backing Up and Restoring the WebLogic Server Configuration 3
Backing Up the WebLogic Server Configuration 4
Before You Back Up the WebLogic Server 4
Setting OSM to Pause Processing of JMS Messages 4
WebLogic Server Domain Directory 5
WebLogic Persistent Store 5
OSM Attachment Directory 5
External Deployments 5


Copyright © 2009, 2025, Oracle and/or its affiliates.

Remote Managed Servers (Clustered Server Only) 6
Restoring the WebLogic Server Configuration 6
Restoring the WebLogic Server Files 6
Setting OSM to Resume Processing of JMS Messages 6

## 10 Monitoring and Managing OSM

About Monitoring and Managing OSM 1
About Monitoring OSM Using WebLogic Remote Console 1
Accessing the WebLogic Remote Console 2
Using the WebLogic Console to Determine the Status of the OSM Application 2
Using the WebLogic Scripting Tool 3
Refreshing OSM Metadata 4
Monitoring and Analyzing Performance 4
Monitoring Performance Using WebLogic Remote Console 4
Monitoring the Managed Server 5
Analyzing Garbage Collection 5
Analyzing Class Loading 5
Coherence Datagram Testing 6
Monitoring WebLogic Server 6
Monitoring the Operating System 6
Gathering OSM Execution Statistics 7
Analyzing Heap Dumps 7
Analyzing Thread Dumps 8
Monitoring the Database 8
Updating Database Schema Statistics 9
Managing Log Files 9
Managing Log Files Using Enterprise Manager Fusion Middleware Control 9
Viewing Log Files and Messages Using Fusion Middleware Control 10
Configuring Logs Using Fusion Middleware Control 10
Managing Log Files Using WebLogic Remote Console 10
Viewing Log Files 11
Log Size and Rotation 11
About Log Severity Levels 11
Configuring Log Levels Using the logging.xml File 12
Managing Logs Using WLST 12
Configuring Log Levels Using WLST 13
Secure vs Non-Secure Log Filtering 13
Managing OSM Metrics 14
Configuring Prometheus for OSM Metrics 14
Viewing OSM Metrics Without Using Prometheus 14
Viewing OSM Metrics in Grafana 15


Copyright © 2009, 2025, Oracle and/or its affiliates.

Exposed Order Metrics 15
Managing WebLogic Server Metrics 17
Deploying WebLogic Monitoring Exporter (WME) in OSM 18
Configuring the Prometheus Scrape Job for WME Metrics 19
Viewing WebLogic Monitoring Exporter (WME) Metrics Without Using Prometheus 20
Viewing WebLogic Monitoring Exporter Metrics in Grafana 20
Managing Database Connections 20
Using JMS Queues to Send Messages 20
Monitoring the Event Queue 21
Sending Data to External Systems Using Plug-Ins 22
About OSM and XA Support 22
Using Work Managers to Prioritize Work 23
Creating and Configuring Work Managers 23
Overriding the Internet Explorer Language in the OMS Web Clients 24
About Metrics Data 25
Viewing Metrics Data 25
About DMS Metrics Tables 26
About WebLogic Metrics Tables 26
About Non-J2EE Metrics Tables 26
About Aggregated Metrics Tables 26
About Metric Rules Files 27
Analyzing Metric Data 27
Discovering a Cluster Database Target 28

## 11 Exporting and Importing OSM Schema Data

About Exporting and Importing OSM Schema Data 1
Exporting and Importing the OSM Model Data Only 1
Exporting OSM Model Data 2
Preparing the Target OSM Schema Before Import 3
Creating the Target OSM Schema 4
Adding Partitions 4
Importing OSM Model Data 5
Exporting and Importing the OSM Model and a Single Order 6
Exporting OSM Order Data 6
Preparing to Export Order Tables for a Single Order 6
Exporting Order Tables That Define an Order Sequence ID 7
Exporting the OSM Model Data 8
Importing the OSM Model and Order Data 8
Exporting and Importing a Range of Orders and the OSM Model 9
Exporting the OSM Order Data Range 9
Preparing to Export Order Tables for a Range of Orders 9


Copyright © 2009, 2025, Oracle and/or its affiliates.

Exporting Order Tables That Define an Order Sequence ID 10
Exporting the OSM Model Data 11
Importing OSM Model and Order Data 11
Exporting and Importing a Range of OSM Orders Only 12
Exporting an Additional Range of Orders 13
Preparing to Export Order Tables for a Range of Orders 13
Exporting a Range of Orders from Order Tables That Define an Order Sequence ID 14
Importing an Additional Range of Orders 15
About Order Export Queries 16
Changing PAR File Parameters 16
About Import Parameters 18
Troubleshooting Export/Import 19

## 12 Configuring Time Zone Settings

Configuring Time Zone Settings 1

## 13 Troubleshooting OSM

Information You Need for Troubleshooting 1
Running Scripts to Automatically Gather Troubleshooting Information 1
About the Scripts 1
Prerequisites for Running the Scripts 2
Running the Scripts 2
getSRInfo.sh 3
gatherWLSData.sh 3
gatherDBData.sh 4
General Checklist for Resolving Problems 4
Diagnosing Some Common Problems with OSM 5
Cannot Log in or Access Certain Functionality 5
System Appears Slow 5
Error: "Java.lang.StackOverflowError" when Using Task Web Client 5
Coherence Configuration Error: [STUCK] ExecuteThread 6
Unexpected Logout from Web Client 6
Error: "Login failed. Please try again." 6
Automation Plug-ins Are Not Getting Called 6
Delayed JMS Messages 7
Error Message For Events From a JMS Topic in a Cluster 7
JMS Message Delivery Failure 7
Unexpected Values for JMS Properties 7
Too Many Open Files 8
Problems When Running Multiple WebLogic Domains on One Host 8


Copyright © 2009, 2025, Oracle and/or its affiliates.

Proxy Fails on a Clustered System 8
Unable to Bring Up Managed Server After Database Failure 8
Orders Are Not Being Created on a Clustered System 8
JBoss Cache Timeouts 8
OSM Fails to Process Orders Because of Metadata Errors 9
Error: "No Backend Servers Available" 9
DataDictionary Expansion Level 9
Quick Fix Button Active During Order Template Conflicts in Design Studio 9
Cannot Create New Orders on a New Cartridge Version 9
Error: "exact fetch returns more than requested number of rows" 10
Error: "unique constraint violated" 10
Exceptions When Purging is in Progress 10
Error: "Ignoring partition" 11
Unexpected Logout from Landing Page 11
Getting Help with OSM Problems 11
Before You Contact Support 11
Reporting Problems 11

14 OSM Log Messages

OSM Catalog Messages 1
Automation Catalog Messages 19

15 Using the XML Import/Export Application

About the XML Import/Export Application 1
About Using the XML Import/Export Application 1
Deploying Cartridges to the OSM Database Using XMLIE 2
About XML Import/Export Batch Scripts and Ant Commands 3
About XML Import/Export Ant Commands and Syntax 3
About XML Import/Export Batch Scripts and Syntax 4
Configuring the XML Import/Export Environment Files 5
Configuring the build.properties File for Ant Commands 5
Configuring the config.bat Script for Batch Scripts 6
Configuring the config.xml File XML Import/Export Nodes and Elements 6
Using SSL Connections 8
About Importing and Exporting Metadata 9
About Exporting Metadata 10
About the Order of Exported Metadata 10
About Export Layout Options 11
Keeping the ID Integrity in SQL Rules 13
Configuring and Running an Export 14


Copyright © 2009, 2025, Oracle and/or its affiliates.

About Importing Metadata 15
Configuring and Running an Import 16
Sample Procedure for Adding a New Workgroup Definition (Role) 17
Sample Procedure for Adding a Task to a Workgroup (Role) 19
About Purging MetaData and Data 20
Undeploying Cartridges and Purging the Database Schema 21
About Purging Orders 23
Purging Orders with the orderPurge.bat Script on Windows 23
Running Ant with the orderPurge.xml file On UNIX or Linux to Purge Orders 26
About Migrating Orders 29
Configuring and Running an Order Migration 30
About Validating the Metadata Model and Data 31
Configuring and Running an XML Document Validation 32
Validating an XML Document During the Import or Export Process 32
About Creating a Graphical Representation of the Metadata Model 33
Configuring and Creating a Graphical Representation of a Metadata Model 33
Viewing the Graphical Representation 34

## A OSM Credential Store API Command Reference

OSM User Security and Credential Store Commands A-1
userAdmin Command A-2
credStoreAdmin Command A-8
OSM User Security and Credential Store API Reference Material A-14
CredStore A-14
PasswordCredStore A-15
CredStoreException A-17
SoapAdapter A-18
ObjectelHTTPAdapter A-19
ViewRuleContext A-20
AutomationContext A-22

## B Tools for Performance Testing, Tuning, and Troubleshooting

WebLogic Remote Console B-1
Java VisualVM B-1
JStack B-1
OSM Task Web Client B-2
Enterprise Manager B-2
SoapUI B-2
Compliance Tool B-2
Design Studio B-3


Copyright © 2009, 2025, Oracle and/or its affiliates.

Software Load Balancer B-3
GCViewer B-3
ThreadLogic B-3
OSW Black Box B-4
Remote Diagnostics Agent B-4
Operating System RDA Report B-5
WebLogic RDA Report B-5
Database RDA Report B-5
Oracle RAC Cluster RDA Report B-5
OSM RDA Report B-6

## C OSM Installed Components

OSM Directory Structure C-1
Productized Cartridges C-2
WebLogic Installed Components C-2
WebLogic Deployments C-2
WebLogic Configuration C-2
Coherence Clusters C-2
Work Managers C-2
JMS Servers C-3
JMS Module C-3
Queues and Topics C-3
Quotas C-4
Connection Factories C-5
Destination Key C-5
JMS Template C-5
File T3 C-6
Data Sources C-6
Users and Groups C-6
Database Configuration C-8


Copyright © 2009, 2025, Oracle and/or its affiliates.

# About This Content

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 a secure manner. This guide assumes that users have a working knowledge of the relevant operating system, Oracle Database, Oracle WebLogic Server, and Java J2EE software.

### Documentation Accessibility

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

### Access to Oracle Support

Oracle customers 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.

### Conventions

The following text conventions are used in this document.

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


Copyright © 2009, 2025, Oracle and/or its affiliates.

1

# 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 task processor. See "Configuring the Task Processor" 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 "Exporting and Importing OSM Schema Data " for more information.

## About OSM System Administration Programs

Use the following programs for OSM system administration:

- Use the Oracle WebLogic Remote 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


Copyright © 2009, 2025, Oracle and/or its affiliates.

> **Note**
>
> Although these functions can be done using the XML Import/Export application, Oracle Communications Design Studio is the preferred application for performing 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 1-1 Placeholders Used in This Guide**

| Placeholder | Directory Description |
|---|---|
| *OSM_SDK* | This is the directory where the **SDK** artifact is extracted. The SDK is available as a seperate downloadable zip file. |
| *MW_home* | The location where Oracle Fusion Middleware was installed. This directory contains the base directory for WebLogic Server and the **user_projects** directory, among other files and directories. |
| *WLS_home* | The base directory for the WebLogic Server core files. It is located in the *MW_home* directory, for example, *MW_home***/wlserver**. |
| *domain_home* | 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. |


Copyright © 2009, 2025, Oracle and/or its affiliates.

2

# Starting and Stopping OSM

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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

# 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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

3

# Changing GUI Application Appearance and Functionality

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 Service Catalog and Design - 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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

- **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* for more information about behaviors.

You can use Design Studio to define additional behaviors.

# Changing Web Client Appearance and Functionality

You can change the behavior of the Order Management web client and Task web client in several ways.

## Changing the Default Timeout Setting

To change the default timeout setting:

1. Unpack the **oms.ear** file. See *OSM Developer's Guide* for more information.
2. Open the *OSM_SDK***/SDK/Customization/osmwebui/WEB-INF/web.xml** file.
3. Search for the following parameter:

   ```
   session-timeout
   ```

4. Change the value of the **session-timeout** parameter to your desired value (in minutes).
5. Save and close the file.
6. Repack **oms.ear**. See *OSM Developer's Guide* for more information.

> **Note**
>
> Changing the **session-timeout** parameter changes the automatic timeout for both the Order Management web client and the Task web client.

## Setting Table Height

You can specify a fixed or variable table height, depending on the value of the **is_tablelayout_height_fixed** parameter in the **oms-config.xml** file. You can set the value of **is_tablelayout_height_fixed** to **True** or to **False**. If **is_tablelayout_height_fixed** is set to **True**, you can also set the **height_of_tablelayout** parameter to specify the fixed height for tables.

The default value of **is_tablelayout_height_fixed** is **True**.


Copyright © 2009, 2025, Oracle and/or its affiliates.

Using fixed table heights can create excessive white space in and between tables. Setting **is_tablelayout_height_fixed** to **False** causes the table height to be solely determined by the number of rows in the table and can help eliminate white space.

To set the table height:

1. Access the oms-config.xml file. See "Working with oms-config.xml" for more information.
2. Set the desired value for the **is_tablelayout_height_fixed** parameter by doing one of the following:
   - To set the table height to fixed, set **is_tablelayout_height_fixed** to **True**.

     You may desire a fixed height that is different from the default. If so, change the fixed height of all tables.

     To change the fixed height of all tables, set the **height_of_tablelayout** parameter to the desired value in pixels. The default is 400 pixels.
   - To set the table height to variable, set **is_tablelayout_height_fixed** to **False**.

     Table height will now vary for each table, depending on the number of rows in the table.
3. Save and close the file.

# Configuring the Data View Performance in the Order Management Web Client

You can change the performance of the data view in the Order Management web client using parameters in the **osm-config.xml** file.

To set the data view performance parameters:

1. Access the oms-config.xml file. See "Working with oms-config.xml" for more information.
2. To set the database fetch limit, which sets a limit to the number of rows that can be requested at one time, set the **jdbc_fetch_size** parameter to the desired number of rows.
3. To set the limit for the number of instances a multi-instance data element may have before its display is forced into table format, set the **oracle.communications.ordermanagement.table-layout.threshold** parameter to the desired maximum number of instances.
4. Save and close the file.

# Configuring the Display Size of Text Fields

You can control the size of the displayed area for text fields, and whether text fields will have scroll bars, using the **no_of_rows_in_textarea_without_scroll** and **max_no_of_rows_in_textarea_with_scroll** parameters in the **oms-config.xml** file.

The **no_of_rows_in_textarea_without_scroll** parameter indicates the maximum number of rows that the field can accommodate without a scroll bar.

The **max_no_of_rows_in_textarea_with_scroll** parameter indicates the maximum number of rows that the field can accommodate if the number of rows exceeds the value of the **no_of_rows_in_textarea_without_scroll** parameter.

However, the scroll bar disappears and the field displays all the rows, if their number exceeds the value of the **max_no_of_rows_in_textarea_with_scroll** parameter.


Copyright © 2009, 2025, Oracle and/or its affiliates.

For example:

- If the value of **no_of_rows_in_textarea_without_scroll** is greater than the value of **max_no_of_rows_in_textarea_with_scroll**, and the number of rows is less than or equal to the value of **no_of_rows_in_textarea_without_scroll**, the field displays all the rows without a scroll bar.
- If the value of **no_of_rows_in_textarea_without_scroll** is greater than the value of **max_no_of_rows_in_textarea_with_scroll**, and the number of rows is greater than the value of **no_of_rows_in_textarea_without_scroll**, the field displays in its text area, rows whose number equals the value of the **no_of_rows_in_textarea_without_scroll** parameter. However, the field accommodates all the other rows and provides a scroll bar.
- If the value of **no_of_rows_in_textarea_without_scroll** is less than the value of **max_no_of_rows_in_textarea_with_scroll**, and the number of rows is less than or equal to the value of **max_no_of_rows_in_textarea_with_scroll**, the field accommodates all the rows and provides a scroll bar.
- If the value of **no_of_rows_in_textarea_without_scroll** is less than the value of **max_no_of_rows_in_textarea_with_scroll**, and the number of rows is greater than the value of **max_no_of_rows_in_textarea_with_scroll**, the field displays all the rows in its text area. In this case, the scroll bar is not displayed.

## Setting Default Action on Orders and Tasks in the Task Web Client

You can specify a particular action as the default action to perform on orders and tasks in the Worklist and Query screens for all users who have not set the default action in the Options page of the Task web client. You can achieve this by configuring a parameter in the **oms-config.xml** file. However, users can change the global default action configured by the administrator to an action of their choice by setting the **Default Action** field in the Options page. By default, the global default action parameter is not set to any action and the application considers the option set by individual users as the default action to perform.

To set a default action on orders and tasks for all users:

1. Access the **oms-config.xml** file. See "Working with oms-config.xml" for more information.
2. Set the **DefaultUserAction** parameter to the desired action.

   Valid values (actions) that you can specify are:

   - NoGlobalDefault (This is the default if no action is set.)
   - AcceptEditTask
   - AddRemark
   - RaiseException
   - ViewOrderData
   - ViewOrderProcessHistory
   - ViewNotificationHistory
   - CopyOrder

   If no action is set for the parameter, the application considers the option (action) that is set by individual users in the Default Action field in the Options screen.
3. Save and close the file.


Copyright © 2009, 2025, Oracle and/or its affiliates.

# Customizing the Appearance of Read-Only Text Fields

You can customize the appearance of the text (field labels). You can change the font style, size, color, and background color. You can customize specific text or all text across the pages.

You can customize specific text in the pages by using VF Style Rules. With VF Style Rules, you can modify the CSS properties (font style, color, and background) for any specific field. When you define a VF style rule for a particular read-only field, only that particular field is customized.

To customize all read-only text:

1. Unpack the **oms.ear** file.
2. Open the **SDK/Customization/oms-war/customScreen.css** file.
3. Add the **oeValueNodeReadOnly** parameter and specify the properties as shown below:

```
.oeValueNodeReadOnly {
FONT-SIZE: 10pt;
color:blue;
}
```

4. Run **PackOMS** to generate the new **oms.ear** file and deploy it to the WebLogic domain.

You can also customize the appearance of read-only fields by using reusable class definitions. With reusable class definitions, you can apply some properties to some text and a different set of properties to the other text fields. For example, you can apply a 10pt font size and blue color to some fields, and a 14pt font style and red color to the other fields.

To customize text fields by using reusable class definitions:

1. Create a list of CSS classes. Specify all the properties for each class as shown below and add the classes to the **customScreen.css** file.

```
.customReadOnlyValueBlue {
FONT-SIZE: 10pt;
color:blue;
}

.customReadOnlyValueRed {
FONT-SIZE: 14pt;
color:red;
}
```

2. In Design Studio, add the Style behavior for the read-only fields that you want to customize:

   For example:

   - For the fields that you want to apply blue, add **customReadOnlyValueBlue** in the **CSS Class Name** field of the **Value** grouping.
   - For the fields that you want to apply red, add **customReadOnlyValueRed** in the **CSS Class Name** field of the **Value** grouping.

3. Update the **oms.ear** file with the modified **customScreen.css** file and deploy the modified cartridges.


Copyright © 2009, 2025, Oracle and/or its affiliates.

4

# Setting Up OSM Security

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 Remote 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"
- Use WebLogic Server's LDAP or another authenticator that is integrated with WebLogic Server. See "Using WebLogic Server Authenticators with OSM."

> **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 credentials securely using the EncryptPasswords utility or the Oracle Fusion Middleware Credential Store Framework (CSF). See "Secure Credential Management."
- Administer users and workgroups Using XML documents. See "Using the XML Import/Export Application to Administer Users and Workgroups."
- Manage workgroups. See *OSM Order Management Web Client User's Guide* for more information.

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

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

## Secure Solution Data Storage

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, 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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

# Adding Users to OSM

In WebLogic Server, a security realm consists of a set of configured security providers, users, groups, security roles, and security policies that protect WebLogic resources. To access WebLogic resources that belong a security realm, a user must be defined in that realm.

To add a user to OSM:

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

For information about the users and groups created by the OSM installer, see "Users and Groups."

## Adding Users to Groups in the WebLogic Remote Console

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

To add users to groups:

1. Log in to the WebLogic Remote Console.

   You must be a WebLogic administrator.
2. Click on **Security Data Tree** and then click on **Realms**.

   The **Summary of Realms** page is displayed.
3. Click the name of your security realm. The default name is **myrealm**.

   The settings for the security realm are displayed.
4. Navigate from **Authentication providers** to **DefaultAuthenticator**, and from here to **Users**.

   A list of users that have been configured is displayed.
5. Click on a user.

   The user's description is displayed. The **Password** and **Membership** tabs are available here. Users are assigned to one or more parent groups that have different levels of access to WebLogic Server resources, depending on their roles and the tasks they can perform. Groups in theWebLogic Server security realm represents the roles.
6. On the page that displays the settings for the selected user, click the **Membership** 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 "Users and Groups" for more information about the groups.
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 OSM Web Service and XML APIs.


Copyright © 2009, 2025, Oracle and/or its affiliates.

> **Note**
>
> You assign individual users to roles using the Administration area of the Order Management web client. See *OSM Order Management Web Client User's Guide* for more information.

Table 4-1 describes the client functions to which you provide access.

**Table 4-1 Client Permissions**

| Function | Description | Applies To |
|---|---|---|
| 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.<br>This permission relates to:<br>• In the Task web client: creating a new order<br>• In the Web Service API: using the CreateOrderBySpecification calls<br>• In the XML API: using the CreateOrder XML API call | Task web client<br>Web Service API<br>XML API |
| Exception Processing | Enables users to alter the flow of a process by applying exception statuses at any time throughout the process.<br>This permission relates to:<br>• In the Task web client: raising exceptions on a task<br>• In the XML API: using the SetException call | Task web client<br>XML API |
| Online Reports | Enables users to view summarized reports on all orders and tasks on the system.<br>This permission relates to:<br>• 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.<br>This permission relates to:<br>• In the Order Management web client: setting the order priority<br>• In the Task web client: setting the task priority<br>• In the Web Service API: changing the order priority using the UpdateOrder call<br>• In the XML API: changing the order or task priority using the UpdateOrder call | Order Management web client<br>Task web client<br>Web Service API<br>XML API |
| Reference Number Modification | Enables users to modify the reference number of an order.<br>This permission relates to:<br>• In the Order Management web client and Task web client: Modifying the reference number of an order<br>• In the Web Service API and XML API: changing the reference number using the UpdateOrder call | Order Management web client<br>Task web client<br>Web Service API<br>XML API |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 4-1 (Cont.) Client Permissions**

| Function | Description | Applies To |
|---|---|---|
| Search View | Enables users to access the order Query function.<br>This permission relates to:<br>• In the Order Management web client: using the main search screen for the client<br>• In the Task web client: using the Search functionality (for example, clicking the Query button from the Worklist)<br>• In the Web Service API: using the FindOrder call<br>• In the XML API: using the Query call | Order Management web client<br>Task web client<br>Web Service API<br>XML API |
| Task Assignment | Enables users to assign tasks to others.<br>This permission relates to:<br>• In the Task web client: assigning a task to another user<br>• In the XML API: using the AssignOrder calls | Task web client<br>XML API |
| Worklist Viewer | Enables users to access the Worklist function.<br>This permission relates to:<br>• In the Task web client: accessing the worklist<br>• In the XML API: using the Worklist call | Task web client<br>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 *Design Studio Modeling OSM Processes* for more information. After you create a role, you must assign permissions to the role entities. See "Role Editor Role Tab" in *Design Studio Modeling OSM Processes* for more information about permissions for role entities.

## Assigning Users to Workgroups

See the discussion about assigning users to a workgroup in *OSM Order Management Web Client User's Guide* for more information.

## Adding Users to WebLogic Server Security Realms

You can add users to WebLogic Server security realms using the Remote console or Oracle WebLogic Scripting Tool commands provided by WebLogic Server, or you can use the userAdmin command. You should add users to a security realm after they have been added to the credential store. For more information about adding users to the credential store, see "Managing Credentials in the Credential Store." However, if you do add a user using userAdmin command and the user has not already been added to the credential store, the userAdmin command will also add the user to the credential store in the **osm** map.

To add users to WebLogic Server security realms using the userAdmin command, see "Using the userAdmin Command." Following is a sample XML data file for adding users:

```
<userConfig xmlns="http://www.metasolv.com/Provisioning/UserConfig"
 xmlns:oms="http://www.metasolv.com/OMS/OrderModel/2002/06/25"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <user name="testOsmUser1">
      <description>OSM test user 1</description >
      <password>password_from_createEncryptPasswords</password>
      <saltstore>/u01/security/testOsmUser1/salt.store</saltstore>
   </user>
```


Copyright © 2009, 2025, Oracle and/or its affiliates.

```
   <user name="testOsmUser2">
      <description>OSM test user 2</description >
      <password>password_from_createEncryptPasswords</password>
      <saltstore>/u01/security/testOsmUser2/salt.store</saltstore>
   </user>
</userConfig>
```

# Using Secure Sockets Layer

OSM supports the following 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.

## Changing Secure Sockets Layer Configuration in OSM

You configure SSL for OSM using the XML parameter file **web.xml** which is stored in the **oms.war** file inside the **oms.ear** file. To edit the **web.xml** file, you must unpack the **oms.ear** file, edit the **web.xml** file, and then repack the **oms.ear** file and re-deploy it to the OSM server. For more information about unpacking and repacking the **oms.ear** file, see *OSM Developer's Guide*.

To change SSL configuration on the OSM server:

1. Locate and extract the **web.xml** file from the **oms.ear/oms.war** file.
2. Add or modify the parameter **secure_login** in the **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.

# Using WebLogic Server Authenticators with OSM

OSM supports using either the WebLogic Server's embedded LDAP directory or another authenticator that is external to, and integrated with 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 Server, all the authentication providers (even if they are configured as optional in WebLogic Server) should be active. If any of the authentication providers is not active, an exception will be raised and the users will not be able to log in to OSM.


Copyright © 2009, 2025, Oracle and/or its affiliates.

## 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 Server. Also, users can be assigned to workgroups in the Administration area of the OSM Order Management web client 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 Server. Also, groups can be assigned to workgroups in the Administration area of the OSM Order Management web client 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 Remote 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 that 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 Server.
- Only users and groups that have been assigned directly to the OMS_client or OSM_automation groups in WebLogic Server will be available to assign to workgroups in the Administration area of the OSM Order Management web client. For example, if UserA1 is a member of GroupA, and GroupA has been assigned to the OMS_client group in WebLogic Server, only GroupA will be displayed in the OSM Order Management web client. 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 Order Management web client. There is no indication which elements are users and which are groups.


Copyright © 2009, 2025, Oracle and/or its affiliates.

- If new users are added to the authenticator, they will not be able to access OSM functionality until the OSM Metadata is refreshed. For information about refreshing OSM metadata, see "Refreshing OSM Metadata".
- Oracle recommends that you assign automation users to the OSM_automation WebLogic Server 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 Server group any users that are specified in the "Run As" property of an automation plug-in.

# 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 Remote Console.
2. In the **Edit Tree**, select **Security**, and from here select **Realms**.

   The Summary of Security Realms page is displayed.
3. Click the name of your security realm. The default name is **myrealm**.

   The settings for the security realm are displayed.

   From this window, you can change the settings for your realm.

For more information about setting up security in WebLogic Server, see *Oracle Fusion Middleware Administering Security for Oracle WebLogic Server*.

# Setting Up OSM for Single Sign-On (SSO) and Single Logout (SLO) Using SAML2

You can set up OSM for Single Sign-on (SSO) and Single Logout (SLO) using SAML2.

See *OSM Security Guide* for detailed information.

# Secure Credential Management

This section describes how to secure credentials for accessing external systems. OSM provides two distinct secure credential management solutions, each appropriate to the type of credential to be secured:

- **EncryptPasswords utility**: This utility is used to secure the credentials required to run other OSM utilities that require those credentials, for example the XML Import/Export tool. Oracle recommends that you use this if you are running utilities unattended. If you are running the utilities attended, Oracle recommends that you provide the required credentials interactively as prompted by the utility, rather than using the EncryptPasswords utility. Because the utility is assumed to run unattended, it must be able to decrypt credentials without user intervention. Therefore, the output of the EncryptPasswords utility should be considered obfuscated and not encrypted. Secure the files containing the output with


Copyright © 2009, 2025, Oracle and/or its affiliates.

appropriate file-system-level restrictions. For more information, see "Using the EncryptPasswords Utility."

- **Credential Store**: This utility is used to secure credentials required to access systems with which your OSM solution interacts. It builds on CSF, adding OSM-specific features. For more information, see "Using the Credential Store."

# Using the EncryptPasswords Utility

You use the **EncryptPasswords** utility to encrypt the user name and password credentials of XML Import/Export application users.

## About the EncryptPasswords Utility

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 about running the utility.

The **EncryptPasswords** utility (located in the **SDK/XMLImportExport** directory) is a password management utility that secures the credentials that the XML Import/Export application uses. The **EncryptPasswords** utility encrypts the credentials, preventing their accidental exposure.

Running 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. It then stores the user names and passwords for these users in encrypted format in the configuration file which the XML Import/Export application uses to provide these credentials (for example, **SDK/XMLImportExport/config/config.xml**). When the XML Import/Export application runs, it decrypts the passwords as part of 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.

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

- **SDK/CartridgeManagementTool/production**
- **SDK/CartridgeManagementTool/development**

The Ant build files have targets corresponding to each of the batch files in the **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

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


Copyright © 2009, 2025, Oracle and/or its affiliates.

To run the EncryptPasswords utility:

1. Create a config file that complies with the **SDK/XMLImportExport/config/config.xsd** file.

   The same directory also contains a sample file: **config_sample.xml**.

   > **Note**
   >
   > The configuration file must contain the <user> and <password> elements, but the values of these elements do not matter because the script will prompt for these values.

2. Do one of the following:

   - **UNIX and Linux:**

     Run the following command:

     ```
     EncryptPasswords.sh XMLConfig [-dbUser] [-xmlapiUser] [-wlsUser]
     ```

     for example:

     ```
     EncryptPasswords.sh config/config.xml -dbUser
     ```

   - **Windows:**

     Run the following command:

     ```
     EncryptPasswords XMLConfig [-dbUser] [-xmlapiUser] [-wlsUser]
     ```

     for example:

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

   where *XMLConfig* indicates the configuration XML file you created in the previous step, and the following optional parameters indicate which passwords should be encrypted:

   - **-dbUser**: the OSM database password
   - **-xmlapiUser**: the XML API interface password
   - **-wlsUser**: the WebLogic domain administration server password

   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.

3. When prompted by the script, enter the user names and passwords that you selected when running the script.

   When you type in passwords, nothing will be displayed on the screen.

## 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 the CreateEncryptPasswords Utility

You use the **CreateEncryptPasswords** utility to encrypt the user name and password credentials of users to put in XML configuration files for the userAdmin and credStoreAdmin


Copyright © 2009, 2025, Oracle and/or its affiliates.

commands. See "userAdmin Command" and "credStoreAdmin Command" for more information about these commands.

The **CreateEncryptPasswords** utility (located in the *OSM_SDK*/**SDK/XMLImportExport** directory) is a password management utility that secures the credentials being administered by the userAdmin and credStoreAdmin commands. The **CreateEncryptPasswords** utility provides an encrypted password and a separate file (a salt file) that are used together to provide password security. You put the encrypted password value and the path and name of the additional file in the XML file you are configuring for the userAdmin or credStoreAdmin command.

## Before Running the CreateEncryptPasswords Utility

Before running the CreateEncryptPasswords utility, you must create or have present a file to contain the encrypted password. This file must be located in the same directory in which you are running the utility. It can be empty, and it does not have to have a **.properties** extension. For example, you can use a file name like **password.txt**.

If you run the utility multiple times, you can store all of the passwords in the same properties file.

## Running the CreateEncryptPasswords Utility

Run the **CreateEncryptPasswords** utility script:

- for each **user** element in the **userConfig** element in the XML file for the userAdmin command. See "Creating the XML Data File for the userAdmin Command" for more information about this file.
- for each **credential** element in the **CredentialConfig** element in the XML file for the userAdmin command. See "Creating the XML Data File for the credStoreAdmin Command" for more information about this file.

To run the **CreateEncryptPasswords** utility:

1. Do one of the following:
   - **UNIX and Linux:**

     Run the following command:

     ```
     CreateEncryptPasswords.sh username password filename label
     ```

     for example:

     ```
     CreateEncryptPasswords.sh username password pass.properties pw
     ```

   - **Windows:**

     Run the following command:

     ```
     CreateEncryptPasswords username password filename label
     ```

     for example:

     ```
     CreateEncryptPasswords username password pass.properties pw
     ```

   where:

   - **username**: is the name of the user to be encrypted
   - **password**: is the plain-text password of the user
   - **filename**: is the name of the properties file in which to store the password


Copyright © 2009, 2025, Oracle and/or its affiliates.

- **label**: is the label to put on the password in the properties file. The label in the file is to differentiate between different passwords. If you reuse an existing label, the password in the file will be overwritten. If you use a new label, the new encrypted password will be added to the beginning of the file. Following is the format of a properties file containing three encrypted passwords created by the **CreateEncryptPasswords** command:

  ```
  #Wed Apr 29 10:13:24 EDT 2015
  pw3=encrypted_password_3
  pw2=encrypted_password_2
  pw=encrypted_password_1
  ```

2. After running the command, rename the file that was generated by the script. When the script is run, it creates the auxiliary file as **salt.store** in the directory from which you ran the script. Since it creates a file with the same name each time, you should rename it to keep it from being overwritten, and also so you know which user it is for. For example, you could rename it **user1_salt.store**.

# Using the Credential Store

You use the credential store to store credentials that OSM uses to interact with external systems.

## 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.

In OSM cloud native, the type of credential store that OSM uses is offered through the Kubernetes Secret.

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 API Command Reference" for more information about 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 about security options and recommendations.

For information about Platform Security Services and managing credentials in the credential store, see "*Oracle Fusion Middleware Securing Applications with Oracle Platform Security Services*."

## How OSM Retrieves Credentials from the Credential Store

The credential store in the WebLogic Server 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


Copyright © 2009, 2025, Oracle and/or its affiliates.

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 then 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 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 about 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 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.

## 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 Platform Security Services, credential store access permission is configured through the Java Platform Security policy.

When you configure the Java Platform Security 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 Java Platform Security policy using map name **osm**. 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 Java Platform Security policy using your own map name.

In a clustered environment, use the administration server URL rather than the managed server when you configure the credential store and the Java Platform Security policy.

To configure the Java Platform Security policy, you can use any of the following methods:

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


Copyright © 2009, 2025, Oracle and/or its affiliates.

- The Fusion Middleware WebLogic Scripting Tool **grantPermission** command. See *Oracle Fusion Middleware Securing Applications with Oracle Platform Security Services* for information about managing policies with the WebLogic Scripting Tool commands.
- The Fusion Middleware Control GUI. Credential store and Java Platform Security policy MBeans can be configured from this GUI. See *Oracle Fusion Middleware Administering Security for Oracle WebLogic Server* for more information. You can also verify the Java security policy configuration from this GUI.

## Configuring the Java Security Policy Using the credStoreAdmin Command

You can configure the Java Security Policy if you have already installed OSM by running the **configDomain/configOSM.sh** and have OSM_SDK zip.

To configure the Java Platform Security policy for the OSM credential store map using the credStoreAdmin command, see "Using the credStoreAdmin Command." Following is a sample XML data file for configuring the security policy:

```
<?xml version="1.0" encoding="UTF-8"?>
<CredentialConfig
 xmlns="http://www.metasolv.com/Provisioning/CredentialConfig"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="…/XMLImportExport/models/CredStoreAdmin.xsd">
   <jpsPolicy operation="add">
      <mapname>osm</mapname>
   </jpsPolicy>
</CredentialConfig>
```

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 the credStoreAdmin Target in Another Ant Script."

## Setting Up Email Notifications Using the credStoreAdmin Command

You can set up OSM to send users notifications by email. When a notification occurs, the system sends a notification ID number. For information about managing notifications, see *OSM Order Management Web Client User's Guide*.

To set up email notifications for a user using the credStoreAdmin command:

1. Add the **oms-automation** user to the workgroup that contains the user who will receive email notifications.

   The **oms-automation** user is the internal automation user account for the processing of OSM automation and email notifications.

2. Make sure that **email_server_ssl** and **email_server_authentication** parameters in the **oms-config.xml** file are set to **true**.

3. If the user's email server requires server authentication, you must store the password for the administrator's email user in OSM credential store. The administrator's email address is specified by the **admin_email_address** parameter (located in the **oms-config.xml** file).

   Run the credStoreAdmin command using the instructions in "Using the credStoreAdmin Command." Following is a sample XML data file for configuring email notifications.

   ```
   <?xml version="1.0" encoding="UTF-8"?>
   <user:CredentialConfig
   xmlns:user=" http://www.metasolv.com/Provisioning/CredentialConfig"
   xmlns:xsi=" http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation=" http://www.metasolv.com/Provisioning/CredentialConfig
   ```


Copyright © 2009, 2025, Oracle and/or its affiliates.

```
../models/CredStoreAdmin.xsd ">
   <user:jpsPolicy operation="add">
     <user:mapname>osm</user:mapname>
   </user:jpsPolicy>
   <user:credential operation="create" overwrite="true">
     <user:mapname>osm</user:mapname>
     <user:keyname>osmUser_{OSM_ADMIN_EMAIL_ADDRESS}</user:keyname>
     <user:user>OSM_admin_email_address</user:user>
     <user:password>OSM_admin_email_password</user:password>
   </user:credential>
 </user:CredentialConfig>
```

4. Delete the XML data file after you are done using it, since it contains a password.

## 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 Remote Console.

Store all OSM users in the default OSM map (**osm**) or in a custom map that you have configured into the Java security policy, 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 Securing Applications with Oracle Platform Security Services* for information about managing credentials in the credential store by using WebLogic Scripting Tool commands.
- Use the credStoreAdmin command in the XML Import/Export application. See "credStoreAdmin Command" for information about this command.

To manage credentials in the credential store using the credStoreAdmin command, see "Using the credStoreAdmin Command." Following is a sample XML data file for managing credentials:

```
<?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" overwrite="true">
      <ns2:mapname>osm</ns2:mapname>
      <ns2:keyname>osmUser_osm1</ns2:keyname>
      <ns2:user>osm1</ns2:user>
      <ns2:password>password_from_createEncryptPasswords</ns2:password>
      <ns2:saltstore>/u01/security/osm1/salt.store</ns2:saltstore>
   </ns2:credential>
   <ns2:credential operation="create" overwrite="true">
      <ns2:mapname>osm</ns2:mapname>
      <ns2:keyname>osmUser_osm2</ns2:keyname>
      <ns2:user>osm2</ns2:user>
      <ns2:password>password_from_createEncryptPasswords</ns2:password>
      <ns2:saltstore>/u01/security/osm2/salt.store</ns2:saltstore>
   </ns2:credential>
</ns2:CredentialConfig>
   <ns2:credential operation="delete">
      <ns2:mapname>osm</ns2:mapname>
```


Copyright © 2009, 2025, Oracle and/or its affiliates.

```
    <ns2:keyname>osmUser_osm3</ns2:keyname>
</ns2:credential>
```

# 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 (except 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 names and passwords 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 "ObjectelHTTPAdapter" 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 API Command Reference" for a description of the OSM credential store APIs.

See "Using 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


Copyright © 2009, 2025, Oracle and/or its affiliates.

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 for 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

You must set up a data provider class in your cartridge if it requires credential information for an external system so it can read the required credentials from the credential store.

## Using the Credential Store with Custom Data Providers

When you add a data provider class in a cartridge that requires credential information for 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 the retrieveInstance() method of the ExternalInstanceAdapter interface.

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:
>
> ```
> String password = context.getOsmCredentialPassword(username)
> ```

```
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
  }
```


Copyright © 2009, 2025, Oracle and/or its affiliates.

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

Oracle provides pre-defined or built-in data provider classes "SoapAdapter" and "ObjectelHTTPAdapter" 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.

Table 4-2 shows the parameters that are required for each adapter type.

**Table 4-2 Parameters for SoapAdapter and ObjectelHTTPAdapter**

| Adapter | Parameter Name | contextType | Default Value |
|---|---|---|---|
| SoapAdapter | oms:credentials.mapname | XPATH | myMap |
| SoapAdapter | oms:credentials.keyname | XPATH | myUser |
| ObjectelHTTPAdapter | obj:mapname | XPATH | osm |
| ObjectelHTTPAdapter | obj:keyname | XPATH | osm |

# 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 about developing cartridges to use the credential store.

In addition to upgrading your cartridge code, provision the credential store for the WebLogic Server domain for OSM applications with required credentials (see "Managing Credentials in the Credential Store") and configure the Java Platform Security policy for the WebLogic Server domain to allow OSM access to the 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 the built-in **SOAPAdapter** class 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 only the user name in the parameter. When you use your custom credential store map, automation plug-in users also 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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

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

The userAdmin command, which is part of the XML Import/Export application, lets you add users to WebLogic Server groups and OSM workgroups using an XML document. The XML document contains the user information you want to add or configure based on the *OSM_SDK***/SDK/XMLImportExport/models/UserAdmin.xsd** schema.

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.

To add users to WebLogic Server security realms using the userAdmin command, see "Using the userAdmin Command." Following is a sample XML data file for adding users:

```
<userConfig xmlns="http://www.metasolv.com/Provisioning/UserConfig"
 xmlns:oms="http://www.metasolv.com/OMS/OrderModel/2002/06/25"
 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>
   <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>
```


Copyright © 2009, 2025, Oracle and/or its affiliates.

5

# Configuring OSM with oms-config.xml

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

## Working with oms-config.xml

You can configure various OSM runtime parameters in the **oms-config.xml** file. The working copy of the **oms-config.xml** file is automatically saved in the *domain_home* directory, or you can define the directory using a start-up configuration parameter.

When you start OSM, the system checks for the OSM WebLogic domain start-up configuration parameter **-Dmslv.oms.config**, which it uses to find a valid **oms-config.xml** file. If the start-up parameter is not present, OSM searches the current OSM WebLogic domain directory, and uses the parameters from that file, if it exists.

If there is no **oms-config.xml** file in the current domain directory, OSM server automatically extracts it from the **oms.ear** file and saves it to the *domain_home* directory. OSM server then loads all parameters from that file.

> **Note**
>
> When you modify **oms-config.xml** parameters, you must ensure that you are using the working copy of the **oms-config.xml** file so that OSM server loads and applies the modified values. The working copy is defined as the following:
>
> - If the **-Dmslv.oms.config** start-up parameter is set, the working copy is the file name location defined in **-Dmslv.oms.config=***file name location*.
> - If no **-Dmslv.oms.config** start-up parameter is defined, the working copy is the **oms-config.xml** file that is saved in the *domain_home* directory.

To modify the working copy of the **oms-config.xml** file:

1. Open the **oms-config.xml** file from the directory where the file is saved, for example, *domain_home*.

   > **Note**
   >
   > Oracle recommends that you back up your **oms-config.xml** file before making any changes to it.

2. Do any of the following. For a list of parameters and their descriptions, see "oms-config.xml Parameters."
   - If the parameter you would like to set is already present in your **oms-config.xml** file, modify the value of the parameter.


Copyright © 2009, 2025, Oracle and/or its affiliates.

- If the parameter you would like to set is in Table 5-1 but is not already present in your **oms-config.xml** file, add an <oms-parameter> element for the parameter under the <oms-configuration> element. It does not matter what order the <oms-parameter> elements are in in the file.

3. Save the **oms-config.xml** file.

# oms-config.xml Parameters

Table 5-1 describes the parameters that can be configured in the **oms-config.xml** file.

**Table 5-1 Description of oms-config.xml Parameters**

| Name | Description | Type | Min | Max | Default |
|---|---|---|---|---|---|
| DefaultUserAction | The default action to perform on orders and tasks in the Worklist and Query screens for all users who have not set the default action in the Options page of the Task web client.<br>Valid values (actions) for the global default action parameter that OSM administrators can specify are:<br>• NoGlobalDefault (This is the default if no action is set.)<br>• AcceptEditTask<br>• AddRemark<br>• RaiseException<br>• ViewOrderData<br>• ViewOrderProcessHistory<br>• ViewNotificationHistory<br>• CopyOrder | string | NA | NA | NoGlobalDefault |
| file_attachment_filter_type | OSM filters the kinds of files that can be given as attachments in order remarks. This parameter indicates if the filter is a list of allowed file types (`Include`) or a list of disallowed file types (`Exclude`). | string | NA | NA | Include |
| exclude_attachment_extensions | This is a list of file types, given as a comma separated list of file extensions. If the `file_attachment_filter_type` is `Exclude`, OSM allows all kinds of files to be attached except for those in this list. Files that conform to this list are refused.<br>Internally, OSM translates this list into the file types canonically associated with each extension, and uses these file types to validate attachment requests based on magic byte analysis.<br>This parameter is ignored if the filter type is `Include`. | string | NA | NA | `<empty>`<br>Note: Leaving the value empty sets the disallowed files to `none`, thereby allowing all attachments if the filter type is `Exclude`. |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 5-1 (Cont.) Description of oms-config.xml Parameters**

| Name | Description | Type | Min | Max | Default |
|---|---|---|---|---|---|
| include_attachment_extensions | This is a list of file types, given as a comma separated list of file extensions. If the `file_attachment_filter_type` is `Include`, OSM treats this list as the only permissible attachments and all other attachments are refused.<br>Internally, OSM translates this list into the file types canonically associated with each extension, and uses these file types to validate attachment requests based on magic byte analysis.<br>This parameter is ignored if the filter type is `Exclude` | string | NA | NA | xml, json, pdf, txt<br>Note: Leaving the value empty sets the permissible files to `none`, thereby disabling all attachments if the filter type is `Include`. |
| EnableDetectAndResumeStuckComponent | This parameter when enabled will detect a stuck order due to a stuck component, and resume the component. | boolean | NA | NA | false |
| DetectAndResumeStuckComponentInterval | This parameter controls how often the detection of stuck components is run, expressed in minutes. | integer | 1 | 720 | 5 |
| Jndi.Lookup.Retry | The maximum number of retries to look up JNDI resource. Valid values include numbers. | integer | 1 | 50 | 5 |
| Jndi.Lookup.Timeout | The delay (in milliseconds) between JNDI lookup retries. | integer | 1000 | 30000 | 10000 |
| node_removal_allowed_on_redeploy | Enables removal of nodes on redeploy. | boolean | NA | NA | false |
| excel_export_row_limit | Maximum number of records (Worklist data and Query results) that can be exported to a CSV file. | integer | 1 | 100000 | 50000 |
| excel_export_cluster_limit | Maximum number or limit of total number of concurrent export requests that can be received on the database | integer | 1 | 100 | NA |
| max_worklist_rows | Maximum number of rows shown in the worklist.<br>Changing this parameter can affect system performance.<br>Oracle recommends that you not change this parameter to a value significantly larger than the default. | integer | 1 | 10000 | 200 |
| max_query_rows | Maximum number of rows returned in the Query Results page in the Task web client.<br>Changing this parameter can affect system performance.<br>Oracle recommends that you not change this parameter to a value significantly larger than the default. | integer | 1 | 10000 | 200 |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 5-1 (Cont.) Description of oms-config.xml Parameters**

| Name | Description | Type | Min | Max | Default |
|---|---|---|---|---|---|
| max_notification_rows | Maximum number notifications returned in one request in the Task web client.<br>Changing this parameter can affect system performance.<br>Oracle recommends that you not change this parameter to a value significantly larger than the default. | integer | 1 | 10000 | 200 |
| worklist_page_size | Maximum number of rows shown on each Worklist page in the Task web client. | integer | 5 | 100 | 20 |
| userlist_page_size | Maximum number of rows shown on each User List page in the Task web client. | integer | 5 | 100 | 20 |
| query_results_page_size | Maximum number of lines shown on one page in the Query Results page in the Task web client. | integer | 5 | 100 | 20 |
| notifications_page_size | Maximum number of lines shown on the Notifications page in the Task web client. | integer | 5 | 100 | 20 |
| jdbc_fetch_size | JDBC fetch size. | integer | 1 | 10000 | 100 |
| max_read_only_field_length | Maximum length of a read-only field in the Order editor. | integer | 1 | 512 | 30 |
| remark_change_timeout_hours | Time in hours after the date and time a remark is added to an order that changes are no longer allowed to the remark. | integer | -1 | 1000 | 1 |
| attachment_file_system_name | Name of the T3 file service configured in WebLogic that manages order attachments for OSM. | string | NA | NA | NA |
| max_attachment_size | Maximum size in MB of documents attached to orders in the Task web client.<br>Changing this parameter impacts OSM WebLogic host machine file system usage. | integer | 1 | 100 | 3 |
| database_timezone_offset | 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.<br>Oracle recommends that you do not change this parameter from the installation setting. | integer | -50400 | 50400 | NA |
| admin_email_address | Default email address to which to send notifications. The default value set by **oms-parameter-default** is ignored.<br>Oracle recommends that you not change this parameter from the installation setting. | string | NA | NA | NA |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 5-1 (Cont.) Description of oms-config.xml Parameters**

| Name | Description | Type | Min | Max | Default |
|---|---|---|---|---|---|
| email_server | IP address or server name of the email server used to send OSM notifications.<br>Oracle recommends that you not change this parameter from the installation setting. | string | NA | NA | 127.0.0.1 |
| email_server_port | Port number of the email server used to send OSM notifications.<br>Oracle recommends that you not change this parameter from the installation setting. | string | NA | NA | 993 |
| email_server_authentication | Enables use of SSL authentication when connecting to the email server when set to **true**. | boolean | NA | NA | true |
| email_server_ssl | The configuration is ignored. The value is hard-coded to **true**. Enables SSL connection to the email server when set to **true**. | boolean | NA | NA | true |
| url_root | The base URL for the OSM web applications. | string | NA | NA | oms |
| create_order_show_namespace | If set to **false**, no list is displayed in the Task web client to select a namespace when creating an order. | boolean | NA | NA | true |
| load_users_from_database | 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). | boolean | NA | NA | false |
| workstream_refresh_interval | In the Task web client, the number of milliseconds to wait before attempting to retrieve the next available task in a workstream.<br>Changing this parameter can affect system performance.<br>Oracle recommends that you do not change this parameter from the installation setting. | integer | 500 | 60000 | 1000 |
| max_workstream_retry | 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.<br>Changing this parameter can affect system performance.<br>Oracle recommends that you do not change this parameter from the installation setting. | integer | 1 | 1000 | 30 |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 5-1 (Cont.) Description of oms-config.xml Parameters**

| Name | Description | Type | Min | Max | Default |
|---|---|---|---|---|---|
| workstream_predefined_status_display_fixed | Controls how task completion status buttons are displayed in the workstream order editor.<br>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.<br>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. | boolean | NA | NA | true |
| hide_dirty_order | 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. | boolean | NA | NA | false |
| disable_edit_on_server_refresh | If set to **true**, if a server refresh occurs while an order is being edited, the edit function becomes disabled. | boolean | NA | NA | true |
| login_screen | Initial screen displayed when the Task web client is started. Valid values include:<br>• about<br>• default<br>• home<br>• query<br>• worklist | string | NA | NA | worklist |
| auto_logout_warning_offset_minutes | 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.<br>Oracle recommends that you do not change this parameter from the installation setting. | integer | 1 | 1440 | 5 |
| com.mslv.oms.util.xml.XMLHelper.DocumentBuilderFactory | Do not use unless instructed to do so by Oracle. | string | NA | NA | oracle.xml.jaxp.JXDocumentBuilderFactory |
| custommenuaction_model_location | Name of the configuration file containing metadata definitions for a custom menu action. | string | NA | NA | NA |
| event_poller_interval | Time in milliseconds OSM waits before polling for new events. | integer | 100 | 60000 | 5000 |
| event_poller_mutex_timeout | Do not use unless instructed to do so by Oracle. | integer | 0 | 60000 | 10000 |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 5-1 (Cont.) Description of oms-config.xml Parameters**

| Name | Description | Type | Min | Max | Default |
|---|---|---|---|---|---|
| enable_log_stacktraces | Enables and disables logging stack traces.<br>If set to **true**, stack trace printing in the log is enabled. If set to **false**, it is disabled. | boolean | NA | NA | true |
| com.mslv.oms.handler.order.cache.OrderCacheManager | Cache manager. | string | NA | NA | com.mslv.oms.handler.order.cache.jboss.JBossOrderCacheManager |
| com.mslv.oms.security.HandlerFactoryRegistry.HandlerFactory | Name of the process that manages the handling of high activity orders in clustered systems. | string | NA | NA | com.mslv.oms.security.HandlerFactory |
| com.mslv.oms.handler.cluster.ClusteredHandlerFactory.HighActivityOrder.CollectionCycle.Enabled | Enables high-activity order collection cycles. | boolean | NA | NA | true |
| com.mslv.oms.handler.cluster.ClusteredHandlerFactory.HighActivityOrder.CollectionCycle.InitialDelay | 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. | integer | 1000 | 3600000 | 10000 |
| com.mslv.oms.handler.cluster.ClusteredHandlerFactory.HighActivityOrder.CollectionCycle.Duration | Time in milliseconds to spend collecting high-activity order statistics per collection cycle. | integer | 1000 | 60000 | 10000 |
| com.mslv.oms.handler.cluster.ClusteredHandlerFactory.HighActivityOrder.CollectionCycle.Interval | The interval in milliseconds between consecutive of the collection cycle. | integer | 1000 | 3600000 | 60000 |
| com.mslv.oms.handler.cluster.ClusteredHandlerFactory.HighActivityOrder.RequestPerSecondThreshold | 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. | integer | 1 | 1000 | 50 |
| com.mslv.oms.security.HandlerCallbackFactoryRegistry.HandlerCallbackFactory | Do not use unless instructed to do so by Oracle. | string | NA | NA | com.mslv.oms.security.HandlerCallbackFactory |
| com.mslv.oms.security.OrderViewAccessProvider | Registers a security callback for the OrderViewAccessException exception. | string | NA | NA | NA |
| com.mslv.oms.cartridgemgmt.DeployCartridgeMDB.CartridgeDeploymentTransactionTimeout | Default transaction timeout interval in seconds for Oracle Communications Service Catalog and Design - Design Studio cartridge deployment. Design Studio can override the default value through the environment property. | integer | 120 | 7200 | 3600 |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 5-1 (Cont.) Description of oms-config.xml Parameters**

| Name | Description | Type | Min | Max | Default |
|---|---|---|---|---|---|
| com.mslv.oms.cartridgemgmt.cache.DeployCartridgeCache.DeployCartridgeRequestTimeToLiveMinutes | 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. | integer | 5 | 360 | 30 |
| com.mslv.oms.cartridgemgmt.cache.DeployCartridgeCache.DeployCartridgeLocalCacheSize | Do not use unless instructed to do so by Oracle. | integer | 10 | 10000 | 1000 |
| com.mslv.oms.logging.ThreadLogAppenderMaxBufferSize | Do not use unless instructed to do so by Oracle. | integer | 1000 | 20000 | 10000 |
| com.mslv.oms.model.transform.OrderTransformer.ModelURL | Do not use unless instructed to do so by Oracle. | string | NA | NA | NA |
| oracle.communications.ordermanagement.automation.MaxMessagePerTransaction | Defines the maximum number of messages to be consumed in one transaction by JMS listener from the destination it is listening to. | integer | 0 | 10 | 10 |
| oracle.communications.ordermanagement.automation.ListenersPerDestinationRatio | Defines number of JMS listener threads per destination. Also used in determining JMS listener thread pool size. | double | 1.0 | 2.0 | 1.0 |
| oracle.communications.ordermanagement.automation.MaxMessageWaitTime | Defines the maximum waiting time for a JMS listener to wait for messages to arrive to the destination it is listening to. | integer | 0 | 5000 | 5000 |
| oracle.communications.ordermanagement.cluster.BusinessRequestBalancer.OrderRelease.Timeout | The time in seconds to wait for a non-exclusive lock to be placed on an order.<br>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. | integer | 200 | 500 | 200 |
| oracle.communications.ordermanagement.cluster.BusinessRequestBalancer.OrderOwnershipLock.Timeout | Timeout interval, in seconds, for acquiring an exclusive cluster-wide lock on an order. | integer | 30 | 60 | 30 |
| oracle.communications.ordermanagement.cluster.BusinessRequestBalancer.ServerState.Scanning.Interval | 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. | integer | 1000 | 10000 | 5000 |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 5-1 (Cont.) Description of oms-config.xml Parameters**

| Name | Description | Type | Min | Max | Default |
|---|---|---|---|---|---|
| oracle.communications.ordermanagement.orchestration.generation.ControlDataLocation | 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. | string | NA | NA | ControlData/OrderItem |
| oracle.communications.ordermanagement.orchestration.generation.TransformedOrderItemLocation | 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. | string | NA | NA | ControlData/TransformedOrderItem |
| oracle.communications.ordermanagement.util.net.CatalogUriResolver.DefaultXmlCatalogsUris | List of URIs specifying the XML Catalogs that are used system-wide. On all OS platforms, entries are separated by a semicolon (;). | string | NA | NA | NA |
| oracle.communications.ordermanagement.config.ModelResourceClasspath | 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 (;). | string | NA | NA | NA |
| order_editor_submit_mode_threshold | 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. | integer | 1 | 9999999 | 9999999 |
| is_tablelayout_height_fixed | 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. | boolean | NA | NA | true |
| height_of_tablelayout | Height in pixels of the table layout. | string | NA | NA | 400px |
| no_of_rows_in_textarea_without_scroll | Maximum number of rows that a text field can accommodate without a scroll bar. | integer | 3 | 1000 | 3 |
| max_no_of_rows_in_textarea_with_scroll | Maximum number of rows that a text field can accommodate with a scroll bar. | integer | 3 | 1000 | 3 |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 5-1 (Cont.) Description of oms-config.xml Parameters**

| Name | Description | Type | Min | Max | Default |
|---|---|---|---|---|---|
| normalize_space_for_lookup_rule | 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 normalize-space XPath function as defined on the World Wide Web Consortium (W3C) website here: http://www.w3.org/TR/xpath/#function-normalize-space. | boolean | NA | NA | true |
| com.mslv.oms.handler.completeorder.CompleteOrderHandlerEJB.OrchPlanLock.Timeout | Time in seconds that OSM will wait while trying to acquire an exclusive lock on an orchestration plan.<br>This lock is required only when OSM detects that all order components within the orchestration plan have completed and the order can complete. | integer | 1 | 3600 | 30 |
| oracle.communications.ordermanagement.orchestration.generation.DumpOrchestrationPlan | If set to **true**, OSM saves a copy of every generated orchestration plan in XML format to the file *orderId_orderType_orchestrationSequence*_**orchestrationPlanOutput.xml**.<br>**Note**: This parameter should be set to **true** only at the request of Oracle Support for diagnostic purposes. | boolean | NA | NA | false |
| webui_order_info_pane_order_item_sort_ascending | Do not use unless instructed to do so by Oracle. | boolean | NA | NA | true |
| webui_order_info_pane_order_component_sort_ascending | Do not use unless instructed to do so by Oracle. | boolean | NA | NA | true |
| time_out_override_for_jms_adapter | The amount of time to wait before throwing the JMS response timeout. | integer | 15000 | 300000 | 15000 |
| oracle.communications.ordermanagement.cache.UserPerferenceCache | 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. | string | NA | NA | near |
| oracle.communications.ordermanagement.RuleDelayTaskPoller.StartupDelay | Do not use unless instructed to do so by Oracle. | integer | 1000 | 600000 | 900000 |
| oracle.communications.ordermanagement.RuleDelayTaskPoller.Interval | Specifies the pause time in milliseconds between consecutive times that the rule is run and delay task processors. | integer | 1000 | 60000 | 5000 |
| oracle.communications.ordermanagement.RuleDelayTaskPoller.MaxRuleTaskProcessors | Specifies the maximum number of rule task processors. | integer | 1 | 50 | 1 |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 5-1 (Cont.) Description of oms-config.xml Parameters**

| Name | Description | Type | Min | Max | Default |
|---|---|---|---|---|---|
| oracle.communications.ordermanagement.RuleDelayTaskPoller.MaxDelayTaskProcessors | Specifies the maximum number of delay task processors. | integer | 0 | 50 | 1 |
| allow_undeploy_component_cartridge_in_solution | By default, component cartridges that are used by composite cartridges in the run-time environment cannot be undeployed. During the development of a composite cartridge, however, you may need to undeploy a component cartridge used by the composite cartridge. To enable the undeployment of one or more component cartridges used by a composite cartridge, set this value to **true**. | boolean | NA | NA | false |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 5-1 (Cont.) Description of oms-config.xml Parameters**

| Name | Description | Type | Min | Max | Default |
|---|---|---|---|---|---|
| oracle.communications.ordermanagement.security.GlobalQueryRoles | Contains the names of the applicable OSM workgroups (known as roles in Design Studio) separated by a semicolon, a comma, and a colon (**;,:**).<br>**Note:** Including a workgroup name in this parameter makes OSM behave (for that workgroup) the way it did prior to OSM 7.0.3.1. It should only be used for backward compatibility.<br>**If a workgroup *is not* included in this parameter**, a user can query an order type/source only if the user is a member of a workgroup with one of the following permissions for the order type/source:<br>• Permission to create the order (creation task)<br>• Permission on at least one task<br>• Permission on at least one flex header where the flex header node is in that order's order template<br>• At least one query view assigned.<br>In addition to the conditions above, the workgroup which provides the user with the permission above must also have the **Search View** permission assigned.<br>**If a workgroup *is* included in this parameter**, the permissions will operate as they did prior to OSM 7.0.3.1: users can query all order types/sources, without meeting any of the conditions above. Having the workgroup included in this parameter also overrides filters defined for the workgroup. So, a user who is a member of a workgroup defined in this parameter is able to query all order types and order sources, regardless of any filters configured.<br>**Note:** Users can always see default and defaultOrchestration cartridges, regardless of any other conditions, and regardless of whether their workgroup is included in this parameter. | string | NA | NA | NA |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 5-1 (Cont.) Description of oms-config.xml Parameters**

| Name | Description | Type | Min | Max | Default |
|---|---|---|---|---|---|
| oracle.communications.ordermanagement.order.UseUnionOfFiltersAcrossRoles | Filters are OR'ed across roles and the results are a union of data across all the results. 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 (**;,:**).<br>This parameter has been added for backward compatibility with older versions of OSM. | string | NA | NA | NA |
| oracle.communications.ordermanagement.ws.CreateOrder.UseTinyTree | Do not use unless instructed to do so by Oracle. | boolean | NA | NA | true |
| oracle.communications.ordermanagement.orchestration.generation.UseTinyTree | Do not use unless instructed to do so by Oracle. | boolean | NA | NA | true |
| oracle.communications.ordermanagement.amendment.DataEnrichmentAware | 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. | boolean | NA | NA | true |
| oracle.communications.ordermanagement.transaction.order.lock | If set to **true**, OSM will attempt to lock the order while processing order fulfillment state updates during an update order request. This is to prevent a race condition between two threads trying to process fulfillment state updates. | boolean | NA | NA | true |
| oracle.communications.ordermanagement.transaction.order.lock.waittime | Wait time in milliseconds for the OSM order lock for oracle.communications.ordermanagement.transaction.order.lock. | integer | 1000 | 30000 | 3000 |
| oracle.communications.ordermanagement.orchestration.generation.CompressOrchestrationPlan | Do not use unless instructed to do so by Oracle. | boolean | NA | NA | true |
| oracle.communications.ordermanagement.order.CompressXMLValues | Do not use unless instructed to do so by Oracle. | boolean | NA | NA | true |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 5-1 (Cont.) Description of oms-config.xml Parameters**

| Name | Description | Type | Min | Max | Default |
|---|---|---|---|---|---|
| soap_adapter_preemptive_authentication | The SOAP Adapter does not support preemptive authentication by default. If preemptive authentication is used incorrectly, it can lead to significant security issues. You must evaluate the potential benefits versus the security risks of preemptive authentication in the context of your specific application environment. If you set this value to **true**, you can configure HttpClient to authenticate preemptively by prepopulating the authentication data cache. | boolean | NA | NA | false |
| oracle.communications.ordermanagement.security.access.summary | 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). | string | NA | NA | * (meaning all workgroups) |
| oracle.communications.ordermanagement.security.access.data | 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). | string | NA | NA | * (meaning all workgroups) |
| oracle.communications.ordermanagement.security.access.orchestration-plan | 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). | string | NA | NA | * (meaning all workgroups) |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 5-1 (Cont.) Description of oms-config.xml Parameters**

| Name | Description | Type | Min | Max | Default |
|---|---|---|---|---|---|
| oracle.communications.ordermanagement.security.access.dependencies | 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). | string | NA | NA | * (meaning all workgroups) |
| oracle.communications.ordermanagement.security.access.amendments | 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). | string | NA | NA | * (meaning all workgroups) |
| oracle.communications.ordermanagement.security.access.activity | 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). | string | NA | NA | * (meaning all workgroups) |
| oracle.communications.ordermanagement.security.access.data-tree | 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). | string | NA | NA | * (meaning all workgroups) |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 5-1 (Cont.) Description of oms-config.xml Parameters**

| Name | Description | Type | Min | Max | Default |
|---|---|---|---|---|---|
| oracle.communications.ordermanagement.table-layout.threshold | 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. | integer | -1 | 64000 | 50 |
| oracle.communications.ordermanagement.table-layout.size | The number of rows displayed in the table in a single view (that is, without scrolling), can be configured. | integer | 0 | 50 | 10 |
| oracle.communications.ordermanagement.table-layout.fetch-size | 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. | integer | 5 | 100 | 25 |
| oracle.communications.ordermanagement.webui.text.cols.size | Do not use unless instructed to do so by Oracle. | integer | 30 | 80 | 48 |
| oracle.communications.ordermanagement.webui.xml.cols.size | Do not use unless instructed to do so by Oracle. | integer | 72 | 144 | 80 |
| oracle.communications.ordermanagement.webui.xml.rows.size | Do not use unless instructed to do so by Oracle. | integer | 4 | 10 | 5 |
| oracle.communications.ordermanagement.webui.max.xmlcomponents.per.row | Do not use unless instructed to do so by Oracle. | integer | 1 | 4 | 2 |
| oracle.communications.ordermanagement.webui.max.auto.expend.tree.levels | Do not use unless instructed to do so by Oracle. | integer | 1 | 128 | 3 |
| oracle.communications.ordermanagement.webui.max.components.per.row | Do not use unless instructed to do so by Oracle. | integer | 1 | 4 | 3 |
| oracle.communications.ordermanagement.webui.sort.component.type | Do not use unless instructed to do so by Oracle. | boolean | NA | NA | false |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 5-1 (Cont.) Description of oms-config.xml Parameters**

| Name | Description | Type | Min | Max | Default |
|---|---|---|---|---|---|
| oracle.communications.ordermanagement.resource.FullScopeAccess | The level of access to system resources. This parameter has one of the following values:<br>• **_strict_access_**<br>This value indicates that no cartridges will have full scope access to system resources.<br>• *****<br>(asterisk) This value indicates that all cartridges will have full scope access to system resources.<br>• *catridgeMnemonic*:*version*,*cartridgeMnemonic*:*version*, etc.<br>This type of value indicates that only the listed cartridges will have full scope access. For example, if the value is:<br>`OSM_Cart1:1.0.0.0.0,OSM_Cart1:2.0.0.0.0`<br>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 | NA | NA | _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.<br>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.<br>If this parameter is not present in the **oms-config.xml** file, the **Activity** tab behaves as though the property is set to **false**. | boolean | NA | NA | false |
| OrderCacheMaxEntries | Maximum number of orders that can be in the cache in a managed server at one time.<br>When there is an attempt to exceed the maximum 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.<br>A value of **0** means the cache size is unlimited. The only way that orders can be removed from the cache when the value of this parameter is **0** is when the value of the OrderCacheInactivityTimeout is reached. | integer | 0 | 86400000 | 1000 |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 5-1 (Cont.) Description of oms-config.xml Parameters**

| Name | Description | Type | Min | Max | Default |
|---|---|---|---|---|---|
| OrderCacheInactivityTimeout | Maximum number of seconds that an order can be inactive (no attempt to access or update it) before being ejected from the cache.<br>A value of **0** means there is no inactivity timeout. | integer | 0 | 86400000 | 3600 |
| ClosedOrderCacheMaxEntries | Maximum number of orders that recently changed from an open state to a closed state that can exist in a managed server at any time.<br>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 the cache. A typical strategy for this value is to set it to a specific fraction of the OrderCacheMaxEntries value.<br>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. | integer | 0 | 86400000 | 50 |
| ClosedOrderCacheTimeout | Maximum number of seconds that recently closed orders remain in the cache after they have changed from an open state to a closed state.<br>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 OrderCacheInactivityTimeout parameter. Typically, this is set to a relatively low value, such as 60 seconds. To be effective, this value should be much smaller than the OrderCacheInactivityTimeout value. | integer | 0 | 86400000 | 60 |
| fast_cartridge_undeploy | If set to **true**, cartridges are undeployed from OSM but cartridge metadata and order data are not purged from the database.<br>After undeploying a cartridge, its status in the OSM database is set to **UNDEPLOYED**.<br>If set to **false**, 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. | boolean | NA | NA | true |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 5-1 (Cont.) Description of oms-config.xml Parameters**

| Name | Description | Type | Min | Max | Default |
|---|---|---|---|---|---|
| cartridge_operation_timeout | Timeout interval, in seconds, for completion of cartridge deployment. Deployment will be terminated when this time is reached. | integer | -1 | 3600 | 1800 |
| oracle.communications.ordermanagement.xml.file.encoding | Specifies the standard text encoding OSM uses to parse XML attachments. | string | NA | NA | UTF-8 |
| 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 | NA | NA | osm_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 | NA | NA | osm_compliance/results |
| oracle.communications.ordermanagement.compliance.evaluator.snapshot.directory | Specifies the directory that contains the generated snapshot files that need to be evaluated.<br>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: **yyyymmdd-hhmmss**. | string | NA | NA | osm_compliance/snapshots |
| oracle.communications.ordermanagement.compliance.evaluator.rule.directory | Specifies the directory that contains the rules that the snapshot is to be evaluated against. Directory rules run in addition to internal rules. | string | NA | NA | osm_compliance/rules |
| oracle.communications.ordermanagement.compliance.configuration.override.file | File to override the configuration file for the compliance tool. | string | NA | NA | NA |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 5-1 (Cont.) Description of oms-config.xml Parameters**

| Name | Description | Type | Min | Max | Default |
|---|---|---|---|---|---|
| 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.<br>For example:<br>`<oms-parameter>`<br>`  <oms-parameter-name>AutomationConcurrencyModels</oms-parameter-name>`<br>`  <oms-parameter-value>file:///Users/sample/automationConcurrencyModel.xml</oms-parameter-value>`<br>`</oms-parameter>`<br><br>You can put the policy file in any reachable location and the policy file can have any name when being referenced from **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.<br>For more information about OACC policy files, see *OSM Developer's Guide*. | list of files | NA | NA | NA |
| oracle.communications.ordermanagement.order.OperationalOverrideFileURLs | In OSM traditional deployments, this is the location of operational jeopardy configuration override files. The format of this parameter is a comma-separated list of file names (including paths). Files should be located in the same file system as the **oms-config.xml** file. For example:<br>`/u01/settings/osm/operational-jeopardy-config1.xml,/u01/settings/osm/operational-jeopardy-config2.xml`<br><br>By default this field is blank, which means that operational jeopardy configuration is not turned on. | list of files | NA | NA | NA |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 5-1 (Cont.) Description of oms-config.xml Parameters**

| Name | Description | Type | Min | Max | Default |
|---|---|---|---|---|---|
| oracle.communications.ordermanagement.scheduler.MaxDelay.OrderJeopardyTimer | Maximum number of milliseconds for the generic timer to wait before triggering items on its list.<br>Changing this parameter can affect system performance. | integer | 0 | 86400000 | 30000 |
| oracle.communications.ordermanagement.scheduler.MaxEventCount.OrderJeopardyTimer | Maximum number of items processed in a single call to the generic timer.<br>Changing this parameter can affect system performance. | integer | 1 | 10000 | 10 |
| oracle.communications.ordermanagement.scheduler.MaxDelay.OrderJeopardyTimerCleanup | Do not use unless instructed to do so by Oracle. | integer | 60000 | 86400000 | 3600000 |
| oracle.communications.ordermanagement.scheduler.MaxPausedTime.OrderJeopardyTimerCleanup | Do not use unless instructed to do so by Oracle. | integer | 1 | 2147483647 | 86400000 |
| adaptivePolling.ceiling | Adaptive event polling determines how the om_jms_event table, which contains records of OSM-generated events, is polled. To increase performance, adaptive polling provides a mechanism for polling only the most active database partitions generally, with only periodic polling of all of the partitions.<br>This parameter specifies how many times to query only the most active partitions before performing the full query again. Set to 0 to turn off adaptive polling and run the full query every time. | integer | 0 | 100 | 10 |
| adaptivePolling.maxPartitions | Adaptive event polling determines how the om_jms_event table, which contains records of OSM-generated events, is polled. To increase performance, adaptive polling provides a mechanism for polling only the most active database partitions generally, with only periodic polling of all of the partitions.<br>This parameter specifies the number of partitions to include in the query of only the most active partitions. | integer | 1 | 30 | 2 |
| oracle.communications.ordermanagement.wm.ProcessRequest.Timeout | Do not use unless instructed to do so by Oracle. | integer | 60000 | 300000 | 60000 |
| oracle.communications.ordermanagement.monitoring.adml.dir | Do not use unless instructed to do so by Oracle. | string | NA | NA | ${common.components.home}/modules/oracle.dms/adml |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 5-1 (Cont.) Description of oms-config.xml Parameters**

| Name | Description | Type | Min | Max | Default |
|---|---|---|---|---|---|
| oracle.communications.ordermanagement.monitoring.date.displayformat | Do not use unless instructed to do so by Oracle. | string | NA | NA | yy-MM-dd HH:mm:ss |
| oracle.communications.ordermanagement.monitoring.movingaverage.windowsize | Do not use unless instructed to do so by Oracle. | integer | 1 | 100000 | 1000 |
| oracle.communications.ordermanagement.order.contingency | Amount of contingency buffer, measured in minutes, that is added to a calculated order duration to determine the projected delivery date of an order. | integer | 0 | 525600 | 0 |
| oracle.communications.ordermanagement.ws.FindOrderRequest.FindOrderMaxOrderThreshold | Maximum number of rows that are returned in the results of the FindOrder web service operation. For more information, see "FindOrder" in *OSM Developer's Guide*. | integer | 1 | 64000 | 1000 |
| oracle.communications.ordermanagement.ws.FindOrderOperationImpl.MaxFlexHeaderCountInOrderBy | The number of flexible headers that can be sent in an order by clause in a web service request. If the number of flexible headers is greater than this value, the web service request will be rejected. | integer | 1 | 100 | 1 |
| oracle.communications.ordermanagement.ui.compatibilityMode | Do not use unless instructed to do so by Oracle. | boolean | NA | NA | false |
| oracle.communications.ordermanagement.OrderPartitioningRealmConfigFileURLs | Location of order partitioning realm configuration files. The format of this parameter is a comma-separated list of file names (including paths). Files should be located in the same file system as the **oms-config.xml** file. For example:<br>`/u01/settings/osm/order_partitioning_realms_config1.xml,/u01/settings/osm/order_partitioning_realms_config2.xml`<br>By default this field is blank, which means that no additional partitioning realms for orders are configured in the system beyond those installed by default.<br>See "Partitioning Realms" for more information about this configuration. | list of files | NA | NA | NA |
| communications.ordermanagement.webui.render_timeline_tab_for_orderdetails | Enables and disables the **Timeline** tab in the Order Management Web Client. For more information about the Order Lifecycle Management UI and the **Timeline** tab, see *OSM Order Management Web Client User's Guide*.<br>If set to **true**, the **Timeline** tab is displayed in the Order Management Web Client UI. If set to **false**, the tab is not displayed. | boolean | NA | NA | true |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 5-1 (Cont.) Description of oms-config.xml Parameters**

| Name | Description | Type | Min | Max | Default |
|---|---|---|---|---|---|
| oracle.communications.ordermanagement.EventPoller.MaxProcessingTime | Time, in seconds, for requests from a single poll of table om_jms_event to process. Events that have not been processed before this time expires will be processed during the next poll. | integer | 1000 | 600000 | 25000 |
| oracle.communications.ordermanagement.EventPoller.MaxEventErrorCount | Number of attempts to retry processing an event when event processing has failed. | integer | 1 | 1000 | 10 |
| oracle.communications.ordermanagement.AutomationResponseMessageParkingEnabled | Determines whether OSM saves external responses for later delivery if the task receiver cannot process them when they are received (for example, the response message is for an **In Progress** order state, but the order state has changed to **Amending** or **Suspending**).<br>For example, assume a task initiated interaction with an external system when order state is **In Progress** but before the response is received the order state changes to **Amending** (revision order received and grace period elapsed), in this situation with this behavior enabled the response message from external system is parked (in table `om_automation_message`) instead of getting processed immediately. Later when the order state changes to **In Progress** the parked message is republished to the queue for processing. Disabling this behavior could cause an order-stuck issue due to loss of response message. | boolean | NA | NA | true |
| application_scope_id | For internal use only. | string | NA | NA | NA |
| sso_enabled | This parameter is used to enable or disable Single Sign On Functionality using SAML2.0 | boolean | N/A | N/A | false |
| user_entitlement_expression | This parameter provides the JSONpath expression to test access authorization via user enablement in SAML2.0 SSO | string (JSONpath expression) | N/A | N/A | Empty (disabling user enablement feature) |
| user_entitlement_expected_result_size | This parameter is used in conjunction with **user_entitlement_expression** to denote the expected return value from the JSONPath. | integer | -1 | 64000 | -1 (indicates any non-zero result is acceptable) |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 5-1 (Cont.) Description of oms-config.xml Parameters**

| Name | Description | Type | Min | Max | Default |
|---|---|---|---|---|---|
| order_editor_parallel_submit_control | This parameter is used for the Task web client, it controls parallel requests coming from Task web client Order Editor.<br>**Note:** Change default value of this parameter only if you understand the impact, or if it is recommended by Oracle.<br>Usage:<br>-1: Simultaneous request is allowed without restriction.<br>0: Simultaneous request is not allowed; subsequent request from the same session fails if the earlier one is not done.<br>1-30: Simultaneous request is allowed, but with restriction; subsequent request from the same session wait for *n* seconds (to allow the earlier request to finish), before it fails. | integer, seconds | -1 | 30 | 2 |

# Configuring Operational Order Jeopardies with Configuration Files

Operational order jeopardies are a method of configuring order jeopardies at runtime. You can configure operational order jeopardies with XML configuration files. These files are referenced in **oms-config.xml** using the **oracle.communications.ordermanagement.order.OperationalOverrideFileURLs** parameter.

A sample operational order jeopardy file is provided in the jeopardy configuration in Design Studio. An order jeopardy configuration file has two main elements.

- **orderJeopardy**: This element contains a jeopardy configuration. The name attribute of this element contains the name of the order jeopardy. If it matches the name of an order jeopardy configured in Design Studio, the definition in this file will override the Design Studio configuration. If the name does not match any configured in Design studio, this jeopardy will be considered a new jeopardy.
- **extendedOrderDefinition**: This element contains information about the jeopardies that should be assigned to a particular order.

Following is a sample **orderJeopardy** element, with an explanation of its contents. In this sample, the jeopardy will be raised if the order spends more than four minutes (80 percent of five minutes) getting from the specified **fromState** to one of the specified **toState** values (not counting any time spent in an **excludedState**).

```
<model:orderJeopardy name="SampleOrderTimeout" namespace="ProvComponent">
   <model:description>Sample Order Timeout</model:description>
   <model:priority>2</model:priority>
   <model:enabled>true</model:enabled>
   <model:condition>
      <model:name>null_rule</model:name>
      <model:namespace>ProvComponent</model:namespace>
   </model:condition>
```


Copyright © 2009, 2025, Oracle and/or its affiliates.

```
   <model:notifyRole>
      <model:name>ProvRole</model:name>
      <model:namespace>ProvComponent</model:namespace>
   </model:notifyRole>
   <model:notificationView>
      <model:name>ActivationOrderQueryTask</model:name>
      <model:namespace>ProvComponent</model:namespace>
   </model:notificationView>
   <model:policy>
      <model:stateConditions>
         <model:fromState>open.running.in_progress</model:fromState>
         <model:toState>open.not_running.canceled</model:toState>
         <model:toState>closed.completed</model:toState>
         <model:toState>closed.aborted</model:toState>
         <model:excludedState>open.not_running.suspended</model:excludedState>
         <model:excludedState>open.not_running.failed</model:excludedState>
      </model:stateConditions>
      <model:threshold xsi:type="model:OrderJeopardyFixedDurationThresholdType">
        <model:percent>80</model:percent>
        <model:duration>P0Y0M0DT0H5M0S</model:duration>
      </model:threshold>
   </model:policy>
</model:orderJeopardy>
```

- **description**: This contains the description of the jeopardy.
- **priority**: This is the priority of the notification. The notification with the highest priority is evaluated first.
- **enabled**: This should be set to **true** if the jeopardy is enabled, or **false** if it is not enabled. Setting this value to **false** is a simple way of disabling a jeopardy with a matching name that was created in Design Studio.
- **condition**: This contains the rule that defines when this jeopardy is active. If the rule returns true, the jeopardy is evaluated.
- **notifyRole**: This contains the name of the role that should be notified if the jeopardy is met.
- **notification view**: This contains the view that should contain any data that is used to determine order jeopardy duration or date/time.
- **policy**: This specifies the actual jeopardy information.
- **stateConditions**: This specifies the states involved in the jeopardy notification, including the state or states the order is transitioning from (**fromState**), the state or states that the order is transitioning to (**toState**), and the states during which the timer should pause (**excludedState**).
- **threshold**: This contains information about duration for the jeopardy.
- **percent**: This is the percentage of the duration to wait before raising the jeopardy.
- **duration**: The duration to wait (taking into consideration the offset) before raising the jeopardy.

Following is another sample of the policy element inside the of the **orderJeopardy** element. In this sample, the jeopardy will be raised if the order does not reach one of the specified states by five minutes before the date/time specified in the **/ControlData/CompleteDate** data element on the order.

```
<model:policy>
  <model:stateConditions>
    <model:toState>closed.completed</model:toState>
    <model:toState>closed.aborted</model:toState>
```


Copyright © 2009, 2025, Oracle and/or its affiliates.

```
  </model:stateConditions>
  <model:threshold xsi:type="model:OrderJeopardyOrderExpressionThresholdType">
    <model:offset>-P0Y0M0DT0H5M0S</model:offset>
    <model:expression>
      <model:orderData>/ControlData/CompleteDate</model:orderData>
      <model:unit>seconds</model:unit>
      <model:default>P0Y0M0DT0H0M0S</model:default>
    </model:expression>
  </model:threshold>
</model:policy>
```

- **stateConditions**: This specifies the states involved in the jeopardy notification. In this example, only the **toState** is included, indicating that the expression element should refer to a date/time value rather than a duration.
- **offset**: This specifies the amount of time to raise the jeopardy before (because the value begins with a minus sign) the date/time specified in the **orderData** element.
- **orderData**: This specifies an order data element that contains a date/time type value.
- **unit**: Because the data element in this case is a date/time, the value of this parameter is ignored.
- **default**: Because the data element in this case is a date/time, the value of this parameter is ignored.

Following is a sample **extendedOrderDefinition** element, with an explanation of its contents:

```
<model:extendedOrderDefinition xsi:type="model:OrderJeopardyAssociationDefinitionType"
name="SampleOrderTimeoutExtendedOrder" namespace="SystemConfig">
   <model:extends>
      <model:orderType>
         <model:name>ActivationSampleOrder</model:name>
         <model:namespace>ProvBase</model:namespace>
      </model:orderType>
      <model:orderSource>
         <model:name>ActivationSampleOrder</model:name>
         <model:namespace>ProvBase</model:namespace>
      </model:orderSource>
      <model:cartridge>
        <oms:namespace>OsmCentralOMExample-Solution</oms:namespace>
        <oms:version>4.0.0.0.0</oms:version>
      </model:cartridge>
   </model:extends>
   <model:jeopardy xsi:type="model:OperationalOrderJeopardyReferenceType">
      <model:name>SampleOrderTimeout</model:name>
      <model:namespace>ProvComponent</model:namespace>
   </model:jeopardy>
</model:extendedOrderDefinition>
```

> **Note**
>
> In this example, **xmlns:model** is equivalent to: http://xmlns.oracle.com/communications/ordermanagement/model; and **xmlns:oms** is equivalent to: http://www.metasolv.com/OMS/OrderModel/2002/06/25

- **extends**: This element contains information about the order being extended.
- **orderType**: This contains the name of the order.


Copyright © 2009, 2025, Oracle and/or its affiliates.

- **orderSource**: This contains the Order Source of the order. This value is set in the Order editor **Details** tab, in the **Order Source** field. If there is no value in this field for the order, the **orderSource** is the same as the **orderType**.
- **cartridge**: This element contains the cartridge namespace and version, which are required when the operational order jeopardy is referenced in the **oms-config.xml** file instead of being defined in the model from Design Studio.
- **jeopardy**: This element contains the name and namespace of the jeopardy that should be evaluated for the specified order. There should be one instance of this element for each jeopardy that applies to the order.


Copyright © 2009, 2025, Oracle and/or its affiliates.

6

# Configuring the Task Processor

This chapter describes how to configure the task processor 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 task processor.

## About Configuring the Task Processor

The task processor evaluates rules, event delays, and timer delays to determine when to transition to the next task in a process. The task processor 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 the Task Processor for Performance

You can use the oms-config parameters to tune the number and type of task processors.

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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

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**.


Copyright © 2009, 2025, Oracle and/or its affiliates.

7

# Managing the OSM Database Schema

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

See "Database Terms" for descriptions about database terms used in this chapter.

## 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 Oracle WebLogic Server 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 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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

- "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.
- "Partitioning Realms": Discusses the use of logical partitions in the OSM schema to assist in purging or dropping orders.
- "Cartridge Management Strategy": Recommends a strategy for managing cartridges.
- "Sizing Partitions": Discusses how to size partitions for order data. Partition sizing depends on your order purge strategy.
- "Online vs. Offline Maintenance": Gives a brief overview of which maintenance operations can be performed online.

# Creating Tablespaces

The OSM DB Installer expects the following permanent database tablespaces specified in the project specification:

```
db:
  type: "STANDARD" # Acceptable values are STANDARD and ADB
  # datasourcesPrimary section is applicable only for STANDARD DB. For ADB,
values will be used from Autonomous Database Serverless  secrets+configMap.
  datasourcesPrimary:
    port: 1521
    # If not using RAC, provide the DB server hostname/IP address
    # If using RAC, comment out "#host:"
    #host: dbserver-ip
    #
    # If using RAC, provide list of SCAN hostname/IP addresses
    # If not using RAC, comment out "#scans:"
    #scans:
    #  - scan1-ip
    #  - scan2-ip
    #
    # If using RAC, provide either a list of VIP hostname/IP addresses
    # or a list of INSTANCE_NAMES
    # If not using RAC, comment these out "#vips:" and "#instances:"
    #
    #vips:
    #  - vip1-ip
    #  - vip2-ip
    # --- OR ---
    #instances:
    #  - instance-1
    #  - instance-2

  # Default log level. Valid value
  #
  # The levels in descending order are:
  #   SEVERE (highest value)
  #   WARNING
  #   INFO
  #   CONFIG
  #   FINE
  #   FINER
```


Copyright © 2009, 2025, Oracle and/or its affiliates.

```
#   FINEST  (lowest value)
#
logLevel: "WARNING"
#
# The remaining parameters must match the values used when the PDB was
# created.  Failure to match will result in dbInstaller errors
#
# The default tablespace name of OSM schema
defaultTablespace: "OSM"
# The temporary tablespace name of OSM schema
tempTablespace: "TEMP"
# The time zone offset in seconds
timezoneOffsetSeconds: "-28800"
# The model data tablespace name of OSM schema
modelDataTablespace: "OSM"
# The model index tablespace name of OSM schema
modelIndexTablespace: "OSM"
# The order data tablespace name of OSM schema
orderDataTablespace: "OSM"
# The order index tablespace name of OSM schema
orderIndexTablespace: "OSM"
```

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

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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

- 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.

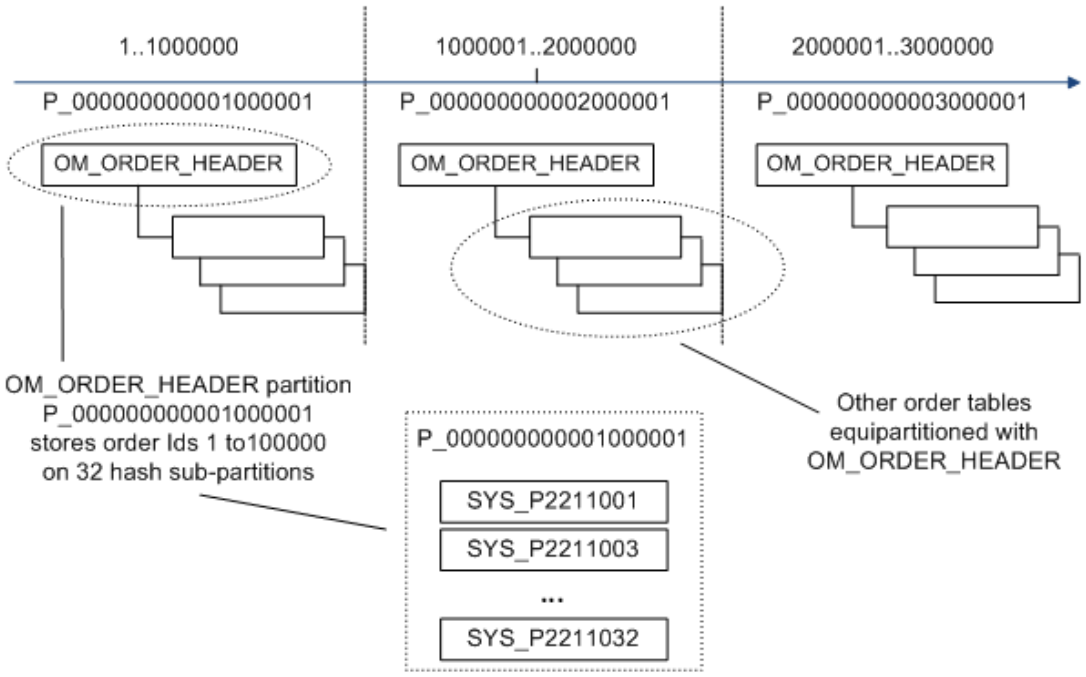
For more information about managing tablespaces, see *Oracle Database Administrator's Guide*.

# Using Partitioning

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

The OSM DB Installer enables partitioning automatically. The following figure provides details about the OSM partition tables that accumulate order-related information using range partitioning based on OSM order ID ranges.

**Figure 7-1 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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

- 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_000000000001000001 is 1,000,001. 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.

You can use the partitioning realms feature to separate orders with different operational characteristics into different partitions. See "Partitioning Realms" for more information.

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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

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. Directing interactions from a given managed server to a specific RAC instance is achieved using JDBC multi data sources.OSM does not support an Active GridLink data source given that this configuration has a severe impact on OSM performance. The reason for this is that, with Active GridLink, OSM cannot direct interactions from a given managed server to a specific primary RAC instance.

Figure 7-2 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 7-3 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_00000000000100001 and P_00000000000200001. 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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Figure 7-2 OSM Data Source Configuration for Active-Active Oracle RAC**

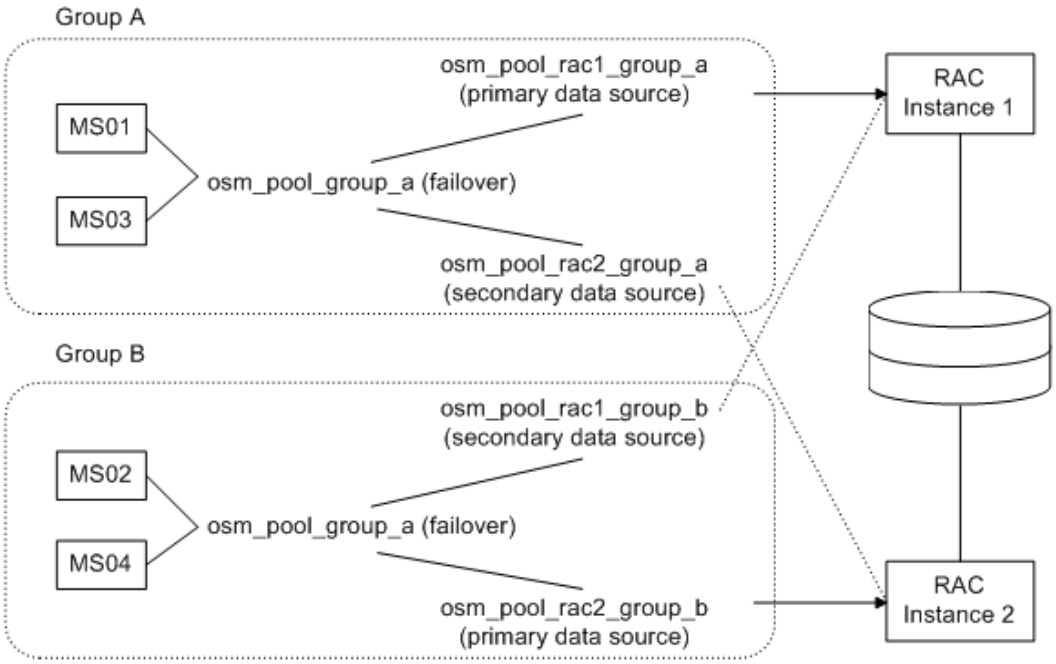
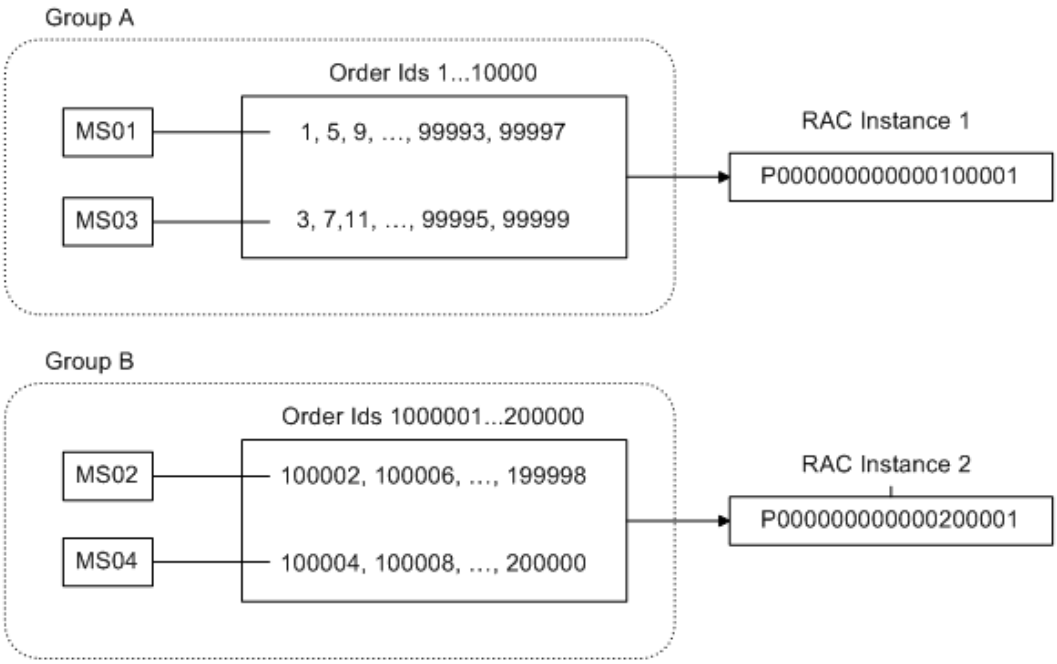



**Figure 7-3 OSM Order ID Generation for Active-Active Oracle RAC**




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


Copyright © 2009, 2025, Oracle and/or its affiliates.

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 affect 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 run in parallel 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


Copyright © 2009, 2025, Oracle and/or its affiliates.

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 runs 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).

## 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


Copyright © 2009, 2025, Oracle and/or its affiliates.

- 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 7-4 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:


Copyright © 2009, 2025, Oracle and/or its affiliates.

- 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 7-4 Partition Purge Maintenance, Window 1**

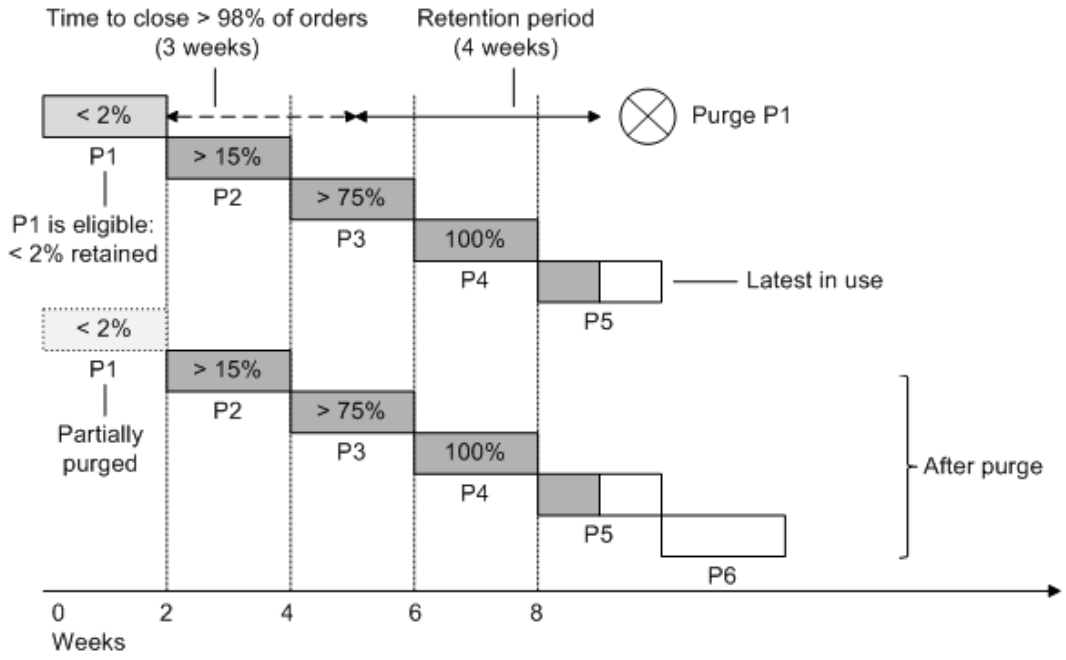



Figure 7-5 shows the second maintenance window (after 2 weeks). Because a partition contains 2 weeks' worth of orders, P5 is exhausted and P6 is half full. As in the previous maintenance, it is cost-effective to purge only one partition that has not yet been purged, which is P2 in this example. You also add at least one partition (not shown). Notice that the number of partitions that are not purged is 3.5 again.


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Figure 7-5 Partition Purge Maintenance, Window 2**

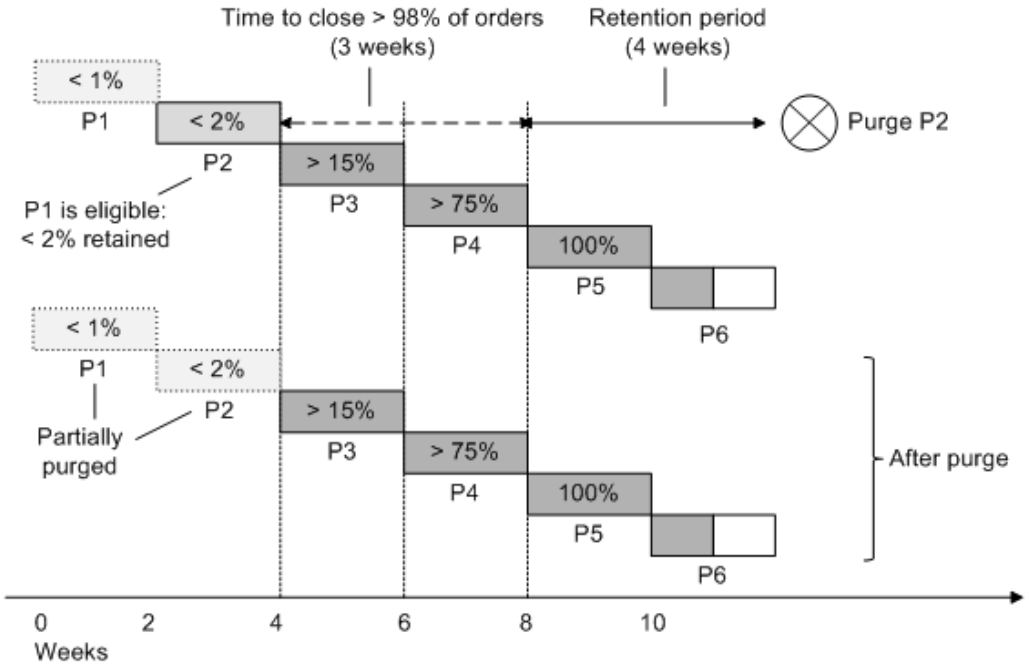
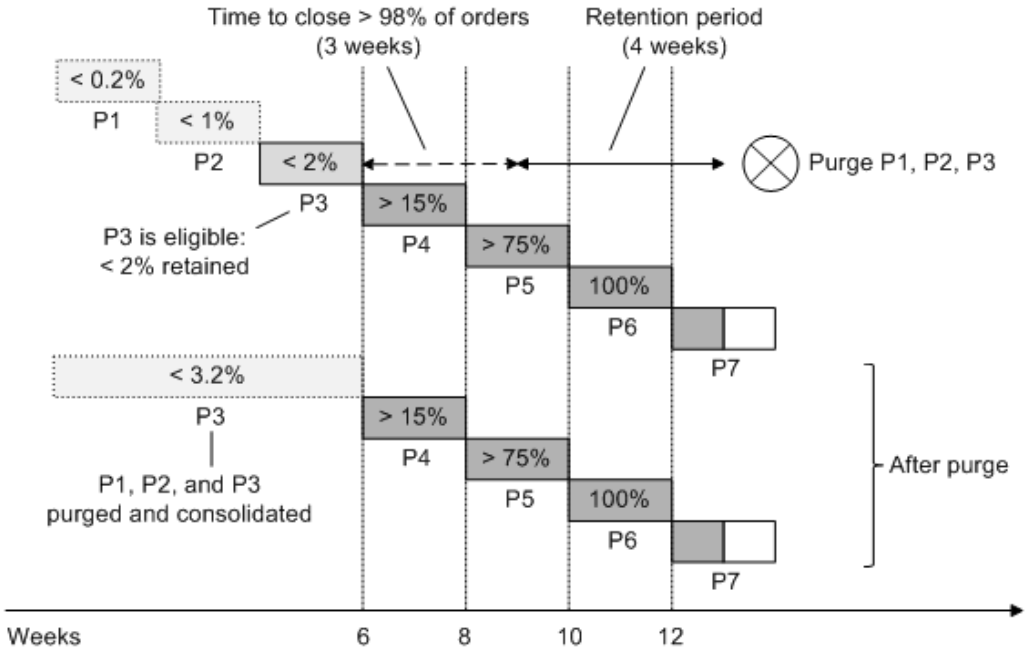



Figure 7-6 shows the third maintenance window. As in the previous maintenance, it is cost-effective to purge only one partition (P3) that has not yet been purged, and 3.5 partitions are not purged. This time, however, the previously purged partitions P1 and P2 are purged again and consolidated with P3 (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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Figure 7-6 Partition Purge Maintenance, Window 3**




## Advantages and Disadvantages of Partition-Based Purge

Advantages of the partition purge strategy:

- 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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

- 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 IO 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 lifetime, 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


Copyright © 2009, 2025, Oracle and/or its affiliates.

(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:

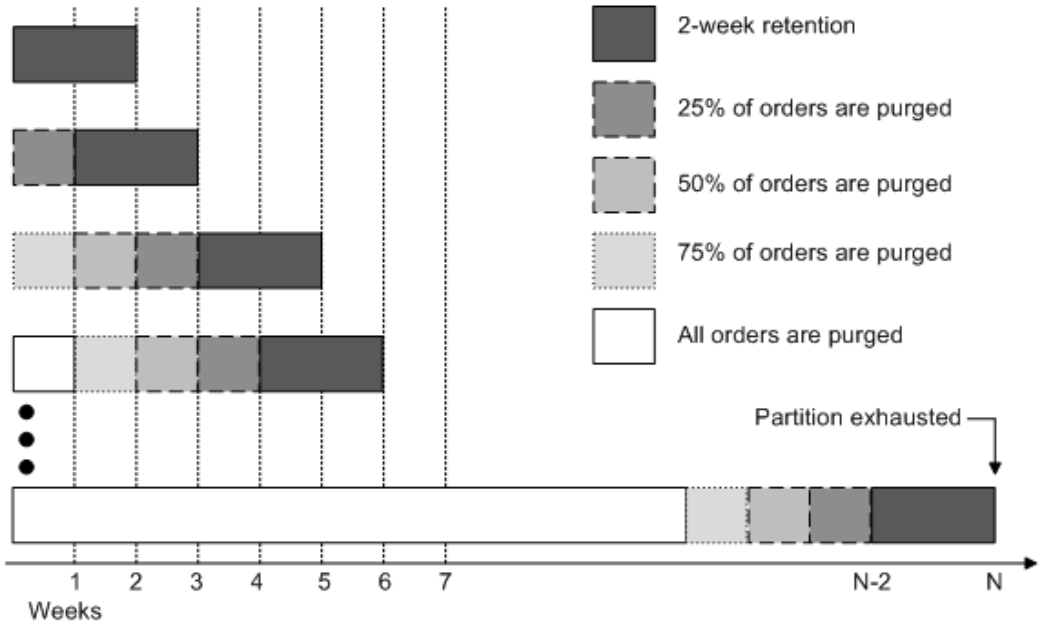
- Row-Based Order Purge Example
- Advantages and Disadvantages of Row-Based Order Purge
- Using Row-Based Order Purge

## Row-Based Order Purge Example

Figure 7-7 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 Nth week. Then it repeats for the next partition.

**Figure 7-7 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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

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."

# Partitioning Realms

You can use the partitioning realms feature to separate orders into different partitions when those orders have different operational characteristics, for example, short-lived orders versus long-lived orders. This enables you to group orders into partitions that can be purged or dropped more effectively.

The partitioning realms feature is used for logical partitioning, which is supported for even non-partitioned schemas.

> **Note**
>
> Oracle does not recommend using this feature. Instead, use online purge.


Copyright © 2009, 2025, Oracle and/or its affiliates.

A new order that is received by OSM requires an **order_seq_id** before it is persisted in the database. Each partitioning realm reserves ranges of order_seq_id values to be used by orders that are assigned to that realm. Mapping rules within each partitioning realm determine which orders are assigned to the partitioning realm. If an order does not map to any partitioning realm it is assigned to a pre-defined default realm. The default realm is called **default_order**.

Orders automatically map to default realms that are pre-defined in the database. You do not have to configure partitioning realms unless you have a specific requirement to do so. For more information, see "Default Partitioning Realm."

> **Note**
>
> Using partitioning realm limits the partition consolidation capability. Partition consolidation happens as part of offline partition-based purge, in which few partitions are dropped to free database file system space, and orders-to-retain are consolidated into fewer partitions. When partitioning realm is used, then partitions that belong to different realms cannot be consolidated as the orders from different realms cannot be consolidated into same partition. For example, if Partition P1 belongs to realm R1 and Partition P2 belongs to realm R2, then P1 and P2 cannot be consolidated into one partition, but P1 and P2 can be purged individually. Also, if P1 and P2 belong to R1 and P3 belongs to R2, then P1 and P2 can be consolidated into one partition and P3 can be purged separately. In other words, these three partitions cannot be consolidated into single partition but into two.

# Partitioning Realms Configuration

Partitioning realms are created and maintained in the system using XML configuration files. Some aspects of partitioning realms that are defined in the configuration files are also saved in the database, which is required for database maintenance operations, such as purging.

The configuration files are referenced in the **oms-config.xml** file using the following parameter:

```
oracle.communications.ordermanagement.OrderPartitioningRealmConfigFileURLs
```

> **Note**
>
> The partitioning realms files must be located in the same file system as the **oms-config.xml** file.

Sample order partitioning realms configuration files are provided in the OSM SDK in the **SDK/Samples/PartitioningRealms** directory. An XML Schema Definition (XSD) for partitioning realms is also located in this directory.

Example 7-1 is a sample partitioning realm file. This sample file contains only one order partitioning realm, but files can contain configuration for multiple realms, if required.

**Example 7-1 Sample Order Partitioning Realms File**

```
<partitioningRealmModel xmlns="http://xmlns.oracle.com/communications/ordermanagement/
partitioningRealms" realmType="ORDER">
  <partitioningRealm name="sample_order_realm" enabled="true">
    <description>Sample for demonstrating realm configuration</description>
    <purgeStrategy>ANY</purgeStrategy>
```


Copyright © 2009, 2025, Oracle and/or its affiliates.

```
    <parameters>
      <rangePartitionSize>500000</rangePartitionSize>
    </parameters>
    <mappings>
      <includes>
        <cartridgeNamespace>MyTestCartridge</cartridgeNamespace>
        <orderName>MyOrder</orderName>
      </includes>
    </mappings>
  </partitioningRealm>
</partitioningRealmModel>
```

Table 7-1 lists and describes the elements and attributes that can be included in the partitioning realms file.

**Table 7-1 Partitioning Realms File Elements and Attributes**

| Element or Attribute | Description |
|---|---|
| **realmType** | The type of realm contained in this configuration file. Valid value is **ORDER**. This element is mandatory. |
| **name** | The name of the realm. The name is not case-sensitive and must be unique in the system. This element is mandatory. |
| **enabled** | Indicates whether the realm is enabled or disabled. Disabled realms are never mapped to orders and do not have new partitions added automatically.<br>The **enabled** attribute is optional and the default value is **true** (enabled) if not present. |
| **description** | This contains the description of the realm. This element is optional. |
| **purgeStrategy** | Indicates the purge strategies supported by this realm. Valid values are **PARTITION-BASED**, **ROW-BASED**, and **ANY**.<br>The element is optional and the default is **ANY** if not present. See "Purge Strategy for Partitioning Realms" for more information. |
| **parameters** | Contains parameters specific to the partitioning realm. These parameters override the values in the om_parameter table. All the parameters are optional. If they are not specified, the value from the om_parameter table is used. Currently supported parameter is rangePartitionSize. |
| **rangePartitionSize** | The range partition size represents the number of orders to use when adding new partitions for this realm. This element maps to the **range_partition_size** parameter in the om_partitioning_realm_param table. This element is optional. If not specified, the **range_partition_size** value from the om_parameter table is used. |
| **mappings** | Contains **includes** rules that map new orders to this partitioning realm. Multiple **includes** elements can be used to map multiple different cartridges or order types to a single realm. The valid child elements are **includes** and **excludes**. See "Mapping Orders to Partitioning Realms" for more information. This parameter is technically optional, but if it is not included, the partition will not be used for any orders. |
| **includes** | Contains the elements used to map new orders to this realm. The valid child elements are: **cartridgeNamespace**, **cartridgeVersion**, and **orderName**. This parameter is optional.<br>For more information about using the **includes** element, see "Mapping Orders to Partitioning Realms." |
| **excludes** | (Not shown in example) Contains the elements used to exclude new orders from this realm. The valid child elements are: **cartridgeNamespace**, **cartridgeVersion**, and **orderName**. This parameter is optional. |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 7-1 (Cont.) Partitioning Realms File Elements and Attributes**

| Element or Attribute | Description |
|---|---|
| **cartridgeNamespace** | A regular expression used to match the cartridge namespace of a new order. This parameter is optional. |
| **cartridgeVersion** | (Not shown in example) A regular expression used to match the cartridge version of a new order. This is valid only when the **cartridgeNamespace** element is also included.<br>Oracle does not recommend using **cartridgeVersion** because that would require the realm mappings need to be updated when cartridges are upgraded to a new version. This parameter is optional. |
| **orderName** | A regular expression used to match the order name (or order type) of a new order. This parameter is optional. |

## Mapping Orders to Partitioning Realms

The **mappings** element of the partitioning realms configuration is used to map new orders to a specific partitioning realm. The mappings associate orders to partitioning realms by comparing order data to match criteria (regular expressions) for each partitioning realm. If a match is NOT found for an order, the order is assigned to the **default_order** partitioning realm. The **default_order** partitioning realm cannot contain any mappings. See "Default Partitioning Realm" for more information.

For example, the following partitioning realms XML configuration sets the partitioning realm to **short_lived_orders** for any order with a cartridgeNamespace that starts with "Mobile." Any order that is not in a cartridge with a name that starts with "Mobile" will be assigned the **default_order** partitioning realm.

```
<partitioningRealmModel realmType="ORDER">
    <partitioningRealm name="short_lived_orders">
        <mappings>
            <includes>
                <cartridgeNamespace>Mobile.*</cartridgeNamespace>
            </includes>
        </mappings>
    </partitioningRealm>
</partitioningRealmModel>
```

The following fields can be used as match criteria in the partitioning realms mapping file:

- Cartridge Namespace
- Cartridge Version
- Order Name


Copyright © 2009, 2025, Oracle and/or its affiliates.

> **Note**
>
> The cartridge namespace and version are the deployed cartridge namespace and version. For standalone cartridges, use the standalone cartridge namespace and version. For composite cartridges, use the composite cartridge namespace and version.
>
> Using the cartridge version in mappings is not recommended because mappings can be broken when cartridges are upgraded and will then require more frequent maintenance.

### About the includes Element

If there are multiple match criteria specified within an **includes** element, ALL the criteria in the **includes** element must match. In other words, the criteria within an **includes** element are part of an AND condition. In this example, only **LTEMobileService** orders from the **Mobile** are mapped to the **short_lived_orders** partitioning realm. Other orders from the same cartridge are not mapped to this partitioning realm.

```
<partitioningRealmModel realmType="ORDER">
    <partitioningRealm name="short_lived_orders">
        <mappings>
            <includes>
                <cartridgeNamespace>Mobile</cartridgeNamespace>
                <orderName>LTEMobileService</orderName>
            </includes>
        </mappings>
    </partitioningRealm>
</partitioningRealmModel>
```

If there are multiple **includes** elements, only one of the **includes** criteria must match in order for the realm to map. In other words, multiple **includes** elements are part of an OR condition.

For example, with the following configuration the **short_lived_orders** partitioning realm is mapped to orders from both **Mobile** and **Broadband** cartridges.

```
<partitioningRealmModel realmType="ORDER">
    <partitioningRealm name="short_lived_orders">
        <mappings>
            <includes>
                <cartridgeNamespace>Mobile</cartridgeNamespace>
            </includes>
            <includes>
                <cartridgeNamespace>Broadband</cartridgeNamespace>
            </includes>
        </mappings>
    </partitioningRealm>
</partitioningRealmModel>
```

When there are multiple partitioning realms defined in a configuration file, they are processed sequentially. The first partitioning realm to match the order data is used. In the following example, the **long_lived_orders** realm would never be mapped because **short_lived_orders** realm maps to all cartridges starting with "Mobile".

```
<partitioningRealmModel realmType="ORDER">
    <partitioningRealm name="short_lived_orders">
        <mappings>
```


Copyright © 2009, 2025, Oracle and/or its affiliates.

```
            <includes>
                <cartridgeNamespace>Mobile.*</cartridgeNamespace>
            </includes>
        </mappings>
    </partitioningRealm>
    <partitioningRealm name="long_lived_orders">
        <mappings>
            <includes>
                <cartridgeNamespace>MobileCartridge</cartridgeNamespace>
            </includes>
        </mappings>
    </partitioningRealm>
</partitioningRealmModel>
```

### About the excludes Element

To exclude an order from getting mapped to a realm you can add an **excludes** element under the mappings. The **excludes** element takes precedence over the **includes** element. In other words, if an order matches both **includes** element criteria and **excludes** element criteria, the order is not mapped to the partitioning realm. In the following example, all orders that start with "Mobile" are mapped to the **short_lived_orders** realm, except **Mobile4GService** orders.

```
<partitioningRealmModel realmType="ORDER">
    <partitioningRealm name="short_lived_orders">
        <mappings>
            <includes>
                <orderName>Mobile.*</orderName>
            </includes>
            <excludes>
                <orderName>Mobile4GService</orderName>
            </excludes>
        </mappings>
    </partitioningRealm>
</partitioningRealmModel>
```

## Enabling and Disabling Partitioning Realms

New realms that you add are disabled. Typically, you add realms, add partitions to those realms, and then enable them. See "Partitioning Realms" and "partition_auto_creation" for more information.

You can enable and disable realms by changing the **enabled** attribute in the partitioning realm configuration to **true** (enabled) or **false** (disabled). If a partitioning realm is disabled, it is no longer used for mapping incoming orders, therefore any partition or order ID block that is assigned to that partitioning realm is no longer used.

Enabled partitioning realms cannot be removed from the partitioning realm configuration file. If an enabled partitioning realm is removed from the configuration file, a validation error occurs the next time you restart the system or refresh OSM metadata.

You cannot disable the default partitioning realm: **default_order**.

To remove a partitioning realm from the configuration file:

1. In the configuration file, set the **enabled** attribute to **false** to disable the partitioning realm.
2. Do one of the following:
   - Restart OSM.
   - Refresh the OSM metadata.

   The partitioning realm is disabled in the database.


Copyright © 2009, 2025, Oracle and/or its affiliates.

3. Remove or comment out the realm XML configuration that you want to remove.

> **Note**
>
> This removes the XML configuration for the realm, however the realm still exists in the database. You cannot remove partitioning realms from the database.

## Renaming a Partitioning Realm

A partitioning realm can be renamed by adding the **oldName** attribute to the partitioning realm configuration and changing the **name** attribute to the new name. For example:

```
<partitioningRealm oldName="MyOldRealmName" name="MyNewRealmName"/>
```

After you change the name in the XML configuration file, refresh the OSM metadata.

You cannot rename the default partitioning realm: **default_order**.

## Refreshing Partitioning Realms Configuration

Refreshing OSM metadata loads and updates the partitioning realms configuration. When you make changes to the partitioning realms configuration, you must refresh the OSM metadata for the change to take effect. There are several ways to refresh the OSM metadata. See "Refreshing OSM Metadata" for more information.

If there is a validation error in the partitioning realms configuration, the behavior depends on the following factors:

- If the server is starting up, an invalid partitioning realms configuration causes a **CRITICAL** health policy error which prevents OSM from starting. You can find the cause of the validation error in the log files. The problem in the partitioning realms configuration must be fixed before you attempt to restart OSM.
- If the server is already running, an invalid partitioning realms configuration causes errors in the logs and the loading of the partitioning realms configuration is abandoned. The OSM server continues to run using the last known good realm configuration. In other words, the partitioning realm configuration changes are ignored. You can find the cause of the validation error in the log files. The problem in the partitioning realm configuration must be fixed before you attempt to refresh the OSM metadata.

# Adding Partitions for New Partitioning Realms

The steps to add a partition for a new partitioning realm depends on whether the **partition_auto_creation** value is enabled or disabled. In production and performance environments, Oracle recommends that you disable the **partition_auto_creation** attribute and add partitions manually before they are required.

When you create new partitioning realms, a new partition is needed to hold orders associated with the new realm. When you start OSM or refresh metadata, all enabled partitioning realms are assigned an order ID generator. The order ID generator is responsible for creating new order_seq_ids for new orders in OSM. When order ID generators are initialized they are assigned an **om_order_id_block**, which represents a range of order_seq_ids reserved for that order ID generator. Each **om_order_id_block** maps to a partition in the database schema. This is why every enabled partitioning realm requires a partition.


Copyright © 2009, 2025, Oracle and/or its affiliates.

## partition_auto_creation Disabled

If **partition_auto_creation** is disabled, partitioning realms must be created in a disabled state before adding partitions.

To add partitions with the **partition_auto_creation** attribute disabled:

1. Create the new partitioning realm by setting the **enabled** attribute to **false** in the partitioning realm XML configuration file.
2. Refresh the OSM metadata or restart OSM.

   This creates the partitioning realm in the system in a disabled state.
3. Add one or more partitions for the new (disabled) realm using the **om_part_maintain.add_partitions** procedure. In addition to the name of the new partitioning realm, you must set the **a_force** argument to **true** to create a partition for a disabled partitioning realm. If you do not enter **true** for the **a_force** argument, an exception is raised because, by default, partitions cannot be added to disabled realms.

   ```
   exec om_part_maintain.add_partitions(a_count, a_tablespace, a_realm_mnemonic, true)
   ```
4. When you are ready to use the new partitioning realm, set the **enabled** attribute to **true** in the partitioning realm XML configuration.
5. Refresh the OSM metadata or restart OSM.

   This enables partitioning realm in the system.

## partition_auto_creation Enabled

If **partition_auto_creation** is enabled, you can create partitioning realms in an enabled state. Partitions are created automatically for the partitioning realm.

To add partitions with the **partition_auto_creation** attribute enabled:

1. Create the new partitioning realm by setting the **enabled** attribute to **true** in the partitioning realm XML configuration file.
2. Refresh the OSM metadata or restart OSM.

   This creates the partitioning realm in the system in an enabled state.

   A partition is automatically created for the new partitioning realm.
3. (Optional) Add one or more partitions for the new realm using the **om_part_maintain.add_partitions** procedure.

   ```
   exec om_part_maintain.add_partitions(a_count, a_tablespace, a_realm_mnemonic)
   ```

# Purge Strategy for Partitioning Realms

The purge strategy associated with a partitioning realm determines the type of purge supported by the realm. Set the purgeStrategy value in the partitioning realms file to specify the purge strategy. See " Partitioning Realms Configuration" for more information about the partitioning realms file.

Valid purgeStrategy values are the following:

- **ROW-BASED**: Only row-based purge operations can be used in this realm. Partition-based purge procedures will ignore partitions with a purge strategy of **ROW-BASED**,


Copyright © 2009, 2025, Oracle and/or its affiliates.

unless they are empty. If a partition is empty, it can be dropped regardless of the configured purge strategy.

- **PARTITION-BASED**: Only partition-based purge operations can be used on this realm. Row-based purge procedures will ignore orders in a realm with a purge strategy of **PARTITION-BASED**. The only exceptions are procedures that let the user specify the orders to purge, such as the **delete_order** and **purge_selected_orders** procedures; these procedures will delete the order regardless of the realm's purge strategy.
- **ANY**: There are no restrictions on the purge method used. This is the default purge strategy.

## Default Partitioning Realm

The default partitioning realm is for all orders that do not map to a user-configured partitioning realm. The following default partitioning realm is created in OSM:

- **default_order**

The default partitioning realm is not included in the configuration files, but can be included if you want to change some properties of the default realm.

When working with the default realm, you **cannot** change the following:

- The default realm cannot be disabled.
- The default realm cannot be renamed.
- The default realm cannot contain any mappings.
- The **realmType** of default realm cannot be changed.

When working with default realms, you **can** change the following:

- The description of default realm can be changed.
- The purge strategy can be changed.
- The **rangePartitionSize** of the default realm can be changed. By default, the **default_order** realm does not have a **rangePartitionSize** and uses the **range_partition_size** parameter in the **om_parameter** table.

## Non-Partitioned Schemas

You can use partitioning realms in non-partitioned environments to improve the performance of row-based purging. Grouping orders based on how long they take to close allows orders to be purged together. Purging sequential orders reduces database IO during the purge.

## Order ID Blocks

Use the **OM_ORDER_ID_BLOCK** table to determine which orders belong to which realms. When new partitions are added, a new row is inserted into this table to track the order ID range and associated partitioning realm.

The following example query lists all the block ranges and their associated partitioning realm:

```
select b.first_order_id,
    b.last_order_id,
    r.mnemonic,
    b.status,
    r.realm_type,
    b.dbinstance
```


Copyright © 2009, 2025, Oracle and/or its affiliates.

```
from om_order_id_block b,
    om_partitioning_realm r
where b.realm_id = r.realm_id
order by b.last_order_id;
```

Where:

- **first_order_id**: The first order ID in the block.
- **last_order_id**: The last order ID in the block.
- **mnemonic**: The name of the partitioning realm associated with the block.
- **status**: One of **USED**, **ACTIVE**, or **AVAILABLE**.
  - **ACTIVE**: A block of IDs that is actively being used for new orders.
  - **USED**: A block of IDs that is no longer being used to generate new IDs.
  - **AVAILABLE**: A new, empty block that is available for use.
- **realm_type**: Must be **ORDER**.
- **dbinstance**: The number of the Oracle RAC instance. A value of -1 means the block is unassigned to an Oracle RAC node.

To determine which realm a specific order is associated with, run the following query (replacing **order_seq_id** with the order that you want to query):

```
select r.mnemonic,
    r.realm_type
from om_order_id_block b,
    om_partitioning_realm r
where order_seq_id between b.first_order_id and b.last_order_id
    and b.realm_id = r.realm_id;
```

# Cartridge Management Strategy

Following are the main components to a deployed cartridge:

- 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 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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

- Consider purging metadata of an undeployed cartridge only if it has no associated orders. If you are purging fast-undeployed cartridges online, you cannot purge any cartridges that have orders.

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.
- **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.

You can configure different **range_partition_size** parameters for each partitioning realm. This allows you to tailor the partition size for the partitioning realm purge strategy. For example, you could use row-based purged with oversized partitions for a realm defined for high-volume, short-lived orders, and partition-based purge with smaller partitions for low-volume, long-lived orders. You configure the realm-specific **range_partition_size** using partitioning realm configuration files. For more information, refer to the configuration section of "Partitioning Realms."

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**


Copyright © 2009, 2025, Oracle and/or its affiliates.

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-


Copyright © 2009, 2025, Oracle and/or its affiliates.

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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

- To estimate the number of partitions that are eligible for purge: F / P x N
- To estimate the number of partitions that are ineligible for purge: (T + R) / P x N
- To estimate the total number of partitions that have never been purged: (F + T + R) / P x N
- To estimate the total space consumed by these partitions: (F + T + R) / P x N x 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 x 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 7-8 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

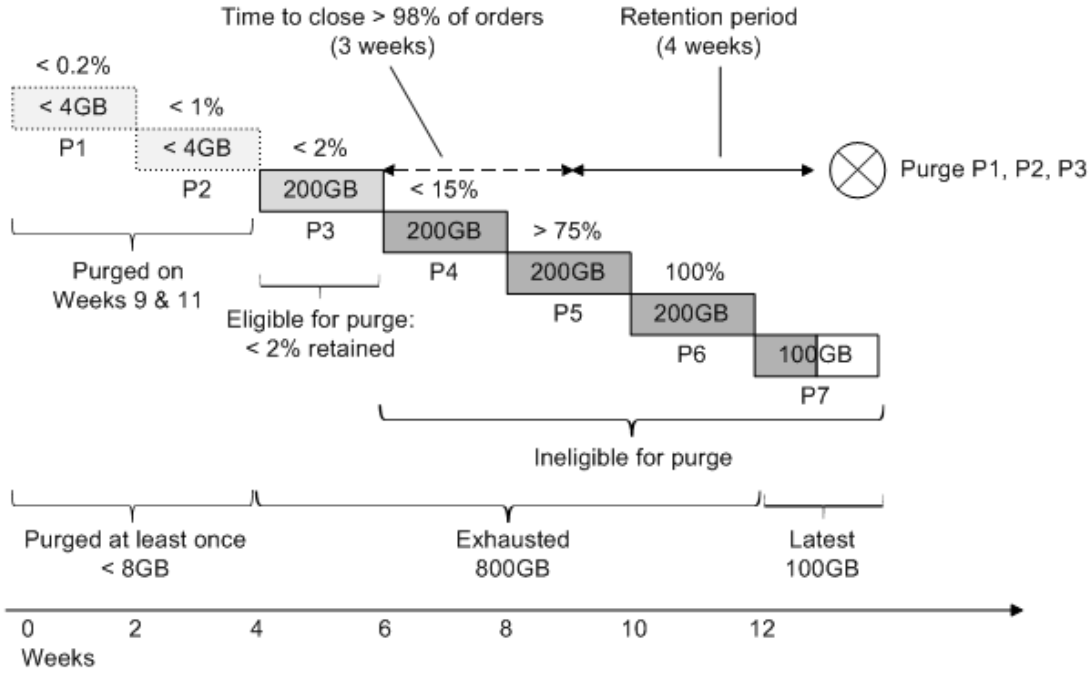
The number of partitions that have never been purged before is:

(F + T + R) / P x N = (2 + 3 + 4) / 2 x 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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Figure 7-8 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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

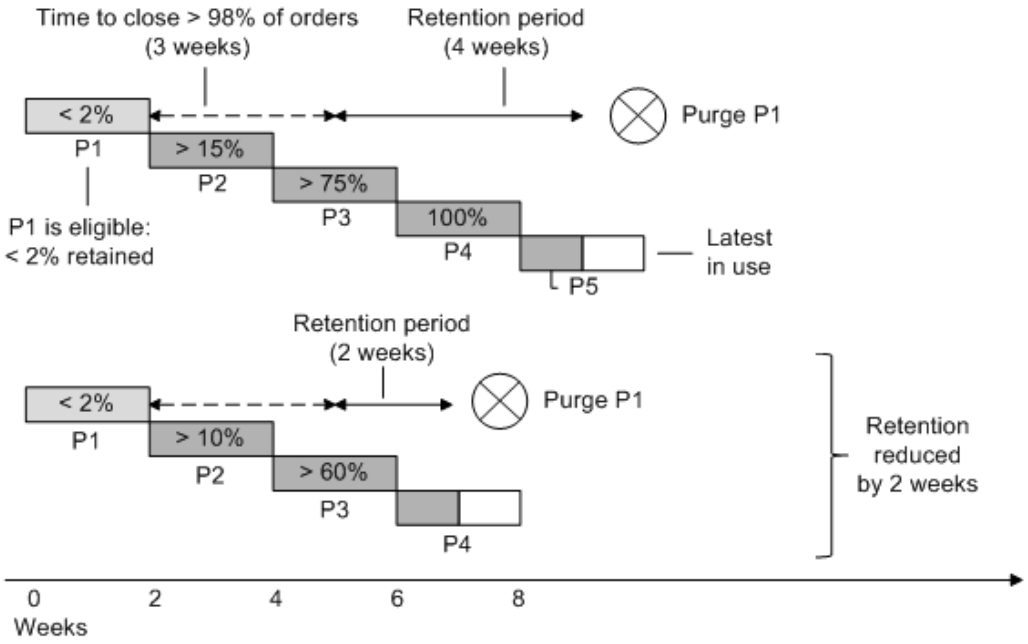
## Retention Policy

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 7-9 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 7-9 Impact of Retention Policy**




## 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 the operation time by increasing parallelism. In general, you should aim to maximize resource utilization in order to improve


Copyright © 2009, 2025, Oracle and/or its affiliates.

purge performance. However, increased parallelism comes with more overhead. For example, you might find out that if you double the parallelism, the operation 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.

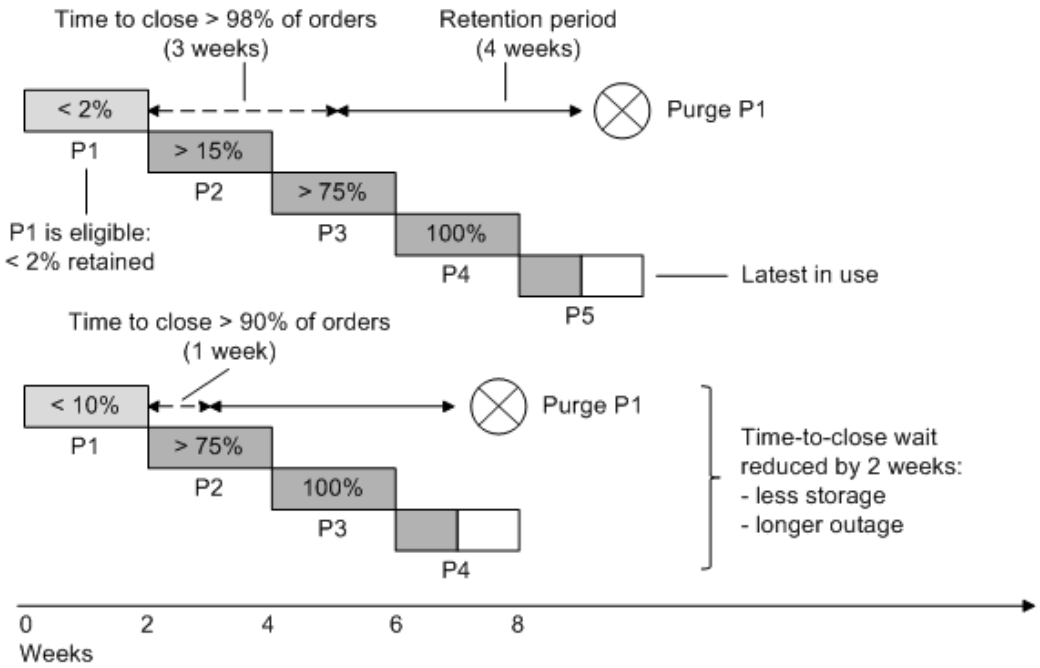
It is easier to use percentages in your initial time-to-close calculations (for example, the time to close 98% of orders). Performance tests help to nail it down to absolute numbers. For example, suppose your acceptable range for copying retained orders (backup and restore) is 15-30 minutes, and that 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 7-10 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 7-10 Impact of Time-to-Close Wait Period**





Copyright © 2009, 2025, Oracle and/or its affiliates.

## 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.

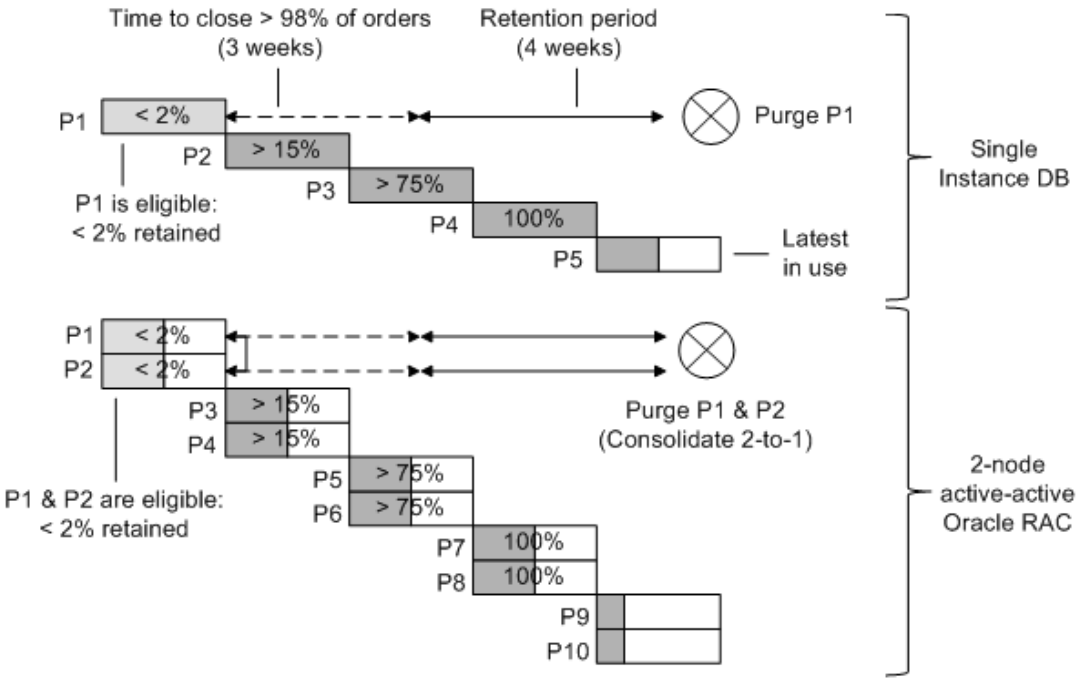
Using Oracle RAC in active-passive mode is similar to using a single instance database. The only difference is that order creation might be switched to another partition and then back to the original in the events of failover and failback, although a switch might not occur right away or even not at all.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

**Example:**

Figure 7-11 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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Figure 7-11 Single-Instance Database vs. Two-Node Active-Active Oracle RAC**




## 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 is 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:

$$(P + T + R) / P \times S = (1 + (T + R) / P) \times S$$

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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

**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:

(P + T + R) / P x S = (P + 3/2P + 2P) / P x S = 4.5 x S

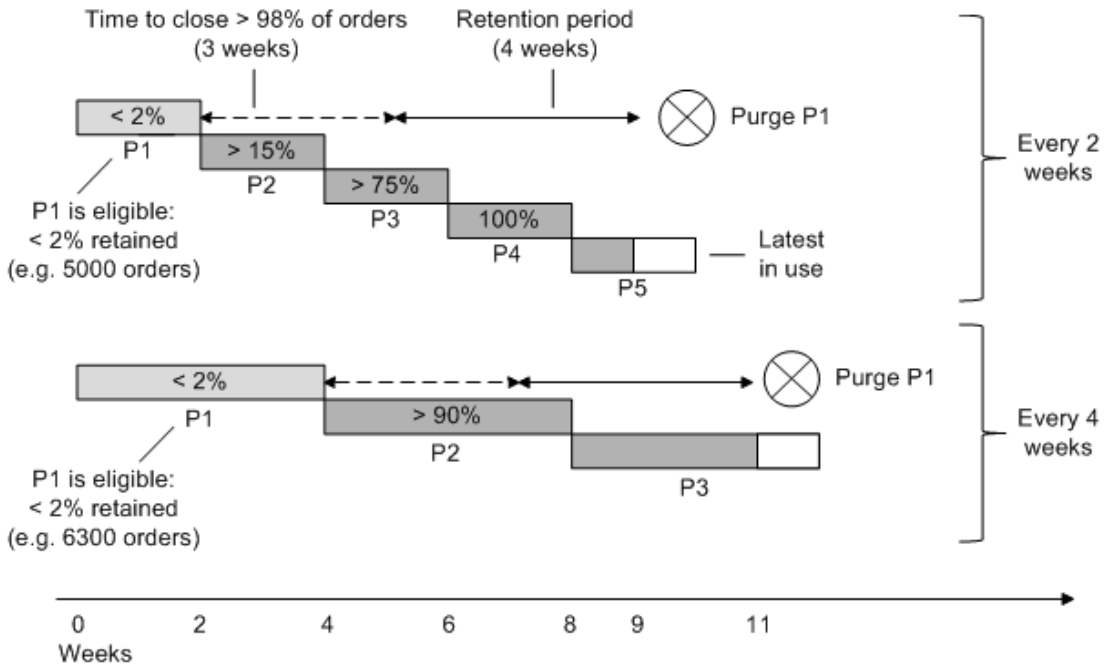
Now suppose that you want to reduce the purge frequency to every 4 weeks. You can double the partition size and estimate the storage capacity:

(P + T + R) / P x S = (P + 3/4P + P) / P x 2 x S =5.5 x 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.

**Figure 7-12 Impact of Doubling Partition Size and Reducing Purge Frequency to Half**




## 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


Copyright © 2009, 2025, Oracle and/or its affiliates.

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

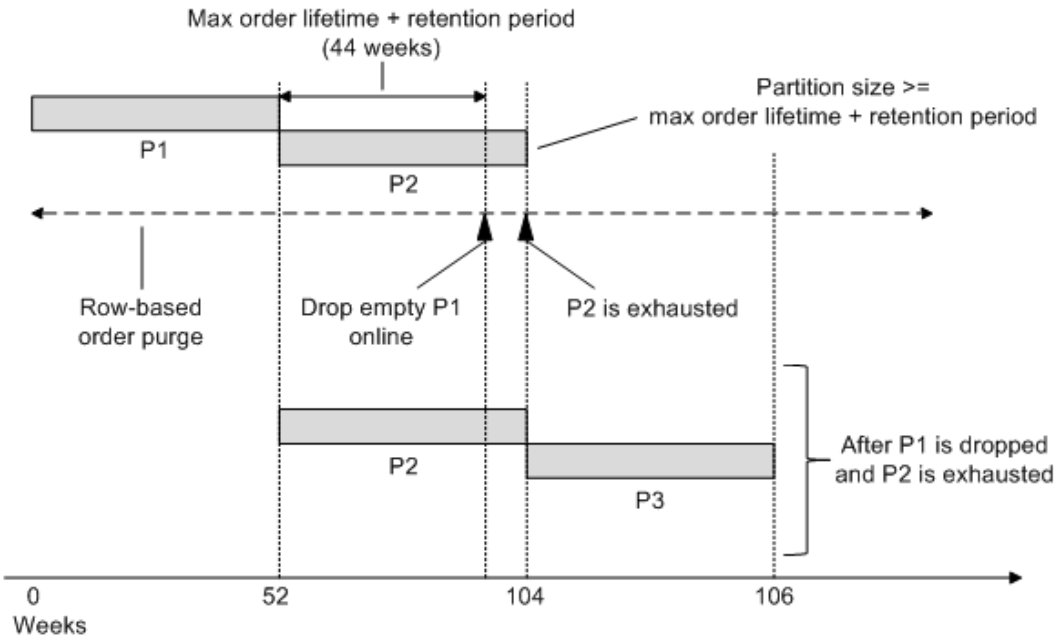
If you cannot afford outages, you can size partitions to avoid outages as follows:

- 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 7-13 illustrates this by example.


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Figure 7-13 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 (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:


Copyright © 2009, 2025, Oracle and/or its affiliates.

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 perform 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 OSM 7.2.0.9, database management procedures stop and restart OSM database jobs automatically.

Table 7-2 summarizes which operations can be performed online. Performing operations offline is always faster than online. If a procedure supports online maintenance operation, it is recommended only under low volume. In particular, performing partition management operations 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.

**Table 7-2 Summary of Online Versus Offline Maintenance Operations**

| Operation | Online | Offline |
|---|---|---|
| Add a partition | Avoid if possible | Recommended |
| Row-based order purge | OK | OK |
| Drop a populated partition | Not supported | OK |
| Drop an empty partition | Not supported | Recommended |
| Partition purge | • Purging a partition partially is not supported online.<br>• 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 | Recommended |
| Cartridge Fast Undeploy | OK, but if done using XML Import/Export (rather than Design Studio), you need to refresh the OSM metadata as well. See "Refreshing OSM Metadata" for more information. | OK |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 7-2 (Cont.) Summary of Online Versus Offline Maintenance Operations**

| Operation | Online | Offline |
|---|---|---|
| Cartridge conventional undeploy | Not supported | OK |

# 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.

- 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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

- 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**.
- (Optional) You can 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.
- (Optional) You can specify the mnemonic of the partitioning realm the partition is associated with as the input argument to **om_part_maintain.add_partition**. If a realm is not specified, the new partition is assigned to the **default_order** realm. If the partitioning realm specified in the argument is disabled, an error will occur and the partition will not be added. To override this error and create the partition for the disabled partitioning realm, set the optional **a_force** argument to **true**.

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.

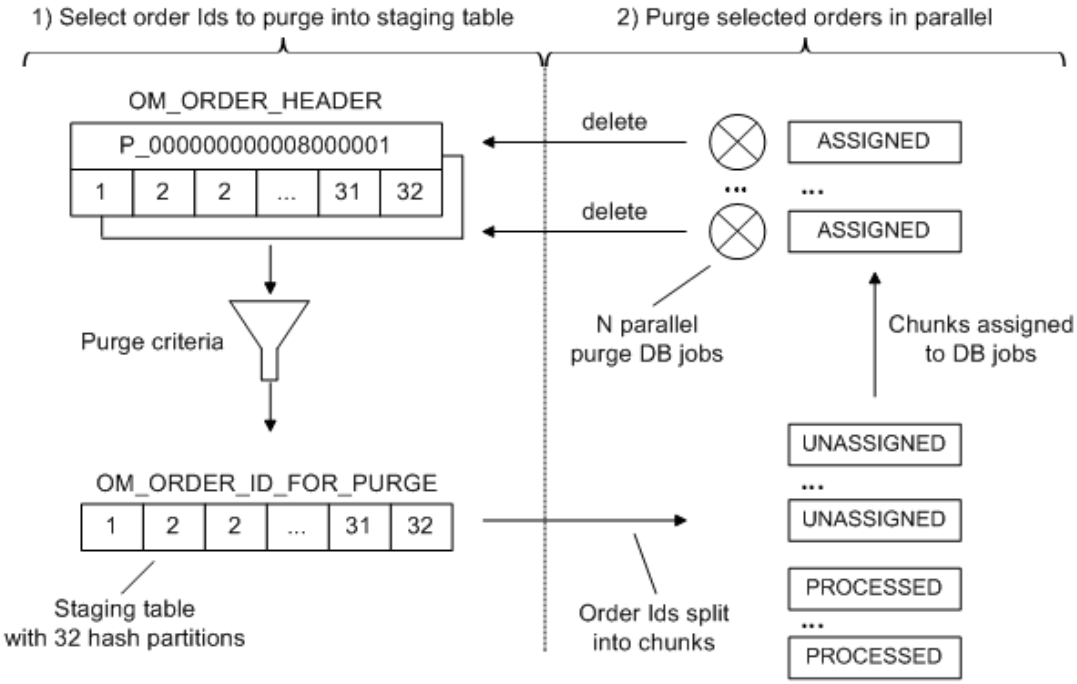
The space freed by deletes in a table or index segment does not lower the high water mark of that segment. That space can be reused by inserts into the same segment but the space is not released to the tablespace. Therefore, you must run order purge as frequently as possible (at least once a day) to 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 7-14 shows how row-based order purge deletes orders in two stages. The API also provides procedures that allow you to run each stage separately, as follows:


Copyright © 2009, 2025, Oracle and/or its affiliates.

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 7-14 How Row-Based Order Purge Works**




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 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 partition. Otherwise, order


Copyright © 2009, 2025, Oracle and/or its affiliates.

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.
- Row-based purge operations are audited. Purge audit views allow you to monitor order purges. 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 run the order selection and order purge steps separately. For example, you can select the orders to purge piecemeal by running **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).


Copyright © 2009, 2025, Oracle and/or its affiliates.

## Scheduling Order Purge

The **om_new_purge_pkg.schedule_order_purge_job** procedure allows you to schedule a one-time order purge operation. 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 order purge operation periodically.

## Stopping and Resuming an Order Purge

The **om_new_purge_pkg.stop_purge** procedure stops an order purge. This might be necessary, for example, if the host is too busy or you want to perform other maintenance operations. This procedure returns immediately. However, the purge stops after all currently assigned chunks are processed, possibly after several minutes.

Later, you can resume the same purge procedure by running **om_new_purge_pkg.resume_purge**. You can also restart the purge procedure with different parameters by running **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 operation.

# Using Partition-Based Order Purge

As discussed in "Partition-Based Order Purge Strategy," 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 threshold. In this case, OSM copies retained orders to the so-called backup tables prior to the partition exchange, and restores them afterwards.


Copyright © 2009, 2025, Oracle and/or its affiliates.

- 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.

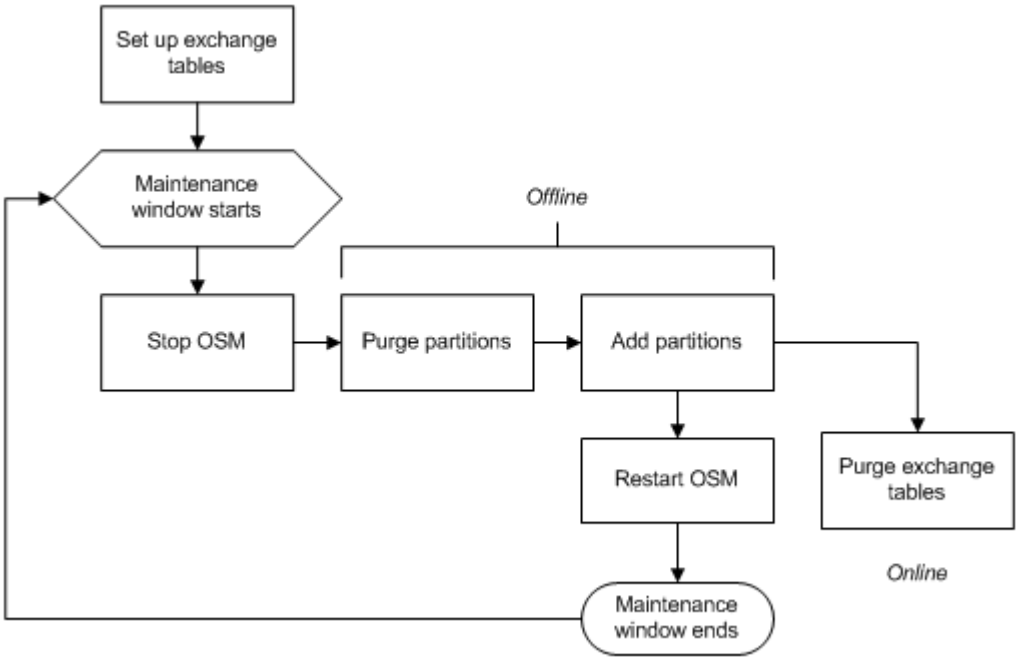
## Purging Partitions (Online or Offline)

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 IO 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."


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Figure 7-15 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**.


Copyright © 2009, 2025, Oracle and/or its affiliates.

## 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 7-16:

- 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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Figure 7-16 Purging Partitions That Contain Orders That Must be Excluded from Purging**

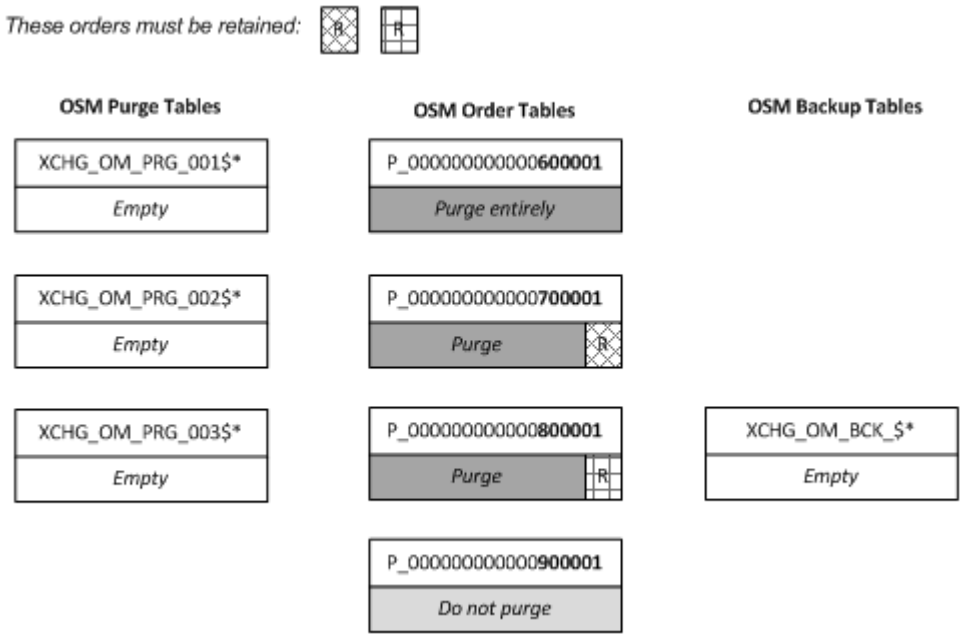



Figure 7-17 to Figure 7-21 show how **purge_partitions** purges these partitions step by step:

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_000000000000600001, P_000000000000700001 and P_000000000000800001 by exchanging them with the purge tables.
3. It drops partitions P_000000000000600001 and P_000000000000700001, 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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Figure 7-17 Step 1: Back Up Retained Orders**

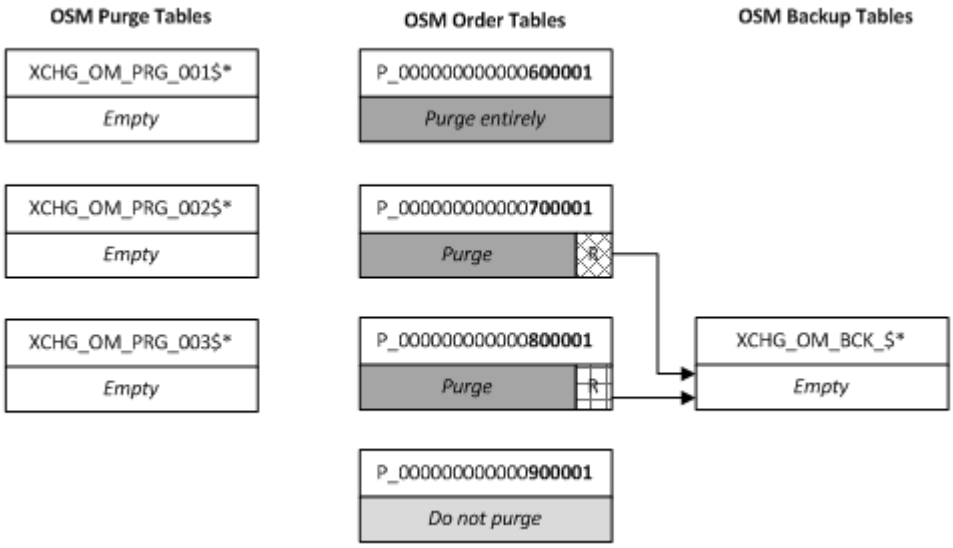
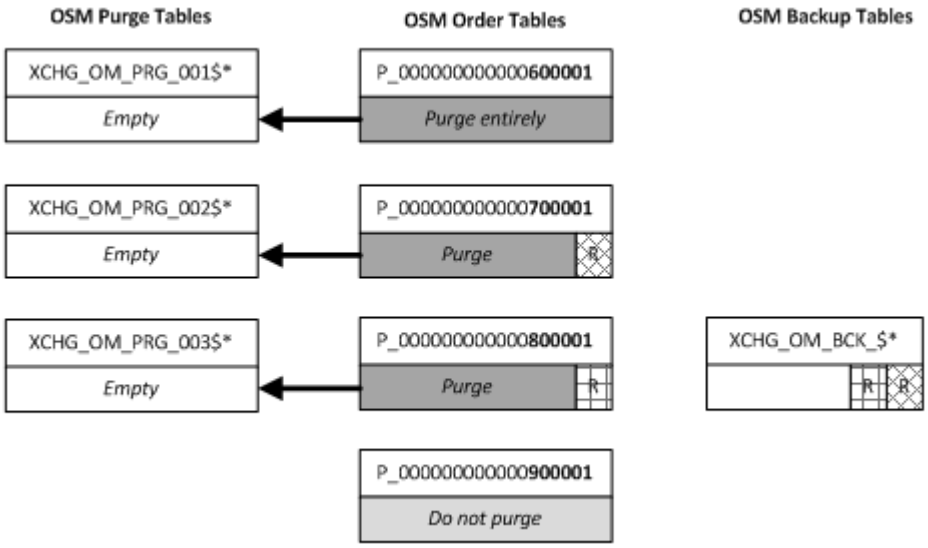



**Figure 7-18 Step 2: Purge the Partitions**





Copyright © 2009, 2025, Oracle and/or its affiliates.

**Figure 7-19 Step 3: Drop the Partitions**

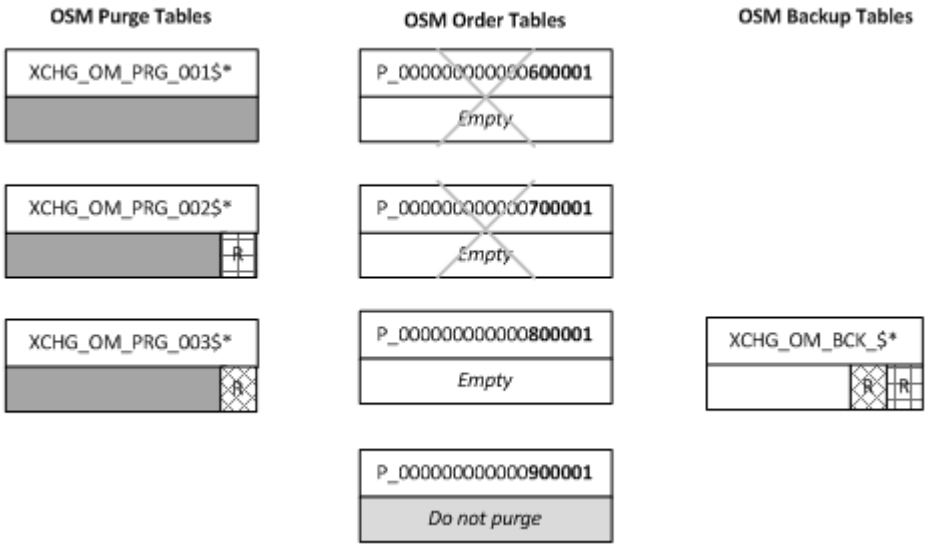
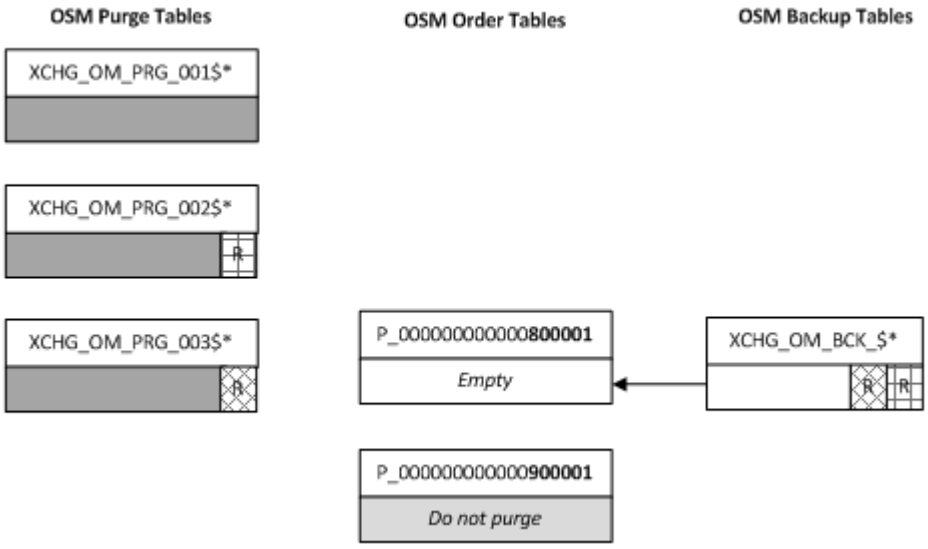


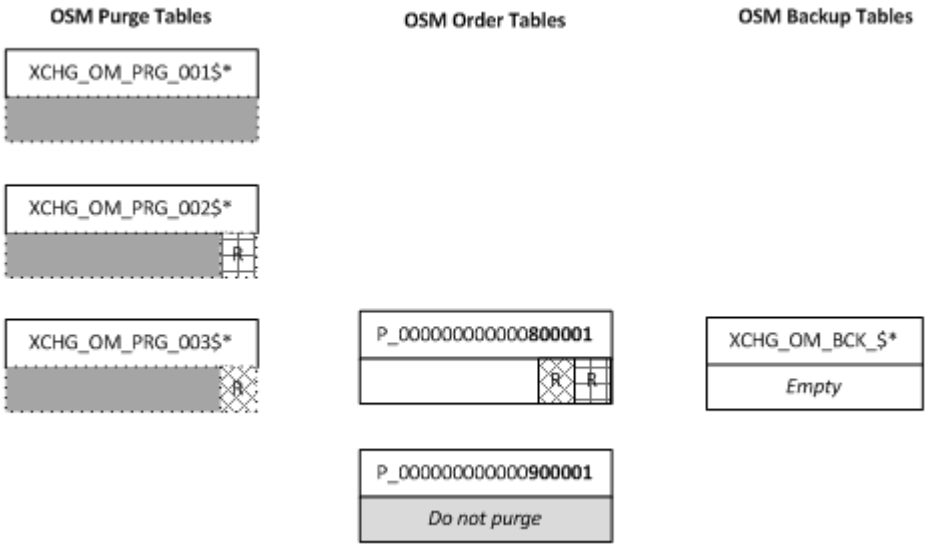

**Figure 7-20 Step 4: Restore the Retained Orders from the Backup Tables**





Copyright © 2009, 2025, Oracle and/or its affiliates.

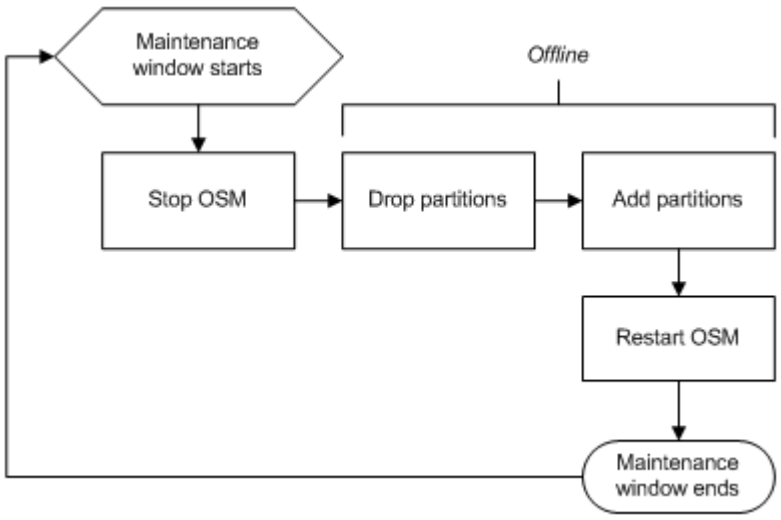
**Figure 7-21 Step5: (Optional) Purge the Purge Tables to Reclaim Space**




## 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 7-22. Exchange tables are not used when you drop partitions.

**Figure 7-22 Dropping and Adding 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


Copyright © 2009, 2025, Oracle and/or its affiliates.

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 (Offline only)."

**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 run 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)."

# Reclaiming Unused Space in Volatile Tables

Some volatile tables in the OSM database can grow to take up an inappropriate amount of space. To determine whether this is happening in your database, you can use the Segment Advisor in the database to determine the tables with the most free space, to help you decide whether to reclaim that free space. For more information about the Segment Advisor, see *Oracle Database Administrator's Guide*.

The tables that are most likely to have free space are:

- OM_ORDER_STATE_PENDING
- OM_JMS_EVENT

If you find that these tables have free space, you can reclaim the space using the following commands:

```
alter table OM_ORDER_STATE_PENDING enable row movement;
alter table OM_ORDER_STATE_PENDING shrink space;

alter table OM_JMS_EVENT enable row movement;
alter table OM_JMS_EVENT shrink space;
```


Copyright © 2009, 2025, Oracle and/or its affiliates.

These commands must be run only offline, when OSM is in a maintenance window. The commands may take several minutes to run.

# 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.

> **Note**
>
> 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:

```
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);
```

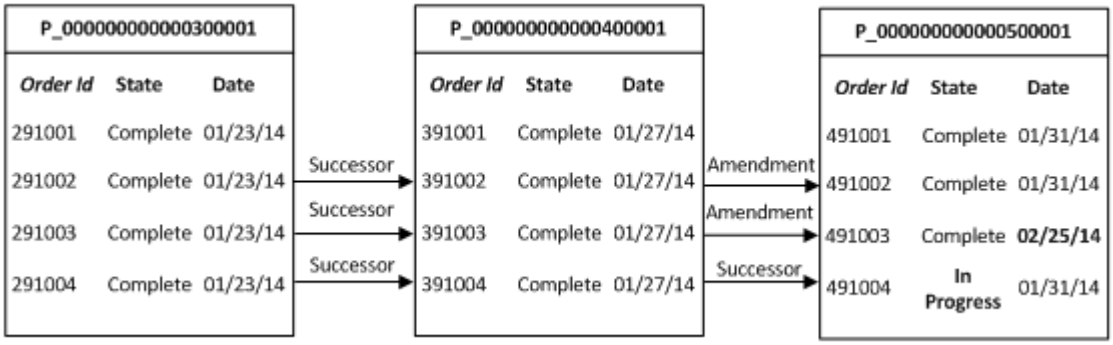
Figure 7-23 shows some of the orders in partition P_000000000000300001 and the related orders:

- 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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

- 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 7-23 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**


Copyright © 2009, 2025, Oracle and/or its affiliates.

- **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;

STATUS  EST_OR_ACTUAL_END_DATE PERCENT_COMPLETE PURGE_RATE PARALLELISM
------  -------------------- ---------------- ---------- -----------
RUNNING    15-09-2015 08:23:39            95.45       1911          16
```

## 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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

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 3xN 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.

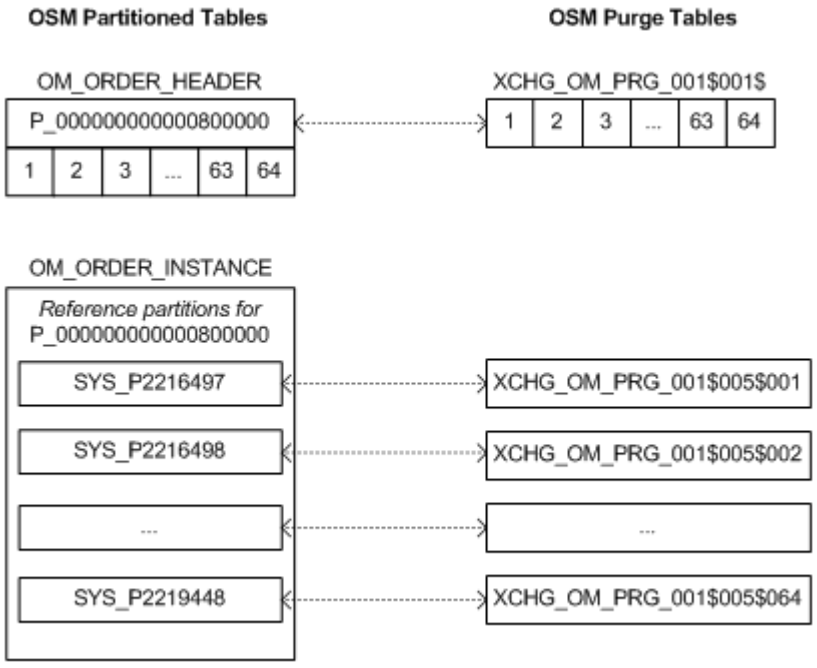

Copyright © 2009, 2025, Oracle and/or its affiliates.

- *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.

**Figure 7-24 Partition to Purge Table Mapping Example**




## 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:


Copyright © 2009, 2025, Oracle and/or its affiliates.

- *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 runs 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 procedure 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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

- 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 running the **estimate_ptn_purged_space** function:

- Exchange tables must be created before calling this function.
- This function can be performed offline or online.
- This function assumes that you will run **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.

```
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).

```
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

The main components to a deployed cartridge are:

- 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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

- The dynamic order data that is populated in the OSM database whenever an order is created and as it is being processed.

> **Note**
>
> OSM does not create ear files for automation plug-ins. The WebLogic server console does not display automation plug-in ear files. Use the console logs to debug issues.

Your primary goals should be to minimize the memory needs and startup time of OSM and 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 consumes memory resources and takes time to initialize on startup. You can minimize the memory needs and startup time of OSM by undeploying old cartridges that are no longer required from the run-time production environment.

To undeploy and redeploy cartridges quickly online, use Fast Undeploy instead of conventional undeploy.

## 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. Use the following statement to query the database for undeployed cartridges:


Copyright © 2009, 2025, Oracle and/or its affiliates.

```
select * from om_cartridge where status = 'UNDEPLOYED';
```

To purge all cartridges that were undeployed using Fast Undeploy, use the following statement:

```
exec om_cartridge_pkg.drop_obsolete_cartridges;
```

You can run this procedure when OSM is online or offline. This procedure does not purge any cartridges associated with orders and it does not purge any component cartridges unless all associated solution cartridges are also selected for purge. The DBMS output displays the cartridges that were purged and, for those cartridges with an UNDEPLOYED status but not purged, the reason the procedure did not purge the cartridge. For more information about DBMS output, see "DBMS Output."

To purge a single undeployed cartridge, use the following statement:

```
exec om_cartridge_pkg.drop_cartridge(cartridge_id);
```

You can run this procedure only when OSM is offline.

# Configuration Parameters

The following configuration parameters affect partition maintenance operations. These parameters are defined in the **om_parameter** table.

> **Note**
>
> You can override the parameters marked with (*) for a specific partitioning realm in the partitioning realm configuration file. OSM persists partitioning realm-specific parameters in the **om_partitioning_realm_param** table. For more information, see "Partitioning Realms."

- 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**


Copyright © 2009, 2025, Oracle and/or its affiliates.

  - **purge_commit_count**
- **deferred_segment_creations** (Oracle Database initialization parameter)

## range_partition_size

This parameter is present in both the **om_partitioning_realm_param** table and **om_parameter** table. The value in **om_partitioning_realm_param** is specific to a partitioning realm and takes precedence over the value in table **om_parameter**. This parameter specifies the size of new partitions.

The initial value in the **om_parameter** table for this parameter is specified during installation. You can change it with the following SQL statement, where N is the new value (for example 100000):

```
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.

The value of this parameter in table **om_partitioning_realm_param** is inserted and updated by changes in partitioning realm configuration xml. Refer the Partitioning Realm section for details.

## 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).

```
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-run **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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

## degree_of_parallelism

Specifies the default degree of parallelism for statements that are run 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


Copyright © 2009, 2025, Oracle and/or its affiliates.

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


Copyright © 2009, 2025, Oracle and/or its affiliates.

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."

> **Note**
>
> 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.

**Example 1 (temporal affinity disabled)**: In this example, the retention period is 10 days and the row-based order purge runs daily. The purge procedure runs at 10:30 PM with the **a_order_states** argument set to closed orders only (**v_closed_orders**) and the **a_delete_before** argument set to **sysdate-10** (the current date/time minus 10 days). This purges all orders that were closed 10 days ago before 10:30 PM. If 60% of orders close within the same day and 30% close on the next day before 10:30 PM then 90% of orders close within 2 days.

If temporal affinity is disabled, closed orders are purged as follows. For simplicity, ignore the 10% of orders that are closed slowly over several days and therefore they are purged at a slower rate.

- No orders are purged on days 1 to 10 because they are all either open or in the 10 day retention period.


Copyright © 2009, 2025, Oracle and/or its affiliates.

- Orders purged at 10:30 PM on day 11: All orders that closed before 10:30 PM on day 1 (60% of the orders created on day 1).
- Orders purged at 10:30 PM on day 12: All orders that closed before 10:30 PM on day 2 (60% of the orders created on day 2 and 30% of orders created on day 1).
- Orders purged at 10:30 PM on day 13: All orders that closed before 10:30 PM on day 3 (60% of the orders created on day 3 and 30% of orders created on day 2).
- …
- Orders purged at 10:30 PM on day N: All orders that closed before 10:30 PM on day N-10 (60% of the orders created on day N-10 and 30% of orders created on day N-11).

**Example 2 (temporal affinity enabled and 90% of the orders closed within 2 days)**: In the previous example, if the **purge_policy_time_to_close_wait**=1 (1 day), purging would be delayed by one day. 90% of the orders created on a day would be purged at the same time as the orders that were created and closed on the same day. The purge procedure runs at 10:30 PM and the same **a_order_states** and **a_delete_before** settings are configured in the same way as example 1. However, this configuration purges all orders that were created 11 days before 10:30 PM and were closed 10 days before 10:30 PM. The creation date criterion is based on 1 day time-to-close wait and 10 days retention period.

- No orders are purged on days 1 to 10 because they are all either open or in the 10 day retention period.
- No orders purged on day 11: The orders closed on day 1 are out of retention but they have to wait an extra day.
- Orders purged at 10:30 PM on day 12: All orders created before 10:30 PM on day 1 and closed before 10:30 PM on day 2 (90% of the orders created on day 1).
- Orders purged at 10:30 PM on day 13: All orders created before 10:30 PM on day 2 and closed before 10:30 PM on day 3 (90% of the orders created on day 2).
- …
- Orders purged at 10:30 PM on day N: All orders created before 10:30 PM on day N-11 and closed before 10:30 PM on day N-10 (90% of the orders created on day N-11).

**Example 3 (temporal affinity enabled and 90% of the orders closed within 3 days)**: This is similar to the previous example, except that we want a 2 day delay instead of 1 (**purge_policy_time_to_close_wait=2**).

- No orders are purged on days 1 to 10 because they are all either open or in the 10 day retention period.
- No orders purged on day 11: The orders closed on day 1 are out of retention but they have to wait.
- No orders purged on day 12: The orders closed on day 1 and 2 are out of retention but they have to wait.
- Orders purged at 10:30 PM on day 13: All orders created before 10:30 PM on day 1 and closed before 10:30 PM on day 3 (90% of the orders created on day 1).
- Orders purged at 10:30 PM on day 14: All orders created before 10:30 PM on day 2 and closed before 10:30 PM on day 4 (90% of the orders created on day 2).
- Orders purged at 10:30 PM on day N: All orders created before 10:30 PM on day N-12 and closed before 10:30 PM on day N-10 (90% of the orders created on day N-12).


Copyright © 2009, 2025, Oracle and/or its affiliates.

## 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 11*g*R2. 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 run the following statements:

```
alter system set deferred_segment_creation=false scope=both sid='*';
execute dbms_space_admin.materialize_deferred_segments('<schema_name>');
```

## purge_commit_count

Specifies how frequently each purge job issues a commit command. For example, the value 10 means that 10 orders are purged before a commit is done. Unless you perform extensive performance purge tests to determine the optimal value for this parameter, Oracle recommends that you leave it at the default value. If not present, the value 10 is used.

# 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 processing times and error traces.

> **Note**
>
> The DBMS output is sent to the client at the end of the operation. Oracle Database does not provide any mechanism to flush output during the procedure.

For example, if you use SQL*Plus:


Copyright © 2009, 2025, Oracle and/or its affiliates.

```
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

Order purge procedures allow you to specify the following purge criteria:

**Table 7-3 Order Purge Criteria**

| Criteria | Parameters | Partition-Based Purge | Row-Based Order Purge |
|---|---|---|---|
| Order state | **a_order_states** | Yes | Yes |
| Order timestamp | **a_delete_before** | Yes | Yes |
| Order ID range | **a_order_id_lt** | Yes | Yes |
| Order type | **a_order_type_mnemonic** | No | Yes |
| Order source | **a_order_source_mnemonic** | No | Yes |
| Order cartridge | **a_namespace_mnemonic**<br>**a_version_mnemonic** | No | Yes |

Table 7-4 and Table 7-5 that follow represent order states and pre-defined aggregate order states that can be supplied to purge script. You can choose to supply order state, pre-defined aggregate order state, or a custom aggregate state to purge script. Order states and predefined aggregate order states are defined in the **om_new_purge_pkg** package.

- Example of an order state to purge cancelling orders:

  **om_new_purge_pkg.v_cancelling_orders**
- Example of a pre-defined aggregate order state to purge all closed and cancelled orders:

  **om_new_purge_pkg.v_closed_or_cancelled_orders**
- Example of a custom aggregate order state to purge failed and aborted orders:

  **om_new_purge_pkg.v_failed_orders + om_new_purge_pkg.v_aborted_orders**

> **Note**
>
> While forming a custom aggregate state, ensure the following:
>
> - Use only order state, but not the pre-defined aggregate order states.
> - Do not use the same state twice.

Table 7-4 shows the order state constants and their corresponding values.

**Table 7-4 Order State Constants and Values**

| Constant | Value |
|---|---|
| v_completed_orders | 1 |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 7-4 (Cont.) Order State Constants and Values**

| Constant | Value |
|---|---|
| v_aborted_orders | 2 |
| v_not_started_orders | 4 |
| v_suspended_orders | 8 |
| v_cancelled_orders | 16 |
| v_wait_for_revision_orders | 32 |
| v_failed_orders | 64 |
| v_waiting_orders | 128 |
| v_in_progress_orders | 256 |
| v_amending_orders | 512 |
| v_cancelling_orders | 1024 |

Table 7-5 shows the pre-defined aggregate order states and their corresponding values.

**Table 7-5 Predefined Aggregate Order States and Values**

| Constant | Value |
|---|---|
| v_closed_orders | 3 (v_completed_orders + v_aborted_orders) |
| v_closed_or_cancelled_orders | 19 (v_completed_orders + v_aborted_orders + v_cancelled_orders) |
| v_not_running_orders | 252 (v_waiting_orders + v_failed_orders + v_wait_for_revision_orders + v_cancelled_orders + v_suspended_orders + v_not_started_orders) |
| v_compensating_orders | 1536 (v_cancelling_orders + v_amending_orders) |
| v_running_orders | 1792 (v_compensating_orders + v_in_progress_orders) |
| v_open_orders | 2044 (v_running_orders + v_not_running_orders) |
| v_all_orders | 2047 (v_open_orders + v_closed_orders) |

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 7-6 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 7-6 Order Purge Based on Timestamp, Order State, and Order Status**

| a_order_states | Order Status | Timestamp |
|---|---|---|
| v_all_orders | N/A | a_ord_creation_date |
| v_closed_orders | 7 (complete) or 9 (aborted) | a_ord_completion_date |
| v_completed_orders | 7 | a_ord_completion_date |
| v_aborted_orders | 9 | a_ord_completion_date |
| v_suspended_orders | 2 | a_ord_txn_completion_date |
| v_cancelled_orders | 3 | a_ord_txn_completion_date |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 7-6 (Cont.) Order Purge Based on Timestamp, Order State, and Order Status**

| a_order_states | Order Status | Timestamp |
|---|---|---|
| v_in_progress_orders | 4 | a_ord_txn_completion_date |
| v_amending_orders | 5 | a_ord_txn_completion_date |
| v_cancelling_orders | 6 | a_ord_txn_completion_date |
| v_wait_for_revision_orders | 8 | a_ord_txn_completion_date |
| v_failed_orders | 10 | a_ord_txn_completion_date |
| v_not_started_orders | 1 (not started) | a_ord_creation_date |

# Parallel Execution

The **om_part_maintain** API performs many operations in parallel:

- Parallel queries and most DML and DDL statements are run in parallel using parallel servers, which apply multiple CPU and I/O resources to a single database operation. Examples include copying orders into the backup tables and rebuilding unusable indexes.
- Some operations are run 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 processing 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 run in parallel.

The degree of parallelism can be:

- Greater than 1: Statements that can be run in parallel are run with the specified degree of parallelism.
- 1: All statements are run serially.
- 0: The degree of parallelism is computed by the database and it can be 2 or greater. Statements that can be run 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_parallelism** unspecified, OSM uses the default parallelism configured by these parameters:

- **degree_of_parallelism**
- **degree_of_parallelism_rebuild_indexes**
- **degree_of_parallelism_rebuild_xchg_indexes**

# Concurrency Restrictions

Exchange table, partition management, and purge procedures acquire an exclusive user lock to prevent concurrent processing of other management procedures, which could result in unrecoverable errors. Each OSM schema uses a different lock specifically for this package.


Copyright © 2009, 2025, Oracle and/or its affiliates.

The lock is released automatically at the end of operation. The database also releases user locks automatically when a session terminates.

Specifically, the following procedures acquire an exclusive lock to prevent concurrent operations:

- **setup_xchg_tables**
- **drop_xchg_tables**
- **purge_xchg_prg_tables**
- **purge_partitions**
- **drop_empty_partitions**
- **drop_partitions**
- **add_partition** and **add_partitions**
- **equipartition**
- **purge_orders**
- **select_orders**
- **purge_selected_orders**
- **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


Copyright © 2009, 2025, Oracle and/or its affiliates.

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, an exception is thrown. 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-run **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 run automatically when you run **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 run 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.

```
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.

```
procedure set_dop_rebuild_indexes(a_parallelism binary_integer);
```


Copyright © 2009, 2025, Oracle and/or its affiliates.

## 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.

```
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.

```
procedure add_partition(a_tablespace varchar2 default null,
                        a_realm_mnemonic varchar2 default null,
                        a_force boolean default false);
```

The implementation is equivalent to this call:

```
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).

```
procedure add_partitions(
        a_count positiven,
        a_tablespace varchar2 default null,
        a_realm_mnemonic varchar2 default null,
        a_force boolean default false);
```

The upper bound of each new partition is the greatest partition upper bound plus the value of the **range_partition_size** parameter. If found, the **range_partition_size** for the realm is used (found in table **om_partitioning_realm_param**). If not found, the **range_partition_size** in table **om_parameter** is used. The upper bound is used in the partition name. For example, if the new partition's upper bound is 100,000, the partition name is P_000000000000100000 (always formatted to 18 characters).

This procedure inserts a new row into table **om_order_id_block** to represent the range of **order_seq_ids** for the new partition. In addition to the order ID range, the order ID block contains the status (for example, **AVAILABLE** for the newly added partition), the dbinstance (-1 until the block changes to **ACTIVE**), and the partitioning realm associated with the block.

You must run this procedure offline. Running this procedure online 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).


Copyright © 2009, 2025, Oracle and/or its affiliates.

- **a_realm_mnemonic**: The partitioning realm mnemonic (case-insensitive) that the new partition belongs to. If the value is **null**, the partition is assigned to the **default_order** realm. If the realm is disabled, an error occurs; you can ignore this error by entering **true** in the **a_force** argument.
- **a_force**: If the value of this parameter is **true**, partitions can be added for disabled realms. This is useful because partitioning realms are often created in a disabled state, therefore partitions can be added for the realm before enabling the partitioning realm.

**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)

```
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."
- The partition belongs to a partitioning realm with a purge strategy of **PARTITION-BASED** or **ANY**. For more information, see the purge strategy section in "Partitioning Realms."

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 running this procedure.

This procedure must be run 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


Copyright © 2009, 2025, Oracle and/or its affiliates.

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.

```
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 run online. However, in this case, you might experience high contention due to exclusive locks acquired by Oracle Database. Oracle recommends that you run 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 the table below.

```
select partition_name, high_value
from user_tab_partitions
where table_name = 'OM_ORDER_HEADER'
order by partition_name;
```


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 7-7 Example: Dropping Empty Partitions**

| PARTITION_NAME | HIGH_VALUE | Empty? |
|---|---|---|
| P_000000000000100001 | 100001 | Yes |
| P_000000000000200001 | 200001 | -- |
| P_000000000000300001 | 300001 | Yes |
| P_000000000000400001 | 400001 | Yes |
| P_000000000000500001 | 500001 | -- |

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:

```
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 the table below.

**Table 7-8 Example: Dropping Newly Added Partitions**

| PARTITION_NAME | HIGH_VALUE | Mapped Order IDs | Empty? |
|---|---|---|---|
| P_000000000000200001 | 200001 | 1-200000 | -- |
| P_000000000000400001 | 400001 | 200001-400000 | Yes |
| P_000000000000500001 | 500001 | 400001-500000 | -- |
| P_000000000000600001 | 600001 | 500001-600000 | Yes |
| P_000000000000700001 | 700001 | 600001-700000 | Yes |
| P_000000000000800001 | 800001 | 700001-800000 | Yes |

Run the following statement:

```
execute om_part_maintain.drop_empty_partitions(
    a_order_id_ge => 500001);
```

## om_part_maintain.purge_partitions (Online or Offline)

```
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:


Copyright © 2009, 2025, Oracle and/or its affiliates.

- If run 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 run 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.
  - The partition belongs to a partitioning realm with a purge strategy of **PARTITION-BASED** or **ANY**. For more information, see the purge strategy section in "Partitioning Realms."

Oracle recommends that you back up the OSM schema prior to running this procedure and that you gather statistics after you finished purging.

If you run this procedure online, you might experience high contention due to exclusive locks acquired by Oracle Database. Oracle recommends that you run this procedure either offline or off-peak.

If you run 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 run in parallel as specified by the **a_parallelism** argument.

If this procedure is run 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_00000000001000001, P_00000000002000001, and P_00000000003000001 then P_00000000001000001 will be purged first from all partitioned tables, then P_00000000002000001 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:


Copyright © 2009, 2025, Oracle and/or its affiliates.

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 run 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 run 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$xchg_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-run **setup_xchg_tables**.

All disabled constraints are re-enabled at the end (with NOVALIDATE for performance reasons).


Copyright © 2009, 2025, Oracle and/or its affiliates.

The parameters are:

- **a_online**: Specifies whether this procedure is being run 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 running this procedure once 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. Run 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. Run the **setup_xchg_tables** procedure.
- **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-run the **setup_xchg_tables** procedure.
- **ORA-20166**: There is another in-progress maintenance operation.


Copyright © 2009, 2025, Oracle and/or its affiliates.

- **ORA-20170**: Failed to suspend database jobs.
- **ORA-20171**: The procedure was run 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 run **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:

```
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:

```
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_partitions_2 (Online or Offline)

```
procedure purge_partitions_2(
        a_online            boolean,
        a_delete_before     date,
        a_order_states      om_t_numbers default
om_t_numbers(om_new_purge_pkg.v_completed_orders, om_new_purge_pkg.v_aborted_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 validates the supplied order states and then calls purge_partitions:

- Valid absolute order states:
  - **om_new_purge_pkg.v_completed_orders**
  - **om_new_purge_pkg.v_aborted_orders**
  - **om_new_purge_pkg.v_not_started_orders**


Copyright © 2009, 2025, Oracle and/or its affiliates.

  - **om_new_purge_pkg.v_suspended_orders**
  - **om_new_purge_pkg.v_cancelled_orders**
  - **om_new_purge_pkg.v_wait_for_revision_orders**
  - **om_new_purge_pkg.v_failed_orders**
  - **om_new_purge_pkg.v_waiting_orders**
  - **om_new_purge_pkg.v_in_progress_orders**
  - **om_new_purge_pkg.v_amending_orders**
  - **om_new_purge_pkg.v_cancelling_orders**
- Valid aggregated order states:
  - **om_new_purge_pkg.v_closed_orders**
  - **om_new_purge_pkg.v_closed_or_cancelled_orders**
  - **om_new_purge_pkg.v_not_running_orders**
  - **om_new_purge_pkg.v_compensating_orders**
  - **om_new_purge_pkg.v_running_orders**
  - **om_new_purge_pkg.v_open_orders**
  - **om_new_purge_pkg.v_all_orders**

> **Note**
>
> You can supply either multiple absolute order states or only one of aggregated order states.

The procedure exits with an error if invalid input is supplied.

The following is an example of valid input, with one or more absolute states:

```
purge_partitions_2(
        a_order_states      om_t_numbers(om_new_purge_pkg.v_completed_orders))
purge_partitions_2(
        a_order_states      om_t_numbers(om_new_purge_pkg.v_completed_orders,
om_new_purge_pkg.v_failed_orders))
purge_partitions_2(
        a_order_states      om_t_numbers(om_new_purge_pkg.v_suspended_orders,
om_new_purge_pkg.v_cancelled_orders, om_new_purge_pkg.v_wait_for_revision_orders,
om_new_purge_pkg.v_failed_orders))
```

The following is an example of valid input, with only one predefined aggregate order state:

```
purge_partitions_2(
        a_order_states      om_t_numbers(om_new_purge_pkg.v_closed_orders))
purge_partitions_2(
        a_order_states      om_t_numbers(om_new_purge_pkg.v_all_orders))
```

The following is an example of invalid input, with one or more absolute states, along with one or more aggregate states:

```
purge_partitions_2(
        a_order_states      om_t_numbers(om_new_purge_pkg.v_completed_orders,
om_new_purge_pkg.v_closed_or_cancelled_orders))
```


Copyright © 2009, 2025, Oracle and/or its affiliates.

The following is an example of invalid input, with more than one aggregate state:

```
purge_partitions_2(
        a_order_states      om_t_numbers(om_new_purge_pkg.v_all_orders,
om_new_purge_pkg.v_closed_orders))
```

## om_part_maintain.purge_entire_partition (Online or Offline)

```
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:

```
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 run 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 running this procedure once 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)

```
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 running **om_part_maintain.purge_partitions**, therefore refer to the purge partitions API reference for a description of the parameters, exit conditions, and possible exceptions.


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Example (estimate the space reclaimed by purging all orders that were closed at least 180 days ago)**:

```
declarebegindbms_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.

```
declarebegindbms_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 run this procedure because backup tables are purged automatically when all order data is restored. The implementation runs 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 runs 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,
```


Copyright © 2009, 2025, Oracle and/or its affiliates.

```
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 null,
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.

Running this procedure is equivalent to running **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 7-9 describes the possible outcomes. The **a_status** output parameter is set accordingly.

**Table 7-9 Possible Outcomes**

| Outcome | a_status | OM_ORDER_ID_FOR_PURGE |
|---|---|---|
| No orders satisfy the purge criteria. | om_new_purge_pkg.v_status_nopurge | Empty |
| The purge finished successfully and all orders that satisfied the purge criteria were purged. | om_new_purge_pkg.v_status_finished | Cleared |
| 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. | N/A | Contains all order Ids that satisfy the purge criteria. |
| The purge finished prematurely because the expiration date specified by **a_stop_date** was reached. | om_new_purge_pkg.v_status_expired | Cleared |
| The purge was stopped by the user (using **stop_purge**), and it can be resumed using **resume_purge**. | om_new_purge_pkg.v_status_stopped | Contains all order Ids that satisfy the purge criteria. |

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 "Specifying Purge Criteria" for more information.
- **a_order_states**: Only orders with one of these states are eligible for purging. See "Specifying Purge Criteria" 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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

- **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_lt** 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 run serially (with a single database job). See "Parallel Execution" for more information.
- **a_commit_count**: Specifies how often each job should issue a commit command. Unless you performed extensive performance purge tests to determine the optimal value for this parameter, it is recommended that you leave it **null**. If the value is **null**, the job uses the commit count that is configured in the **purge_commit_count** parameter.

**Example**: The following procedure 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.

```
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)

```
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 null,
    a_parallelism binary_integer default null);
```


Copyright © 2009, 2025, Oracle and/or its affiliates.

This procedure schedules an operation of **purge_orders** using the **dbms_job** package.

The **a_start_date** parameter specifies the start date and time.

The **a_version_mnemonic** parameter 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.

The **a_cartridge_id** parameter cannot be used in combination with **a_namespace_mnemonic**. That is, both **a_cartridge_id** and **a_namespace_mnemonic** cannot have '**not null**' values at the same time.

## om_new_purge_pkg.select_orders (Online or Offline)

```
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 operation of **purge_orders**, and you do not want to run multiple purges. In this case:

- You can populate **OM_ORDER_ID_FOR_PURGE** piecemeal by running **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, run **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).

```
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;
```


Copyright © 2009, 2025, Oracle and/or its affiliates.

## om_new_purge_pkg.purge_selected_orders (Online or Offline)

```
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 null,
    a_parallelism  binary_integer default null);
```

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 a commit command. Unless you performed extensive performance purge tests to determine the optimal value, Oracle recommends that you leave it **null**. If it is **null**, it uses the commit count value that is configured in the **purge_commit_count** parameter.
- **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 run serially (with a single database job). See "Parallel Execution" for more information.

## 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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

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.

## om_handler_pkg.abort_order

You can abort an order using below procedure:

```
om_handler_pkg.abort_order

procedure abort_order
(
    a_order_seq_id          IN  integer,
    a_author                IN  varchar2 DEFAULT 'Dummy',
    a_reason                IN  CLOB DEFAULT 'Aborted order');
```

The parameters are:

- `a_order_seq_id` (mandatory): order id to be aborted.
- `a_author` (optional): user name aborting order.
- `a_reason` (optional): reason to abort order

After performing the **om_handler_pkg.abort_order** procedure, do an explicit commit. Aborting an order removes active threads, automation context, amendment key and so on for the given


Copyright © 2009, 2025, Oracle and/or its affiliates.

order, and changes the order state to aborted (order state 9). Note that abort only moves the order to aborted state, it does not purge the order from the system.

The following are some usage examples:

- ```
  execute om_handler_pkg.abort_order(a_order_seq_id => 10); commit;
  ```

- ```
  begin
   om_handler_pkg.abort_order(a_order_seq_id => 10);
   commit;
  end;
  ```

- ```
  declare
   c_size integer := 1000;
   c_count integer := 0;
  begin
   for o in (select order_seq_id from om_order_header
  partition(P_00000000000000011))
   loop
    om_handler_pkg.abort_order(a_order_seq_id => o.order_seq_id);
    c_count := c_count + 1;
    if mod(c_count, c_size) = 0 then
     commit;
    end if;
   end loop;
  end;
  ```

> **Note**
>
> Aborting an order can be executed both online and offline.

# Advanced Procedures

This section provides information about advanced procedures.

## om_part_maintain.backup_selected_ords (Offline)

```
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.


Copyright © 2009, 2025, Oracle and/or its affiliates.
```

**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.

```
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.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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

```
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.

```
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)

```
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 operation of **purge_partitions** or **drop_partitions**.

## om_part_maintain.rebuild_unusable_indexes (Online or Offline)

```
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);
```


Copyright © 2009, 2025, Oracle and/or its affiliates.

```
procedure rebuild_unusable_indexes(
        a_indexes dbms_sql.varchar2s,
        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**.

## 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)


Copyright © 2009, 2025, Oracle and/or its affiliates.

- **om_part_maintain.c_rebuild_idx_recreate_global** (RECREATE GLOBAL).

For more information see **purge_policy_rebuild_unusable_indexes**.

## om_part_maintain.sys$undo_restore_table (Offline)

This is an internal procedure that should be used strictly for recovery purposes.

```
procedure sys$undo_restore_table(
        a_table_name          varchar2,
        a_parallelism binary_integer default null);
```

> **Note**
>
> 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$undo_restore_table** to delete the partially restored rows from that table. This leaves the table in a clean state and prevents unique key violations when another restore attempt is made. The OM_SQL_POINTER table shows the error and points to the line in OM_SQL_LOG. That line includes a call to **om_part_maintain.sys$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, run **om_part_maintain.sys$undo_restore_table** manually.

## 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 7-10 lists and describes the columns in the OM_AUDIT_PURGE_ALL table.


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 7-10 OM_AUDIT_PURGE_ALL Table**

| Column | Datatype | NULL | Description |
|---|---|---|---|
| PURGE_SEQ_ID | NUMBER(18) | NOT NULL | The system-generated unique ID assigned to the purge operation. |
| OPERATION_NAME | VARCHAR(64) | NOT NULL | The name of the operation (for example, **om_new_purge_pkg.purge_orders)** |
| STATUS | VARCHAR2(20) | NOT NULL | The purge status.<br>• STARTED: This short-lived initial status occurs when the purge operation is selecting the orders to be purged.<br>• RUNNING: This status occurs when the purge operation is purging the orders.<br>• STOPPED: This status occurs when the **om_new_purge_pkg.stop_purge** stops a purge operation.<br>• EXPIRED: This status occurs when the purge exceeds the end of the purge window specified by the **a_stop_date** argument.<br>• FINISHED: This status occurs when the purge finishes successfully. All orders that satisfy the purge criteria are purged.<br>• FINISHED_WITH_ERROR: This status occurs when the purge finishes with some errors. Some orders that satisfy the purge criteria are not purged.<br>• FAILED: This status occurs when the purge failed. No orders are purged.<br>• NO_PURGE: This status occurs when no order satisfies the purge criteria. |
| START_DATE | DATE | NOT NULL | The start date and time of the purge. |
| **PURGE_DURATION_MINUTES** | NUMBER(9) | NOT NULL | 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. |
| EST_OR_ACTUAL_END_DATE | DATE | N/A | 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:<br>• STARTED: The estimated date and time is NULL because the purge rate is unknown.<br>• 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.<br>• 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.) |
| PURGE_WINDOW_END_DATE | DATE | N/A | The end of the purge window as specified by the **a_stop_date** argument of the purge procedure. |
| STOPPED_DATE | DATE | N/A | The last date and time when the purge was stopped. |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 7-10 (Cont.) OM_AUDIT_PURGE_ALL Table**

| Column | Datatype | NULL | Description |
|---|---|---|---|
| RESUMED_DATE | DATE | N/A | The last date and time when the purge was resumed. |
| SELECTED_ORDER_COUNT | NUMBER(9) | NOT NULL | The number of orders selected for purge. |
| SELECTED_ORDER_PURGE_COUNT | NUMBER | NOT NULL | The number of orders selected for purge that have been purged so far (cascaded deletes are excluded). |
| PERCENT_COMPLETE | NUMBER | NOT NULL | The completion percent based on SELECTED_ORDER_COUNT and SELECTED_ORDER_PURGED_COUNT. |
| ORDERS_PURGED_PER_MINUTE | NUMBER | NOT NULL | The purge rate per minute. |
| ORDERS_INJECTED_PER_MINUTE | NUMBER | NOT NULL | The rate per minute that orders are created in OSM while the order purge runs. This value allows you to identify any purge overlaps with high order volume periods. |
| PARALLELISM | NUMBER | NOT NULL | 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. |
| ERROR_MESSAGE | VARCHAR2(4000) | N/A | The reason of failure if the purge failed or finished with errors. |

## OM_AUDIT_PURGE_LATEST

The **OM_AUDIT_PURGE_LATEST** views is identical 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 7-11 describes the **OM_AUDIT_PURGE** table. This table is partitioned by START_DATE. Each partition corresponds to a different month.

**Table 7-11 OM_AUDIT_PURGE Table**

| Column | The system-generated unique ID assigned to the purge operation. | NULL | Description |
|---|---|---|---|
| PURGE_SEQ_ID | The name of the operation (for example, **om_new_purge_pkg.purge_orders)** | NOT NULL | A system-generated unique ID assigned to the purge operation. |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 7-11 (Cont.) OM_AUDIT_PURGE Table**

| Column | The system-generated unique ID assigned to the purge operation. | NULL | Description |
|---|---|---|---|
| OPERATION_NAME | VARCHAR2(64) | NOT NULL | The name of the purge operation (for example, **om_new_purge_pkg.purge_orders**). |
| STATUS | VARCHAR2(20) | NOT NULL | The purge status.<br>• STARTED: This short-lived initial status occurs when the purge operation is selecting the orders to be purged.<br>• RUNNING: This status occurs when the purge operation is purging the orders.<br>• STOPPED: This status occurs when the **om_new_purge_pkg.stop_purge** stops a purge operation.<br>• EXPIRED: This status occurs when the purge exceeds the end of the purge window specified by the **a_stop_date** argument.<br>• FINISHED: This status occurs when the purge finishes successfully. All orders that satisfy the purge criteria are purged.<br>• FINISHED_WITH_ERROR: This status occurs when the purge finishes with some errors. Some orders that satisfy the purge criteria are not purged.<br>• FAILED: This status occurs when the purge failed. No orders are purged.<br>• NO_PURGE: This status occurs when no order satisfies the purge criteria. |
| START_DATE | DATE | NOT NULL | The start date and time of the purge. |
| PURGE_DURATION_SEC | NUMBER(9) | NOT NULL | The purge operation 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 time excluding idle periods. |
| INJECTED_ORDER_COUNT | NUMBER(9) | NOT NULL | The number of orders injected while the purge was running online. This value helps identify any purge overlaps with high order volume periods. |
| SELECTED_ORDER_COUNT | NUMBER(9) | NOT NULL | The number of orders selected for purge. |
| END_DATE | DATE | N/A | The date when the purge ended. This is set when the status is FINISHED, FINISHED_WITH_ERROR, EXPIRED or FAILED. |
| STOP_REQUESTED_DATE | DATE | N/A | The last date and time when the user submitted a purge stop request. |
| STOPPED_DATE | DATE | N/A | The last date and time when the purge was stopped. |
| RESUMED_DATE | DATE | N/A | The last date and time when the purge was resumed. |
| ERROR_MESSAGE | VARCHAR2(4000) | N/A | The reason of failure if the purge failed or finished with errors. |


Copyright © 2009, 2025, Oracle and/or its affiliates.

# OM_AUDIT_PURGE_ORDER

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 7-12 describes the **OM_AUDIT_PURGE_ORDER** table. This table is reference-partitioned with **OM_AUDIT_PURGE** as the parent table.

**Table 7-12 OM_AUDIT_PURGE_ORDER Table**

| Column | Datatype | NULL | Description |
|---|---|---|---|
| PURGE_SEQ_ID | NUMBER(18) | NOT NULL | The system-generated unique ID assigned to the purge operation. |
| ORDER_SEQ_ID | NUMBER(18) | NOT NULL | The order ID. |
| CASCADED_DELETE | VARCHAR2(1) | N/A | Indicates whether the order was deleted because of a cascaded delete. For example, whether this is an amendment order of a deleted base order. |
| DELETED_DATE | DATE | NOT NULL | The date and time when the order was deleted. |
| ORDER_TYPE_ID | NUMBER(9) | NOT NULL | Copied from **OM_ORDER_HEADER**. |
| ORDER_SOURCE_ID | NUMBER(9) | NOT NULL | Copied from **OM_ORDER_HEADER**. |
| REFERENCE_NUMBER | VARCHAR2(255) | NOT NULL | Copied from **OM_ORDER_HEADER**. |
| ORD_STATE_ID | NUMBER(9) | NOT NULL | Copied from **OM_ORDER_HEADER**. |
| CARTRIDGE_ID | NUMBER(6) | NOT NULL | Copied from **OM_ORDER_HEADER**. |
| ORD_CREATION_DATE | DATE | NOT NULL | Copied from **OM_ORDER_HEADER**. |
| ORD_START_DATE | DATE | NOT NULL | Copied from **OM_ORDER_HEADER**. |
| ORD_COMPLETION_DATE | DATE | NOT NULL | Copied from **OM_ORDER_HEADER**. |
| ORD_TXN_COMPLETION_DATE | TIMESTAMP | N/A | Copied from **OM_ORDER_HEADER**. |
| VERSION | NUMBER(9) | N/A | Copied from **OM_ORDER_HEADER**. |

# 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**,


Copyright © 2009, 2025, Oracle and/or its affiliates.

**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 ran 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 7-13 describes the **OM_AUDIT_PURGE_PARAM** table. This table is reference-partitioned with **OM_AUDIT_PURGE** as the parent table.

**Table 7-13 OM_AUDIT_PURGE_PARAM Table**

| Column | Datatype | NULL | Description |
|---|---|---|---|
| PURGE_SEQ_ID | NUMBER(18) | NOT NULL | The system-generated unique ID assigned to the purge operation. |
| PARAMETER_NAME | VARCHAR2(254) | NOT NULL | The parameter name. |
| PARAMETER_TYPE | VARCHAR2(1) | NOT NULL | The parameter type:<br>• P: This is a parameter of the purge procedure that specifies a purge criterion.<br>• U: This is a parameter of the purge procedure other than a purge criterion.<br>• S: This is a database session parameter.<br>• C: This is a purge-related configuration parameter. |
| PARAMETER_VALUE | VARCHAR2(255) | | The parameter value. |

# Troubleshooting and Error Handling

> **Note**
>
> In the event of failure during a purge operation, Oracle strongly recommends that you stop OSM and perform all troubleshooting and recovery operations 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 run SQL statements so that processing 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:

- **sid**: The session ID. The default value is the current session ID, for example, **sys_context**('USERENV', 'SID'). This allows concurrent processing.
- **name**: This is usually the name of the procedure that generated the SQL statement. It is useful to Oracle Support.


Copyright © 2009, 2025, Oracle and/or its affiliates.

- **line**: This is a line number used for ordering the SQL statements to be run.
- **sql_text**: The SQL statement.

SQL statements persisted in **om_sql_log** are run by **om_sql_log_pkg.exec**. This procedure runs 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. When run, the line number of the current statement is updated in the **om_sql_pointer** table. This allows you to monitor the procedure. When successful, 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-run, it resumes with the failed statement.

Therefore, you can troubleshoot and recover from a failed partition maintenance operation even if it was run 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 running the following statement:

```
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 the operation by retrieving from **om_sql_log** the current statement of each session. If the **error_code** section is populated, the operation failed.

```
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:

```
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 run 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. Run **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, run **om_sql_log_pkg.exec*(sid)*** to finish partition creation. Then run **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, run **om_part_maintain.is_equipartitioned** to check whether the schema is partitioned equally. If it is not, you can run **om_part_maintain.equipartition** to fix it.


Copyright © 2009, 2025, Oracle and/or its affiliates.

# Error Handling for drop_partitions

In the event of error, **drop_partitions** re-enables disabled foreign keys and throws the exception again. However, 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-run 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 run **rebuild_unusable_indexes** to ensure indexes are usable.
  - (Optional) You can run **purge_orphan_order_data** to delete orphan order data. Otherwise, orphan data is deleted by the next execution of **purge_partitions** or **drop_partitions**.
  - Run **om_job_pkg.resume_jobs** to resume database jobs.
- When in doubt, run **is_equipartitioned** to check whether the schema is equi-partitioned. If it is not, you can run **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, 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, 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 the procedure.

A failure may occur in these procedures:

- **sys$bpdr_backup**: Copies the orders that do not satisfy the purge criteria into the backup tables.
- **sys$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 run:
  - 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**.


Copyright © 2009, 2025, Oracle and/or its affiliates.

- **sys$purge_orphan_order_data**: Purges orphan order data (run at the end).
- **sys$purge_xchg_prg_tables**: If the **a_purge_xchg_prg_tables** argument is true, it is run at the end to purge the purge tables. It may also be run 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, the procedure failed in **sys$bpdr_purge**. Partitions are in an inconsistent state.
  - If **om_sql_log** includes INSERT statements into the backup tables, the procedure failed in **sys$bpdr_backup**. Partitions are in a consistent state.
  - If **om_sql_log** includes calls to **sys$restore_table**, the procedure failed in **sys$bpdr_restore**. Partitions contain partially restored orders.
  - If **om_sql_log** includes statements to rebuild indexes, the procedure 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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

- 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*, the procedure failed in **sys$bpdr_restore** while restoring data into the *tableName* table (assuming the previous procedure 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$bpdr_backup**, the partitions are in a consistent state. Run **om_part_maintain.purge_xchg_bck_tables** and **om_sql_log_pkg.remove**(*SID*) to purge the backup tables, **om_sql_log** and **om_sql_pointer**.
- If the failure occurred during **sys$bpdr_purge**, the partitions are in an inconsistent state (partially purged):
  1. Run **om_sql_log_pkg.exec**(*SID*) to finish the purge (exchange), where *SID* is the session ID of the failed operation (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 Nth 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, run **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 Nth partition, drops N-1 partitions, and restores the retained orders into the Nth partition.
  - If the **om_sql_log** table contains the DROP PARTITION statements, run **om_sql_log_pkg.exec*(SID)***, where *SID* is the session ID of the failed operation (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, run these statements:

    ```
    ALTER TABLE OM_ORDER_HEADER DROP PARTITION partition_name;
    ALTER TABLE OM_SEQUENCE DROP PARTITION partition_name;
    ```

  - If the backup tables are not empty, run **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, Oracle recommends 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. To resume the restore operation from the point of failure:
  1. Run **om_part_maintain.disable_ptned_fks** to disable foreign keys of partitions tables.


Copyright © 2009, 2025, Oracle and/or its affiliates.

2. In the event of failure, **sys$bpdr_restore** automatically deletes the partially restored data from the last partitioned table in order to ensure that a second attempt will not fail due to unique key violations. However, Oracle recommends that you run **om_part_maintain.sys$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. Run **om_sql_log_pkg.exec**(*SID*) to finish the restore, where *SID* is the session ID of the failed operation (the SID is recorded in **om_sql_pointer** together with the error message).
4. Run **om_part_maintain.reenable_ptned_fks** to re-enable foreign keys.

- If the root cause lies with the data to be restored, you must run **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 run **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 run **om_part_maintain.rebuild_unusable_indexes**.
- (Optional) In any case, you can run **om_part_maintain.purge_orphan_order_data** to delete orphan order data. Otherwise, the orphan data will be deleted by the next operation of **purge_partitions** or **drop_partitions**.
- Run **om_job_pkg.resume_jobs** to resume database jobs.

## Error Handling for rebuild_unusable_indexes

This procedure logs the DDL statements to be run in the **om_sql_log** table. In the event of an error, it is important that you re-run **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 run 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-run **setup_xchg_tables** with **a_force**=true.
- Run **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


Copyright © 2009, 2025, Oracle and/or its affiliates.

- 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.

   ```
   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:

   ```
   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:

   ```
   exec om_part_maintain.backup_selected_ords(16);
   ```

4. Purge the backup tables:

   ```
   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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

- **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 cause execution to be in parallel. 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_$001$** 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):


Copyright © 2009, 2025, Oracle and/or its affiliates.

```
insert into xchg_om_bck_$001$
    (select * from om_order_header where order_seq_id between x and y) ;
```

3. Repeat the following procedures 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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

- **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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

8

# Managing Optimizer Statistics

This chapter contains best practices for gathering optimizer statistics for the Oracle Communications Order and Service Management (OSM) product. Using the best practices in this chapter will result in better and more stable execution plans for SQL objects in the OSM database.

## About Optimizer Statistics

Optimizer statistics are a collection of data that describe the database and the objects in the database. These statistics are used by the optimizer to choose the best execution plan for each SQL statement. Statistics are stored in the data dictionary and can be accessed using data dictionary views such as USER_TAB_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 Oracle Database documentation.

Knowledge article 1369591.1, *Master Note: Optimizer Statistics*, has links to many other useful sources of information about optimizer statistics. It is available on the Oracle support website at:

https://support.oracle.com

## Gathering Optimizer Statistics

This section discusses methods of gathering optimizer statistics for OSM.

### Gathering Statistics Online

The procedures provided by the **om_db_stats_pkg** package and native Oracle database statistics procedures can all be run online. However, when gathering statistics during peak hours you should temporarily lower the value of the DEGREE preference as described in "Using Parallel Collection for Gathering Statistics."

### Automated Optimizer Statistics Collection

The automatic database optimizer statistics collection maintenance task is enabled by default. This typically triggers launch of an automatic database optimizer statistics collection job during a predefined maintenance window.

Partition statistics can be locked, and Oracle recommends that you:

- Leave the automatic statistics collection enabled.
- Some maintenance operations (such as purging partitions and deploying, undeploying, and purging cartridges) render the statistics stale. Schedule these maintenance operations to complete before automatic optimizer statistics collection starts.

For a list of steps and procedures that can be used to bootstrap and maintain the OSM Database Optimizer Statistics Management process with OSM releases that support locking of


Copyright © 2009, 2025, Oracle and/or its affiliates.

partition statistics, see knowledge article 1925539.1, *New OSM Database Optimizer Statistics Management,* on the Oracle support website for additional information:

https://support.oracle.com

You can determine if the automatic optimizer statistics collection maintenance task is enabled by running the following commands as a SYSDBA user:

```
set serveroutput on
SELECT client_name, status FROM dba_autotask_operation;
```

You can disable the automatic optimizer statistics collection maintenance task by running the following commands as a SYSDBA user:

```
BEGIN
  DBMS_AUTO_TASK_ADMIN.DISABLE(
    client_name => 'auto optimizer stats collection',
    operation => NULL,
    window_name => NULL);
END;
/
```

You can enable the automatic optimizer statistics collection maintenance task by running the following commands as a SYSDBA user:

```
BEGIN
  DBMS_AUTO_TASK_ADMIN.ENSABLE(
    client_name => 'auto optimizer stats collection',
    operation => NULL,
    window_name => NULL);
END;
/
```

## Gathering Fixed Object Statistics

When fixed object statistics are missing, the database optimizer uses predefined default values that may not be adequate for your system, and this could lead to less than optimal execution plans. For example, RMAN, Data Guard, Streams, and Grid Control make heavy use of fixed tables through DBA and V$ views and they often experience performance issues when fixed object statistics have not been collected. Another common symptom is extreme TEMP space usage driven by poor plans against fixed tables.

The automatic database optimizer statistics collection job does not gather fixed object statistics.

Oracle recommends that you gather fixed object statistics:

- When bootstrapping the OSM database optimizer statistics process
- After an OSM upgrade
- After deploying or undeploying cartridges
- Following a significant increase in order volume
- Following a change in partition size

You can gather fixed object statistics when there is a representative load on the system (ideally at peak utilization) by running the following commands as a SYSDBA user:

```
execute DBMS_STATS.GATHER_FIXED_OBJECTS_STATS;
```


Copyright © 2009, 2025, Oracle and/or its affiliates.

# Gathering System Statistics

System statistics enable the optimizer to more accurately determine the cost of 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.

The automatic database optimizer statistics collection job does not gather system statistics.

Oracle recommends that you gather system statistics:

- When bootstrapping the OSM database optimizer statistics process.
- Following a significant increase in order volume.
- Following changes in your database CPU speed or IO subsystem.

You can gather system statistics when there is a representative load on the system (ideally at peak utilization) by running the following commands as a SYSDBA user:

```
execute DBMS_STATS.GATHER_SYSTEM_STATS;
```

# Gathering Cartridge Metamodel Statistics

OSM stores cartridge metamodel information in database tables and statistics on these tables should be kept up-to-date to ensure optimal order processing performance. Gather cartridge metamodel statistics:

- When bootstrapping the OSM database optimizer statistics process.
- After deploying or undeploying cartridges.
- After an OSM upgrade.

You can cartridge metamodel statistics by running the following commands as an OSM order management user:

```
set serveroutput on
execute om_db_stats_pkg.gather_cartridge_stats;
```

# Gathering Order Statistics

The procedure to follow to gather statistics on OSM order tables varies depending on table volatility. The contents of a volatile order table are primarily impacted by the current order volume. For these tables, once an order has been processed, the associated data is automatically deleted.

## High Volatility Order Tables

The following order tables are always highly volatile because their contents are very short-lived. These tables may have thousands of rows at peak workload, but have little or no data when there is reduced order activity in the system.

- OM_JMS_EVENT
- OM_JMS_THREAD
- OM_ORDER_STATE_PENDING
- OM_ORDER_STATE_EVENT_PENDING
- OM_COORD_NODE_INSTANCE


Copyright © 2009, 2025, Oracle and/or its affiliates.

You should not enable incremental statistics on highly volatile tables. However, Oracle recommends that you lock statistics for these tables, using the following command:

```
execute om_db_stats_pkg.lock_volatile_order_stats;
```

## 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.

## Medium Volatility Order Tables

Some order tables have both partitions that retain order data and volatile partitions from which data is deleted after it is processed. For historical reasons, these tables are configured with a medium level of volatility. However, due to process improvements that have been identified over time, all medium volatility tables should be reconfigured either as high volatility or low volatility tables based on the characteristics of your solution.

By default, the following tables are configured with a medium level of volatility:

- OM_ORDER_FLOW
- OM_AUTOMATION_CTX
- OM_AUTOMATION_CORRELATION
- OM_ORDER_POS_INPUT
- OM_UNDO_BRANCH_ROOT
- OM_ORCH_DEPENDENCY_PENDING

If most of your orders complete in less than 1 hour, it is recommended that you manage these tables in the same manner as high volatility tables. Otherwise, it is recommended that you manage them in the same manner as low volatility tables.

In other words, if most of your orders complete in less than 1 hour, when bootstrapping the OSM database optimizer statistics process, run the following as the OSM order management user:

```
execute om_db_stats_pkg.unlock_volatile_order_stats;
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);
execute om_db_stats_pkg.lock_volatile_order_stats;
```

However, if most of your orders complete in more than 1 hour, run the following instead:

```
execute om_db_stats_pkg.unlock_volatile_order_stats;
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);
```


Copyright © 2009, 2025, Oracle and/or its affiliates.

```
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);
execute om_db_stats_pkg.lock_volatile_order_stats;
```

## Enabling Incremental Statistics

Once table volatility has been properly reconfigured, and during the bootstrapping the OSM database optimizer statistics process, it is recommended that, as the OSM order management user, 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);
```

You should then confirm that INCREMENTAL_STALENESS is configured, using the following command:

```
SELECT dbms_stats.get_prefs(pname=>'INCREMENTAL_STALENESS',
tabname=>'OM_ORDER_INSTANCE') FROM dual;
```

If the query above generates an error or doesn't return USE_STALE_PERCENT, USE_LOCKED_STATS, use the following commands to set the value:

```
set serveroutput on
BEGIN
  FOR T IN
  (SELECT IT.TABLE_NAME
    FROM OM_$INSTALL$TABLE IT, USER_TABLES UT
    WHERE IT.VOLATILITY = om_const_pkg.v_volatility_low AND UT.TABLE_NAME =
IT.TABLE_NAME AND UT.PARTITIONED = 'YES'
  )
  LOOP
    dbms_stats.set_table_prefs(user, T.TABLE_NAME, 'INCREMENTAL_STALENESS',
'USE_STALE_PERCENT, USE_LOCKED_STATS');
  END LOOP;
END;
/
```

If this generates an error, review the list of database patches installed on your system. Otherwise, you can confirm that INCREMENTAL_STALENESS is now configured correctly by re-running the confirmation command you ran earlier, that is:

```
SELECT dbms_stats.get_prefs(pname=>'INCREMENTAL_STALENESS',
tabname=>'OM_ORDER_INSTANCE') FROM dual;
```

## Gathering High-Volatility-Table Statistics

Gather high-volatility-table statistics:

- When bootstrapping the OSM database optimizer statistics process
- After deploying or undeploying cartridges
- After an OSM upgrade
- Following a significant increase in order volume


Copyright © 2009, 2025, Oracle and/or its affiliates.

- After changing partition size
- When a new partition becomes active

As the OSM order management user, gather high-volatility-table statistics when the workload is representative, preferably as near to the peak as possible, given that the database must have spare resources:

```
set serveroutput on
execute DBMS_STATS.SET_SCHEMA_PREFS(user, 'DEGREE', 2);
execute om_db_stats_pkg.gather_volatile_order_stats(a_force_volatile => true)
execute DBMS_STATS.SET_SCHEMA_PREFS(user, 'DEGREE', 'DBMS_STATS.AUTO_DEGREE');
```

## Gathering Low-Volatility-Table Statistics

Gather low-volatility-table statistics:

- When bootstrapping the OSM database optimizer statistics process
- "On a regular basis (for example, once a week) if the automatic database optimizer statistics collection maintenance task is disabled

As the OSM order management user, gather low-volatility-table statistics during a period of low system activity:

```
set serveroutput on
exec om_db_stats_pkg.gather_order_stats(a_volatility => om_const_pkg.v_volatility_low,
a_force => false);
```

# Preparing a New Partition

Before using a new (blank) partition, you should pre-populate the partition with optimizer statistics and then lock the statistics.

## Populating New Partition Statistics

Pre-populating statistics on new partitions is of critical importance for avoiding the issues that will otherwise arise when the partition becomes active. For more information about this issue, see the following Oracle Blog entry:

https://blogs.oracle.com/optimizer/maintaining-statistics-on-large-partitioned-tables

## 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.

Copy statistics from a mature partition to empty partitions:

- When bootstrapping the OSM database optimizer statistics process
- When creating new partitions

To avoid copying partial statistics, make sure that statistics are not being collected when copying statistics from another partition. You can check whether an automatic optimizer statistics collection job is in progress by running the following commands as a SYSDBA user:

```
set serveroutput on
declare
```


Copyright © 2009, 2025, Oracle and/or its affiliates.

```
v_gathering_status integer;
begin
  begin
    select 1
      into v_gathering_status
      from dba_autotask_client_job
      where client_name = 'auto optimizer stats collection'
        and job_scheduler_status = 'RUNNING'
        and rownum = 1;
      dbms_output.put_line('Auto optimizer stats collection is running');
      return;
  exception
  when no_data_found then
    null;
  end;
  dbms_output.put_line('Auto optimizer stats collection is not running');
end;
/
```

If statistics collection is running, wait until it completes.

Then you can copy statistics to a new partition by running the following commands as the OSM order management user:

```
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;
```

If you have changed the partition size and want to copy stats from an older partition of different size, you can use scale factor to scale the stats. Use the following commands:

Example of scaling up stats into destination partition:

```
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',
       scale_factor => '2');
end;
```

Example of scaling down stats into destination partition:

```
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',
       scale_factor => '0.5');
end;
```


Copyright © 2009, 2025, Oracle and/or its affiliates.

> **Note**
>
> Because the **scale_factor** parameter is a varchar2 argument, it must be provided in single quotes. The value can be any positive decimal number.

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:

```
declare
    v_copied boolean;
begin
    om_db_stats_pkg.copy_order_ptn_stats(v_copied);
end;
```

## 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**.

After you save partition statistics to a file, that file can be transferred to the **DATA_PUMP_DIR** directory on another system 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 pre-production system.

## Using Default Statistics Package

The default statistics package includes scripts, partition statistics seed data, and SQL templates. The package is installed by the OSM installer in the *installation_root*/**Database/partition-statistic/processPartitionStatistic.sh** folder.

To load the default partition statistics, run the shell script under *installation_root*/**Database/partition-statistic/processPartitionStatistic.sh**. The **processPartitionStatistic.sh** script imports partition-level DB optimizer statistics with SQL Plus and SQL Loader, which must be installed in the path.

> **Note**
>
> The SQL Plus and SQL Loader tools are obtained by installing Oracle Instant Client. Refer to the *OSM Compatibility Matrix* for more information about which version to use.


Copyright © 2009, 2025, Oracle and/or its affiliates.

The following abstract shows the parameters to use with the script:

```
 Usage:
    processPartitionStatistic.sh parameters
      -u <OSM User Name>         : OSM schema name, optional, if not
specified,
                                   value obtained from environment variable
                                   OSM_MAIN_USERNAME.
      -t <Tablespace Name>       : Tablespace name, optional, if not
specified,
                                   value obtained from environment variable
                                   OSM_DEFAULT_TABLESPACE.
      -i <Database URL>          : Database URL, optional, if not specified,
                                   value obtained from environment variable
                                   OSM_DB_URL. Format is host:port/service.
      -d <SQLLDR URL>            : Database URL for sqlldr command, optional,
                                   if not specified, it defaults to -i
                                   parameter if given, otherwise to the
                                   environment variable OSM_DB_URL.
      -s <State Table Name>      : Statistics Table Name, optional, if not
set,
                                   value obtained from environment variable
                                   OSM_ORDER_STAT_TABLE. When -s is not given
                                   and OSM_ORDER_STAT_TABLE is undefined, the
                                   default is OM_ORDER_STAT_TABLE.
      -a <Source partition Name> : Optional, source partition name, if not
                                   specified, value obtained from environment
                                   variable OSM_SOURCE_PARTITION.
                                   If OSM_SOURCE_PARTITION is not set,
                                   partition statistic seed data is taking
                                   from partition seed data. If defined, seed
                                   data is taking from the specified
partition.
      -b <Target partition Names>: Optional, comma delimited list of target
                                   partition name, if not specified, value
                                   obtained from environment variable
                                   OSM_TARGET_PARTITION.
                                   If OSM_TARGET_PARTITION is not set,
                                   partition statistic seed data is copied to
                                   all partition found in OSM schema.
                                   Otherwise, partition statistic seed data
                                   will be copied to the specified partitions.
      -l <Log file Names>:         Optional, SQL Loader (sqlldr) log file
name.

    or just -h for help
```

The following is an example of the script:

```
cd installation_root/Database/partition-statistic
./processPartitionStatistic.sh -u osm750 -t OSM -i hostOrIp:port/
pdb_service_name -s OM_ORDER_STAT_TABLE
```


Copyright © 2009, 2025, Oracle and/or its affiliates.

## Locking Partition Statistics

After copying statistics to a new empty partition, statistics should be locked on that partition if the automatic optimizer statistics collection maintenance task is enabled. Also note that this capability is not available in some OSM releases.

You can manually lock partition statistics using the om_db_stats_pkg.lock_order_ptn_stats procedure. For example, as the OSM order management user:

```
execute om_db_stats_pkg.lock_order_ptn_stats ('P_000000000000400001');
```

You should lock statistics on an empty partition after copying statistics into that partition and you should leave statistics locked on that partition when it becomes active.

# Configuring a Partition When It Is No Longer the Active Partition

You can manually unlock partition statistics using the om_db_stats_pkg.unlock_order_ptn_stats procedure. For example, as the OSM order management user:

```
execute om_db_stats_pkg.unlock_order_ptn_stats ('P_000000000000400001');
```

You should unlock statistics on a partition when it matures (that is, once it is no longer active). This should be done following a switch to a new active partition.

# Optimizer Statistics Error Handling Using Datapump

Optimizer statistics management error handling is available for automated copy partition statistics jobs and 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, you can determine if datapump jobs are stuck by running the following commands as a SYSDBA user:

```
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 the OSM order management 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 commands:

```
declare
    dpj number;
begin
    dpj := dbms_datapump.attach('IMPORT_ORDER_PTN_STATS', user);
```


Copyright © 2009, 2025, Oracle and/or its affiliates.

```
    dbms_datapump.stop_job(dpj, 1, 0);
end;
```

# Optimizer Statistics Management Performance Tuning

This section presents various ways to tune optimizer statistics management performance.

## Using Parallel Collection for Gathering Statistics

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 partitions. 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.

To do this, run the following command as the OSM order management user:

```
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 (for example, when gathering statistics for high volatility tables), 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 for large objects.

> ⚠ **Caution**
>
> Do not change the degree of parallelism attribute of any OSM table or index. This is not supported.

## Cursor Invalidations

When statistics are modified by DBMS_STATS, new cursors that are not yet cached in the shared pool use updated statistics 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. This results in 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. For this reason, 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.

```
alter system set "_optimizer_invalidation_period"=600 scope=both;
```

If 10 minutes turns out to be too short to avoid significant spikes caused by parsing in your environment, increase the value accordingly.


Copyright © 2009, 2025, Oracle and/or its affiliates.

# 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.

```
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.

```
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.

```
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.

Parameters:


Copyright © 2009, 2025, Oracle and/or its affiliates.

- **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.

```
procedure gather_cartridge_stats;
```

## om_db_stats_pkg.gather_order_stats

This procedure gathers statistics for order tables configured with the specified volatility level.

```
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 gathers statistics for volatile order tables.

```
procedure gather_order_stats(
    a_force boolean default false);
```

Parameters:

- **a_force**: Specifies whether you want statistics to be gathered on volatile order tables even if their statistics are locked. The default is false. An order table is deemed volatile if its volatility level is set to om_const_pkg.v_volatility_high.

## om_db_stats_pkg.copy_order_ptn_stats

This procedure copies order partition statistics from the specified source order partition to the specified destination order partition.

```
procedure copy_order_ptn_stats(
    a_copied out boolean,
    a_dst_partition_name varchar2 default null,
    a_src_partition_name varchar2 default null);
```

Parameters:


Copyright © 2009, 2025, Oracle and/or its affiliates.

- **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.lock_order_ptn_stats

This procedure locks order partition statistics for the specified partition. Statistics of the corresponding partitions of reference partition tables are also locked.

```
procedure lock_order_ptn_stats(
    a_partition_name varchar2);
```

Parameters:

- **a_partition_name**: Specifies the name of the order partition to lock.

Exceptions:

- ORA-20165: Illegal argument: Partition does not exist.

## om_db_stats_pkg.unlock_order_ptn_stats

This procedure unlocks order partition statistics for the specified partition. Statistics of the corresponding partitions of reference partition tables are also unlocked.

```
procedure unlock_order_ptn_stats(
    a_partition_name varchar2);
```

Parameters:

- **a_partition_name**: Specifies the name of the order partition to unlock.

Exceptions:

- ORA-20165: Illegal argument: Partition does not exist.

# Advanced Procedures

This section describes advanced procedures.


Copyright © 2009, 2025, Oracle and/or its affiliates.

## 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.

```
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.

```
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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

- ORA-20144: Function returned unexpected value. Internal error. Contact support: Cannot find the newest partition.

## om_db_stats_pkg.expdp_order_ptn_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.

```
procedure expdp_order_ptn_stats(
    a_saved out boolean,
    a_stat_table_name varchar2 default c_om_order_stat_table);
```

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: Invalid table name.
- **ORA-20165**: Illegal argument: Table does not exist.
- **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_ptn_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.

```
procedure impdp_order_ptn_stats(
    a_loaded out boolean,
    a_stat_table_name varchar2 default c_om_order_stat_table);
```

> **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. If it is not a statistics table, the table is not dropped and an exception is raised.

Exceptions:

- **ORA-20165**: Illegal argument: Invalid table name.


Copyright © 2009, 2025, Oracle and/or its affiliates.

- **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 Procedures

This section describes the troubleshooting procedures.

## 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;
```

## om_db_stats_pkg.get_order_ptn_stats

This procedure lists statistics for table partitions that match the given filter criteria.

```
procedure get_order_ptn_stats;
```

## om_db_stats_pkg.list_order_ptn_stats

This procedure outputs statistics for table partitions that match the given filter criteria to dbms_output.

```
procedure list_order_ptn_stats;
```

## om_db_stats_pkg.check_order_ptn_stats

This procedure validates the statistics on the schema to check for any errors. There are two versions of this procedure: one outputs it to dbms_output and the other returns it as a collection of strings (for external use).

```
procedure check_order_ptn_stats;
```

This procedure looks for the following conditions and creates the appropriate level message (in brackets) if the condition is violated:

- Missing statistics on Order Data Tables (CRITICAL)
- Empty or active partitions with unlocked statistics (CRITICAL)
- Mature partitions with locked statistics (WARNING)
- Statistics are locked with 0 rows on tables that should never have 0 rows when a partition is used to store orders (e.g., OM_ORDER_HEADER) (CRITICAL)
- Statistics are locked with a small number of rows on tables that should have a large number of rows in order to be representative of a partition used to store a large number of orders (e.g., OM_ORDER_HEADER) (CRITICAL)
- Incremental statistics do not work with locked partition statistics (CRITICAL)
- Incremental statistics are not enabled on low or medium volatility tables (MAJOR)
- Incremental statistics are enabled on high volatility tables (MAJOR)
- Statistics are not locked on high-volatility tables (CRITICAL)


Copyright © 2009, 2025, Oracle and/or its affiliates.

- Statistics are locked on low or medium volatility tables (WARNING)
- Attempting to check for partition statistics problems while statistics are being collected (WARNING)

# Recovery Procedures

This section describes the recovery procedures.

## 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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

9

# Backing Up and Restoring OSM Files and Data

This chapter helps you understand how Oracle Communications Order and Service Management (OSM) is related to the Oracle Database backup and restore procedures.

## 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.

### Backup and Restore Overview

Consider OSM information from the Oracle Database for backing up and restoring OSM.

### Backup and Restore Schedule

#### OSM_CFG_Home Directory

The directory in which OSM configuration files are generated or modified by OSM. This directory contains various configuration files for OSM DB schema and Weblogic domain. Oracle strongly recommends that you manage this directory using source control. Otherwise, take backups. The contents of this directory are subject to change whenever any of the installation tools, such as discovery, configDB, configOSM, and configDomain are run.

#### 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 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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

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:

- 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 *OSM_home* directory.

## Backing Up the OSM 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.

## Restoring the OSM Files

To restore the OSM files, remove the contents of the *OSM_home* directory, and extract the contents of the backed-up **tar** file into the *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 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. 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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

## 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 back up 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


Copyright © 2009, 2025, Oracle and/or its affiliates.

- 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 Remote 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 Remote console Home window. See "Accessing the WebLogic Remote Console."
2. Navigate from**Edit Tree** to **Services** to **JMS Services**.

   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 in the General tab. Click on **Show Advanced Fields** to display more options.
5. Change the following options to true:
   - Insertion Paused At Startup
   - Production Paused At Startup
   - Consumption Paused At Startup
6. Click **Save**.
7. Exit the WebLogic Remote 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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

## 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 Remote Console. For information about accessing the WebLogic Console, see "Accessing the WebLogic Remote Console."

From the Home window of the WebLogic Console, navigate from**Edit Tree** to **Services** and then to **File Stores**. The **File Stores** window is displayed containing a list of the file stores that have been defined along with their Targets. Click the name of any file store. The resulting window displays the location of the directory.

> **Note**
>
> For JDBC Stores navigate from **Edit Tree** to **Services** and from there to **JDBC Stores**. 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

OSM database (OM_ATTACHMENT table) is the placeholder for order attachments.

## 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 Remote Console."

From the **Home** window of the WebLogic Console, navigate from **Edit Tree** to **Deployments** and from there to **App Deployments**. 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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

## 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 Remote 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 Remote console **Home** window. See "Accessing the WebLogic Remote Console."
2. Navigate from **Edit Tree** to **Services** and from here to **JMS Services**.

   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 in the **General** tab selecte. Click on **Show Advanced Fields** to display more options.
5. Change the following options to false:
   - Insertion Paused At Startup
   - Production Paused At Startup
   - Consumption Paused At Startup
6. Click **Save**.
7. Exit the WebLogic Remote console.


Copyright © 2009, 2025, Oracle and/or its affiliates.

10

# Monitoring and Managing OSM

This chapter describes how to monitor and manage the Oracle Communications Order and Service Management (OSM) system using the Oracle WebLogic Remote Console.

This chapter provides guidelines and best practices for monitoring an OSM deployment. This includes functional monitoring of particular orders or processes, and performance monitoring to assist in tuning. In order to effectively monitor OSM, you require a broad knowledge of many components, such as the OSM Managed Server, the Java Virtual Machine (JVM), the WebLogic Server, and the Oracle Database.

## About Monitoring and Managing OSM

Many OSM monitoring tasks must be performed on various schedules. Some tasks should be performed daily, whereas others can be done weekly or even monthly. Many tasks can be done automatically, by configuring thresholds that raise warnings when the thresholds are exceeded. The output of many of the tasks can be interpreted as snapshots, whereas others should be interpreted only in the context of a series of data.

For each monitoring task in this chapter, there is a description of the item that the task monitors, why the item should be monitored, what tool(s) you should use to monitor the item, and information about what to monitor, including guidance on selecting values, such as thresholds.

The monitoring tasks are grouped into monitoring the application, monitoring input/output (I/O), and monitoring the host.

### About Monitoring OSM Using WebLogic Remote Console

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 Remote 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.

Oracle WebLogic Remote Console is a graphical user interface that administers Oracle WebLogic Server domains.

WebLogic Remote Console is available in two formats:

- Desktop WebLogic Remote Console, a desktop application installed on your computer.
- Hosted WebLogic Remote Console, a web application deployed to an administration server and accessed through a browser.

See the Oracle WebLogic documentation for more information.


Copyright © 2009, 2025, Oracle and/or its affiliates.

# Accessing the WebLogic Remote Console

1. In Desktop WebLogic Remote console do one of the following:
   - If you are not connecting via Secure Socket Layer (SSL), enter the following URL into your Admin Server Connection Provider:

     `http://host:port`

     where *port* is the port on which the Administration server is listening for requests.
   - If you are connecting via SSL, enter the following URL into your Admin Server Connection Provider:

     `https://host:SSLport`

     where *SSLport* is the port on which the Administration server is listening for SSL requests. This is a different port from the one used for non-SSL requests.
2. In Hosted WebLogic Remote Console do one of the following:
   - If you are not connecting via Secure Socket Layer (SSL), enter the following URL into your browser:

     `http://host:port/rconsole`

     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.
   - If you are connecting via SSL, enter the following URL into your browser:

     `http://host:SSLport/rconsole`

     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. This is a different port from the one used for non-SSL requests.

For more details on Hosted WebLogic Remote Console, see Start Hosted WebLogic Remote Console

When started, the WebLogic Remote Console prompts for a password. This should be the password for a user that is a member of the **Administrators** group in WebLogic. One such user is the WebLogic administration user that was created when the domain was created. By default, the name of this user is **weblogic**.

After you have successfully logged in, the WebLogic Console Home window is displayed.

# 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. Log in to the WebLogic Remote Console.
2. Navigate from **Monitoring Tree** to **Environment**, and then to **Servers**.


Copyright © 2009, 2025, Oracle and/or its affiliates.

The Summary of Servers window is displayed. Server statuses are contained in the Health column of the table.

3. Navigate from **Monitoring Tree** to **Deployments** and then to **Application Management**.

   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.

# Using the WebLogic Scripting Tool

Several OSM features use the WebLogic Scripting Tool (WLST). For security, Oracle recommends that you configure secure communication for the administration or managed server on which you want to run WLST.

To configure secure communication for WLST on the managed server:

1. Configure the managed server to use a custom keystore.

> **Note**
>
> The use of the Demo Keystore (including DemoIdentity or DemoTrust) is no longer supported in OSM 8.0. The DemoIdentity keystore now has an encrypted and inaccessible passphrase, making it unusable as a placeholder. For information about configuring custom keystores, see *OSM Installation Guide* for the topic about setting up secure HTTPS connections.

2. Set the **CLASS_JMV_ARG**, as in the following example:

```
export CONFIG_JVM_ARGS="-Dweblogic.security.SSL.ignoreHostnameVerification=true -
Dweblogic.security.TrustKeyStore=CustomTrust -
Dweblogic.security.CustomTrustKeyStoreFileName=/OSM-QA/oracle/Middleware1213/
user_projects/domains/OSM_CS_Cluster_4VM_6Nodes_2RAC/config/security/jks/osm.jks"
```

> **Note**
>
> You need to add other parameters to the **CLASS_JVM_ARGS** according to different keystore and password requirements. For example, to use the trusted Certificate Authorities (CA) from another keystore, specify the following:
>
> ```
> -Dweblogic.security.CustomTrustKeyStoreType=type
> -Dweblogic.security.CustomTrustKeyStorePassPhrase=passphrase
> ```
>
> where *type* is the default keystore type specified in the JDK **java.security** file.
>
> Passphrase defaults to no passphrase. which is allowed by some keystore types to grant read-only access. Other keystore types always require a keystore passphrase.


Copyright © 2009, 2025, Oracle and/or its affiliates.

3. To create a WLST secure communication session, run the `connect` command, as in following example:

   ```
   connect('WLS_Admin_Username', 'WLS_Admin_Password', 't3s://hostname:port')
   ```

   where *hostname* is the hostname of the WebLogic Server, and *port* is the SSL port number of the WebLogic Server.

# Refreshing OSM Metadata

When you are working with OSM, you may come across instructions to refresh the OSM metadata. There are different ways to do this. You can use whichever of them is easiest for you, as long as you meet any prerequisites.

To refresh the OSM metadata, do any **one** of the following:

- If you have access to the Order Management web client, click **Refresh Server Cache** in the Administration area.
- If you are using a clustered WebLogic deployment, start a managed server that is running OSM.
- If you have configured an external **oms-config.xml** file, touch the file.
- Shut down and restart OSM.

# 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 Remote Console

The WebLogic Remote Console provides a real-time view of system performance.

> **Note**
>
> For more information about tuning OSM production systems, see *OSM Installation Guide*.

To access the performance monitor:

1. Start the WebLogic Remote Console.
2. Click **Monitoring Tree**, then expand **Environment**, then **Servers**, and then select a server from the list.

   This displays the **General Configuration** page for the selected server.
3. From the selected server, click **Deployments** and then **Application Runtimes** 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 OK, Warn, Overloaded, Critical, and Failed.
4. In the selected server navigate from **Environment** to **JVM Runtime** to view the JVM memory utilization statistics for the servers.


Copyright © 2009, 2025, Oracle and/or its affiliates.

Refresh the screen several times to sample values for Health Free Percent. If memory usage remains elevated (for example, not falling to 50% or less while the system is under load), analyze the garbage collection logs and follow the recommendations described in the *OSM Memory Tuning Guidelines* (Doc ID: 2028249.1) knowledge article on My Oracle Support. For example, undeploy cartridges that are no longer needed or add a new managed server to the WebLogic Server cluster.

5. Click on the selected server then click the **Threads** tab to monitor thread activity for the server. An important column to monitor is Min Threads Constraints Pending. A count of zero is optimal, meaning no user requests are stuck or waiting to be processed.
6. Click the **Requests** tab to monitor the Work Managers configured for the server.
7. In the selected server, navigate from **Scheduling** to **Work Managers Runtime** to monitor detailed Work Managers Runtime information.
8. Check Deferred Requests in the **Max Threads Constraint** table in **Scheduling**. If **Deferred Requests** does not go back to **0**, you may need to tune the system (for example, add a new managed server to the WebLogic Server cluster or allocate more threads to OSM work managers).
9. In the selected server navigate from **Services** to **Transactions** and then to **JTA runtime**. Here, click the **Transactions** tab to monitor transaction activity on the server. Ideally, there should be no Roll Back statistics in the summary; if they are, 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:

- In the Weblogic Remote Console, ensure all servers are marked as **Active** 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 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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

## 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. For information about using WLST over a secure connection, see the security section of *OSM System Administrator's Guide.*

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 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".


Copyright © 2009, 2025, Oracle and/or its affiliates.

## 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:

   UNIX/Linux:

   ```
   jconsole -J-Djava.class.path=$JAVA_HOME/lib/jconsole.jar:$JAVA_HOME/lib/
   tools.jar:$WLS_HOME/lib/wljmxclient.jar -J-
   Djmx.remote.protocol.provider.pkgs=weblogic.management.remote service:jmx:rmi:///
   jndi/iiop://hostname:port/weblogic.management.mbeanservers.runtime
   ```

   Windows:

   ```
   jconsole -J-Djava.class.path=%JAVA_HOME%\lib\jconsole.jar;%JAVA_HOME%
   \lib\tools.jar;%WLS_HOME%\lib\wljmxclient.jar -J-
   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 user name 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/


Copyright © 2009, 2025, Oracle and/or its affiliates.

> **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.

> **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 Input/Output (I/O) wait times are low (nothing more than few percent of I/O waits are acceptable).

You can monitor the database by using Oracle Enterprise Manager, a web-based tool that allows you to manage Oracle software. For more information about Enterprise Manager, see Oracle Enterprise Manager Cloud Control documentation.


Copyright © 2009, 2025, Oracle and/or its affiliates.

# 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_PERCENT 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*.

```
BEGIN
DBMS_STATS.GATHER_SCHEMA_STATS (
OWNNAME=>USER,
ESTIMATE_PERCENT=>percent,
GRANULARITY =>'ALL',
CASCADE =>TRUE,
BLOCK_SAMPLE=>TRUE);
END;
```

# Managing Log Files

Oracle Diagnostic Logging (ODL) is the system logging service that is used by most Oracle Fusion Middleware applications. The system writes diagnostic messages to log files. Each message includes such information as the time, component ID, and user.

You can manage ODL log files using Oracle Enterprise Manager, the WebLogic Remote Console, or WebLogic Scripting Tool (WLST) commands. For more information about understanding ODL messages and log files, see *Oracle Fusion Middleware Administrator's Guide*.

## Managing Log Files Using Enterprise Manager Fusion Middleware Control

Using Enterprise Manager, you can view and search log files, and configure settings for log files. For more information about managing log files and diagnostic data, see *Oracle Fusion Middleware Administrator's Guide*.

View and manage log files using the Enterprise Manager Fusion Middleware Control plug-in. For information about enabling the Enterprise Manager template during OSM installation, see the topic about creating the WebLogic server domain in *OSM Installation Guide*.

To access Enterprise Manager, enter the following URL into your browser:

```
http://host:port/em
```

where *host* is the IP address of the computer on which the WebLogic administration server is installed, and *port* is the port for the administration server domain.


Copyright © 2009, 2025, Oracle and/or its affiliates.

## Viewing Log Files and Messages Using Fusion Middleware Control

You can view log files and messages, as well as search and filter log information, by using the Log Messages page. For information about searching and filtering log information, see the topic about viewing log files and their messages in *Oracle Fusion Middleware Administrator's Guide*.

To open the Log Messages page:

1. From the navigation pane, expand **WebLogic Domain**, and then expand the domain.
2. Right-click the server on which you want to view log files and messages, and select **Logs**, and then **View Log Messages**.

   The Log Messages page is displayed.

## Configuring Logs Using Fusion Middleware Control

You can change the log settings of managed servers and Java components using Fusion Middleware Control. For information, see the topic about configuring settings for log files in *Oracle Fusion Middleware Administrator's Guide*.

> **Note**
>
> You must apply log configuration settings to each server that you want to view logs for. Fusion Middleware Control allows you to view logs and log messages for each server in your OSM deployment.

From the Log Configuration page, you can configure the following:

- **Log Levels**: Allows you to configure the log level for persistent and runtime loggers. For more information, see "About Log Severity Levels."
- **Log Files**: Allows you to create and view log handlers, or edit, view, and delete the configuration of a log handler.
- **QuickTrace**: Allows you to create and view QuickTrace handlers, or edit, view, and delete the configuration of a QuickTrace handler. Using QuickTrace, you can trace messages from specific loggers and store the messages in memory.

To open the Log Configuration page:

1. From the navigation pane, expand **WebLogic Domain**, and then expand the domain.
2. Right-click the server where you want to create and edit log file configuration, and select **Logs**, and then **Log Configuration**.

   The Log Configuration page is displayed.

# Managing Log Files Using WebLogic Remote Console

Use the WebLogic Remote Console to view log files, configure the log view, and change log size and rotation. For more information about configuring logging, refer to *WebLogic Remote Console Online Help*.


Copyright © 2009, 2025, Oracle and/or its affiliates.

## Viewing Log Files

You can view the OSM log files using WebLogic Remote Console.

To view the most recent messages in the log files:

1. Log in to the WebLogic Remote Console.
2. Navigate from **Monitoring Tree** to **Diagnostics** and then to **Logs and Archives**.
3. From the Logs list, select the name of the log you want, and then click on the checkbox to download the log file.

## Log Size and Rotation

It is important to manage the log file's maximum size and rotations to prevent the logs from consuming disk resources. Rotation refers to log files that are rotated into archived log files.

To manage log settings:

1. Log in to the WebLogic Remote Console.
2. Click **Edit Tree/Services/JMS Servers/server**.

   The **General Configuration** page for the selected server is displayed.
3. Click the **Logging** tab.

   A page that displays the logging settings for the selected server is displayed.
4. Modify the **Rotation file size** to 25,000 kilobytes and be sure to limit the number of **Files to retain**. Modify these settings for both tabs, General and HTTP.

# About Log Severity Levels

To control the volume of log messages generated and written to the ODL log files, you assign severity levels to the various areas of the application that generate their own discrete messages. Some diagnostic message types have levels within them that you can set for finer logging granularity. If you do not assign a severity level, the severity level is **NOTIFICATION:1**.

The following is a descending list of the severity levels, starting with the most severe:

- **ERROR**: (Most severe) A serious problem that requires immediate attention from the administrator and is not caused by a bug in the product
- **WARNING**: A potential problem that should be reviewed by the administrator
- **NOTIFICATION**: A major lifecycle event, such as the activation or deactivation of a primary sub-component or feature. NOTIFICATION:1 is the default level. NOTIFICATION:16 provides a finer level of granularity for reporting normal events.
- **TRACE**: (Least severe) Trace or debug information for events that are meaningful to administrators, such as public API entry or exit points. For example, TRACE:16 provides detailed trace or debug information that can help diagnose problems with a particular subsystem.

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), ODL does not generate the message.


Copyright © 2009, 2025, Oracle and/or its affiliates.

## Configuring Log Levels Using the logging.xml File

You can change logging level configuration by modifying the **logging.xml** file. You can modify both log handlers and loggers. For more information about working with the **logging.xml** file, see the topic about log handler and logger configuration in *Fusion Middleware Administrator's Guide for Oracle Identity Manager*.

The **logging.xml** file has the following basic structure:

```
<logging configuration>
  <log_handlers>
    <log_handler name='console-handler' level="NOTIFICATION:16"></log_handler>
    <log_handler name='odl-handler'></log_handler>
    <!--Additional log_handler elements defined here....-->
  </log_handlers>
  <loggers>
    <logger name="example.logger.one" level="NOTIFICATION:16">
      <handler name="console-handler"/>
    </logger>
    <logger name="example.logger.two" />
    <logger name="example.logger.three" />
    <!--Additional logger elements defined here....-->
  </loggers>
</logging_configuration>
```

To change log levels using the **logging.xml** file:

1. Open the **logging.xml** file, which is in the following location:

   *domain_home***/config/fmwconfig/servers/***server name***/logging.xml**

2. Add or change the log level for the appropriate log handler.

   For example, change:

   ```
   <log_handler name='odl-handler' class='oracle.core.ojdl.logging.ODLHandlerFactory'
   filter='oracle.dfw.incident.IncidentDetectionLogFilter'>
   ```

   To the following:

   ```
   <log_handler name='odl-handler' class='oracle.core.ojdl.logging.ODLHandlerFactory'
   level='TRACE:32' filter='oracle.dfw.incident.IncidentDetectionLogFilter'>
   ```

   This log handler is now able to log debug messages.

> **Note**
>
> Even though the **logging.xml** file is configured to allow the handler to log debug messages, you must also enable the corresponding OSM Java class to the correct logging level.

## Managing Logs Using WLST

You can view and query server logs by using WebLogic Scripting Tool (WLST) commands, as well as changing log levels. You can list the log files for an Oracle WebLogic Server domain, server, an Oracle instance, or component using the WLST `listLogs` command. You can search the log files by using the WLST `displayLogs` command and narrow the search by specifying criteria, such as time, component ID, or message type. For example:


Copyright © 2009, 2025, Oracle and/or its affiliates.

```
displayLogs(target='opmn:asinst_1/ohs1', last=5)
```

Logs are automatically aggregated across all managed servers in a cluster. Log viewing commands work whether you are connected or not connected to a WebLogic Server.

To run WLST in interactive mode:

1. Do one of the following:
   - For UNIX and Solaris platforms, enter the following command:

     *MW_home*/Middleware/oracle_common/common/bin/wlst.sh
   - For Windows platforms, enter the following command:

     *MW_home*/Middleware/oracle_common/common/bin/wlst.cmd

> **Note**
>
> Most of the WLST logging commands require that you are running in the domainRuntime tree. For example, to connect and to run in the domainRuntime tree, use the following commands:
>
> ```
> connect('WLS_username', 'WLS_password', 'WLS_hostname:WLS_port_number')
> domainRuntime()
> ```

For more information about managing ODL files by using WLST, see the topic about managing log files and diagnostic data in *Oracle Fusion Middleware Administrator's Guide*.

## Configuring Log Levels Using WLST

To configure log levels using a WLST command:

1. Enter WLST in interactive mode.
2. Run the `setLogLevel` command. For example:

   ```
   setLogLevel(target='soa_server1', logger='oracle.soa', level='WARNING')
   ```

   In this example, the **soa_server1** server logging level is set to **WARNING**. For more information, see "About Log Severity Levels."

   For more information about managing ODL files by using WLST, see the topic about managing log files and diagnostic data in *Oracle Fusion Middleware Administrator's Guide*.

# Secure vs Non-Secure Log Filtering

A stack trace is a list of the method calls that an application is in the middle of at the time an exception is thrown. Running OSM with stack trace logging enabled can be important for debugging an application during runtime.

Using the **oms-config.xml** parameter, **enable_log_stacktraces**, you can enable or disable logging stack traces. By default, this parameter is enabled. If there is a security concern about having log stack traces enabled, you can disable this parameter. For more information about setting this parameter, see "Configuring OSM with oms-config.xml".


Copyright © 2009, 2025, Oracle and/or its affiliates.

# Managing OSM Metrics

OSM provides a sample Grafana dashboard that can be used to visualize OSM metrics available from a Prometheus data source. OSM relies on Prometheus to scrape and expose these metrics.

See the following topics for further details:


- Configuring Prometheus for OSM Metrics
- Viewing OSM Metrics Without Using Prometheus
- Viewing OSM Metrics in Grafana
- Exposed Order Metrics


## Configuring Prometheus for OSM Metrics

Configure the scrape job in Prometheus for OSM as follows:

```
- job_name: 'job_name'
    # metrics_path defaults to '/metrics'
    # scheme defaults to 'http'
    metrics_path: /OrderManagement/metrics
    static_configs:
    - targets:
      - MS1_hostname:MS1_port
      - MS2_hostname:MS2_port
      - MSn_hostname:Msn_port
    # HTTP basic authentication information
      basic_auth:
        username: metrics_username
        password: metrics_user_password
```

where:

- *job_name* refers to a particular job. For example, COM_Production, SOM_UAT, and so on. Use this to distinguish the various OSM instances such as COM_Production, COM_Pre-prod, SOM_QA, SOM_UAT, and so on. Each of these instances will have its own job in the scrape configuration.
- *MS1* refers to managed server 1, *MS2* refers to managed server 2 and so on.

> **Note**
>
> During the installation, the installer creates a metrics user who is authorized to view OSM metrics and WebLogic server metrics.

## Viewing OSM Metrics Without Using Prometheus

The OSM metrics can be viewed at:

```
http://hostname:port/OrderManagement/metrics
```


Copyright © 2009, 2025, Oracle and/or its affiliates.

This only provides metrics of the managed server that is serving the request. It does not provide the consolidated metrics for the entire cluster. Only Prometheus Query and Grafana dashboards can provide the consolidated metrics.

# Viewing OSM Metrics in Grafana

OSM metrics scraped by Prometheus can be made available for further processing and visualization.

The OSM installer comes with sample Grafana dashboards to get you started with visualizations. Import the dashboard JSON files from *OSM_SDK/***samples/grafana** into your Grafana environment.

The sample dashboards are:

- **OSM by Instance**: This dashboard provides a view of OSM metrics for one or more instances in the selected project namespace.
- **OSM by Server**: This dashboard provides a view of OSM metrics for one or more managed servers for a given instance in the selected project namespace.
- **OSM by Order Type**: This dashboard provides a view of OSM metrics for one or more order types for a given cartridge version in the selected instance and project namespace.
- **OSM and Weblogic by Server**: This dashboard provides a view of OSM and WebLogic Monitoring Exporter metrics for one or more managed servers for a given instance in the selected project namespace.

# Exposed Order Metrics

The following OSM metrics are exposed via Prometheus APIs.

> **Note**
>
> - All metrics are per managed server. Prometheus Query Language can be used to combine or aggregate metrics across all managed servers.
> - All metric values are short-lived and indicate the number of orders (or tasks) in a particular state since the managed server was last restarted.
> - When a managed server restarts, all the metrics are reset to **0**. These metrics do not refer to the exact values, which can be queried via OSM APIs such as Web Services and XML API.

### Order Metrics

The following table lists order metrics exposed via Prometheus APIs.

**Table 10-1 Order Metrics Exposed via Prometheus APIs**

| Name | Type | Help Text | Notes |
|---|---|---|---|
| osm_order_items | Histogram | Histogram that tracks the number of order items in an order with buckets for 0, 10, 25, 50, 100, 250, 1000, and 5000 order items. | N/A |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 10-1 (Cont.) Order Metrics Exposed via Prometheus APIs**

| Name | Type | Help Text | Notes |
|---|---|---|---|
| osm_orders_created | Counter | Counter for the number of Orders Created | N/A |
| osm_orders_completed | Counter | Counter for the number of Orders Completed | N/A |
| osm_orders_failed | Counter | Counter for the number of Orders Failed | N/A |
| osm_orders_cancelled | Counter | Counter for the number of Orders Cancelled | N/A |
| osm_orders_aborted | Counter | Counter for the number of Orders Aborted | N/A |
| osm_orders_in_progress | Gauge | Gauge for the number of orders currently in the In Progress state | N/A |
| osm_orders_amending | Gauge | Gauge for the number of orders currently in the Amending state | N/A |
| osm_short_lived_orders | Histogram | Histogram that tracks the duration of all orders in seconds with buckets for 1 second, 3 seconds, 5 seconds, 10 seconds, 1 minute, 3 minutes, 5 minutes, and 15 minutes.<br>Enables focus on short-lived orders. | Buckets for 1 second, 3 seconds, 5 seconds, 10 seconds, 1 minute, 3 minutes, 5 minutes, and 15 minutes. |
| osm_medium_lived_orders | Histogram | Histogram that tracks the duration of all orders in minutes with buckets for 5 minutes, 15 minutes, 1 hour, 12 hours, 1 day, 3 days, 1 week, and 2 weeks.<br>Enables focus on medium-lived orders. | Buckets for 5 minutes, 15 minutes, 1 hour, 12 hours, 1 day, 3 days, 7 days, and 14 days. |
| osm_long_lived_orders | Histogram | Histogram that tracks the duration of all orders in days with buckets for 1 week, 2 weeks, 1 month, 2 months, 3 months, 6 months, 1 year and 2 years.<br>Enables focus on long-lived orders. | Buckets for 7 days, 14 days, 30 days, 60 days, 90 days, 180 days, 365 days, and 730 days. |
| osm_order_cache_entries_total | Gauge | Gauge for the number of entries in the cache of type order, orchestration, historical order, closed order, and redo order. | N/A |
| osm_order_cache_max_entries_total | Gauge | Gauge for the number of maximum entries in the cache of type order,orchestration, historical order, closed order, and redo order. | N/A |


Copyright © 2009, 2025, Oracle and/or its affiliates.

### Labels For All Order Metrics

The following table lists labels for all order metrics.

**Table 10-2 Labels for All Order Metrics**

| Label Name | Sample Value | Notes | Source of the Label |
|---|---|---|---|
| cartridge_name_version | view_framework_demo_1.0.0.0.0 | Combined Cartridge Name and Version | OSM Metric Label Name/Value |
| order_type | vf_demo_web | OSM Order Type | OSM Metric Label Name/Value |
| server_name | ms1 | Name of the Managed Server | OSM Metric Label Name/Value |
| instance | 10.244.0.198:8081 | Indicates the IP address and port of the host from which this metric is being scraped. | Prometheus Service Discovery |
| job | COM_Production | Job name in Prometheus configuration which scraped this metric. | Prometheus Service Discovery |

### Task Metrics

The following metrics are captured for Manual or Automated Task Types only. All other Task Types are currently not being captured.

**Table 10-3 Task Metrics Captured for Manual or Automated Task Types Only**

| Name | Type | Help Text |
|---|---|---|
| osm_tasks_created | Counter | Counter for the number of Tasks Created |
| osm_tasks_completed | Counter | Counter for the number of Tasks Completed |

### Labels for all Task Metrics

A task metric has all the labels that an order metric has. In addition, a task metric has two more labels.

**Table 10-4 Labels for All Task Metrics**

| Label | Sample Value | Notes | Source of Label |
|---|---|---|---|
| task_name | enter_account_information | Task Name | OSM Metric Label Name/Value |
| task_type | A | **A** for Automated<br>**M** for Manual | OSM Metric Label Name/Value |

# Managing WebLogic Server Metrics

OSM provides a sample Grafana dashboard that you can use to visualize WebLogic server metrics available from a Prometheus data source. You use WebLogic Monitoring Exporter (WME) to expose the WebLogic server metrics. WebLogic Monitoring Exporter is part of the WebLogic Kubernetes toolkit. It is an open source project, based at: https://github.com/oracle/weblogic-monitoring-exporter. While the metrics are available via WME Restful Management


Copyright © 2009, 2025, Oracle and/or its affiliates.

API endpoints, OSM relies on Prometheus to scrape and expose these metrics. This version of OSM supports WME 1.3.0. See the WME documentation for details on configuration and the exposed metrics.

The following topics describe a sample integration:

- Deploying WebLogic Monitoring Exporter (WME) in OSM
- Configuring the Prometheus Scrape Job for WME Metrics
- Viewing WebLogic Monitoring Exporter (WME) Metrics Without Using Prometheus
- Viewing WebLogic Monitoring Exporter Metrics in Grafana

# Deploying WebLogic Monitoring Exporter (WME) in OSM

To deploy WebLogic Monitoring Exporter:

1. Generate the WME WAR file by running the following commands:

```
mkdir -p ~/wme
cd ~/wme

curl -x $http_proxy -L https://github.com/oracle/weblogic-monitoring-
exporter/releases/download/v1.3.0/wls-exporter.war -o wls-exporter.war
curl -x $http_proxy https://raw.githubusercontent.com/oracle/weblogic-
monitoring-exporter/v1.3.0/samples/kubernetes/end2end/dashboard/exporter-
config.yaml -o config.yml

jar -uvf wls-exporter.war config.yml
```

> **Note**
>
> WME requires the configuration file to be named `config.yml` for automatic configuration. If you specify a different file name, then you have to manually update the configuration.

The commands update the **wls-exporter.war** file with the **exporter-config.yaml** configuration file.

2. Deploy the WME WAR file by running the following command:

```
java -cp path_to_weblogic_server_lib weblogic.Deployer -adminurl t3://
host_name:adminserver_port -user wls_admin_username -password
wls_admin_password -deploy -name name_of_the WME_WAR_file -source  path_to
MWE_WAR_file -targets wls_server_targets_list

### Example :
java -cp /../../Oracle/WLS/12_2_1_4/wlserver/server/lib/weblogic.jar
weblogic.Deployer -adminurl t3://localhost:7001 -user weblogic -password
password -deploy -name wls-exporter -source  /../../wme/wls-exporter.war -
targets AdminServer,Cluster_Name
```


Copyright © 2009, 2025, Oracle and/or its affiliates.

# Configuring the Prometheus Scrape Job for WME Metrics

Configure the scrape job in Prometheus as follows:

```
- job_name: 'wme_job_name'
  metrics_path: wls-exporter/metrics
  basic_auth:
    username: weblogic_username
    password: weblogic_password
  static_configs:
  - targets: [AdminServer_hostname:AdminServer_port]
    labels:
      # The namespace label enables multiple related instances to be grouped.
      # For a given WebLogic server, the namespace used in a WME Prometheus
job must match
      # the namespace used for the corresponding OSM Prometheus job.
      # In a sample Grafana dashboard, the specified namespace is displayed
under the Project
      # drop-down menu.
      namespace: namespace
      # The weblogic_domainUID label uniquely identifies an OSM instance
within a given namespace.
      # For a given WebLogic server, the weblogic_domainUID used in a WME
Prometheus job must match
      # the weblogic_domainUID used for the corresponding OSM Prometheus job.
      # In a sample Grafana dashboard, the specified weblogic_domainUID is
displayed under the
      # Instance drop-down menu.
      weblogic_domainUID: weblogic_domainUID
      # The weblogic_serverName label must match the server name configured
in WebLogic.
      weblogic_serverName: AdminServer

  # Repeat this pattern for each managed server
  - targets: [MSn_hostname:Msn_port]
    labels:
      namespace: namespace
      weblogic_domainUID: weblogic_domainUID
      weblogic_serverName: MSn
```

The `namespace` label enables multiple related instances to be grouped. For a given WebLogic server, the namespace used in a WME Prometheus job must match the namespace used for the corresponding OSM Prometheus job. In a sample Grafana dashboard, the specified namespace is displayed under the Project drop-down menu.

The `weblogic_domainUID` label uniquely identifies an OSM instance within a given namespace. For a given WebLogic server, the `weblogic_domainUID` label used in a WME Prometheus job must match the weblogic_domainUID used for the corresponding OSM Prometheus job. In a sample Grafana dashboard, the specified weblogic_domainUID is displayed under the Instance drop-down menu.

The `weblogic_serverName` label must match the server name configured in WebLogic.

To enable correlation with WME metrics, the `namespace` and `weblogic_domainUID` labels have been added to the corresponding scrape job definition for OSM metrics. Note that, with these


Copyright © 2009, 2025, Oracle and/or its affiliates.

new labels, the dashboard's JSON files can be reused between OSM traditional and OSM cloud native deployments.

```
- job_name: 'osm_job_name'
  # metrics_path defaults to '/metrics'
  # scheme defaults to 'http'
  metrics_path: OrderManagement/metrics
  static_configs:
  - targets: [MSn_hostname:Msn_port]
    labels:
      namespace: namespace
      weblogic_domainUID: weblogic_domainUID
```

## Viewing WebLogic Monitoring Exporter (WME) Metrics Without Using Prometheus

To view WME metrics of the admin server without using Prometheus, access the following URL:

```
http://adminserver_host:adminserver_port/wls-exporter/metrics
```

To view WME metrics of managed servers without using Prometheus, access the following URL:

```
http://managedServerN_host:managedServerN_port/wls-exporter/metrics
```

## Viewing WebLogic Monitoring Exporter Metrics in Grafana

OSM provides sample Grafana dashboards to get you started with visualizations. The sample **OSM and WebLogic by Server** dashboard provides a combined view of OSM cloud native and WebLogic Monitoring Exporter metrics for one or more managed servers for a given instance in the selected project namespace.

Import the dashboard JSON file from **OSM_SDK/samples/Grafana** into your Grafana environment, selecting Prometheus as the data source.

# Managing Database Connections

In OSM, database connections are managed through database pools that are set up in WebLogic Server.

The database pool connections can be configured through the WebLogic RemoteConsole by clicking **Edit Tree/Services/Data Sources/<oms_pool>**. Here, open the **Connection Pool** tab.

Use the parameters on the Connection Pool page to modify and tune your connection pool settings.

# 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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

> **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 the WebLogic Remote Console

   Click **Edit Tree/Services/Messaging/JMS Runtime/JMS Servers/server/Destination**.

   The **General Configuration** page is displayed.

2. Click **Monitoring Tree/Services/Messaging/JMS Runtime/JMS Servers/server/Destinations**.

   A list of active destinations targeted to the server is displayed.

> **Note**
>
> The default view of the table does not contain the **Consumers Current** column. We recommend that you customize the table using the **Customize** link to include this column, along with any other customizations you may want to make.

The **Consumers Column** column defines the current number of listeners on the destination. If a destination does not have any listeners, the external system does not receive the messages.

The **Messages Current** column defines the current number of unprocessed messages in the JMS destination. A large number of messages (for example, 10,000) in this destination is a problem. It means that the system is not keeping up, 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.

OSM has the following JMS destinations 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
- 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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

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_SDK*\SDK\Samples\DatabasePlugin\map**automationmap.xml** file and make sure that all of the plug-ins are 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, 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 still uses the non-XA emulated two-phase commit. Note that this also applies to JMS queues that support Application Integration Architecture (AIA) cartridges.

> **Note**
>
> When you are using **JTA with XA-enabled messaging** with either SAF (Store-and-Forward) messaging or JMS bridge messaging (with *Exactly-once* QoS), it is recommended to enable Cross Domain Trust instead of Global Trust.
>
> Enabling Global Trust introduces security risks, especially for anonymous users and has been observed to cause JMS message rollbacks with errors. To ensure secure and reliable message delivery, always configure Cross Domain Trust.


Copyright © 2009, 2025, Oracle and/or its affiliates.

# Using Work Managers to Prioritize Work

You use WebLogic Remote 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 performed. 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.

> **Note**
>
> See the *OSM default Work Managers, Constraints related to upgrade process (Doc ID 3019290.1)* knowledge article on My Oracle Support, if you are upgrading from OSM version 7.3.1 or lower (source version) to OSM version 7.3.5 or higher (target version).

Table 10-5 shows the default work managers and components that are created and configured by the installer.

**Table 10-5 Default Work Managers and Components**

| Work Manager or Component | Configuration Properties | Description |
|---|---|---|
| Maximum Threads Constraint | **Name**: osmMaxThreadConstraint<br>**Count**: 90% of the number of connections in the OSM datasource | Used by all OSM Work Managers to prioritize work. |
| Minimum Threads Constraint | **Name**: osmMinThreadConstraint<br>**Count**: 10 | Used by the work managers osmTaskClientWorkManager and osmOmClientWorkManager to guarantee that some threads are reserved for the web clients. |
| Work Manager | **Name**: osmAutomationWorkManager<br>**Maximum Threads Constraint**: osmMaxThreadConstraint | Used to process work performed by automation tasks. |
| Work Manager | **Name**: osmXmlWorkManager<br>**Maximum Threads Constraint**: osmMaxThreadConstraint | Used to process requests coming from external clients for the OSM XML API. |
| Work Manager | **Name**: osmTaskClientWorkManager<br>**Maximum Threads Constraint**: osmMaxThreadConstraint<br>**Minimum Threads Constraint**: osmMinThreadConstraint<br>**Request Class:** osmFairShareReqClass | Used to process requests from manual users using the Task web client. |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 10-5 (Cont.) Default Work Managers and Components**

| Work Manager or Component | Configuration Properties | Description |
|---|---|---|
| Work Manager | **Name**: osmWsJmsWorkManager<br>**Maximum Threads Constraint**: osmMaxThreadConstraint | Used to process OSM JMS Web Service requests. |
| Work Manager | **Name**: osmWsHttpWorkManager<br>**Maximum Threads Constraint**: osmMaxThreadConstraint | Used to process OSM HTTP Web Service requests. |
| Work Manager | **Name**: osmOmClientWorkManager<br>**Maximum Threads Constraint**: osmMaxThreadConstraint<br>**Minimum Threads Constraint**: osmMinThreadConstraint | Used to process requests from manual users using the Order Management web client. |

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 Remote Console.
2. Click on **Edit Tree**, then **Scheduling**, and then **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. Enter a name for the new Work Manager, and then click **Create**.
5. Click on the **Targets** tab and select the deployment target from the list of available targets
6. Click **Save**.

   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 are displayed 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. For more information about working with **oms.ear** file, see *OSM Developer's Guide*.


Copyright © 2009, 2025, Oracle and/or its affiliates.

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 are displayed in Czech. Removing the **resources_cs.properties** file causes the web client prompts to be displayed in English, even though the browser language setting is still configured to the Czech language.

# 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

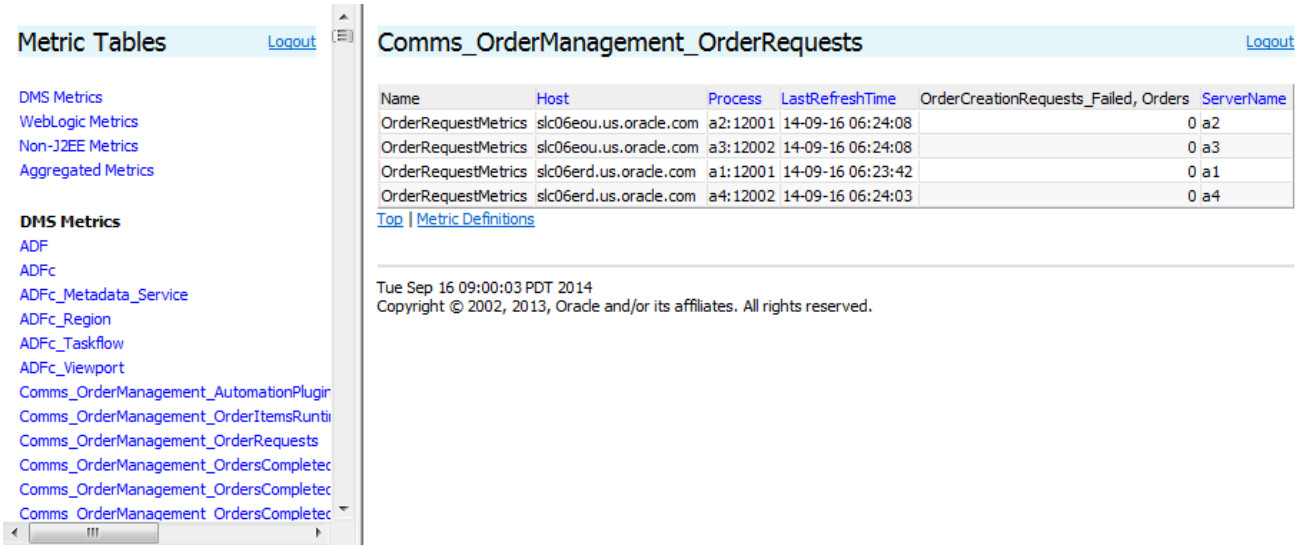
Metrics data is gathered and displayed using tables. You view OSM-specific metric data, which is displayed in the DMS Metrics tables and Aggregated Metrics tables. You can also use the DMS interface to view non-OSM metric data, for example, WebLogic and non-J2EE metrics.

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 10-1 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.

**Figure 10-1 DMS Order Requests Metrics Table**




To view metrics tables using DMS:

1. In a browser, enter one of the following URLs:


Copyright © 2009, 2025, Oracle and/or its affiliates.

`http://hostname:port/dms`

`https://hostname.dnsdomainname.tld:sslport/dms`

where *hostname* is the name of the WebLogic administration server, *port* is the port for the administration server domain, and *sslport* is the secure port for the administration 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


Copyright © 2009, 2025, Oracle and/or its affiliates.

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

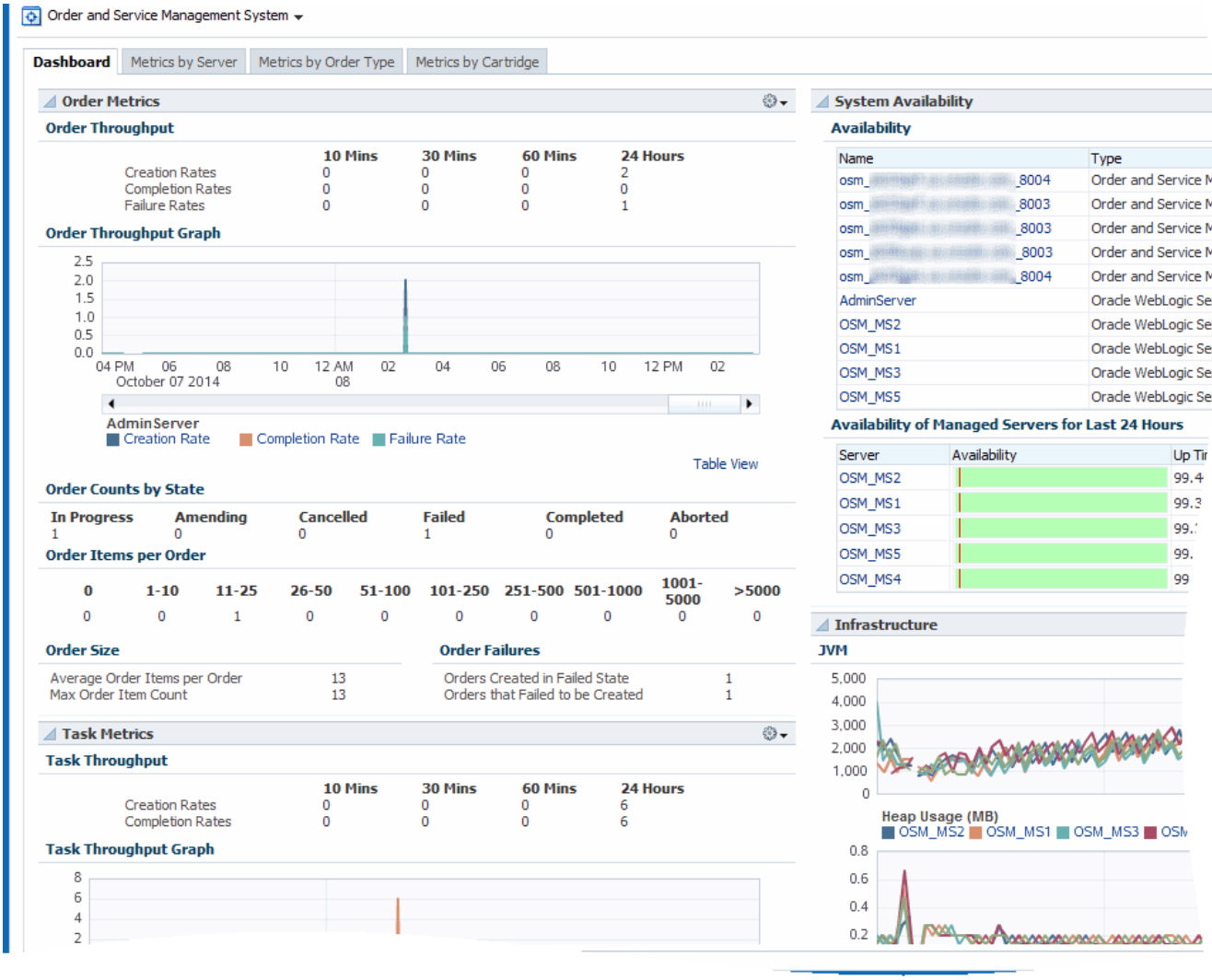
You can view the metric data that is gathered by the system about your environment by accessing the DMS interface that is provided by default with your OSM installation. The tables provide information about various aspects of your OSM environment, and you can compare the pieces of data over time to determine 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 Communications System Administrator's Guide*.

Figure 10-2 shows the Application Management Pack dashboard for an OSM system. The interface also displays metrics by server, order type, and cartridge.


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Figure 10-2 OSM System Data in the AMP Dashboard**




# Discovering a Cluster Database Target

You can use Enterprise Manager to associate a Oracle RAC database to an OSM target, and then view the association in the Topology page.

To associate a Oracle RAC database to an OSM target, you must first discover the Oracle RAC database as a cluster database target. Discovering the Oracle RAC database as a cluster database target is a two-step process of adding a cluster target, and then discovering the cluster database target by selecting the one that you added.

For more information about discovering targets, see *Enterprise Manager Cloud Control Administrator's Guide*.

To add a cluster target:

1. In Enterprise Manager, click **Setup**, **Add Target**, and then click **Add Target Manually**.
2. Select **Add Targets Using Guided Process**.
3. In the **Target Types** list, select **Oracle Cluster and High Availability Service**.


Copyright © 2009, 2025, Oracle and/or its affiliates.

4. Click **Add Using Guided Process**, and discover the Oracle RAC database.
5. Save the Oracle RAC database as a cluster target.

   The Oracle RAC database is added as a cluster target.

To discover and associate the cluster database target:

1. In Enterprise Manager, click **Setup**, **Add Target**, and then click **Add Target Manually**.
2. Select **Add Targets Using Guided Process**.
3. In the **Target Types** list, select **Oracle Database, Listener, and Automatic Storage Management**.
4. Click **Add Using Guided Process**, and select the added cluster target.
5. Follow the steps in the wizard to add it as a cluster database target.

   The Oracle RAC database is added as a cluster database target.
6. On the dashboard page of the OSM target, in Enterprise Manager, click **Associate RAC Database**.

> **Note**
>
> To successfully associate the Oracle RAC database to the cluster target, ensure that the OSM target servers are running.

The Oracle RAC database is associated to the target that you specified.


Copyright © 2009, 2025, Oracle and/or its affiliates.

11

# Exporting and Importing OSM Schema Data

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

You can export an OSM schema using standard Oracle Database tools. 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. 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.

This chapter provides information about exporting and importing order and model data from one database to another using the OSM export/import utility package. Exporting only the data that is causing an issue allows support to more quickly locate and resolve issues. You can use the utilities in the package to dynamically generate the files 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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

> **Note**
>
> Model tables that are system-managed, or that 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.

# 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:

   ```
   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.

   > **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. Open another terminal and take the OSM server offline.
4. Return to the original terminal and allow the output to be generated by running the following command:

   ```
   set serveroutput on
   ```

5. 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.

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:

   ```
   spool osm_expdp_model_parfile.txt
   ```

7. Export the model by running the **om_exp_imp_util_pkg.generate_expdp_model_parfile** procedure:

   ```
   exec om_exp_imp_util_pkg.generate_expdp_model_parfile
   ```

8. Stop capturing the generated output by running the following command:


Copyright © 2009, 2025, Oracle and/or its affiliates.

```
spool off
```

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. 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 the "Oracle Data Pump Export" section in *Oracle Database Utilities*.

14. Restart the OSM server you exported the data from.

# 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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

## 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. If you require a fully functional environment, you must also manually create any Weblogic resources such as users, groups, any custom queues, and so on.

## 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.

If the source schema uses partitioning realms (for example, for short-lived orders), the size and order of the partitions and the partitioning realms that they are associated with on the source system, must be duplicated on the target system. Partitions associated with partitioning realms must be added after OSM model data is imported because the partitioning realms are imported with OSM model data. See "Partitioning Realms" for more information.

If the source schema has the partitions and partitioning realms that are shown in Table 11-1, for example, the same size of partitions and realms must be created on the target schema. For this example, you run the following commands after you create the partitioning realms in the target schema using OSM model import:

```
exec om_part_maintain.add_partition(a_tablespace, 'default_order');
exec om_part_maintain.add_partition(a_tablespace, 'short_lived_orders');
```

where *a_tablespace* is the tablespace for 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 if the **a_tablespace** value 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).

Table 11-1 shows an example of a set of partitions that use the partitioning realms feature.

**Table 11-1 Example Set of Partitions That Use Partitioning Realms**

| Partition Name | Partition Size | order_seq_id Lower Bound | order_seq_id Upper Bound | Partitioning Realm |
|---|---|---|---|---|
| P_000000000000100001 | 100000 | 1 | 100000 | default_order |
| P_000000000010600001 | 500000 | 10100001 | 10600000 | short_lived_orders |

For information about using partitioning realms, see "Partitioning Realms". 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:

   ```
   exec om_part_maintain.add_partitions(n, tablespace, realm_mnemonic)
   ```

   where:

   *(n)* is the number of partitions that you want to add


Copyright © 2009, 2025, Oracle and/or its affiliates.

(*tablespace*) is the tablespace for the new partition(*realm_mnemonic*) is the mnemonic of the partitioning realm associated with this partition. This value is used only for schemas that use partitioning realms. The default realm is **default_order** if this value is not provided.

3. Query the **user_tab_partitions** table to view the newly added partitions.

```
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 "Adding Partitions (Online or Offline)" 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. Open a terminal and take the OSM server offline.
2. (Optional) Purge the existing OSM model 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 there are no constraint violations when importing duplicate data.

3. Open another terminal, log in to SQL*Plus as the OSM core schema user.
4. Disable constraints and triggers, and stop jobs by running the following command:

```
exec om_exp_imp_util_pkg.pre_impdp_setup
```

   This command ensures that there are no errors when importing OSM data.

5. From another terminal, import the model tables. For example:

```
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
```

6. 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.

7. Restart the OSM server.


Copyright © 2009, 2025, Oracle and/or its affiliates.

# 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:

```
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:

```
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 is displayed:

```
There are no open orders
```


Copyright © 2009, 2025, Oracle and/or its affiliates.

If the order specified is open and the server should be taken offline, the following message is displayed:

```
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.
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
   ```

8. Restart the OSM server.


Copyright © 2009, 2025, Oracle and/or its affiliates.

# 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. Stop the OSM server that you want to import to.
2. (Optional) Purge existing OSM orders by running the following command:

   ```
   exec om_exp_imp_util_pkg.purge_orders
   ```

   > **Note**
   >
   > The orders must match the model. Purging the data before importing it to the target schema ensures there are no constraint violations when importing duplicate data.

3. (Optional) Purge the existing OSM model 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 there are no constraint violations when importing duplicate data.

4. Log in to SQL*Plus as the OSM core schema user.
5. 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.

6. Import the model data by running the following command:


Copyright © 2009, 2025, Oracle and/or its affiliates.

```
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".

7. 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".

8. 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.

9. Restart the OSM server.

# 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, "Exporting OSM Order Data".

## 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:

```
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 **configDB.sh** script to create it.


Copyright © 2009, 2025, Oracle and/or its affiliates.

3. Verify that none of the orders that you want to export are open by running the following SQL commands:

   ```
   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".

   The following message is displayed:

   ```
   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 is displayed:

   ```
   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.
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 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:

   ```
   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:

   ```
   exec om_exp_imp_util_pkg.generate_expdp_order_parfile(a_min_order_id => 100000);

   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=(
   ```


Copyright © 2009, 2025, Oracle and/or its affiliates.

```
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:

   ```
   spool off
   ```

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. Restart the OSM server.

# 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. Stop the OSM server that you want to import to.

2. (Optional) Purge existing OSM orders by running the following command:

   ```
   exec om_exp_imp_util_pkg.purge_orders
   ```


Copyright © 2009, 2025, Oracle and/or its affiliates.

> **Note**
>
> The orders must match the model. Purging the data before importing it to the target schema ensures there are no constraint violations when importing duplicate data.

3. (Optional) Purge the existing OSM model 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 there are no constraint violations when importing duplicate data.

4. Log in to SQL*Plus as the OSM core schema user.
5. 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.

6. 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".

7. 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".

8. 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.

9. Restart the OSM server.

# 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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

> **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 **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 (**om_exp_imp_util_pkg**) exists by running the following command:

```
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 orders that you want to export are not open by running the following SQL commands:

```
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 is displayed:

```
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 is displayed:


Copyright © 2009, 2025, Oracle and/or its affiliates.

```
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.
5. Allow the output to be generated by running the following command:

   ```
   set serveroutput on
   ```

6. Prevents 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 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:

   ```
   spool osm_expdp_orders_parfile.txt
   ```

3. Export orders by running the **om_exp_imp_util_pkg.generate_expdp_order_parfile** procedure. For example:

   ```
   exec om_exp_imp_util_pkg.generate_expdp_order_parfile(a_min_order_id => 100000,
   a_max_order_id => 200000);

   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
   )
   ```


Copyright © 2009, 2025, Oracle and/or its affiliates.

```
PL/SQL procedure successfully completed.
```

4. Stop capturing the generated output by running the following command:

   ```
   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:

   ```
   expdp SourceOSMCoreSchemaUserName PARFILE=osm_expdp_orders_parfile.txt
   ```

7. Restart the OSM server.

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. Stop the target OSM server.
2. Log in to SQL*Plus as the OSM core schema user.
3. 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.

4. 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".

5. 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".

6. Finalize the target OSM schema by running the following command:

   ```
   exec om_exp_imp_util_pkg.post_impdp_setup
   ```


Copyright © 2009, 2025, Oracle and/or its affiliates.

This enables the constraints and triggers, and resubmits the jobs that were disabled and stopped before importing the OSM data.

7. Restart the OSM server.

# About Order Export Queries

You can use a variety of queries to select the orders that you want to export. Table 11-2 provides examples of queries that you can use to select the orders to export.

**Table 11-2 Example Queries for Selecting Orders to Export**

| Example Query | Data to Export |
|---|---|
| `exec om_exp_imp_util_pkg.generate_expdp_order_parfile;` | Exports all orders when exporting order tables that define an order sequence ID.<br>You can also use this example and edit the PAR file to customize the QUERY option. |
| `exec om_exp_imp_util_pkg.generate_expdp_order_parfile(a_max_order_id => 2000);` | This example exports all orders below order ID 2000 for order tables that define an order sequence ID. |
| `exec om_exp_imp_util_pkg.generate_expdp_order_parfile(a_min_order_id => 1000);` | This example exports all orders above order ID 1000 for order tables that define an order sequence ID. |
| `exec om_exp_imp_util_pkg.generate_expdp_order_parfile(a_min_order_id => 1000, a_max_order_id => 2000);` | This example exports all orders between order ID 1000 and order ID 2000 for order tables that define an order sequence ID. |

# 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 11-3 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 "Oracle Data Pump Export" section in *Oracle Database Utilities*.


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 11-3 PAR File Parameters**

| PAR File Parameter | Default | Description |
|---|---|---|
| CONTENT | ALL | Enables you to filter what export unloads: data only, metadata only, or both.<br>**Note**: If you change this parameter in any of the export scenarios in this chapter, the export will **not** be successful. |
| DIRECTORY | DATA_PUMP_DIR | Specifies the directory to which export writes the dump file and the log file.<br>**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 "Oracle RAC Considerations" section in the Oracle Database Data Pump documentation.<br>**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.<br>The following sqlplus commands create and set permissions on a new variable PDB directory variable:<br>`create directory pdb_variable_name as 'path';`<br>`grant read, write on directory pdb_variable_name to osm_core_schema;`<br>where:<br>• *pdb_variable_name* is the name of the PDB directory variable<br>• *path* is the path to the data pump directory (for example /samplepath/pdbdatapumpdir)<br>• *osm_core_schema* is the core OSM schema (for example ordermgmt) |
| 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.<br>**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. |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 11-3 (Cont.) PAR File Parameters**

| PAR File Parameter | Default | Description |
|---|---|---|
| CLUSTER | Y | This parameter is not available in the generated PAR file. This parameter determines whether Data Pump can use Oracle Real Application Cluster's (Oracle RAC) resources and start workers on other Oracle RAC instances. To force Data Pump Import to use only the instance where the job is started, add CLUSTER=N in the PAR file. Otherwise, ignore this parameter. In Oracle database for import, the default value is Y. |

# About Import Parameters

You can add parameters in-line to data pump import command (impdp).

Table 11-4 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 "Oracle Data Pump Import" section in *Oracle Database Utilities Guide*.

**Table 11-4 Import In-line Parameters**

| PAR File Parameter | Description |
|---|---|
| DIRECTORY | Specifies the directory to which import finds the dump file and the log file generated by the export.<br>**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.<br>**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 impdp with must have its own data pump directory.<br>The following sqlplus commands create and set permissions on a new variable PDB directory variable:<br>`create directory pdb_variable_name as 'path';`<br>`grant read, write on directory pdb_variable_name to osm_core_schema;`<br>where:<br>• *pdb_variable_name* is the name of the PDB directory variable<br>• *path* is the path to the data pump directory (for example /samplepath/pdbdatapumpdir)<br>• *osm_core_schema* is the core OSM schema (for example ordermgmt) |
| 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. |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 11-4 (Cont.) Import In-line Parameters**

| PAR File Parameter | Description |
|---|---|
| TRANSFORM | Specifies whether the types being created should be assigned a new object identifier (OID). For example:<br>`TRANSFORM=oid:n` |
| REMAP_SCHEMA | This parameter specifies the source schema and the target schema from which all objects are loaded to.<br>When importing, if the source and target schema are the same, the **REMAP_SCHEMA** option does not need to be specified. |
| REMAP_TABLESPACE | This parameter remaps all objects selected for import with persistent data in the source tablespace to be created in the target tablespace.<br>When importing, if the source and target schema are the same, the **REMAP_TABLESPACE** option does not need to be specified. |
| CLUSTER | This parameter is not available in the generated PAR file. This parameter determines whether Data Pump can use Oracle Real Application Cluster's (Oracle RAC) resources and start workers on other Oracle RAC instances. To force Data Pump Import to use only the instance where the job is started, add CLUSTER=N in the PAR file. Otherwise, ignore this parameter. In Oracle database for import, the default value is Y. |

# Troubleshooting Export/Import

Errors might occur during the process of exporting or importing data.

Table 11-5 lists some common export errors and their solutions. For more information about troubleshooting errors that might occur when exporting or importing data, see the troubleshooting section of the knowledge article about exporting and importing data [Doc ID 1594152.1], available from the Oracle support website:

`https://support.oracle.com`

For information about Oracle RAC and data pump import, see the "Oracle RAC Considerations" and "Oracle Data Pump Import" sections in *Oracle Database Utilities*.

For information about predefined roles in an Oracle Database installation, and about guidelines for securing user and accounts privileges, see *Oracle Database Security Guide*.

**Table 11-5 Common Export Errors**

| Error | Cause | Solution |
|---|---|---|
| ORA-39097: Data Pump job encountered unexpected error -12801<br>ORA-39065: unexpected master process exception in MAIN<br>ORA-12801: error signaled in parallel query server PZ99, instance <instanceDetails> (4)<br>ORA-01460: unimplemented or unreasonable conversion requested | There is an issue with the Oracle Data Pump export tool and Oracle RAC databases.<br>For more information, see the knowledge article about the issue [Doc ID 13099577.8], available from the Oracle support website:<br>`https://support.oracle.com` | Update the following database parameters:<br>`ALTER SYSTEM set parallel_force_local=true`<br>`ALTER SYSTEM set parallel_max_servers=0`<br>`ALTER SYSTEM set parallel_min_servers=0` |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 11-5 (Cont.) Common Export Errors**

| Error | Cause | Solution |
|---|---|---|
| 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, run the following command in SQL Plus before generating the options PAR files:<br>`SQL> SET TRIMSPOOL ON` |
| ORA-39002: invalid operation<br>ORA-39070: Unable to open the log file.<br>ORA-29283: invalid file operation<br>ORA-06512: at "SYS.UTL_FILE", line 536<br>ORA-29283: invalid file operation | The location specified for the export DIRECTORY option is not accessible. | Update the DIRECTORY option specified in the export command.<br>For exports or imports performed in an Oracle RAC environment using Automatic Storage Management, the DIRECTORY option should be updated to point to the shared location.<br>For more information, see the Data Pump: Oracle RAC Considerations section of Oracle Database documentation. |
| ORA-39001: invalid argument value<br>ORA-39000: bad dump file specification<br>ORA-31641: unable to create dump file "+DATA/osm_expdp_model.dmp"<br>ORA-17502: ksfdcre:4 Failed to create file +DATA/osm_expdp_model.dmp<br>ORA-15005: name "osm_expdp_model.dmp" is already used by an existing alias | A previously generated version of the dmp file already exists. | Remove the previously generated version of the dmp file.<br>or<br>Specify the following option in the export PAR file to overwrite it:<br>`REUSE_DUMPFILES=YES` |
| ORA-31693: Table data object "<OSMCoreSchemaUserName>"."OM_ORDER_HEADER":"P_<br>000000000000005001"."SYS_SUBP612617" failed to load/unload and is being skipped due to error:<br>ORA-06502: PL/SQL: numeric or value error: character string buffer too small<br>ORA-01403: no data found<br>ORA-01403: no data found | The data pump export job name is too long.<br>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.<br>For more information, see the knowledge article about the issue [Doc ID 1502119.1], available from the Oracle support website:<br>https://support.oracle.com | Specify the JOB_NAME option in the export PAR file. Make sure the value specified for the option is less than 26 characters long.<br>`JOB_NAME=osm_expdp_job` |

Table 11-6 lists some common import errors and their solutions.


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 11-6 Common Import Errors**

| Error | Cause | Solution |
|---|---|---|
| ORA-31693: Table data object "OSMCoreSchemaUserName"."OM_ORCH_PLAN":"SYS_SUBP44607" failed to load/unload and is being skipped due to error:<br>ORA-29913: error in executing ODCIEXTTABLEOPEN callout<br>ORA-29400: data cartridge error<br>ORA-39779: type "<OSMCoreSchemaUserName>"."OM_T_ORCH_PROCESS" not found or conversion to latest version is not possible | The types being created should be assigned a new object identifier (OID).<br>For more information, see the knowledge article about the issue [Doc ID 351519.1], available from the Oracle support website:<br>`https://support.oracle.com` | Make sure the following option is specified when importing. This option assigns new OIDs.<br>`TRANSFORM=oid:n`<br>For more information, see the Data Pump: Import TRANSFORM section of Oracle Database documentation. |
| ORA-31693: Table data object "<OSMCoreSchemaUserName>"."<OSMTableName>":"SYS_P44941" failed to load/unload and is being skipped due to error:<br>ORA-29913: error in executing ODCIEXTTABLEFETCH callout<br>ORA-14400: inserted partition key does not map to any partition | The order IDs being imported are greater than the greatest partition upper bound. | Resolve this error by adding partitions.<br>For more information, see "Adding Partitions". |
| ORA-31693: Table data object "<OSMCoreSchemaUserName>"."<OSMTableName>" failed to load/unload and is being skipped due to error:<br>ORA-00001: unique constraint (<OSMCoreSchemaUserName>.<OSMTableName>) violated | The table already contains the data that is being imported. | Before you import, clean up the table using one or more of the following purge commands:<br>`SQL> exec om_exp_imp_util_pkg.purge_orders`<br>`SQL> exec om_exp_imp_util_pkg.purge_model`<br>`SQL> exec om_exp_imp_util_pkg.purge_schema` |
| ORA-39001: invalid argument value<br>ORA-39046: Metadata remap REMAP_TABLESPACE has already been specified. | The same source tablespace has been specified more than once for the REMAP_TABLESPACE option. | 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. |
| ORA-00932: inconsistent datatypes: expected OM_T_ORCH_PROCESS got OM_T_ORCH_PROCESS | There is a known issue with data pump import that causes imports with REMAP_SCHEMA and TYPE definitions to generate this error. | 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 > 0. |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 11-6 (Cont.) Common Import Errors**

| Error | Cause | Solution |
|---|---|---|
| UDI-31626: operation generated ORACLE error 31626<br>ORA-31626: job does not exist<br>ORA-39086: cannot retrieve job information<br>ORA-06512: at "SYS.DBMS_DATAPUMP", line 3326<br>ORA-06512: at "SYS.DBMS_DATAPUMP", line 4551<br>ORA-06512: at line 1 | This is an issue with data pump import privileges.<br>For more information, see the knowledge article about the issue [Doc ID 1459430.1], available from the Oracle support website:<br>https://support.oracle.com | Apply Patch 13715680<br>Or<br>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.<br>https://support.oracle.com<br>The missing privileges are:<br>`SQL> GRANT lock any table TO datapump_imp_full_database;`<br>`SQL> GRANT alter any index TO datapump_imp_full_database;` |
| ORA-31693: Table data object "<InvalidSourceOSMCoreSchemaUserName>"."OM_SQL_LOG" failed to load/unload and is being skipped due to error:<br>ORA-00001: unique constraint (<InvalidSourceOSMCoreSchemaUserName>.XPKOM_SQL_LOG) violated<br>ORA-31693: Table data object "<InvalidSourceOSMCoreSchemaUserName>"."OM_MODEL_CLOB" failed to load/unload and is being skipped due to error:<br>ORA-29913: error in executing ODCIEXTTABLEFETCH callout<br>ORA-00001: unique constraint (<InvalidSourceOSMCoreSchemaUserName>.XPKOM_MODEL_CLOB) violated | Import fails if the wrong source schema name is specified.<br>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. | Verify that the correct source OSM core schema user name is specified in the import command:<br>`impdp` *`TargetOSMCoreSchemaUserName`* `DIRECTORY=DATA_PUMP_DIR DUMPFILE=osm_expdp_orders%U.dmp LOGFILE=osm_impdp_orders.log REMAP_SCHEMA=`*`SourceOSMCoreSchemaUserName:TargetOSMCoreSchemaUserName`*<br>`REMAP_TABLESPACE=`*`SourceOSMTablespace:TargetOSMTablespace`*<br>`TRANSFORM=oid:n` |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 11-6 (Cont.) Common Import Errors**

| Error | Cause | Solution |
|---|---|---|
| ORA-31631: privileges are required<br>ORA-39122: Unprivileged users may not perform REMAP_SCHEMA remappings. | The schema user being used for import does not have the **imp_full_database** role.<br>This role is required for imports only if you cannot use the system user because of the REMAP_SCHEMA bug.<br>For more information, see the knowledge article about the issue [Doc ID 1367290.1], available from the My Oracle support website:<br>https://support.oracle.com<br>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). | Grant **imp_full_database** role to the OSM core schema user. The following command grants this role:<br>`SQL> GRANT imp_full_database TO <OSMCoreSchemaUserName>;`<br>**Caution**: This role provides a user with access to any data in any schema in the database. Use caution when granting this role to users.<br>For more information, see "Predefined Roles in an Oracle Database Installation" in *Oracle Database Security Guide*.<br>Oracle recommends removing this role after the import is complete. For more information, see "Guidelines for Securing User Accounts and Privileges" in *Oracle Database Security Guide*.<br>The following command removes this role:<br>`SQL> REVOKE imp_full_database FROM <OSMCoreSchemaUserName>;` |


Copyright © 2009, 2025, Oracle and/or its affiliates.

12

# Configuring Time Zone Settings

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 database that OSM uses should always be set to a time zone that does not use daylight savings time. See "Installing and Configuring the Oracle RAC Database" in *OSM Installation Guide* for more information.

The oms_timezone parameter, in the om_parameter table of the database, must also be set to a time zone that does not use daylight savings time. This setting must be defined in seconds offset to UTC (Coordinated Universal Time, formerly Greenwich Mean Time) for the time zone for your OSM database. For example, if you are setting this parameter for Eastern Standard Time, use the value **-18000** (60 seconds x 60 minutes = 3600 seconds–or one hour–x 5 for five hours offset from UTC/GMT). The oms_timezone parameter is used by the OSM database.

The OSM server uses the database_timezone_offset parameter in the **oms-config.xml** file. This must be set to be exactly the value used for the oms_timezone parameter in the om_parameter table.


Copyright © 2009, 2025, Oracle and/or its affiliates.

13

# Troubleshooting OSM

This chapter provides guidelines to help you troubleshoot problems with your Oracle Communications Order and Service Management (OSM) system.

## Information You Need for Troubleshooting

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.
- Oracle WebLogic Server administration server logs and output files.
- WebLogic Server managed server logs and output files.
- WebLogic Server node manager's logs and output files.
- JVM garbage collector logs.
- JVM heap dumps.
- JVM thread dumps (several in succession). See "Accessing Thread Dumps in OSM Cloud Native" for details.
- 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."
- Java Flight Recorder (JFR) recordings.

For OSM traditional deployments, OSM provides scripts for gathering this information automatically. See “Running Scripts to Automatically Gather Troubleshooting Information” for more information.

## Running Scripts to Automatically Gather Troubleshooting Information

Oracle provides scripts that help you gather the required data mentioned in the previous section automatically and quickly. This eliminates the need for gathering data manually.

The following sections provide information about the scripts and how the required data is gathered.

### About the Scripts

The ***OSM_SDK*/Samples/SRDataGathering** folder contains the following scripts:

- **getSRInfo.sh**
- **gatherWLSData.sh**
- **gatherDBData.sh**


Copyright © 2009, 2025, Oracle and/or its affiliates.

Along with these scripts, the OSM installer also places a **config.sh** file in the *OSM_SDK*/**Samples/SRDataGathering** folder. Update this file with accurate information before running the scripts. The instructions for updating the **config.sh** file are provided within the **config.sh** file itself.

You can copy the **config.sh** file to any location. If you do this, provide the path as an argument to the **getSRInfo.sh** script. You can also maintain and store multiple configuration files in the same location, for different systems (for production and testing) by renaming the **config.sh** file.

## Prerequisites for Running the Scripts

- The host from where you run the scripts must:
  - Support running shell scripts.
  - Have the **sshpass** utility pre-installed, if you are using password-based authentication to access the WebLogic host or the database host.
  - Be able to connect to WebLogic Server Hosts (where the admin server and the managed server are running) and the database hosts (where the database instances are running).
- Update the *OSM_SDK*/**Samples/SRDataGathering/config.sh** file appropriately as per the instructions provided within the file. The **config.sh** file must also be made available for the scripts.
- The user who runs the scripts should have permissions for accessing the WebLogic Server domain and the database servers.
- The usernames of the user on the WebLogic hosts must be the same as the usernames provided in the DOMAIN_USERNAME and FMW_USERNAME parameters in the **config.sh** file. If the usernames are different, then both the users must be part of the same operating system group.
- If you use password-based authentication, instead of ssh key-based authentication for logging into the WebLogic hosts, the ssh password should be the same for all hosts.
- The usernames of the user on the database hosts must be the same as the usernames provided in the DBHOST_USERNAME and DB_ADMIN parameters in the **config.sh** file. If the usernames are different, then both the users must be part of the same operating system group.
- If you use password-based authentication, instead of ssh key-based authentication for logging into database hosts, the ssh password should be the same for all hosts.
- Download the latest operating system information and the OSM database information collections scripts from My Oracle Support and update the path of the scripts in the **config.sh** file. Refer to the following KM articles on My Oracle Support:
  - SRDC - Collect OS/hardware specific static data related to the product (Doc ID 2496949.1)
  - SRDC - Collecting OSM Database Information Script (Doc ID 2165700.1)

## Running the Scripts

Before running the scripts, ensure that the **config.sh** file is updated with accurate information. The instructions for updating the **config.sh** file are provided within the **config.sh** file itself.

Run the scripts in the following order:

1. Run the **getSRInfo.sh** first.


Copyright © 2009, 2025, Oracle and/or its affiliates.

2. Run the following scripts parallelly or one after the other, as necessary:
   - **gatherWLSData.sh**
   - **gatherDBData.sh**

See the following sections for details about the scripts.

## getSRInfo.sh

Run this script to provide *SR-ID*, *BEGIN_TimeStamp* and *END_Timestamp* of the incident that you want to report.

This script does the following:

- Prompts you to specify *SR-ID*, *BEGIN_TimeStamp*, and *END_Timestamp* of the incident.
- Validates the **config.sh** file for errors.
- Upon successful validation, updates the **config.sh** file with the *Incident Timestamps* and *SR-ID*.
- Creates a directory with the SR-ID as the name, where all log files (both database and WebLogic server) will be stored.

If the WebLogic server and the database are on the same time zone, run this script once and then proceed with running the other scripts (**gatherWLSData.sh** and **gatherDBData.sh**).

If the WebLogic server and the database are on different time zones, run the **getSRInfo.sh** script twice separately; once before running **gatherWLSData.sh** and once again before running **gatherDBData.sh**, so that the respective time zones can be provided for gathering information.

## gatherWLSData.sh

This script takes the parameter values specified in the **config.sh** file and connects to the hosts where the WebLogic server is running, to collect the WebLogic server data.

If SSH key authentication is available, the script connects to the host directly and collects the required data.

If SSH key authentication is disabled, the script prompts for the SSH password for the username with which you are connecting to the WebLogic server host. This script also prompts for the password for the WebLogic server.

Once the required information is obtained from the updated **config.sh** file, which is generated after running the **getSRInfo.sh** script, the **gatherWLSData.sh** script collects the following data in the order listed:

- Managed server GC logs
- WebLogic server domain configuration
- OPatch information
- OSM configuration files
- Operating system information of the WebLogic server host
- Managed server logs
- OSM Compliance data


Copyright © 2009, 2025, Oracle and/or its affiliates.

> **Note**
>
> For collecting OSM compliance data, the managed server that is configured with Coherence MBean should be up before running the **gatherWLSData.sh** script. The details of the server for which this parameter is enabled can be found in the **$*DOMAIN_HOME*/config/config.xml** file.

After the successful collection, the data is copied to the specified directory individually. For example, after the GC logs are collected, they are copied to the specified directory before proceeding with collecting the Weblogic Domain Configuration data. This saves time by copying data as it becomes available so that triage can start before all data becomes available.

### gatherDBData.sh

This script connects to the host where the database is running and obtains the required values from the updated **config.sh** file, which is generated after running the **getSRInfo.sh** script, to collect the logs related to the OSM database.

If SSH Key authentication is available, then the script connects to the host directly and collects the required data. If SSH Key authentication is disabled, then the script prompts for the SSH password for the username with which you are connecting to the database host.

If the database host user (DBHOST_USERNAME in the **config.sh** file) does not have sufficient permissions for connecting to the database, the script considers the user specified for the DB_ADMIN variable in the **config.sh** file. This user must have sufficient permissions, such as permissions required for accessing $ORACLE_HOME.

Provide the password if the script considers the database administrator user to collect the required data. In addition, provide the OSM schema password when the script prompts.

Once the above information is obtained from the **config.sh** file, the script collects the following data in the order given below:

- AWR reports
- OPatch information
- Operating system information
- OSM database information

## 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?


Copyright © 2009, 2025, Oracle and/or its affiliates.

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 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 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 performance (for example, using AWR reports).
- 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.
- WebLogic server is paging JMS message bodies to disk. You can verify and confirm this by logging into the WebLogic Remote Console and checking the value of **Messages Paged Out Total** on individual JMS servers. If messages are being paged, check if JMS messages have been left to accumulate in error queues. Another option is to tune the message buffer size for a JMS server. By default, this is set to 512 megabytes for a production OSM system and can be increased (for example, to 1 gigabyte) if required.

## Error: "Java.lang.StackOverflowError" when Using Task Web Client

You may see the error "Java.lang.StackOverflowError" in the log files. If this happens, you can address the problem by tuning the thread stack size parameter.


Copyright © 2009, 2025, Oracle and/or its affiliates.

> **Note**
>
> The procedures below set the value to 2 MB. This is a suggested value to start with, but you should adjust the value if necessary, according to your needs.

In your instance, project or shape specification file, add or append the following parameter and adjust the value as necessary:

```
shape:
   user_mem_args: "-Xss2m"
```

## Coherence Configuration Error: [STUCK] ExecuteThread

The following thread error can occur in the OSM WebLogic server console when running an order:

```
[STUCK] ExecuteThread: '2' for queue: 'weblogic.kernel.Default (self-tuning)'" waiting
for lock java.util.concurrent.locks.ReentrantReadWriteLock$FairSync@5d7fc269 WAITING
```

The osm-invocation and osm-distributed thread-count values are set too low. See the discussion of configuring and monitoring coherence threads in *OSM Installation Guide* for more information about increasing these settings.

## Unexpected Logout from Web Client

If your system is running on a WebLogic Server cluster, and the following conditions apply:

- a user is viewing an order in the Order Management web client or Task web client
- that order is hosted on a managed server that fails or is shut down

the user will be logged out of the web client and will have to log back in. See the discussion of order affinity in *OSM Installation Guide* for more information about orders being hosted on a particular managed server.

## 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.

To resolve this issue, log in to the WebLogic Remote Console using the administrator account. Make sure you have been added to the group **OMS_client**. Try to log in again.

## 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?


Copyright © 2009, 2025, Oracle and/or its affiliates.

## Delayed JMS Messages

Clock synchronization issues may cause JMS request and response messages to remain in a queue in a delayed state.

When a message is sent, it is stamped with the time on the sender's machine. When the message arrives at the JMS destination, if the recipient's machine has a clock that is running several minutes slower, the timestamp is displayed as a future time and the WebLogic server decides to delay the message until the future time arrives.

To prevent this problem, use network time protocol (NTP) servers to synchronize clocks across all machines in a cluster.

## Error Message For Events From a JMS Topic in a Cluster

The following message can occur when attempting to process an event from a JMS topic:

```
<Message-Driven EJB: YourCartridgeName_1.0.0.0.0_YourPluginName_orderCompleteEventMDB's
transaction was rolled back. The transaction details are: ....
```

OSM does not support JMS topics within an OSM clustered environment. For more information about OSM queue configuration, see the discussion of OSM integration with external systems in *OSM Installation Guide*.

## JMS Message Delivery Failure

If OSM drops and does not process JMS messages, ensure that the messages are not using uniform distributed queue (UDQ) format. OSM supports only weighted distributed queues. For more information about OSM queue configuration, see the discussion of OSM integration with external systems in *OSM Installation Guide*.

## Unexpected Values for JMS Properties

There are some situations in which OSM may set the JMS properties on a message to values that you do not expect.

- Sometimes messages in the queue that were received from external systems will have JMS properties that were not set by the external system.

  This is because when OSM sends a request to an external system, any messages received in response must be correlated back to an appropriate automator. When a message is first received, before it is placed on the queue, OSM finds the handling task's context based on the correlation data in the message. OSM adds this context, and some additional data associated with the message-driven bean (MDB) that received the message, to the message using additional JMS message properties. You can see the context data by browsing the messages in the queue using the WebLogic Remote Console.

  When OSM runs in a cluster, this message is sometimes redirected to a message queue hosted on a different managed server from the one that sent the request.

- Sometimes messages have values that seem wrong. For example, a message might have JMS properties with pluginJndiName=X and cartridgeNamespace=Y, but plug in X is not in a cartridge with namespace Y.

  These property values are implementation details and will not necessarily match what you expect. For example, because different plug-ins can share the same MDB, the


Copyright © 2009, 2025, Oracle and/or its affiliates.

pluginJndiName property may not contain the name of the plug in that actually handles the message.

## 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 the following:

- From the WebLogic Remote Console, select **Edit Tree**, then **Servers** and then **Server**. From here, click on the **Protocols** tab, and then the **HTTP** sub-tab. 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.

## Problems When Running Multiple WebLogic Domains on One Host

If you are running multiple WebLogic domains on one host and you see errors such as web clients failing to load, JSP errors, or errors indicating that JSP pages can't be recompiled, you may not have set umask values properly to protect files in one WebLogic instance from other WebLogic instances.

## 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.

## Unable to Bring Up Managed Server After Database Failure

OSM fails to start if the Oracle database is unreachable. In addition, if the database is Oracle RAC, both database nodes specified by the data source URLs must be reachable. When WebLogic Server starts, failure of a database node is handled gracefully (processing fails over to the other node).

## 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.

## JBoss Cache Timeouts

Long full garbage collections can cause JBoss timeout errors to appear in the log files.


Copyright © 2009, 2025, Oracle and/or its affiliates.

## 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.

## Error: "No 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.

## DataDictionary Expansion Level

If you are having issues deploying cartridges, the cause may be related to the DataDictionary expansion level. In Oracle Communications Design Studio, under Windows preferences, increase the DataDictionary expansion level to 10. In some cases, you may need to increase the level to more than 10.

## 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:

```
<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>
```


Copyright © 2009, 2025, Oracle and/or its affiliates.

To resolve this issue:

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.

## Error: "exact fetch returns more than requested number of rows"

You may see the error "exact fetch returns more than requested number of rows" in the log files if there are memory issues relating to very large orders causing contention issues in orchestration XQuery calls when multiple orchestration plans are running at the same time. The default orchestration plan concurrency level is 3. You can reduce this value as described below.

To resolve this issue, decrease the orchestration plan concurrency level in your project specification file.

```
export JAVA_OPTIONS="${JAVA_OPTIONS} -
Doracle.communications.ordermanagement.orchestration.generation.model.Concurre
ncyLevel=2
```

## Error: "unique constraint violated"

You may see the "unique constraint violated" error in the log files if you retry to purge order data that you already tried to purge once but failed.

```
ORA-00001: unique constraint ...violated
ORA-06512: at "<database_schema>.OM_SQL_LOG_PKG", line 335
ORA-06512: at "<database_schema>.OM_PART_MAINTAIN", line 5012
ORA-06512: at "<database_schema>.OM_PART_MAINTAIN", line 5599
ORA-06512: at "<database_schema>.OM_PART_MAINTAIN", line 5778
ORA-06512: at "<database_schema>.OM_PART_MAINTAIN", line 6191
ORA-06512: at "<database_schema>.OM_PART_MAINTAIN", line 6360
ORA-06512: at "<database_schema>.OM_PART_MAINTAIN", line 6886
ORA-06512: at line 1
```

This error occurs because there are non-empty exchange tables that are created by the failed purge operation that you performed the first time.

To resolve this issue, you must purge the exchange tables manually before you retry purging.

## Exceptions When Purging is in Progress

When purging is in progress, you may encounter a number of order not found exceptions. This happens because when purging is in progress, there are some automated tasks in JMS queue. As a result, exceptions such as automation context not found and order not found occur.

To resolve this issue:

1. Log in to Weblogic Remote Console.
2. Navigate to **Monitoring Tree/Environment/Services/Messaging/JMS Runtime/JMS Servers/<Managed Server>/Destinations/<queue>**.


Copyright © 2009, 2025, Oracle and/or its affiliates.

3. Delete all the messages related to the orders that have been purged.

> **Note**
>
> Do not delete messages related to existing orders. To know which messages are related to existing orders, select * from om_order_header, where order_seq_id=x

4. Restart the OSM server.

## Error: "Ignoring partition"

You may encounter the following error:

"Ignoring partition <*partition_number*>: The number of OM_ORDER_HEADER subpartitions does not match the number of XCHG_OM_PRG_001$001$ partitions".

This error occurs when you try to purge partitions without setting up or resetting the exchange table in the upgraded or new schema.

To resolve this issue, drop the existing exchange tables and create new exchange tables.

## Unexpected Logout from Landing Page

If you are logged out of the Order Operations and Fallout Dashboard UX immediately after logging in, please verify if you have the required user entitlements to access OSM.

# Getting Help with OSM Problems

If you cannot resolve your problems with OSM, contact Oracle Technical Support.

## Before You Contact Support

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.
- 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.
- Have ready all specification files (project, instance and shape) used to create the OSM instance.

When you are ready, report the problem to Oracle.


Copyright © 2009, 2025, Oracle and/or its affiliates.

14

# OSM Log Messages

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 14-1 shows OSM Catalog messages.

**Table 14-1 OSM Catalog Messages**

| Error Code | Severity | Description |
|---|---|---|
| 600000 | error | **Message Body**<br>Order type/source is not found. The order type={1} /source={0} either does not exist, or is not available to the user.<br>**Message Details**<br>The order type={1}/ source={0} either does not exist, or is not available to the user.<br>**Method**<br>logSourceTypeNotFound(String source, String type) |
| 600001 | error | **Message Body**<br>Order is not found. The order with orderid={0} and orderHistId={1} does not exist, or is not available to the user.<br>**Message Details**<br>The order with orderid={0} and orderHistId={1} does not exist, or is not available to the user.<br>**Cause**<br>The orderHistId might not be up to date.<br>**Action**<br>Refresh server.<br>**Method**<br>logOrderNotFound(String orderId, String orderHistId) |
| 600002 | error | **Message Body**<br>Order template is not found. The order template does not exist, or is not available to the user.<br>**Message Details**<br>The order template does not exist, or is not available to the user.<br>**Method**<br>logOrderTemplateNotFound() |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 14-1 (Cont.) OSM Catalog Messages**

| Error Code | Severity | Description |
|---|---|---|
| 600003 | error | **Message Body**<br>Remark is not found. The given remark (orderid={0}, histid={1} remarkid={2}) does not match a remark in OMS.<br>**Message Details**<br>The given remark (orderid={0}, histid={1} remarkid={2}) does not match a remark in OMS.<br>**Cause**<br>Remark might have been deleted from the server or cannot be found in a specified location.<br>**Action**<br>Contact your local administrator.<br>**Method**<br>logRemarkNotFound(String orderId, String orderHistId, String remarkId) |
| 600004 | error | **Message Body**<br>Header for mnemonic path is not found. The header for mnemonic path={0} does not exist, or is not available to the user.<br>**Message Details**<br>The header for mnemonic path={0} does not exist, or is not available to the user.<br>**Method**<br>logHeaderNotFound(String mnemonicPath) |
| 600005 | error | **Message Body**<br>The format of the order data is not correct. The message details the error location. Invalid data = {0}<br>**Message Details**<br>Invalid data = {0}<br>**Method**<br>logOrderDataInvalid(String orderdata) |
| 600006 | error | **Message Body**<br>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}.<br>**Message Details**<br>Order (orderid={0} histid={1}) has not been accepted by user={2}<br>**Cause**<br>An attempt to update an order was made without accepting the order.<br>**Action**<br>You should accept the order first, then update the order.<br>**Method**<br>logOrderNotAcceptedByUser(String orderId, String orderHistId, String userid) |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 14-1 (Cont.) OSM Catalog Messages**

| Error Code | Severity | Description |
|---|---|---|
| 600007 | error | **Message Body**<br>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}<br>**Message Details**<br>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}<br>**Cause**<br>Data format error.<br>**Action**<br>Make sure all your data are in correct format and comply with their masks.<br>**Method**<br>logOrderUpdateFailed(String orderId, String histid, String data) |
| 600008 | error | **Message Body**<br>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}.<br>**Message Details**<br>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}<br>**Cause**<br>Not all mandatory fields are filled.<br>**Action**<br>Fill in data for all mandatory fields.<br>**Method**<br>logMandatoryCheckFailed(String orderID, String orderHistID, String num, String firstNodeID, String firstNodeType) |
| 600009 | error | **Message Body**<br>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.<br>**Message Details**<br>The order (orderid={0} histid={1}) cannot be transitioned to state={2}. Use ListStatesNStatuses.Request to get a list of valid states.<br>**Cause**<br>Cannot transition to the selected state.<br>**Action**<br>Use the ListStatesNStatuses XML API request to get a list of valid states. See *OSM Developer's Guide* for information.<br>**Method**<br>logTransistionInvalid(String orderid, String histid, String state) |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 14-1 (Cont.) OSM Catalog Messages**

| Error Code | Severity | Description |
|---|---|---|
| 600010 | error | **Message Body**<br>Unable to accept order. When retrieving an order for update, the order (orderid={0} histid={1}) cannot be accepted by the current user={2}<br>**Message Details**<br>When retrieving an order for update, the order (orderid={0} histid={1}) cannot be accepted by the current user={2}.<br>**Cause**<br>The orderid and histid are not up to date, or the order has been accepted by another user.<br>**Action**<br>Refresh the server to get new orderid and histid. If the order is currently accepted by another user, you cannot perform Accept operation.<br>**Method**<br>logUnableToAccept(long orderid, long histid, String userid) |
| 600011 | error | **Message Body**<br>User is not found. The order orderid={0} histid={1} cannot be assigned to userid={2}.<br>**Message Details**<br>The order orderid={0} histid={1} cannot be assigned to userid={2}.<br>**Cause**<br>User is not found.<br>**Action**<br>Try to assign the order to another user.<br>**Method**<br>logUserNotFound(String orderid, String histid, String userid) |
| 600012 | error | **Message Body**<br>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.<br>**Message Details**<br>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.<br>**Cause**<br>You are calling suspendOrder with an invalid user defined state.<br>**Action**<br>Supply a valid user defined state, or try to use other appropriate requests.<br>**Method**<br>logInvalidStateMnemonic(String orderid, String histid, String state) |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 14-1 (Cont.) OSM Catalog Messages**

| Error Code | Severity | Description |
|---|---|---|
| 600013 | error | **Message Body**<br>Invalid status mnemonic. The order (orderid={0} histid={1}) cannot be completed with the given status mnemonic (status={2}).<br>**Message Details**<br>The order (orderid={0} histid={1}) cannot be completed with the given status mnemonic (status={2}).<br>**Cause**<br>The status might not be valid for the current task.<br>**Action**<br>Supply a valid status.<br>**Method**<br>logInvalidStatusMnemonic(String orderid, String histid, String status) |
| 600014 | error | **Message Body**<br>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})<br>**Message Details**<br>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})<br>**Method**<br>logRemarkCannotBeModified(String orderid, String histid, String remarkid, String userid) |
| 600015 | error | **Message Body**<br>Request Unknown. The request type could not be identified. Type given={0}<br>**Message Details**<br>The request type could not be identified. Type given={0}<br>**Method**<br>logRequestUnkown(String type) |
| 600016 | error | **Message Body**<br>Request parameter error. A parameter for the request is missing or invalid. The message details the parameter in question Parameter = {0}, request type = {1}.<br>**Message Details**<br>A parameter for the request is missing or invalid. The message details the parameter in question Parameter = {0}, request type = {1}.<br>**Method**<br>logRequestParameterError(String parameter, String requestType) |
| 600017 | error | **Message Body**<br>Not authorized. The user={0} is not authorized to make the request={1}.<br>**Message Details**<br>The user={0} is not authorized to make the request={1}.<br>**Cause**<br>The user is not authorized to perform the operation.<br>**Method**<br>logNotAuthorized(String userid, String request) |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 14-1 (Cont.) OSM Catalog Messages**

| Error Code | Severity | Description |
|---|---|---|
| 600018 | error | **Message Body**<br>Database connection failed.<br>**Method**<br>logDatabaseConnectionFailed() |
| 600019 | error | **Message Body**<br>Security violation by user = {0}<br>**Cause**<br>User is not authorized to perform this operation.<br>**Method**<br>logSecurityViolation(String userid) |
| 600020 | error | **Message Body**<br>Naming exception was thrown while looking up JNDI name={0}.<br>**Cause**<br>JNDI name might not exist.<br>**Action**<br>Contact your local administrator.<br>**Method**<br>logNamingException(String name) |
| 600021 | error | **Message Body**<br>Remote exception thrown, while access object = {0}<br>**Message Details**<br>Remote exception was thrown while working with an EJB.<br>**Method**<br>logRemoteException(String name) |
| 600022 | error | **Message Body**<br>EJB Create exception thrown while creating object={0}.<br>**Message Details**<br>EJB Create exception thrown.<br>**Method**<br>logEJBCreateException(String name) |
| 600023 | error | **Message Body**<br>Unknown exception thrown, message={0}.<br>**Message Details**<br>Unknown exception thrown, message={0}.<br>**Method**<br>logUnknownException(String message) |
| 600024 | error | **Message Body**<br>Cannot deliver JMS message to queue = {0}<br>**Message Details**<br>Cannot deliver JMS message.<br>**Cause**<br>JMS Queue might be down.<br>**Method**<br>logCannotDeliverMessage(String arg) |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 14-1 (Cont.) OSM Catalog Messages**

| Error Code | Severity | Description |
|---|---|---|
| 600025 | error | **Message Body**<br>Invalid XML document, doc={0}.<br>**Message Details**<br>Invalid XML document, doc={0}.<br>**Cause**<br>XML document has syntax errors.<br>**Action**<br>Fix the syntax errors.<br>**Method**<br>logInvalidXMLDocument(String doc) |
| 600028 | debug | **Message Body**<br>SQL: {0}<br>**Message Details**<br>SQL statement execution<br>**Method**<br>logSQL(String sql) |
| 600029 | debug | **Message Body**<br>{0}<br>**Message Details**<br>Log of a request with all its parameters, based on a toString()<br>**Method**<br>logRequestWithParameters(com.nortel.oms.request.Request request) |
| 600030 | debug | **Message Body**<br>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}<br>**Message Details**<br>Node information was sent to the database<br>**Cause**<br>An order node has been created<br>**Action**<br>None<br>**Method**<br>logNodeCreate(String mnemonicPath, long nodeID, String nodeType, long nodeIndex, long parentIndex, double doubleValue, String value, String parentNew) |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 14-1 (Cont.) OSM Catalog Messages**

| Error Code | Severity | Description |
|---|---|---|
| 600031 | debug | **Message Body**<br>Deleting Order Node: node ID {0}, node type {1}, node index {2}, old double value {3}, old text value {4}<br>**Message Details**<br>Node information was sent to the database<br>**Cause**<br>An order node has been deleted<br>**Action**<br>None<br>**Method**<br>logNodeDelete(long nodeID, String nodeTypeCode, long nodeIndex, double oldDoubleValue, String oldValue) |
| 600032 | | **Message Body**<br>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}<br>**Message Details**<br>Node information was sent to the database<br>**Action**<br>None<br>**Method**<br>logNodeUpdate(String mnemonicPath, long nodeID, String nodeTypeCode, long nodeIndex, double oldDouble, String oldText, double newDouble, String newText) |
| 600035 | debug | **Message Body**<br>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}<br>**Message Details**<br>Node (name={0} nodeid={1} nodetype={2} nodeIndex={3} nodeDataType={4} mask={5}) does not comply with mask. Order ID={6} and user={7}<br>**Cause**<br>Data supplied do not comply with their masks.<br>**Action**<br>Must supply data with correct format.<br>**Method**<br>logOrderEditorDataValidationFailed(String name, String nodeId, String nodeType, String nodeIndex, String nodeDataType, String mask, String orderID, String user) |
| 600036 | debug | **Message Body**<br>Order editor - Create a node.<br>**Message Details**<br>Order editor: create a node (nodeId={0} nodeType={1} parentWebID={2})<br>**Method**<br>logOrderEditorCreateNode(String nodeId, String nodeType, String parentWebID) |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 14-1 (Cont.) OSM Catalog Messages**

| Error Code | Severity | Description |
|---|---|---|
| 600037 | debug | **Message Body**<br>Order editor - delete a node<br>**Message Details**<br>Order editor - delete a node (webID={0})<br>**Method**<br>logOrderEditorDeleteNode(String webID) |
| 600038 | debug | **Message Body**<br>Notification Engine - create message.<br>**Message Details**<br>Creation of event message<br>**Method**<br>logNotificationEngineCreateMsg(String arg) |
| 600039 | debug | **Message Body**<br>Could remove an event from the DB. Event Id = {0}<br>**Method**<br>logEventEngineRemoveEvent(long eventID) |
| 600055 | warning | **Message Body**<br>An exception occurred while removing the session for user {0}. Reason: {1}.<br>**Message Details**<br>While logging out user {0}, an exception was thrown when calling EJBObject.remove().<br>**Cause**<br>The server does not allow the session to be removed, or a communication error occurred.<br>**Method**<br>logEJBRemoveException(String username, Throwable th) |
| 600063 | error | **Message Body**<br>Error loading screen definitions from {0}. Reason: {1}.<br>**Message Details**<br>The screen definition file {0} could not be loaded. The screen definition file is required to construct the web pages.<br>**Cause**<br>The OMS application has not been deployed or built properly.<br>**Action**<br>Make sure that the screendefinitions.xml file is in the oms.ear file: oms.ear/oms.war/WEB-INF/conf/<br>**Method**<br>logCouldNotLoadScreenDefinitions(String url, Throwable th) |
| 600066 | warning | **Message Body**<br>Poller cannot locate listener {0}. Reason: {1}.<br>**Message Details**<br>**Cause**<br>Poller Servlet was deployed before listener {0}.<br>**Action**<br>None<br>**Method**<br>logGetPollerListener(String listener, Throwable th) |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 14-1 (Cont.) OSM Catalog Messages**

| Error Code | Severity | Description |
|---|---|---|
| 600069 | error | **Message Body**<br>Could not unsubscribe for {0} event.<br>**Message Details**<br>Event unsubscription for {0} event has failed.<br>**Cause**<br>JMS queue might be down.<br>**Action**<br>Use the WebLogic Server Console to verify JMS deployment.<br>**Method**<br>logEventTypeUnsubscribeException(String eventType) |
| 600071 | debug | **Message Body**<br>Filter value {1} does not have the proper format ({3}) for {0}.<br>**Message Details**<br>A filter could not be generated using the value {0}, formatter for {1} and mask {3}<br>**Cause**<br>User error<br>**Action**<br>Expected. You must retry.<br>**Method**<br>logFilterFormatError(String headerName, String filterValue, String mask) |
| 600072 | debug | **Message Body**<br>{3} invalid. Wildcards are not permitted for operation {1} for header {0}<br>**Message Details**<br>{3} invalid. Wildcards are not permitted for operation {1} for header {0}<br>**Cause**<br>User input. Expected error.<br>**Action**<br>You must retry.<br>**Method**<br>logFilterWildcardError(String headerName, String operationType, String value) |
| 600073 | error | **Message Body**<br>SSL is not enabled.<br>**Message Details**<br>SSL is not enabled.<br>**Cause**<br>SSL is not enabled.<br>**Action**<br>You must have SSL enabled for your server through WebLogic console.<br>**Method**<br>logSSLDisabled() |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 14-1 (Cont.) OSM Catalog Messages**

| Error Code | Severity | Description |
|---|---|---|
| 600076 | error | **Message Body**<br>Unable to connect to remote file system for accessing attachments.<br>**Message Details**<br>Unable to connect to T3 remote file system for accessing attachments.<br>**Cause**<br>Targeted server might be down.<br>**Action**<br>Contact your local administrator.<br>**Method**<br>logT3ConnectionException() |
| 600077 | error | **Message Body**<br>Unable to add attachment with id={0} in remote file system.<br>**Message Details**<br>Unable to add attachment with id={0} in remote file system using T3 file services.<br>**Cause**<br>IO Exception.<br>**Method**<br>logT3AddAttachmentException(String attachmentID) |
| 600078 | error | **Message Body**<br>Unable to delete attachment with id={0} in remote file system. Attachment with id={0} does not exist.<br>**Message Details**<br>Unable to delete attachment with id={0} in remote file system using T3 file services. Attachment with id={0} does not exist.<br>**Cause**<br>The attachment does not exist.<br>**Method**<br>logT3DeleteAttachmentNotFoundException(String attachmentID) |
| 600079 | error | **Message Body**<br>Unable to read attachment with id={0} in remote file system.<br>**Message Details**<br>Unable to read attachment with id={0} in remote file system using T3 file services.<br>**Method**<br>logT3ReadAttachmentException(String attachmentID) |
| 600086 | | **Message Body**<br>Retrieve PendingOrdersReport<br>**Message Details**<br>Retrieve PendingOrdersReport with orderTypesID={0}, orderSourceID={1}, summaryLevel={2}<br>**Method**<br>logPendingOrdersReport(String orderTypesID, String orderSourceID, String summaryLevel) |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 14-1 (Cont.) OSM Catalog Messages**

| Error Code | Severity | Description |
|---|---|---|
| 600088 | error | **Message Body**<br>Unable to add attachment with id={0} in remote file system. Attachment with id={0} already exists.<br>**Message Details**<br>Unable to add attachment with id={0} in remote file system using T3 file services. Attachment with id={0} already exists.<br>**Cause**<br>The attachment already exists.<br>**Action**<br>None<br>**Method**<br>logT3AddAttachmentAlreadyExists(String attachmentID) |
| 600089 | error | **Message Body**<br>Unable to add attachment with id={0} in remote file system. Attachment with id={0} exceeds maximum file size specified in the configuration file.<br>**Message Details**<br>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.<br>**Cause**<br>Your attachment size is too big.<br>**Action**<br>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.<br>**Method**<br>logT3AddAttachmentExceedMaxFileSize(String attachmentID) |
| 600090 | error | **Message Body**<br>Unable to read attachment with id={0} in remote file system. Attachment with id={0} does not exist.<br>**Message Details**<br>Unable to read attachment with id={0} in remote file system using T3 file services. Attachment with id={0} does not exist.<br>**Cause**<br>The attachment does not exist.<br>**Action**<br>None<br>**Method**<br>logT3ReadAttachmentNotFound(String attachmentID) |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 14-1 (Cont.) OSM Catalog Messages**

| Error Code | Severity | Description |
|---|---|---|
| 600091 | | **Message Body**<br>SSL port is not found.<br>**Message Details**<br>SSL port is not found.<br>**Cause**<br>Either server is not found, or there is no SSL port set up for the current server.<br>**Action**<br>Make sure SSL port was set up through WebLogic console.<br>**Method**<br>logSSLPortNotFound() |
| 600092 | error | **Message Body**<br>MBean home is not found.<br>**Message Details**<br>MBeanHome.ADMIN_JNDI_NAME cannot be found.<br>**Cause**<br>MBeanHome.ADMIN_JNDI_NAME cannot be found.<br>**Action**<br>None<br>**Method**<br>logMBeanHomeDisabled() |
| 600093 | error | **Message Body**<br>DataSource connection to database could not be found.<br>**Message Details**<br>The EJB requesting a connection to the database does not have a DataSource configured. This caused the data access object to fail.<br>**Cause**<br>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<br>**Action**<br>Determine if the database is available. Determine if EJB has correct deployment descriptor<br>**Method**<br>logDataSourceNotFound() |
| 600098 | error | **Message Body**<br>The current server name cannot be found<br>**Message Details**<br>The current server name cannot be found<br>**Cause**<br>MBean cannot find current run time server<br>**Action**<br>None<br>**Method**<br>logserverNameNotFound() |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 14-1 (Cont.) OSM Catalog Messages**

| Error Code | Severity | Description |
|---|---|---|
| 600099 | error | **Message Body**<br>Get max_read_only_field_length property from oms-config.xml has failed.<br>**Message Details**<br>Get max_read_only_field_length property from oms-config.xml has failed.<br>**Cause**<br>max_read_only_field_length entry does not exist in oms-config.xml.<br>**Action**<br>Make sure max_read_only_field_length entry exists in oms-config.xml.<br>**Method**<br>logGetMaxReadOnlyLengthError() |
| 600103 | error | **Message Body**<br>OMS is not enabled to send {0} event to the automated agent.<br>**Cause**<br>Either it is disabled in oms-config.xml file or the {1} key is not found in the file.<br>**Action**<br>Make sure it is {1} key in the oms-config.xml file is set to true.<br>**Method**<br>logDisabledJMSEvent(String event, String key) |
| 600104 | error | **Message Body**<br>Exception thrown while trying to enable or disable {0} event.<br>**Method**<br>logExceptionFindListenerForJMSEvent(String event) |
| 600105 | warning | **Message Body**<br>Cannot send messages. JMS connection is down.<br>**Cause**<br>JMS connection is down.<br>**Action**<br>Use the WebLogic Server Console to verify JMS deployment.<br>**Method**<br>logJMSTopicConnectionDown() |
| 600108 | error | **Message Body**<br>An OMS Exception has been thrown. Reason: {0}<br>**Method**<br>logOMSException(Throwable th) |
| 600109 | debug | **Message Body**<br>Lock successful<br>**Message Details**<br>Exclusive access to events table granted<br>**Method**<br>logEventTableLockSuccess() |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 14-1 (Cont.) OSM Catalog Messages**

| Error Code | Severity | Description |
|---|---|---|
| 600110 | debug | **Message Body**<br>Lock failed<br>**Message Details**<br>Access to events table denied<br>**Cause**<br>Other instance of Poller servlet holds the lock<br>**Action**<br>Wait for the current operation to end, and try again.<br>**Method**<br>logEventTableLockFailure() |
| 600111 | error | **Message Body**<br>Invalid value ({1}) is specified for ({0}) in configuration file. Defaulting to ({2}).<br>**Message Details**<br>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.<br>**Cause**<br>Value out of range - nonexisting path - invalid timezone id - incorrect boolean value<br>**Action**<br>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-conifg-defaults.xml file<br>**Method**<br>logInvalidValueForParameter(String property, String value, String default) |
| 600112 | error | **Message Body**<br>Invalid value ({1}) is specified for ({0}) in configuration file and no default value found for it in oms-config-defaults.xml file.<br>**Message Details**<br>Provided value for parameter is missing or is invalid. A default value is not found in oms-config-defaults.xml to default to.<br>**Cause**<br>oms-config-default.xml file cannot be accessed or it does not provide necessary information for the property in question<br>**Action**<br>verify oms-config.xml and oms-config-defaults.xml files. check the installation. check the spelling of parameters in oms-config.xml file<br>**Method**<br>logInvalidValueAndNoDefault(String property, String value) |
| 600114 | error | **Message Body**<br>Error reading configuration parameters XML file(URL: {0}) - {1} - {2}<br>**Message Details**<br>An error happened while accessing or reading the indicated configuration file.<br>**Cause**<br>passing error - a malformed URL F119G120E120- G120I/O exception<br>**Method**<br>logErrorLoadingConfigXMLFile(String url, String cause, String msg) |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 14-1 (Cont.) OSM Catalog Messages**

| Error Code | Severity | Description |
|---|---|---|
| 600115 | error | **Message Body**<br>In class {0}, an SQL Statement was not closed after use. It will now be closed.<br>**Cause**<br>AbstractProxy.close() was not called after using SQL connection<br>**Action**<br>Report problem to development<br>**Method**<br>logStatementNotClosed(String classname) |
| 600116 | error | **Message Body**<br>In class {0}, an SQL Connection was not closed after use. It will now be closed.<br>**Cause**<br>AbstractProxy.close() was not called after using SQL connection<br>**Action**<br>Report problem to development<br>**Method**<br>logConnectionNotClosed(String classname) |
| 600117 | warning | **Message Body**<br>The view node {0} of type {1} has an invalid default value of {2}. The value has been ignored.<br>**Cause**<br>The default value is not entered correctly, or is incorrect for the data type.<br>**Action**<br>Change default value.<br>**Method**<br>logInvalidDefaultOrderNodeValue(String mnemonicPath, String nodeType, String default) |
| 600118 | error | **Message Body**<br>Unable to locate resource bundle.<br>**Cause**<br>Resource does not exist.<br>**Action**<br>Make sure there exist a resource file named {0}_ {1}.properties with key(s) {2}.<br>**Method**<br>logI18NMissingResourceException(String baseName, String locale, String key) |
| 600119 | error | **Message Body**<br>Poller failed in init, message={0}.<br>**Method**<br>logPollerInitFailure(String msg) |
| 600120 | error | **Message Body**<br>Processing of timeout event failed, message={0}.<br>**Method**<br>logPollerProcessTimeoutFailed(String msg) |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 14-1 (Cont.) OSM Catalog Messages**

| Error Code | Severity | Description |
|---|---|---|
| 600121 | error | **Message Body**<br>Unexpected system exception processing XML request: {0}<br>**Cause**<br>Unexpected system exception processing XML request: {0}<br>**Action**<br>Report to development<br>**Method**<br>logXMLAPIProcessorError(Throwable th) |
| 600122 | debug | **Message Body**<br>XMLAPI Servlet, message={0}.<br>**Method**<br>logXMLAPIServletTrace(String msg) |
| 600123 | error | **Message Body**<br>Could not load XMLAPI Servlet request mappings. Servlet will be unavailable.<br>**Cause**<br>xml-request-mappings.xml file is missing.<br>**Action**<br>check web application WEB-INF/conf directory.<br>**Method**<br>logRequestMappingUnavailable(Exception reason) |
| 600127 | warning | **Message Body**<br>Error reading XML Document: {0}<br>**Method**<br>logErrorReadingXML(Exception cause) |
| 600128 | warning | **Message Body**<br>An error occurred processing XML request: {0} {1} : message {2}<br>**Method**<br>logXMLApplicationException(int code, String desc, String msg, Exception ex) |
| 600129 | warning | **Message Body**<br>SQL Exception {0} : {1}<br>**Method**<br>logSQLException(int errorCode, String message, Throwable th) |
| 600132 | error | **Message Body**<br>Unable to add attachment. Attachment exceeds maximum file size={0} specified in the configuration file.<br>**Message Details**<br>Unable to add attachment. Attachment exceeds maximum file size={0} specified in the configuration file.<br>**Method**<br>logAddAttachmentExceedMaxFileSize(String maxFileSize) |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 14-1 (Cont.) OSM Catalog Messages**

| Error Code | Severity | Description |
|---|---|---|
| 600133 | error | **Message Body**<br>Invalid oms configuration on attachments. Attachment file system must not be null or empty. Maximum attachment size must be bigger than zero.<br>**Message Details**<br>Invalid oms configuration on attachments. Attachment file system must not be null or empty. Maximum attachment size must be bigger than zero.<br>**Method**<br>logInvalidOmsConfigurationOnAttachments() |
| 600134 | error | **Message Body**<br>Missing T3 file service with name={0}.<br>**Message Details**<br>Missing T3 file service with name={0}.<br>**Cause**<br>Missing configuration of T3 file server name in WebLogic console.<br>**Action**<br>Contact your local administrator to set T3 file server.<br>**Method**<br>logMissingT3FileServiceName(String fileSystemName) |
| 600135 | | **Message Body**<br>T3 file service with name={0} must target exactly to one server.<br>**Message Details**<br>T3 file service with name={0} must target exactly to one server.<br>**Cause**<br>T3 file service was targeted to more than one server.<br>**Action**<br>Contact your local administrator to set up T3 file server properly.<br>**Method**<br>logDuplicateT3FileServiceTarget(String fileSystemName) |
| 600136 | error | **Message Body**<br>Invalid T3 file service path.<br>**Message Details**<br>Invalid T3 file service path.<br>**Cause**<br>Invalid T3 file service path.<br>**Action**<br>Make sure T3 file service path is valid.<br>**Method**<br>logInvalidT3FileServicePath() |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 14-1 (Cont.) OSM Catalog Messages**

| Error Code | Severity | Description |
|---|---|---|
| 600138 | error | **Message Body**<br>Targeted server = [{0}] for the T3 File Service with name = [{1}] is not running.<br>**Message Details**<br>Targeted server = [{0}] for the T3 File Service with name = [{1}] is not running.<br>**Cause**<br>Targeted server = [{0}] for the T3 File Service with name = [{1}] is not running.<br>**Action**<br>Make sure the targeted server is running.<br>**Method**<br>logTargetedserverForT3FileServiceNotRunning(String t3TargetserverName, String fileSystemName) |
| 600139 | debug | **Message Body**{0} |

# Automation Catalog Messages

Table 14-2 shows Automation Catalog messages.

**Table 14-2 Automation Catalog Messages**

| Error Code | Severity | Description |
|---|---|---|
| 700002 | debug | **Message Body**<br>parseXmlData(String xmlData)<br>**Method**<br>Parse the XML data {0} |
| 700004 | debug | **Message Body**<br>lookupOMSBean(String jndiName)<br>**Method**<br>Lookup the OMSSession Bean with jndiName={0} |
| 700007 | debug | **Message Body**<br>receiveTask(long orderId, long orderHistoryId, String taskMnemonic)<br>**Method**<br>Receive Task with orderId={0} orderHistoryId={1} and task mnemonic={2} |
| 700008 | debug | **Message Body**<br>completeTaskOnExit(long orderId, long orderHistId, String taskMnemonic,String comepletStat)<br>**Method**<br>Complete task On Exit with orderId={0}, orderHistoryId={1}, taskMnemonic={2} and Completion Status={3} |
| 700009 | debug | **Message Body**<br>assignTask(long orderId,long orderHistoryId, String taskMnemonic, String userId)<br>**Method**<br>Assign Task to {3} with orderId={0} orderHistId ={1} and TaskMnemonic ={2} |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 14-2 (Cont.) Automation Catalog Messages**

| Error Code | Severity | Description |
|---|---|---|
| 700010 | debug | **Message Body**<br>suspendTask(long orderId,long orderHistoryId, String taskMnemonic, String suspendState)<br>**Method**<br>SuspendTask to state={3} with OrderID={0}, OrderHistId={1} and Task Mnemonic={2} |
| 700011 | debug | **Message Body**<br>acceptTask(long orderId, long orderHistoryId, String taskMnemonic)<br>**Method**<br>Accept Task with OrderId={0} orderHistryId={1} and taskMnemonic={2} |
| 700012 | debug | **Message Body**<br>updateOrderData1(long orderId,long orderHistId,String TaskMnem, String updateData)<br>**Method**<br>UpdateOrderData with orderId={0} orderHistoryId={1}, TaskMnemonic={2} and XML Data={3} |
| 700014 | debug | **Message Body**<br>The orderData id={0} and Data:{1}<br>**Method**<br>printOrderDataAsXML(long OrderId, String OrderData) |
| 700016 | debug | **Message Body**<br>Lookup the Automator: type={0} and jndiName={1}<br>**Method**<br>lookupAutomator(String type, String jndiName) |
| 700019 | debug | **Message Body**<br>Get a JMS Message with EventType={0}, Mnemonic={1}<br>**Method**<br>getAMessage(String type, String mnemonic) |
| 700020 | debug | **Message Body**<br>Set the outMessage Correlation to={0} the correlationId={1}<br>**Method**<br>setCorrelationId(String correlation, String correlationId) |
| 700021 | debug | **Message Body**<br>Send Notification Email message to :{0}, subject:{1}, MessageBody{2}<br>**Method**<br>sendEmailNotification(String to, String subject, String msgBody) |
| 700023 | error | **Message Body**<br>An automation exception has occurred At {0}, the reason is :{1}<br>**Cause**<br>See message body.<br>**Method**<br>AutomationException(String msg, Throwable th) |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table 14-2 (Cont.) Automation Catalog Messages**

| Error Code | Severity | Description |
|---|---|---|
| 700024 | error | **Message Body**<br>Naming Exception has occurred at {0} reason {1}<br>**Cause**<br>Cannot find user plug-in from JNDI tree.<br>**Method**<br>namingException(String name, Throwable ex) |
| 700033 | error | **Message Body**<br>{0} {1}<br>**Cause**<br>See message body for details.<br>**Method**<br>logAutomationException(String desc, Throwable exception) |


Copyright © 2009, 2025, Oracle and/or its affiliates.

15

# 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 within the OSM SDK artifact, which can be downloaded as a .zip file. For more information about OSM SDK, see *OSM Modeling 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 model and 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 running 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 **SDK/XMLImportExport/build.properties** file. See "Configuring the build.properties File for Ant Commands."


Copyright © 2009, 2025, Oracle and/or its affiliates.

- For batch scripts, configure the **SDK/XMLImportExport/config.bat** script. See "Configuring the config.bat Script for Batch Scripts."

2. Copy the **SDK/XMLImportExport/config/config_sample.xml** file and rename it to **SDK/XMLImportExport/config/config.xml** in the same directory.

> **Note**
>
> The **SDK/XMLImportExport/config/config_sample.xml** file is 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 **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 an **.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.

# Deploying Cartridges to the OSM Database Using XMLIE

You can deploy cartridges to the OSM DB directly using XMLIE while OSM is still running or shut down. This approach is strongly recommended for controlled environments such as


Copyright © 2009, 2025, Oracle and/or its affiliates.

production, pre-production or UAT, and for environments managed using a CI/CD pipeline. It is also the preferred approach for semi-formal environments, such as test systems.

You can perform certain cartridge deployment actions on your OSM running instance while orders from a cartridge that you deployed earlier are still being processed.

For instructions and more information about deploying cartridges to the OSM DB directly using XMLIE in offline and online modes, see "Deploying Cartridges to the OSM Database Using XMLIE" in *OSM Modeling Guide*.

For instructions about deploying cartridges using Design Studio, see *Design Studio Modeling OSM Processes*.

# 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 exportAll**: exports all cartridges.
- **ant exportNamespace**: exports all cartridges that match the given namespace.
- **ant exportCartridge**: exports one specific cartridge.
- **ant import**: imports the XML model.
- **ant validate**: validates the XML document.
- **ant migrate**: migrates orders from one cartridge version to another version.
- **ant migrateadvanced**: migrates orders by source and type.
- **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 the pending orders.
- **ant fastUndeploy**: fast undeploys cartridge if no dependent orders exist.
- **ant forceFastUndeploy**: fast undeploys cartridge and dependent orders.
- **ant immediateOrderPurge**: purges order immediately.
- **ant scheduleOrderPurge**: schedules an order purge.
- **ant removeOrderPurge**: removes a scheduled order purge job.
- **ant listOrderPurges**: lists all deployed cartridge statuses and deployment states.
- **ant refresh**: refreshes server metadata.
- **ant convert**: upgrades the old XML model to the 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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

- **ant htmlModel**: converts an XML metadata document to HTML and graphical format.
- **ant credStoreAdmin**: configures credential store in Weblogic.
- **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 use the following scripts to encrypt passwords:

- **EncryptPasswords.sh**: This encrypts passwords in the **config.xml** file. For more information, see "Using the EncryptPasswords Utility."
- **CreateEncryptPasswords.sh**: This encrypts passwords for use in the userAdmin and credStoreAdmin credential store commands. For more information, see "Using the CreateEncryptPasswords Utility."

# About XML Import/Export Batch Scripts and Syntax

OSM supports the following batch scripts for Windows systems in the *OSM_SDK*/**SDK**/**XMLImportExport** folder:

- **listCartridges.bat**: Displays a list of all deployed cartridge names, versions, in default and the deployment state.
- **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.
- **validate.bat**: Validates an XML model document.
- **modeldoc.bat**: Converts an XML metadata document to HTML and graphical format.
- **EncryptPasswords.bat**: Encrypts passwords contained in the **config.xml** file. For more information, see "Using the EncryptPasswords Utility."
- **CreateEncryptPasswords.bat**: Encrypts passwords for use in the userAdmin and credStoreAdmin credential store commands. For more information, see "Using the CreateEncryptPasswords Utility."
- **convertmodel.bat**: Migrates a model from older versions and uses the following syntax:

  ```
  convertmodel.bat xmlModelFile
  ```

  > **Note**
  >
  > This script is deprecated. Use the **import.bat** script instead because it automatically upgrades models during an import.


Copyright © 2009, 2025, Oracle and/or its affiliates.

- **userAdmin.bat**: Adds users to WebLogic from WebLogic groups and OSM workgroups based on an XML definition.

  For more information about the **userAdmin.bat** script, see "Using the XML Import/Export Application to Administer Users and Workgroups."

> **Note**
>
> Passing an unencrypted password as a command line argument to the XML Import/Export tool scripts is no longer possible. The **-p** *db_password* and **-clientpassword** *xmlAPI_password* arguments, which were previously deprecated, have now been removed. You must either use encrypted passwords in the config.xml file (using the EncryptPassword utility) or interactively provide the unencrypted password when prompted. For security recommendations, see "Configuring the config.xml File XML Import/Export Nodes and Elements."

Before you can run these scripts, you must configure the environment **config.bat** script 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."

# 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 **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 **SDK/XMLImportExport/build.properties** file.
2. Update the following variables:

```
middlewareHome=MW_home
java.maxmemory=java_max_memory
xmlie.root.dir=root_dir
xmlie.root.modelDocument=model_document
xmlie.root.configDocument=config_document
xmlie.root.namespace=namespace
xmlie.root.version=version
xmlie.root.htmlDir=html_dir
```

   where:

   - *MW_home* is the location where Oracle Fusion Middleware was installed
   - *java_max_memory* is the maximum heap to be used by JVM
   - *root.dir* is the path of XMLIE directory
   - *model_document* is the path for XML model document
   - *config_document* is the path for **config.xml**


Copyright © 2009, 2025, Oracle and/or its affiliates.

- *namespace* is the OSM cartridge namespace
- *version* is the cartridge version
- *html_dir* is the path for HTML model directory

> **Note**
>
> You can also refer to the **README.txt** file found under **SDK/XMLImportExport** for information on configuring the **build.properties** file for Ant scripts.

For example:

```
middlewareHome=C:/oracle/middleware
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. If you want 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. If you want to configure cartridge migration parameters in the **build.properties** file, see "Configuring and Running an Order Migration."

## Configuring the config.bat Script for Batch Scripts

To configure the batch script environment, go to the **SDK/XMLImportExport** directory and configure the user-configurable variables (for example, JAVA_OPTS) in the **config.bat** file.

# Configuring the config.xml File XML Import/Export Nodes and Elements

To configure the **config.xml** file:

1. Open the **SDK/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.

```
<databaseConnection>
   <user>osm_schema_username</user>
   <password>osm_schema_password</password>
   <dataSource>jdbc:oracle:thin:@ip_address:osm_db_port:osm_db_sid</dataSource>
</databaseConnection>
```


Copyright © 2009, 2025, Oracle and/or its affiliates.

where:

- *osm_schema_username* and *osm_schema_password* are the OSM schema user name and password.
- *ip_address*, *osm_db_port*, and *osm_db_sid* are the OSM database IP address, port number, and SID.

> **Note**
>
> To export or import the symbols, change the character set and database connection to something like the following sample:
>
> ```
> <encoding>UTF-8</encoding>
> <dataSource> jdbc:oracle:oci:(description=(address=(host=host1.example.com)
> (protocol=tcp)(port=1521)) (connect_data=(SID=ORASID)))
> </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.

```
<xmlAPIConnection>
   <user>weblogic_username</user>
   <password>weblogic_password</password>
   <url>http://ip_address:port</url>
</xmlAPIConnection>
```

where

- *weblogic_username* and *weblogic_password* 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:

```
<j2eeAdminConnection>
   <j2eeServiceName>weblogic</j2eeServiceName>
   <user>weblogic_username</user>
   <password>weblogic_password</password>
   <protocol>protocol</protocol>
   <hostname>ip_address</hostname>
   <port>port</port>
</j2eeAdminConnection>
```

where

- *weblogic_username* and *weblogic_password* are the user name and password of the WebLogic with Administrator privileges.
- *protocol* is the protocol to use to connect to the WebLogic server. If not included, the t3 protocol will be used. If you would like to connect to the WebLogic server using SSL, set this value to **t3s**.
- *ip_address*, and *port* are the WebLogic Administration server IP address and port number.


Copyright © 2009, 2025, Oracle and/or its affiliates.

6. Configure the XMLIE log file location node:

   ```
   <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 exporting metadata from an OSM database, configure the export node. See "About Exporting Metadata."
   - For importing metadata to the OSM database, configure the import node. See "About Importing and 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. For more information, see "Using the EncryptPasswords Utility."

> **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.

# Using SSL Connections

You can use a secure connection in XMLIE. To do this, configure the **config.xml** file for a secure connection (by setting the **j2eeAdminConnection** element in the **config.xml** file to use the "t3s" protocol), and then make the certificate available to the XMLIE scripts.


Copyright © 2009, 2025, Oracle and/or its affiliates.

To configure the **config.xml** file for a secure connection, see step 5 in "Configuring the config.xml File XML Import/Export Nodes and Elements."

To make the certificate available to the XMLIE scripts on UNIX and Linux, run the following command, located in the *OSM_SDK*/**SDK/XMLImportExport** directory:

```
ImportCertificate.sh server port
```

where *server* is the name of the host that the WebLogic administration server is running on and *port* is the SSL port for that Weblogic administration server.

To make the certificate available to the XMLIE scripts on Windows:

1. Do one of the following to get a copy of the certificate:
   - Download and use a tool like OpenSSL
   - Connect to the SSL port for the administration server in your browser. You can connect by accessing either the WebLogic Remote Console or one of the OSM web clients. Once you have connected, the certificate will be stored in your browser. Consult the documentation for your browser for information about how to export a certificate. Export the certificate in Base-64-encoded X.509 format (sometimes known as X.509 PEM format).
2. Import the certificate into your Java keystore by doing the following:
   a. Set your JAVA_HOME environment variable to the location of the instance of Java that you will be using for the credStoreAdmin or userAdmin command.
   b. Run the **keytool** command (located in **%JAVA_HOME%/bin**) with the following syntax:

      ```
      keytool -import -trustcacerts -keystore "%JAVA_HOME%\jre\lib\security\cacerts" -
      alias osmalias -file filename
      ```

      where:

      *osmalias* is an alias for the certificate that is unique within the instance of Java

      *filename* is the path and name of the file to which you exported the certificate.

      For example:

      ```
      keytool -import -trustcacerts -keystore "%JAVA_HOME%\jre\lib\security\cacerts" -
      alias osmserver -file C:\OSMSec\osmdomainCert.crt
      ```

      You will be prompted for the password for your keystore.

# 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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

> **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 **ant export** 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 15-1, the definition of entities (for example, each state) is now sorted alphabetically by name:

**Example 15-1 Definition of Entities (each state) Sorted Alphabetically by Name**

```
<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>
```


Copyright © 2009, 2025, Oracle and/or its affiliates.

```
</state>
...
```

In Example 15-2, the references to entities (for example, each state) is now sorted alphabetically by name:

**Example 15-2 References to Entities (states or statuses) Sorted Alphabetically by Name:**

```
<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 15-3, 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 15-3 Not Sorted By Name**

```
<masterOrderTemplate>
<dataNode element="account_information">
<dataNode element="first_name"/>
<dataNode element="last_name"/>
<dataNode element="country">
<viewRule xsi:type="eventRuleType">
<event>value-changed</event>
<action>refresh</action>
</viewRule>
</dataNode>
```

## About Export Layout Options

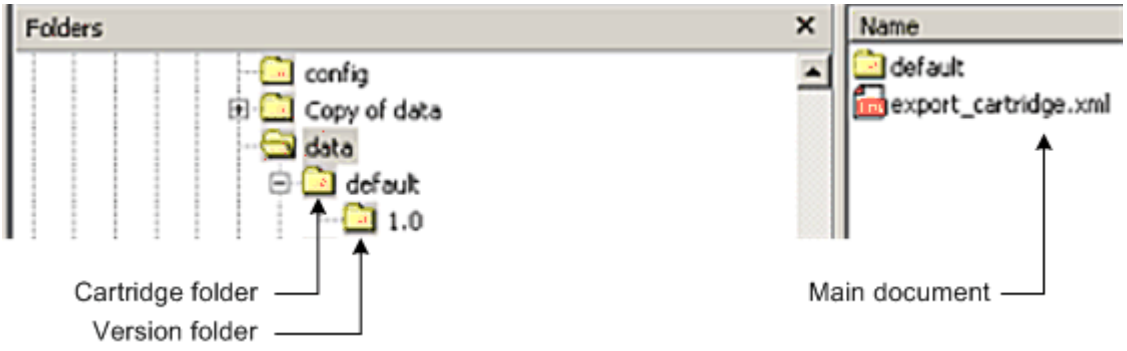
When you perform a selective export, you can specify different export layouts by using the export command to create one or multiple files, and where to put the files. You have the following options:

- 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 15-1 shows the **cartridge** layout.


Copyright © 2009, 2025, Oracle and/or its affiliates.

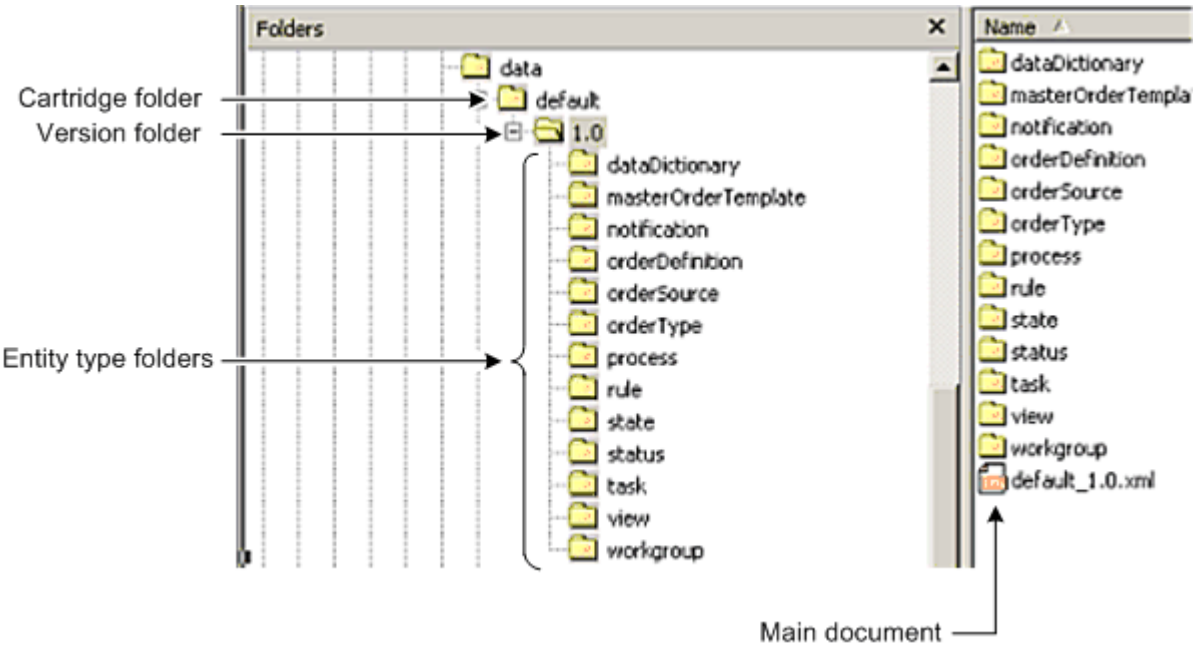
**Figure 15-1 Cartridge Layout**




- 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 15-2 shows the **entity** layout.

**Figure 15-2 Entity Layout**





Copyright © 2009, 2025, Oracle and/or its affiliates.

# 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 XMLIE, to false, thereby reducing the model document size.

> **Note**
>
> This is a function of the export behavior.

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):

```
 /*$entityType*/
```

**Example 15-4 Original Rule**

```
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 &lt;= (sysdate - 2/24)) then
:rule_result := 'true';
else
:rule_result := 'false';
end if;
end;
```

**Example 15-5 Modified Rule**

The import operation detects IDs and replaces them with new IDs.


Copyright © 2009, 2025, Oracle and/or its affiliates.

```
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 &lt;= (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 **ant export** command.

   ```
   <export validateModel="validation" exposeEntityID="entityID"
   layout="layout_action" readOnlyFileAction="readOnly_action"
   exportOriginalModel="true">
   </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.
   - *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.
   - *exportOriginalModel* : Options are


Copyright © 2009, 2025, Oracle and/or its affiliates.

  - **true**: The original imported PAR or Model XML file will be exported and all other export attributes and selection settings except for the **readOnlyFileAction** will be ignored. The exported PAR or XML file's name format is **$CartridgeName-$CartridgeVersion.[par/xml]**. It is stored in the column **resource_location** of the table **OM_MODEL_CLOB**.

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:

```
    <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 are using Ant, run the following command:

      ```
      ant export
      ```

   b. If you are 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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

> **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:

   ```
   <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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

- **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:

```
<import validateModel="validation"
nonEmptyDatabaseAction="database_action"entityConflictAction="entity_action">
     <selection>/oms:model/oms:cartridge[@namespace="namespace" and
@version="version"]/oms:entity</selection>
     <selection>/oms:model/oms:system_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 using Ant, run the following command:

      ```
      ant import
      ```

   b. If you using a batch script, run the following command:

      ```
      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 **SDK\XMLImportExport\config\config.xml** file, add a selection element to an export node that targets the workgroupDefinition entity.

   For example:

   ```
   <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 **SDK\XMLImportExport** folder, create an XML document to store the XML workgroup information from the cartridge you are targeting. For example: **workgroupDefinition.xml**.


Copyright © 2009, 2025, Oracle and/or its affiliates.

4. Run the **SDK\XMLImportExport\export.bat** script:

   For example:

   ```
   export.bat C:\osminstall\SDK\XMLImportExport\worgrkoupDefinition.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'?>
   <model xmlns="http://www.metasolv.com/OMS/OrderModel/2002/06/25"
    xmlns:osm="http://xmlns.oracle.com/communications/ordermanagement/model"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.metasolv.com/OMS/OrderModel/2002/06/25
    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 **SDK\XMLImportExport\import.bat** script:

   For example:

   ```
   import.bat C:\osminstall\SDK\XMLImportExport\worgrkoupDefinition.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 **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>
> ```


Copyright © 2009, 2025, Oracle and/or its affiliates.

# 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 **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:

   ```
   <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 **SDK\XMLImportExport** folder, create an XML document to store the XML workgroup information from the cartridge you are targeting. For example: **workgroup.xml**.
4. Run the **SDK\XMLImportExport\export.bat** script:

   For example:

   ```
   export.bat C:\osminstall\SDK\XMLImportExport\worgrkoup.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 version = '1.0' encoding = 'ISO-8859-2'?>
   <model xmlns="http://www.metasolv.com/OMS/OrderModel/2002/06/25"
    xmlns:osm="http://xmlns.oracle.com/communications/ordermanagement/model"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.metasolv.com/OMS/OrderModel/2002/06/25
    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>
   ```


Copyright © 2009, 2025, Oracle and/or its affiliates.

```
        <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 **SDK\XMLImportExport\import.bat** script:

   For example:

   ```
   import.bat C:\osminstall\SDK\XMLImportExport\worgrkoup.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 **SDK\XMLImportExport\config\config.xml** file that targets the workgroup entity in the model file.
>
> For example:
>
> ```
> <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**, **ant undeploy**, and **ant 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 **ant purge** command, you can remove all the data. The **purgeOrder.bat** script and **ant purge** command do not affect metadata.


Copyright © 2009, 2025, Oracle and/or its affiliates.

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 running the **purge** command or an exception is thrown.
>
> The **purge.bat** script and **ant purge** 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 **ant purge** 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. Consult "Online vs. Offline Maintenance" to ensure that the system is in the appropriate state (online or offline) for the operation you intend to perform.
2. 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:


Copyright © 2009, 2025, Oracle and/or its affiliates.

```
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
  ```

3. If you used the fast undeploy option and OSM was online, refresh the metadata for OSM. See "Refreshing OSM Metadata" for more information.

To purge the entire schema (metadata and orders) or undeploy a specific cartridge using batch scripts:

1. Consult "Online vs. Offline Maintenance" to ensure that the system is in the appropriate state (online or offline) for the operation you intend to perform.
2. 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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

- 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.

3. If you used the fast undeploy option and OSM was online, refresh the metadata for OSM. See "Refreshing OSM Metadata" for more information.

# 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 "Dropping Partitions (Offline Only)" 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
- 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 syntax 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=order_id_upper" "orderIdGreaterOrEq=order_id_lower"
   "parallelism=op_number"
   ```


Copyright © 2009, 2025, Oracle and/or its affiliates.

> **Note**
>
> All date parameters must be specified in the format:
>
> **yyyy-mm-ddThh24:mi:ss Z**
>
> 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**
  - **aborted**
  - **in_progress**
  - **waiting_for_revision**
  - **waiting**
  - **failed**

  For example: `"not_started,completed"`


Copyright © 2009, 2025, Oracle and/or its affiliates.

- *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.
- *order_id_upper*: The order ID number used with the **orderIdLessThan** parameter which specifies the exclusive upper order ID bound to purge.
- *order_id_lower*: The order ID number used with the **orderIdGreaterOrEq** parameter which specifies the inclusive lower order ID bound to purge.
- *op_number*: The number of purge operations to run in the database in parallel. The value can be a power of two of the set 1, 2, 4, 8, 32, or 64.

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=10"
"orderIdGreaterOrEq=50" "parallelism=64"
```

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=***order_id_upper***" "orderIdGreaterOrEq=***order_id_lower***"**
**"parallelism=***op_number***" "stop_date=***stop_date***" "start_date=***start_date***"**

Where the parameters match the ones in step 1, with the addition of the following:

- *stop_date*: The time when the purge should stop, even if all orders satisfying the purge criteria have been purged (for example, 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 "start_date=2007-01-01T23:59:59 EST"
```


Copyright © 2009, 2025, Oracle and/or its affiliates.

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 **SDK/XMLImportExport/build.properties** file.
2. If you want to perform an immediate purge of some or all orders before a certain date using the **immediateOrderPurge** attribute for the **ant purge** command, edit the following arguments:

   ```
   xmlie.orderPurge.purgeBefore=before_purge
   xmlie.orderPurge.orderStates=order_states
   xmlie.orderPurge.namespace=namespace
   xmlie.orderPurge.version=version
   xmlie.orderPurge.orderType=order_type
   xmlie.orderPurge.orderSource=order_source
   xmlie.orderPurge.commitCount=commitCount
   xmlie.orderPurge.orderIdLessThan=order_id_upper
   xmlie.orderPurge.orderIdGreaterOrEq=order_id_lower
   xmlie.orderPurge.parallelism=op_number
   ```

   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**


Copyright © 2009, 2025, Oracle and/or its affiliates.

  - **running**
  - **not_started**
  - **suspended**
  - **cancelled**
  - **compensating**
  - **amending**
  - **cancelling**
  - **closed**
  - **completed**
  - **aborted**
  - **in_progress**
  - **waiting_for_revision**
  - **waiting**
  - **failed**

  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.
- *order_id_upper*: The order ID number used with the **orderIdLessThan** parameter which specifies the exclusive upper order ID bound to purge.
- *order_id_lower*: The order ID number used with the **orderIdGreaterOrEq** parameter which specifies the inclusive lower order ID bound to purge.
- *op_number*: The number of purge operations to run in the database in parallel. The value can be a power of two of the set 1, 2, 4, 8, 32, or 64.

For example:

```
xmlie.orderPurge.purgeBefore=2006-06-30T23:59:59 EST
xmlie.orderPurge.orderStates=COMPLETED
xmlie.orderPurge.namespace=default
xmlie.orderPurge.version=1.0.0.0.0
xmlie.orderPurge.orderType=ot
xmlie.orderPurge.orderSource=os
xmlie.orderPurge.commitCount=10
xmlie.orderPurge.orderIdLessThan=10
xmlie.orderPurge.orderIdGreaterOrEq=50
xmlie.orderPurge.parallelism=64
```


Copyright © 2009, 2025, Oracle and/or its affiliates.

3. If you want to schedule a purge of some or all orders on a certain date using the **scheduleOrderPurge** attribute for the **ant purge** command, edit the following additional arguments:

   **xmlie.orderPurge.startDate=***start_date*
   **xmlie.orderPurge.stopDate=***stop_date*

   Where:

   - *stop_date*: The time when the purge should stop, even if all orders satisfying the purge criteria have been purged (for example, 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:

   ```
   xmlie.orderPurge.purgeBefore=2006-06-30T23:59:59 EST
   xmlie.orderPurge.orderStates=COMPLETED
   xmlie.orderPurge.namespace=default
   xmlie.orderPurge.version=1.0.0.0.0
   xmlie.orderPurge.orderType=ot
   xmlie.orderPurge.orderSource=os
   xmlie.orderPurge.commitCount=10
   xmlie.orderPurge.orderIdLessThan=10
   xmlie.orderPurge.orderIdGreaterOrEq=50
   xmlie.orderPurge.parallelism=64
   xmlie.orderPurge.startDate=2007-01-01T00:01:01 EST
   xmlie.orderPurge.stopDate=2007-12-31T23:59:59 EST
   ```

4. If you want to remove an scheduled order purge that has not started using the **removeOrderPurge** attribute for the **ant purge** command, edit the following arguments:

   **xmlie.orderPurge.jobId=***job_id*

   where *job_id* is the job ID of the scheduled purge.

   For example:

   ```
   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:

   **ant immediateOrderPurge**

6. Use the following syntax to run a scheduled order purge:

   **ant scheduleOrderPurge**

7. Use the following syntax to list all scheduled order purges that have not started:

   **ant listOrderPurges**


Copyright © 2009, 2025, Oracle and/or its affiliates.

8. Use the following syntax to remove an order purge that has not started:

```
ant removeOrderPurge
```

# About Migrating Orders

XMLIE provides the **ant migrate** 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:


Copyright © 2009, 2025, Oracle and/or its affiliates.

- 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 **migrate** and **userInteraction** elements.

   ```
   <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:

   ```
   migrate.bat config/config.xml -sourcenamespace namespace -sourceversion version
   sourceordertype type -sourceordersource source -targetversion version
   ```

   where:


Copyright © 2009, 2025, Oracle and/or its affiliates.

- *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 migrate** command, do the following:
   a. Open the **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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

- **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.

```
<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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

# 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 **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 **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 %JAVAOPTS% -classpath %CLASSPATH% %APP_PROPERTIES%
   com.mslv.oms.metadatahandler.operation.ModelDocOperator %XML_MODEL% %GRAPHVIZ% ./
   modeldoc
   ```

   > **Note**
   >
   > The **modeldoc.bat** script, due to external limitations, is not case sensitive. Because of this, it does not work if two entities' documents go into the same directory and their names are differentiated only by capitalization, for example two tasks in the same process with names like **IsDebugOn** and **isDebugon**.
   >
   > This limitation is only for the **modeldoc.bat** script, not for the **ant htmlModel** command, used on UNIX and Linux systems.

4. If you are using the **ant htmlModel** command, configure the **graphiz** property in the **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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

A

# OSM Credential Store API Command Reference

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 Oracle Fusion Middleware Credential Store Framework (CSF).

The OSM credential store APIs and credential store-related classes are listed in Table A-1:

**Table A-1 Credential Store API Commands and Classes**

| Command or Class | Description |
|---|---|
| userAdmin Command | This command adds an OSM user to a WebLogic Server security realm. If the user does not already exist in the credential store, this script will add it with default values. |
| credStoreAdmin Command | This command to configures the Java Platform Security policy for the credential store and manages credentials in the credential store. |
| CredStore | This is the credential store object, which is the domain credential store class and contains a single instance of the CredentialStore object. |
| PasswordCredStore | This is the password credential store object. |
| CredStoreException | This is the credential store exception object. |
| SoapAdapter | The attributes in this class provide the attributes for the credential store when you define SOAP data provider instances in your cartridges. |
| ObjectelHTTPAdapter | The attributes in this class provide the attributes for the credential store when you define Objectel HTTP data provider instances in your cartridges. |
| ViewRuleContext | This interface object provides operations for the credential store. |
| AutomationContext | This interface object provides operations for retrieving information from the credential store in automations. |

## OSM User Security and Credential Store Commands

To develop OSM cartridges to use the credential store offered through CSF (see "Using 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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

# userAdmin Command

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

You can do the following with the userAdmin command:

- add users to WebLogic Server security realms
- add users to WebLogic Server groups
- add users to OSM workgroups (roles)

You should add users to a security realm *after* they have been added to the credential store. For more information about adding users to the credential store, see "Managing Credentials in the Credential Store." However, if you do add a user using userAdmin command and the user has not already been added to the credential store, you can configure the userAdmin command also to add the user to the credential store with the keyname in the format **osmUser**_*username* in the **osm** map. For more information, see "Creating the Configuration XML File for the userAdmin Command."

For detailed instructions on using the userAdmin command, see "Using the userAdmin Command."

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

### Schema File

The schema file for the credStoreAdmin command is *OSM_SDK***/SDK/XMLImportExport/models/UserAdmin.xsd**

### Creating the XML Data File for the userAdmin Command

The following is an example XML data file for the userAdmin command (for example, **user.xml**).

```
<userConfig xmlns="http://www.metasolv.com/Provisioning/UserConfig"
 xmlns:oms="http://www.metasolv.com/OMS/OrderModel/2002/06/25"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <user name="test_osm_user1">
      <description>OSM test user 1</description >
      <password>password_from_createEncryptPasswords</password>
      <saltstore>/u01/security/testOsmUser1/salt.store</saltstore>
   </user>
   <user name="test_osm_user2">
      <description>OSM test user 2</description >
      <password>password_from_createEncryptPasswords</password>
      <saltstore>/u01/security/testOsmUser2/salt.store</saltstore>
   </user>
   <clientGroup>
      <user>testOsmUser1</user>
      <user>testOsmUser2</user>
   </clientGroup>
   <automationGroup>
      <user>testOsmUser1</user>
      <user>testOsmUser2</user>
   </automationGroup>
   <wsAPIGroup>
      <user>testOsmUser1</user>
      <user>testOsmUser2</user>
```


Copyright © 2009, 2025, Oracle and/or its affiliates.

```
   </wsAPIGroup>
   <xmlAPIGroup>
      <user>testOsmUser1</user>
      <user>testOsmUser2</user>
   </xmlAPIGroup>
   <workgroup name="testRole1">
      <user>testOsmUser1</user>
   </workgroup>
   <workgroup name="testRole2">
      <user>testOsmUser2</user>
   </workgroup>
</userConfig>
```

Following are descriptions of some of the elements in the XML data file:

- **userConfig/user**: The name of the user to add to the security realm. Use the **name** attribute to contain the name of the user. If this user does not exist in the credential store, the user will also be added to the credential store with the map name **osm** and the key name **osmUser**_*username*.
- **password**: Enter the password that was output from the **createEncryptPassword** command. For more information, see "Using the CreateEncryptPasswords Utility."
- **saltstore**: Enter the current name and location of the **salt.store** file that was created when you ran the **createEncryptPassword** command. For more information, see "Using the CreateEncryptPasswords Utility."
- **clientGroup**: The name of the workgroup (role in Oracle Communications Design Studio) to which the contained users should be added.
- **clientGroup/user**: The name of a user to add to the workgroup/role.

### Creating the Configuration XML File for the userAdmin Command

You should create a configuration file that contains information about your environment.

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

1. Copy the sample XML Import/Export application configuration file *OSM_SDK*/**SDK/XMLImportExport/config/config_sample.xml** and rename it (for example, to **config.xml)**.
2. Edit the **j2eeAdminConnection** section of the file with your installation information.

   Following is the format of the **j2eeAdminConnection** section:

   ```
   <j2eeAdminConnection>
      <user>admin_user</user>
      <password>password</password>
      <protocol>t3</protocol>
      <hostname>host_name</hostname>
      <port>port</port>
   </j2eeAdminConnection>
   ```

   where *admin_user* is an administrative WebLogic Server user, *host_name* is the name of the server that WebLogic Server is running on, and *port* is the listen port for WebLogic Server. The value of the password element does not matter. Do not include your real plain-text password here.

   If you want to connect to the WebLogic server using SSL, see "Using SSL Connections."
3. Edit the **log** tag of the file so that the **logFileUrl** attribute points to a valid location on your system.


Copyright © 2009, 2025, Oracle and/or its affiliates.

4. If you want the userAdmin command to add users to the credential store if not already present, edit the **credentialStore** tag to define the **addUser** attribute as **true**:

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

   This enables the userAdmin command to perform credential store updates.

5. If you configure workgroups using the userAdmin command (the XML data file contains **workgroup** sections), edit the **databaseConnection** section.

   Following is the format of the databaseConnection section:

   ```
   <databaseConnection>
      <user>db_user_name</user>
      <password>password</password>
      <dataSource>jdbc:oracle:thin:@hostname:port:SID</dataSource>
   </databaseConnection>
   ```

   where *db_user_user* is the OSM database user, *hostname* is the name of the server that the database is running on, *port* is the listener port for the database, and *SID* is the identifier for your database. The value of the password element does not matter. Do not include your real plain-text password here.

6. Use the EncryptPasswords utility to insert the encrypted passwords in the file. See "Using the EncryptPasswords Utility" for instructions on using this utility.

### Editing the Configuration Script for the userAdmin Command

You should edit the configuration script with the appropriate values for your environment.

To edit the configuration script for UNIX or Linux:

1. If the following are all true, you do not need to perform this step:
   - In the terminal where you are going to run userAdmin, you have the JAVA_HOME and MIDDLEWARE_HOME environment variables set to the correct values.
   - You are running the **userAdmin.sh** command from the *OSM_SDK*/**SDK/XMLImportExport** directory.
2. If the conditions in step 1 are not all true, edit the *OSM_SDK*/**SDK/XMLImportExport/config.sh** file, and make the following changes:
   - If the JAVA_HOME environment variable is not set, at the very beginning of the "user configuration below here" section, add the following line:

     ```
     export JAVA_HOME="path"
     ```

     where *path* is the path to the appropriate version of Java.
   - If the MIDDLEWARE_HOME variable is not set, uncomment and update the following line so that it points to the location where Oracle Fusion Middleware is installed:

     ```
     #export MIDDLEWARE_HOME=/Oracle/middleware
     ```
   - If you are not running the userAdmin script from the *OSM_SDK*/**SDK/XMLImportExport** directory, update the following line so that it points to the *OSM_SDK*/**SDK/XMLImportExport** directory:

     ```
     export APP_ROOT="$( cd "$( dirname "${BASH_SOURCE[0]}" )" && pwd )"
     ```

     For example:

     ```
     export APP_ROOT="/u01/osmdir/SDK/XMLImportExport"
     ```


Copyright © 2009, 2025, Oracle and/or its affiliates.

3. If you are not running the userAdmin script from the *OSM_SDK*/**SDK/XMLImportExport** directory, move the **config.sh** file to the directory from which you are running the **userAdmin.sh** script.

To edit the configuration script for Windows:

1. If the following are all true, you do not need to perform this step:
   - You have the JAVA_HOME and MIDDLEWARE_HOME environment variables set to the correct values.
   - You are running the **userAdmin.bat** command from the *OSM_SDK*/**SDK/XMLImportExport** directory.
2. If the conditions in step 1 are not all true, edit the *OSM_SDK*/**SDK/XMLImportExport/config.bat** file, and make the following changes:
   - If the JAVA_HOME environment variable is not set, edit the following line to point to the location of the appropriate version of Java:

     ```
     set JAVA_HOME=%JAVA_HOME%
     ```

     For example:

     ```
     set JAVA_HOME=%"C:\Program Files\Java\Java21"%
     ```
   - If the MIDDLEWARE_HOME variable is not set, uncomment and update the following line so that it points to the location where Oracle Fusion Middleware is installed:

     ```
     @rem set MIDDLEWARE_HOME=%MIDDLEWARE_HOME%
     ```
   - If you are not running the userAdmin script from the *OSM_SDK*/**SDK/XMLImportExport** directory, update the following line so that it points to the *OSM_SDK*/**SDK/XMLImportExport** directory:

     ```
     set APP_ROOT=%~sdp0
     ```

     For example:

     ```
     set APP_ROOT="C:\Oracle\OSM\SDK\XMLImportExport"
     ```
3. If you are not running the userAdmin script from the *OSM_SDK*/**SDK/XMLImportExport** directory, move the **config.bat** file to the directory from which you are running the **userAdmin.bat** script.

### Using the userAdmin Command

You can do the following with the userAdmin command:

- add users to WebLogic Server security realms
- add users to WebLogic Server groups
- add users to OSM workgroups (roles)

You should add users to a security realm after they have been added to the credential store. For more information about adding users to the credential store, see "Managing Credentials in the Credential Store." However, if you do add a user using userAdmin command and the user has not already been added to the credential store, the userAdmin command will also add the user to the credential store in the **osm** map.

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 EncryptPasswords utility should be run. See "Using the EncryptPasswords Utility" for more information.


Copyright © 2009, 2025, Oracle and/or its affiliates.

The userAdmin command can be used in two ways. You can run a script on either UNIX/Linux or Windows, or you can run the command as an Ant target. The instructions for both methods are below.

To run the userAdmin command as a script:

1. Encrypt the user passwords for use in the XML data file. See "Using the CreateEncryptPasswords Utility" for more information.
2. Create an XML data file for the userAdmin command. See "Creating the XML Data File for the userAdmin Command."
3. Create a configuration file for the userAdmin command. See "Creating the Configuration XML File for the userAdmin Command."
4. Edit the configuration script with information about your environment. See "Editing the Configuration Script for the userAdmin Command."
5. Run the following command:

   UNIX or Linux:

   ```
   userAdmin.sh data_file config_file
   ```

   Windows:

   ```
   userAdmin.bat data_file config_file
   ```

   where *data_file* is the XML data file that you created in step 2 and *config_file* is the configuration file you created in step 3. This script is ordinarily located in the *OSM_SDK*/**SDK/XMLImportExport** directory, but it is possible to run it from somewhere else. For information about running the script from another location, see "Editing the Configuration Script for the userAdmin Command."

To run userAdmin as an Ant target:

1. Create an XML data file for the userAdmin command. See "Creating the XML Data File for the userAdmin Command."
2. Create a configuration file for the userAdmin command. See "Creating the Configuration XML File for the userAdmin Command."
3. From the *OSM_SDK*/**SDK/XMLImportExport** directory, set the following properties in the **build.properties** file:
   - **xmlie.root.dir**: The XML Import/Export application directory relative to the OSM SDK installation (for example, **/XMLImportExport**).
   - **middlewareHome**: The Fusion Middleware installation directory.
   - **xmlie.root.modelDocument**: The XML data file that you created in step 1.
   - **xmlie.root.configDocument**: The configuration file that you created in step 2.

   The other properties in the **build.properties** file are not used by the userAdmin target.
4. Run the following command:

   ```
   ant userAdmin
   ```

### Calling the 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}"
```


Copyright © 2009, 2025, Oracle and/or its affiliates.

```
    target="userAdmin">
      <property name="wls_admin_user" value="weblogic"/>
      <property name="wls_admin_password" value="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="password">
   <input message=" Enter WebLogic Admin User Password:  "
    addproperty="wls.password">
      <handler classname="org.apache.tools.ant.input.SecureInputHandler"/>
   </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 Server and add the user to appropriate J2EE groups. It can also add the user in the WebLogic Server 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 Server, 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*


Copyright © 2009, 2025, Oracle and/or its affiliates.

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 "osm2", then the map and key values used for it will be:

- map="osm"
- key="osmUser_osm2"

# credStoreAdmin Command

Use the credStoreAdmin command to configure the Java Platform Security 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 run time; this is the recommended method of using the credStoreAdmin command because entering the password at run time is a more secure approach.

For detailed instructions on using the credStoreAdmin command, see "Using the credStoreAdmin Command."

See "Configuring the Java Security Policy for the OSM Credential Store" for instructions on configuring the Java Platform Security 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.

You must create encrypted passwords to use in this script before running it. See "Using the CreateEncryptPasswords Utility" for more information.

**Schema File**

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


Copyright © 2009, 2025, Oracle and/or its affiliates.

### Creating the XML Data File for the credStoreAdmin Command

The following is an example XML data file for the credStoreAdmin command (for example, **credential.xml**). This example uses the map name **osm**. It is a good idea to create this map, since some operations use that as the default value for the map name.

```
<?xml version="1.0" encoding="UTF-8"?>
<CredentialConfig
 xmlns="http://www.metasolv.com/Provisioning/CredentialConfig"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="…/XMLImportExport/models/CredStoreAdmin.xsd">
   <jpsPolicy operation="add">
      <mapname>osm</mapname>
   </jpsPolicy>
   <credential operation="create" overwrite="true">
      <mapname>osm</mapname>
      <keyname>osmUser_osm1</keyname>
      <user>user1</user>
      <password>password_from_createEncryptPasswords</password>
      <saltstore>/u01/security/testOsmUser1/salt.store</saltstore>
   </credential>
</CredentialConfig>
```

Following are descriptions of some of the elements in the XML data file:

- **jpsPolicy**: This element contains the name of a map to add or remove. Use the **operation** attribute to determine the action to perform on the map. Available values for the **operation** attribute are:
  - **add**: Add a new credential store map. It is a good idea to use **osm** as the name of the primary (or only) map.
  - **remove**: Remove an existing credential store map.
- **credential**: This element contains information about a user to be added, modified, or removed from the credential store. Use the **operation** attribute to determine the action to perform on the user. Available values for the **operation** attribute are:
  - **create**: Add a new user.
  - **modify**: Change the password for an existing user.
  - **delete**: Delete an existing user from the credential store.

  Set the **overwrite** attribute to **true**.
- **mapname**: The name of the credential store to which the user should be added.
- **keyname**: The key value for the user in the credential store. Oracle recommends using **osmUser**_*username* as the key value, where *username* is the name of the OSM user.
- **user**: The name of the user to add to the credential store.
- **password**: Enter the password that was output from the **createEncryptPasswords** command. For more information, see "Using the CreateEncryptPasswords Utility."
- **saltstore**: Enter the current name and location of the **salt.store** file that was created when you ran the **createEncryptPasswords** command. For more information, see "Using the CreateEncryptPasswords Utility."

### Creating the Configuration File for the credStoreAdmin Command

You should create a configuration file that contains information about your environment.


Copyright © 2009, 2025, Oracle and/or its affiliates.

To create the configuration file (for example, **config.xml**):

1. Copy the sample XML Import/Export application configuration file *OSM_SDK*/**SDK/XMLImportExport/config/config_sample.xml** and rename it (for example, to **config.xml)**.
2. Edit the **j2eeAdminConnection** section of the file with your installation information.

   Following is the format of the **j2eeAdminConnection** section:

   ```
   <j2eeAdminConnection>
      <user>admin_user</user>
      <password>password</password>
      <protocol>t3</protocol>
      <hostname>host_name</hostname>
      <port>port</port>
   </j2eeAdminConnection>
   ```

   where *admin_user* is an administrative WebLogic Server user, *host_name* is the name of the server that WebLogic Server is running on, and *port* is the listen port for WebLogic Server. The value of the password element does not matter. Do not include your real plain-text password here.

   If you want to connect to the WebLogic server using SSL, see "Using SSL Connections."
3. Edit the **log** tag of the file so that the **logFileUrl** attribute points to a valid location on your system.
4. Use the EncryptPasswords utility to insert the encrypted passwords in the file. See "Using the EncryptPasswords Utility" for instructions on using this utility.

### Editing the credStoreAdmin Script

You should edit the credStoreAdmin script with the appropriate values for your environment.

To edit the script for UNIX or Linux:

1. Edit the *OSM_SDK*/**SDK/XMLImportExport/credStoreAdmin.sh** file, and make the following changes:
   - If the JAVA_HOME environment variable is not set, modify the following line to contain the path to the appropriate version of Java:

     ```
     export JAVA_HOME=$JAVA_HOME
     ```
   - Modify the following line so that it points to the *OSM_SDK*/**SDK/XMLImportExport** directory:

     ```
     export APP_ROOT=$APP_ROOT
     ```

     For example:

     ```
     export APP_ROOT="/u01/osmdir/SDK/XMLImportExport"
     ```
   - If the MIDDLEWARE_HOME variable is not set, modify the following line so that it points to the location where Oracle Fusion Middleware is installed:

     ```
     export MIDDLEWARE_HOME=$MIDDLEWARE_HOME
     ```

To edit the script for Windows:

1. If the following are all true, you do not need to perform this step:
   - You have the JAVA_HOME and MIDDLEWARE_HOME environment variables set to the correct values.


Copyright © 2009, 2025, Oracle and/or its affiliates.

- You are running the **credStoreAdmin.bat** command from the *OSM_SDK*/**SDK/XMLImportExport** directory.

2. If the conditions in step 1 are not all true, edit the *OSM_SDK*/**SDK/XMLImportExport/credStoreAdmin.bat** file, and make the following changes:
   - If the JAVA_HOME environment variable is not set, edit the following line to point to the location of the appropriate version of Java:

     ```
     set JAVA_HOME=%JAVA_HOME%
     ```

     For example:

     ```
     set JAVA_HOME=%"C:\Program Files\Java\Java21"%
     ```

   - If you are not running the **credStoreAdmin.bat** script from the *OSM_SDK*/**SDK/XMLImportExport** directory, update the following line so that it points to the *OSM_SDK*/**SDK/XMLImportExport** directory:

     ```
     set APP_ROOT=%~sdp0
     ```

     For example:

     ```
     set APP_ROOT="C:\Oracle\OSM\SDK\XMLImportExport"
     ```

   - If the MIDDLEWARE_HOME variable is not set, uncomment and update the following line so that it points to the location where Oracle Fusion Middleware is installed:

     ```
     @rem set MIDDLEWARE_HOME=%MIDDLEWARE_HOME%
     ```

### Using the credStoreAdmin Command

You can do the following with the credStoreAdmin command:

- add credential maps
- add users to the credential store
- change the password for users in the credential store
- remove users from the credential store

For more information about adding users to the credential store, see "Managing Credentials in the Credential Store."

The credStoreAdmin command can be used in two ways. You can run a script on either UNIX/Linux or Windows, or you can run the command as an Ant target. The instructions for both methods are below.

To run the credStoreAdmin command as a script:

1. Encrypt the user passwords for use in the XML data file. See "Using the CreateEncryptPasswords Utility" for more information.
2. Create an XML data file for the credStoreAdmin command. See "Creating the XML Data File for the credStoreAdmin Command."
3. Create a configuration file for the credStoreAdmin command. See "Creating the Configuration File for the credStoreAdmin Command."
4. Edit the configuration script with information about your environment. See "Editing the credStoreAdmin Script."
5. Run the following command:

   UNIX or Linux:


Copyright © 2009, 2025, Oracle and/or its affiliates.

```
credStoreAdmin.sh data_file config_file
```

Windows:

```
credStoreAdmin.bat data_file config_file
```

where *data_file* is the XML data file that you created in step 2 and *config_file* is the configuration file you created in step 3. This script is ordinarily located in the *OSM_SDK*/**SDK/XMLImportExport** directory, but it is possible to run it from somewhere else. For information about running the script from another location, see "Editing the credStoreAdmin Script."

To run credStoreAdmin as an Ant target:

1. Create an XML data file for the credStoreAdmin command. See "Creating the XML Data File for the credStoreAdmin Command."
2. Create a configuration file for the credStoreAdmin command. See "Creating the Configuration File for the credStoreAdmin Command."
3. From the *OSM_SDK*/**SDK/XMLImportExport** directory, set the following properties in the **build.properties** file:
   - **xmlie.root.dir**: The XML Import/Export application directory relative to the OSM SDK installation (for example, **/XMLImportExport**).
   - **middlewareHome**: The Fusion Middleware installation directory.
   - **xmlie.root.modelDocument**: The XML data file that you created in step 1.
   - **xmlie.root.configDocument**: The configuration file that you created in step 2.

   The other properties in the **build.properties** file are not used by the credStoreAdmin target.
4. Run the following command:

   ```
   ant credStoreAdmin
   ```

### Calling the 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:

```
<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:  "
    addproperty="wls.password">
      <handler classname="org.apache.tools.ant.input.SecureInputHandler"/>
```


Copyright © 2009, 2025, Oracle and/or its affiliates.

```
   </input>
</target>
```

### 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 Java Platform Security policy for your custom credential store map and to manage credentials in the credential store.

**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 credStoreAdmin command to configure the OSM user in the credential store. (See "credStoreAdmin Command" for more information.
Example credential data in the XML file with credential information:

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

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

  Example credential data in XML file with Java Platform Security policy information:


Copyright © 2009, 2025, Oracle and/or its affiliates.

```
……
   <jpspolicy operation="add">
      <mapname>osm_systemAmap</mapname>
   </jpspolicy >
……
```

# OSM User Security and Credential Store API Reference Material

To develop OSM cartridges to use the credential store offered through CSF (see "Using 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.

## 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 CSF look up the single instance of the CredentialStore object.

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

**Package name**

com.mslv.oms.security.credstore

**Attributes**

Name: store

Type: oracle.security.jps.service.credstore.CredentialStore

Description: A reference object to the Java Platform Security 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.

**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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

### Error Conditions

Improper Java Platform Security 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


Copyright © 2009, 2025, Oracle and/or its affiliates.

**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 "user name/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


Copyright © 2009, 2025, Oracle and/or its affiliates.

**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 Java Platform Security 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**

```
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**

```
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


Copyright © 2009, 2025, Oracle and/or its affiliates.

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 "Modeling Behaviors" in *OSM Modeling 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:


Copyright © 2009, 2025, Oracle and/or its affiliates.

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.

# ObjectelHTTPAdapter

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

For detailed information on data provider adapters, see "Modeling Behaviors" in *OSM Modeling Guide*.

**Description**

Built-in adapter. Objectel HTTP adapter.

**Attributes**

- CREDENTIAL_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".

- CREDENTIAL_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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

### 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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

**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 "user name/password".

Details on operation getCredential():

```
/**
 * Get user name and password values in string format from credential store,
 * given map and key values.
 *
 * @param map
 *     Map name of the credential stored in domain credential store.
 * @param key
 *     Key name of the credential stored in domain credential store.
 * @return A String that contains user name and password values, separated by "/"
 * @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.
 */
    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 Java Platform Security configuration can cause creation of PasswordCredStore to fail.

Message Name: ViewRuleFailedException


Copyright © 2009, 2025, Oracle and/or its affiliates.

Message Text: "Fail to create PasswordCredStore."

**Usage Notes**

This API is often used in XQuery scripts.

# AutomationContext

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 Credentials."

**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
 *     Map name of the credential stored in domain credential store.
 * @param key
 *     Key name of the credential stored in domain credential store.
 *
 * @return User name and password for the user in this XML format:
 *     <Credential xmlns=\"urn:com:metasolv:oms:xmlapi:1\">
 *         <Username>username</Username>
 *         <Password>password</Password>
 *     </Credential>
 * @throws CredStoreException
 *     If the application cannot access credential store, or if there is no
```


Copyright © 2009, 2025, Oracle and/or its affiliates.

```
 *       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
AutomationException, RemoteException;
```

**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 Java Platform Security 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**

> **Note**
>
> This example assumes your map name is (**osmTest**).

```
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()
```


Copyright © 2009, 2025, Oracle and/or its affiliates.

B

# Tools for Performance Testing, Tuning, and Troubleshooting

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 Remote Console

Oracle WebLogic Remote Console is a web-based, graphical user interface that enables you to monitor the WebLogic server.

WebLogic Remote Console is available in two formats:

- Desktop WebLogic Remote Console, a desktop application installed on your computer.
- Hosted WebLogic Remote Console, a web application deployed to an Administration Server and accessed through a browser.

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:

https://visualvm.github.io/

## 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
```


Copyright © 2009, 2025, Oracle and/or its affiliates.

> **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 11*g* 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:

http://www.soapui.org/

# 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


Copyright © 2009, 2025, Oracle and/or its affiliates.

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 *OSM Installation Guide*.

# 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 website:

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-1.36.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.

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:

http://www.ateam-oracle.com/analyzing-thread-dumps-in-middleware-part-4-2/


Copyright © 2009, 2025, Oracle and/or its affiliates.

# 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 website:

https://support.oracle.com

To run OSW Black Box as a background process, run the following command:

```
nohup ./startOSWbb.sh &
```

To stop OSW Black Box processes, run the following command:

```
./stopOSWbb.sh
```

To generate an OSW Black Box Analyzer report, run the following command:

```
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, 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 website:


Copyright © 2009, 2025, Oracle and/or its affiliates.

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, 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, 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, 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, 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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

For more information about Oracle RAC Cluster RDA reports, see the knowledge article [Doc ID 359395.1], available from the Oracle support website:

https://support.oracle.com

To set up a Oracle RAC Cluster RDA report, 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 website:

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.


Copyright © 2009, 2025, Oracle and/or its affiliates.

C

# OSM Installed Components

When you run the various installers and follow the configuration procedures, the system creates and populates a directory structure and installs the components that OSM requires. For information about installing Oracle Communications Order and Service Management (OSM), see *OSM Installation Guide*.
This appendix describes the OSM directory structure and the components that are automatically configured by OSM Installer.

## OSM Directory Structure

Table C-1 lists and describes the OSM directories.

**Table C-1 OSM Directories**

| Directory | Description |
|---|---|
| *OSM_home*/**bin** | Contains copies of the **oms.ear** file and cartridge management web service (**cartridge_management_ws**) application. |
| *OSM_home*/**Database** | Contains the following database installation files, in the **/Database/osm-db-installer-core/install** directory:<br>• om_$install$plan_actions.csv<br>• dbInstaller.log<br>• omsmodel_l10n.jar |
| *OSM_home*/**ProductCartridges** | Contains the product cartridges. |
| *OSM_SDK*/**SDK** | Contains the SDK and SDK samples (optional). The following sub-directories are available in the SDK:<br>• **Automation**: Used by Design Studio for compiling when there are automation plug-ins.<br>• **CartridgeManagementTool**: Used for compiling and deploying task assignment.<br>• **Compliance**: Needed to run the compliance tool. See *OSM Installation Guide* for more information.<br>• **Customization**: Needed to manage the **oms.ear** file when updates are required for localization files.<br>• **Reporting**: Needed to run reporting queries.<br>• **Samples**: Contains development samples, for example, web services samples.<br>• **WebService**: Contains web service definition language (WSDL) files.<br>• **XMLImportExport**: Contains command line utilities for metadata management.<br>**Note**: Oracle recommends using Design Studio for cartridge management.<br>• **XMLSchema**: Contains schema definitions for the XML API. |


Copyright © 2009, 2025, Oracle and/or its affiliates.

# Productized Cartridges

The OSM DB Installer deploys the following cartridge:

- Job Order cartridge: Enables the job control order feature. For information about using job orders, see *OSM Modeling Guide*.

# WebLogic Installed Components

This section lists and describes the OSM WebLogic installed components and configurations.

## WebLogic Deployments

Table C-2 lists the WebLogic deployments.

**Table C-2 WebLogic Deployments**

| Deployment Name | Description |
|---|---|
| cartridge_management_ws | Application used to deploy, undeploy, and query cartridges. |
| oms | The Order and Service Management application. |

## WebLogic Configuration

This section lists and describes the WebLogic configuration.

### Coherence Clusters

The Installer configures the following coherence cluster:

- osmCoherenceCluster*uniqueid*

  where *uniqueid* is a random four digit number used to generate a unique string.

For information about configuring Oracle coherence for an OSM cluster, see *OSM Installation Guide*.

### Work Managers

See "Using Work Managers to Prioritize Work" for information about work managers.

> **Note**
>
> Also, see the *OSM default Work Managers, Constraints related to upgrade process (Doc ID 3019290.1)* knowledge article on My Oracle Support, if you are upgrading from OSM version 7.3.1 or lower (source version) to OSM version 7.3.5 or higher (target version).

The following table lists and describes the work managers that are configured.


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table C-3 Configured Work Managers**

| Name | Type | Description |
|---|---|---|
| osmAutomationWorkManager | Work manager | Used to process work performed by automation tasks. |
| osmOmClientWorkManager | Work manager | Used to process requests from manual users using the Order Management web client. |
| osmTaskClientWorkManager | Work manager | Used to process requests from manual users using the Task web client. |
| osmWsHttpWorkManager | Work manager | Used to process OSM HTTP WebService requests. |
| osmWsJmsWorkManager | Work manager | Used to process OSM JMS WebService requests. |
| osmXmlWorkManager | Work manager | Used to process requests coming in from external clients for the OSM XML API. |
| osmAutomationMaxThreadConstraint | Maximum Thread Constraint | Used to ensure that the maximum thread size is not more than the JDBC connection pool size. |
| osmGuiMaxThreadConstraint | Maximum Thread Constraint | N/A |
| osmHttpApiMaxThreadConstraint | Maximum Thread Constraint | N/A |
| osmJmsApiMaxThreadConstraint | Maximum Thread Constraint | N/A |
| osmGuiMinThreadConstraint | Minimum Thread Constraint | N/A |

## JMS Servers

The following JMS server is configured:

- oms_jms_server

## JMS Module

The following JMS module is configured:

- oms_jms_module: Contains the JMS system resources for OSM.

## Queues and Topics

### Distributed Destinations in a Cluster

In a clustered WebLogic deployment, a distributed destination is created for each queue and topic. This distributed destination then has members created (a queue or topic) on each of the managed servers.

For example, in a clustered deployment that contains two managed servers called **MS1** and **MS2**, the **osm_behavior_queue** queue has the following:

- A distributed destination called **osm_behavior_queue**


Copyright © 2009, 2025, Oracle and/or its affiliates.

- Two members called **oms_jms_server@ms1@osm_behavior_queue** and **oms_jms_server@ms2@osm_behavior_queue**

**Table C-4 Queues and Topics Configured for OSM**

| Name | Type | JNDI | Description |
|---|---|---|---|
| osm_behavior_queue | Queue | mslv/oms/oms1/internal/jms/behaviors | Used to customize task assignment. |
| osm_cartridge_deploy | Queue | mslv/provisioning/internal/ejb/deployCartridgeQueue | Internal use only. |
| osm_events | Queue | mslv/oms/oms1/internal/jms/events | Internal use only. |
| osm_order_events | Topic | mslv/provisioning/external/orderevents | Where order lifecycle events are published. |
| osm_order_updates | Queue | mslv/provisioning/internal/ejb/orderupdates | Internal use only. |
| osm_signal_topic | Topic | mslv/oms/oms1/internal/jms/InternalSignalTopic | Internal use only. |
| osm_ws_cluster_correlates | Queue | oracle/communications/ordermanagement/WebServiceClusterCorrelateQueue | Used to route OSM service requests, in the context of an existing order in a WebLogic Server cluster, to the managed server that owns the order. |
| osm_ws_cluster_requests | Queue | oracle/communications/ordermanagement/WebServiceClusterRequestQueue | Used to receive requests for order amendments in a WebLogic Server cluster. Can also be used for new order creation requests. |
| osm_ws_cluster_responses | Queue | oracle/communications/ordermanagement/WebServiceClusterResponseQueue | Used to fetch a correlated message from the web service cluster correlate queue, analyze the original JMS properties, and forward the response to the original JMSReplyTo. |
| osm_ws_requests | Queue | oracle/communications/ordermanagement/WebServiceQueue | Used to receive order creation and amendment requests in a single-server WebLogic Server environment. Can also be used for new order creation requests for a WebLogic Server cluster. |
| osmErrorQueue | Queue | oracle/communications/ordermanagement/osm/ErrorQueue | N/A |
| ErrorQueue | Queue | oracle/error_queue | N/A |

## Quotas

Quotas are used to control the allotment of system resources available to OSM destinations (queues or topics).

Table C-5 lists and describes the quotas that are configured.


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table C-5 Configured Quotas**

| Quota Name | Destination Name |
|---|---|
| oms_behavior_queue.Quota | oms_behavior_queue |
| oms_cartridge_deploy.Quota | oms_cartridge_deploy |
| oms_events.Quota | oms_events |
| oms_order_events.Quota | oms_order_events |
| oms_order_updates.Quota | oms_order_updates |
| oms_signal_topic.Quota | oms_signal_topic |
| oms_ws_cluster_correlates.Quota | oms_ws_cluster_correlates |
| oms_ws_cluster_requests.Quota | oms_ws_cluster_requests |
| oms_ws_cluster_responses.Quota | oms_ws_cluster_responses |
| oms_ws_requests.Quota | oms_ws_requests |
| OrchestrationDependeciesQueue.Quota | OrchestrationDependeciesQueue |
| osmErrorQueue.Quota | osmErrorQueue |

## Connection Factories

The following connection factories are configured:

- osm_connection_factory
- osmExternalClientConnectionFactory: Use this connection factory to submit OSM Web Service and XML API requests (including order creation) from an external system.
- oms_events_connection_factory

## Destination Key

The following destination key is configured:

- osmDescendingPriorityDestinationKey

For information about configuring destination keys, see Oracle WebLogic Remote Console Online Help.

## JMS Template

A JMS template for the OSM destinations is created. This template applies recommended defaults for the following settings:

- Redelivery Limit
- Redelivery Delay
- Error Destination

The following JMS template is configured:

- osmJmsTemplate

For information about JMS templates, see Oracle WebLogic Remote Console Online Help.


Copyright © 2009, 2025, Oracle and/or its affiliates.

## File T3

The installer configures the following file:

- Oms_Remote_File_System

For information about configuring the T3 file path to support attachments, see *OSM Installation Guide*.

## Data Sources

OSM Installer configures the following types of data sources:

- Active Active
- Active Passive

The Installer also configures the following standalone data source:

- oms_pool

Table C-6 lists the load balancing data sources configured by the Installer. For information about connecting Oracle RAC with JDBC multi data sources, see *OSM Installation Guide*.

**Table C-6 Load Balancing Data Sources Configured by OSM Installer**

| Name | Type | Description |
|---|---|---|
| oms_pool_group | Multi | One oms_pool_group is required for each target server for failover. |
| osm_pool_rac_group | Generic | One osm_pool_rac_group is required for each target server. For example:<br>**osm_pool_rac2_group_b** |

Table C-7 lists the failover data sources configured by the installer.

**Table C-7 Failover Data Sources Configured by OSM Installer**

| Name | Type | Description |
|---|---|---|
| oms_pool | Multi | One oms_pool data source is configured for the cluster. |
| osm_pool_rac | Generic | |

# Users and Groups

This section describes the users and groups that are created by the OSM Installer. For information about the users and groups that are created by the WebLogic Server Installer, see *Fusion Middleware Securing Resources Using Roles and Policies for Oracle WebLogic Server*.

A security realm consists of a set of configured security providers, users, groups, security roles, and security policies that protect WebLogic resources. To access WebLogic resources that belong a security realm, a user must be defined in that realm.

Add users to groups by using WebLogic Remote Console. For more information, see "Adding Users to Groups in the WebLogic Remote Console."


Copyright © 2009, 2025, Oracle and/or its affiliates.

When OSM is installed, several default accounts are created in WebLogic Security Realm.

Table C-8 and Table C-9 list the users, groups, and their descriptions.

**Table C-8 Default Application Accounts**

| Default Account | Description | Assigned Groups |
|---|---|---|
| Default OSM Administrator account | The OSM Administrator account name and password are specified during OSM installation. The default is **admin**. | OMS_user_assigner<br>OMS_workgroup_manager<br>OMS_client<br>OSM_fallout<br>OMS_xml_api<br>OMS_ws_api<br>OMS_ws_diag |
| oms-automation | Used for processing OSM automation and email notifications. | OSM_automation |
| osm-internal | Used for internal processing when operation must be performed on behalf of the application rather than on behalf of the user. | Administrators<br>OSM_automation |
| Default Design Studio administrator account | The Design Studio administrator account and password for managing OSM cartridges are specified during OSM installation. The default is **deployAdmin**. | Cartridge_Management_WebService |

**Table C-9 Default Groups in the Server Security Realm**

| Group Name | Description |
|---|---|
| Cartridge_Management_WebService | OSM cartridge management access. This group also allows access to the features in the View Cartridges option in the Order Management web client. See *OSM Order Management Web Client User's Guide* for information about this feature.<br>This belongs to the Administrator group. |
| OMS_cache_manager | Monitoring and control of order cache. |
| OMS_client | Provides access to Order Management, Task, and Order Lifecycle Manager web clients. |
| OMS_user_assigner | Users in this group can do the following:<br>• Associate users with workgroups<br>• Manage notifications<br>• Manage system events<br>• Refresh the metadata cache |
| OMS_workgroup_manager | Users in this group can do the following:<br>• Modify workgroup schedules<br>• Delete workgroups<br>• Manage notifications<br>• Manage system events<br>• Refresh the metadata cache |
| OMS_ws_api | WebService API access |
| OMS_ws_diag | WebService diagnostic access |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table C-9 (Cont.) Default Groups in the Server Security Realm**

| Group Name | Description |
|---|---|
| OMS_xml_api | XML API access |
| OSM_automation | This role is granted to the user that is allowed to run the automation plug-in.<br>The Installer automatically assigns the oms-automation user to this group, and Design Studio automatically assigns users specified in the Run-As property of the automation plug-in. |
| OSM_USER_manager | Manage user accounts |
| osmEntityClientGroup | Entity UI access |
| osmRestApiGroup | OSM REST API |
| OMS_log_manager | Manage log settings |
| OSM_fallout | Users in this group can do the following:<br>• View and access fallout orders from the Fallout Orders Dashboard<br>• Manage fallout orders, perform investigative and write actions on fallout orders |

# Database Configuration

The OSM DB Installer creates the following database schemas:

- Core schema
- Rule engine schema
- Reporting schema

Table C-10 shows the roles and permissions that are granted to database schema users.

**Table C-10 Roles and Permissions Granted to Database Schema Users**

| Schema | Roles and Permissions |
|---|---|
| Core | resource<br>create session<br>create table<br>create view<br>create synonym<br>create sequence<br>alter session<br>unlimited tablespace<br>create job<br>create any context<br>execute on dbms_lock<br>select on v_$parameter |


Copyright © 2009, 2025, Oracle and/or its affiliates.

**Table C-10 (Cont.) Roles and Permissions Granted to Database Schema Users**

| Schema | Roles and Permissions |
|---|---|
| Rule engine | resource<br>create session<br>create table<br>create view<br>create synonym<br>create sequence<br>alter session<br>unlimited tablespace |
| Reporting | resource<br>create session<br>create view<br>create synonym<br>unlimited tablespace |


Copyright © 2009, 2025, Oracle and/or its affiliates.